



# 概述

Casdoor 是一个基于 OAuth 2.0、OIDC、SAML 和 CAS 的，UI 优先的[身份和访问管理\(IAM\)](#)/[单点登录 \(SSO\)](#) 平台。

You need to enable JavaScript to run this app.

Casdoor 可为网页和应用程序用户的登录请求提供服务。

# Casdoor 的特性：

1. Casdoor 遵循前后端分离架构，采用 Golang 进行开发。它支持高同步，提供基于网页的用户界面管理，并支持10多种语言的本地化。
2. Casdoor 支持第三方应用登录，如 GitHub、谷歌、QQ、微信等，并支持通过插件扩展第三方登录。
3. Casdoor 支持基于 [Cassbin](#) 的授权管理。它支持 ACL、RBAC、ABAC 和 RESTful鉴权管理模式。
4. Casdoor 提供了手机验证码、电子邮件验证码以及重置密码的功能。
5. Casdoor 支持日志的审计和记录。
6. Casdoor 可以使用阿里云、腾讯云、七牛云提供的图片CDN云存储功能。
7. Casdoor 允许自定义注册、登录以及找回密码页面。
8. 通过数据库同步支持与现有系统的集成，从而能够顺利过渡到 Casdoor。
9. Casdoor 支持主流数据库: MySQL、PostgreSQL、SQL Server 等, 并支持扩展插件以支持新的数据库。

# 它如何工作?



## 步骤0（前置知识）

1. Casdoor的授权程序基于OAuth 2.0协议。强烈建议简要了解下 OAuth 2.0 的运行原理。你可以通过此处了解 [OAuth 2.0 简介](#)。

## Abstract Protocol Flow



### 步骤 1 (授权请求)

您的应用（或网站等）应该以 `endpoint/login/oauth/authorize?client_id=xxx&response_type=code&redirect_uri=xxx&scope=read&state=xxx` 这种格式来编写 URL。并使用你的 Casdoor 的网站 URL 替换 `endpoint`，以及使用您的信息替换 `xxx`。

#### ① 提示

对于 `xxx` 的部分需要写上什么？

- 对于 `client_id`: 您可以在每个应用程序里找到它
- 对于 `redirect_uri`: 您应该将此设置为您自己的应用程序回调 URL。Casdoor 将使用此信息在授权后发送回响应。
- 对于 `state`: 您应该在此处填写您的应用程序名称。

应用程序将会提示用户：“嘿，我需要一些资源，且该资源需要您的许可我才能访问。您是否可以转到这个网址并为我输入您的用户名和密码？”

使用正确组合的 URL，应用就会向此 URL 发起请求，完成 授权请求。

## 步骤 2（授权认证）

This step is straightforward: the user is redirected to the URL composed in Step 1, and the user will see the login page from Casdoor. 通过在登录页面输入正确的用户名和认证信息，Casdoor 现在能成功识别用户，将发送 `code` 和 `status` 返回第 1 步中设置的回调URL。

用户打开网址并向Casdoor提供凭据。 Casdoor will say: "*Looking good ~ this is the user who is authorizing the Application to get the code and state. I know this user in my database, and I will send the code and state back to the Application using the callback URL (redirect\_uri)*"

With these two pieces of information sent back to your Application, the authorization is granted to the app, and the `Authorization Grant` is completed.



提示

Casdoor also provides third-party logins. In this case, instead of seeing the credential entry page, you will see a list of third-party providers. You can log in to your app using these providers, with Casdoor acting as a middle layer (middleware).

## 步骤 3（授权认证）

In this step, your Application already has the code from Step 2, and it will speak to Casdoor: "*Hey, the user agreed to give me the code. Can you verify this code and give me the access\_token ?*"

## 步骤 4（访问令牌）

Casdoor responds to your Application: "*You know what, this code seems legit. You must be the right Application. Here's the access\_token for you.*" With this `code`, Casdoor confirms that it is an authorized Application (authorized by the correct user in Step 2) trying to obtain the `access_token` (which will be used later to access more resources).

## 步骤 5 (访问令牌)

In this step, your Application says: "Nice! I just got the fresh-and-tasty `access_token`. Now I can use it to access something more valuable from the `Resource Server`!"

Your Application then turns to the `Resource Server` and says: "Hey buddy, can you check out this `access_token`? I received it from Casdoor. Do you want to verify if this is the correct token you issued to Casdoor?"

## 步骤 6 (受保护资源)

The `Resource Server` responds to your Application: "Not bad. It seems just like the one I issued to Casdoor, and Casdoor says whoever holds this `access_token` can access these `Protected Resources`. So go ahead and take it!"

And that's basically how Casdoor works with your Application.

### ⓘ 提示

Casdoor can act as both an `Authorization Server` and a `Resource Server`. In other words, Casdoor authorizes your Application to access resources, usually the currently logged-in user's information, from Casdoor's database.

## 在线演示

### Casdoor

这里是一个由Casbin部署的在线演示。

- [Casdoor官方演示](#)

全局管理员登录：

- 用户名：`admin`

- 密码:

## Casbin-OA

Casbin-OA 是 Casbin 的 web 应用程序之一。它使用 Casdoor 进行身份验证。

- [Casbin-OA](#)
- 源码地址: <https://github.com/casbin/casbin-qa>

## Casnnode

Casnnode 是由 Casbin 社区开发的官方论坛。

它使用 Casdoor 作为认证平台并管理成员。

- [Casnnode](#)
- 源码地址: <https://github.com/casbin/casnnode>

## 结构

Casdoor 包括以下两部分:

名称	描述	语言	源代码
前端	Casdoor的前端 Web界面	JavaScript + React	<a href="https://github.com/casdoor/casdoor/tree/master/web">https://github.com/casdoor/casdoor/tree/master/web</a>
后端	Casdoor后端 RESTful API	Golang + Beego + SQL	<a href="https://github.com/casdoor/casdoor">https://github.com/casdoor/casdoor</a>

# 核心概念

As a Casdoor administrator, you should be familiar with at least four core concepts: Organization, User, Application, and Provider.



提示

In the following parts, we will use the demo site <https://door.casdoor.com> as an example.

## 组织

在Casdoor中，组织是用户和应用程序的容器。For example, all the employees of a company or all the customers of a business can be abstracted as one organization. The Organization class definition is shown below:

```
type Organization struct {
    Owner      string `xorm:"varchar(100) notnull pk" json:"owner"`
    Name       string `xorm:"varchar(100) notnull pk" json:"name"`
    CreatedTime string `xorm:"varchar(100)" json:"createdTime"`

    DisplayName     string `xorm:"varchar(100)" json:"displayName"`
    WebsiteUrl     string `xorm:"varchar(100)" json:"websiteUrl"`
    Favicon        string `xorm:"varchar(100)" json:"favicon"`
    PasswordType   string `xorm:"varchar(100)" json:"passwordType"`
    PasswordSalt   string `xorm:"varchar(100)" json:"passwordSalt"`
    PhonePrefix    string `xorm:"varchar(10)" json:"phonePrefix"`
    DefaultAvatar  string `xorm:"varchar(100)" json:"defaultAvatar"`
    Tags           []string `xorm:"mediumtext" json:"tags"`
    MasterPassword string `xorm:"varchar(100)" json:"masterPassword"`
    EnableSoftDeletion bool `json:"enableSoftDeletion"`
    IsProfilePublic bool `json:"isProfilePublic"`

    AccountItems []*AccountItem `xorm:"varchar(2000)" json:"accountItems"`
}
```

## 用户

In Casdoor, a user can log into an application. Each user can belong to only one organization but can log into multiple applications owned by the organization. Currently, there are two types of users in Casdoor:

- built-in organization users, such as `built-in/admin`: global administrators who have full administrative power on the Casdoor platform.
- Other organizations' users, such as `my-company/alice`: normal users who can sign up, sign in, sign out, change their own profile, etc.

In the Casdoor API, a user is typically identified as `<organization_name>/<username>`. For example, the default administrator of Casdoor is denoted as `built-in/admin`. Additionally, the User class definition includes an `id` property, which is a UUID like `d835a48f-2e88-4c1f-b907-60ac6b6c1b40` and can be chosen as a user's ID by an application.



提示

For applications that are only for one organization, it's possible to use `<username>` instead of `<organization_name>/<username>` as the user ID across the application for simplicity.

Here's the User class definition:

```
type User struct {
```

# 应用程序

An **application** represents a web service that needs to be protected by Casdoor, such as a forum site, an OA system, or a CRM system.

```
type Application struct {
    Owner          string      `xorm:"varchar(100) notnull pk" json:"owner"`
    Name           string      `xorm:"varchar(100) notnull pk" json:"name"`
    CreatedTime   string      `xorm:"varchar(100)" json:"createdTime"`
    DisplayName    string      `xorm:"varchar(100)" json:"displayName"`
    Logo           string      `xorm:"varchar(100)" json:"logo"`
    HomepageUrl  string      `xorm:"varchar(100)" json:"homepageUrl"`
    Description    string      `xorm:"varchar(100)" json:"description"`
    Organization   string      `xorm:"varchar(100)" json:"organization"`
    Cert           string      `xorm:"varchar(100)" json:"cert"`
    EnablePassword bool       `json:"enablePassword"`
    EnableSignUp   bool       `json:"enableSignUp"`
    EnableSigninSession bool     `json:"enableSigninSession"`
    EnableCodeSignin bool     `json:"enableCodeSignin"`
    Providers      []*ProviderItem `xorm:"mediumtext" json:"providers"`
    SignupItems    []*SignupItem  `xorm:"varchar(1000)" json:"signupItems"`
    OrganizationObj *Organization `xorm:"-" json:"organizationObj"`
    ClientId       string      `xorm:"varchar(100)" json:"clientId"`
    ClientSecret   string      `xorm:"varchar(100)" json:"clientSecret"`
    RedirectUris   []string    `xorm:"varchar(1000)" json:"redirectUris"`
    TokenFormat    string      `xorm:"varchar(100)" json:"tokenFormat"`
    ExpireInHours  int        `json:"expireInHours"`
    RefreshExpireInHours int        `json:"refreshExpireInHours"`
    SignupUrl      string      `xorm:"varchar(200)" json:"signupUrl"`
    SigninUrl      string      `xorm:"varchar(200)" json:"signinUrl"`
    ForgetUrl      string      `xorm:"varchar(200)" json:"forgetUrl"`
    AffiliationUrl string      `xorm:"varchar(100)" json:"affiliationUrl"`
    TermsOfUse      string      `xorm:"varchar(100)" json:"termsOfUse"`
    SignupHtml      string      `xorm:"mediumtext" json:"signupHtml"`
    SigninHtml      string      `xorm:"mediumtext" json:"signinHtml"`
}
```

Each application can have its own customized sign-up page, sign-in page, and more. The root login page `/login` (e.g., <https://door.casdoor.com/login>) is the sign-in page only for Casdoor's built-in application: `app-built-in`.

应用程序是用户登录到Casdoor的“入口”或“界面”。A user must go through one application's sign-in page to log into Casdoor.

应用程序	Sign-up page URL	Sign-in page URL
内置应用 程序	<a href="https://door.casdoor.com/signup">https://door.casdoor.com/ signup</a>	<a href="https://door.casdoor.com/login">https://door.casdoor.com/login</a>
casnode 论坛系统	<a href="https://door.casdoor.com/signup/app-casnode">https://door.casdoor.com/ signup/app-casnode</a>	<a href="https://door.casdoor.com/login/oauth/&lt;br/&gt;authorize?client_id=014ae4bd048734ca2dea&amp;response_type=code&amp;redirect_uri=http://localhost:9000/&lt;br/&gt;callback&amp;scope=read&amp;state=casdoor">https://door.casdoor.com/login/oauth/ authorize?client_id=014ae4bd048734ca2dea&amp;response_type=code&amp;redirect_uri=http://localhost:9000/ callback&amp;scope=read&amp;state=casdoor</a>
casbin 的OA系 统	<a href="https://door.casdoor.com/signup/app-casbin-&lt;br/&gt;oa">https://door.casdoor.com/ signup/app-casbin- oa</a>	<a href="https://door.casdoor.com/login/oauth/&lt;br/&gt;authorize?client_id=0ba528121ea87b3eb54d&amp;response_type=code&amp;redirect_uri=http://localhost:9000/&lt;br/&gt;callback&amp;scope=read&amp;state=casdoor">https://door.casdoor.com/login/oauth/ authorize?client_id=0ba528121ea87b3eb54d&amp;response_type=code&amp;redirect_uri=http://localhost:9000/ callback&amp;scope=read&amp;state=casdoor</a>

## 登录 URL

It's very easy to log into Casdoor via Casdoor's built-in application; simply visit Casdoor server homepage (e.g., <https://door.casdoor.com> for demo site) and it will automatically redirect you to `/login`. But how do you get the URLs for other applications in frontend and backend code?

You can either concatenate strings manually or call some utility functions provided by Casdoor SDKs to get the URLs:

## 1. Manually concatenating strings

- Sign-up page URL
  - 注册指定的应用程序: <your-casdoor-hostname>/signup/<your-application-name>
  - Signup by OAuth: <your-casdoor-hostname>/signup/oauth/authorize?client\_id=<client-id-for-your-application>&response\_type=code&redirect\_uri=<redirect-uri-for-your-application>&&scope=read&state=casdoor
  - 自动注册: <your-casdoor-hostname>/auto-signup/oauth/authorize?client\_id=<client-id-for-your-application>&response\_type=code&redirect\_uri=<redirect-uri-for-your-application>&&scope=read&state=casdoor
- Sign-in page URL
  - Sign-in for the specified organization: <your-casdoor-hostname>/login/<your-organization-name>
  - Sign-in by OAuth: <your-casdoor-hostname>/login/oauth/authorize?client\_id=<client-id-for-your-application>&response\_type=code&redirect\_uri=<redirect-uri-for-your-application>&&scope=read&state=casdoor

## 2. Using frontend SDK (for frontend JavaScript code using React, Vue, or Angular)

`getSignupUrl()` and `getSigninUrl()`: casdoor-js-sdk

## 3. Using backend SDK (for backend code using Go, Java, etc.)

`GetSignupUrl()` and `GetSigninUrl()`: casdoor-go-sdk

# 提供商

Casdoor is a federated single sign-on system that supports multiple identity providers via OIDC, OAuth, and SAML. Casdoor can also send verification codes or other notifications to users via email or SMS. Casdoor uses the concept of `Provider` to manage all these third-party connectors.

A list of all providers supported by Casdoor can be found at [provider/overview](#).

```
type Provider struct {
    Owner      string `xorm:"varchar(100) notnull pk" json:"owner"`
    Name       string `xorm:"varchar(100) notnull pk" json:"name"`
    CreatedTime string `xorm:"varchar(100)" json:"createdTime"`

    DisplayName  string `xorm:"varchar(100)" json:"displayName"`
    Category    string `xorm:"varchar(100)" json:"category"`
    Type        string `xorm:"varchar(100)" json:"type"`
    Method      string `xorm:"varchar(100)" json:"method"`
    ClientId    string `xorm:"varchar(100)" json:"clientId"`
    ClientSecret string `xorm:"varchar(100)" json:"clientSecret"`
    ClientId2   string `xorm:"varchar(100)" json:"clientId2"`
    ClientSecret2 string `xorm:"varchar(100)" json:"clientSecret2"`

    Host     string `xorm:"varchar(100)" json:"host"`
    Port     int     `json:"port"`
    Title    string `xorm:"varchar(100)" json:"title"`
    Content  string `xorm:"varchar(1000)" json:"content"`

    RegionId   string `xorm:"varchar(100)" json:"regionId"`
    SignName    string `xorm:"varchar(100)" json:"signName"`
    TemplateCode string `xorm:"varchar(100)" json:"templateCode"`
    AppId      string `xorm:"varchar(100)" json:"appId"`

    Endpoint    string `xorm:"varchar(1000)" json:"endpoint"`
    IntranetEndpoint string `xorm:"varchar(100)" json:"intranetEndpoint"`
    Domain     string `xorm:"varchar(100)" json:"domain"`
    Bucket      string `xorm:"varchar(100)" json:"bucket"`}
```

## Casdoor是如何自我管理的?

Upon running Casdoor for the first time, some built-in objects are created to facilitate its management:

- 一个内置的命名为 `built-in` 的组织。
- `built-in` 组织下用户名为 `admin` 的用户。
- A built-in application named `app-built-in`, administered by the `built-in` organization, representing Casdoor itself.

All users under the `built-in` organization, including `admin`, will have full administrator privileges on the Casdoor platform. Therefore, if there are multiple administrators, it is advisable to create new accounts under the `built-in` organization. Alternatively, the sign-up channel for the `app-built-in` application should be closed to prevent unwanted access.

### ⚠ 注意事项

It is not possible to rename or delete the built-in objects via both the web UI or the RESTful API. Casdoor has hardcoded these reserved names in many places; attempting to rename or delete them by modifying the DB may cause the entire system to crash.



# 服务器安装

## 安装要求

### 操作系统

All major operating systems, including Windows, Linux, and macOS, are supported.

### 环境

- Go 1.17+
- Node.js LTS (18)
- Yarn 1.x

#### ① 信息

我们强烈建议使用 [Yarn 1.x](#) 来运行和构建Casdoor前端。 使用NPM可能会导致界面样式问题。 欲了解更多详情, 请见: [casdoor#294](#)。

#### ⚠ 注意事项

如果您的网络无法直接同步Go依赖包, 您需要配置GOPROXY环境变量。 我们强烈建议使用: <https://goproxy.cn/>

### 数据库

Casdoor使用 [XORM](#) 与数据库进行交互。 基于 [Xorm Drivers](#), 当前支持的数据库包括:

- MySQL
- MariaDB
- PostgreSQL
- CockroachDB
- SQL Server
- Oracle
- SQLite 3
- TiDB

## 下载

Casdoor 的源代码托管在 GitHub: <https://github.com/casdoor/casdoor>。 转到后端代码和 React 前端代码都包含在一个仓库中。

名称	描述	语言	源代码
前端	Casdoor的网页前端界面	JavaScript + React	<a href="https://github.com/casdoor/casdoor/tree/master/web">https://github.com/casdoor/casdoor/tree/master/web</a>
后端	Casdoor的ResTful API 后端	Golang + Beego + XORM	<a href="https://github.com/casdoor/casdoor">https://github.com/casdoor/casdoor</a>

Casdoor支持 Go Modules。 To download the code, simply clone the code using git:

```
cd /文件夹路径/  
git clone https://github.com/casdoor/casdoor
```

# 配置

## 配置数据库

Casdoor 支持MySQL、MSSQL、SQLite3和PostgreSQL。默认情况下，Casto使用 MySQL。

### MySQL

Casdoor 将会把users, nodes 和 topics 信息存储在一个名为 casdoor 的 MySQL 数据库中。如果数据库不存在，则需手动创建。可以在以下位置指定数据库连接字符串  
<https://github.com/casdoor/blob/master/conf/app.conf>

```
driverName = mysql
dataSourceName = root:123456@tcp(localhost:3306)-
dbName = casdoor
```

### PostgreSQL

在运行 Casdoor 之前，您需要手动准备 PostgreSQL 数据库，因为 Castor 需要在使用 xorm 打开 Postgres 时指定一个数据库。

Assuming you have already prepared a database called casdoor, you should specify app.conf like this:

```
driverName = postgres
dataSourceName = "user=postgres password=postgres host=localhost
port=5432 sslmode=disable dbname=casdoor"
dbName = casdoor
```

**!** FOR POSTGRESQL, ENSURE THAT `dataSourceName` HAS A NON-EMPTY `dbName` AND ALSO DUPLICATE THE DATABASE NAME FOR THE `dbname` FIELD AS SHOWN IN THE EXAMPLE ABOVE. :::

### CockroachDB

CockroachDB can also be used with the PostgreSQL driver and has the same configuration as PostgreSQL.

```
driverName = postgres
dataSourceName = "user=postgres password=postgres
host=localhost port=5432 sslmode=disable dbname=casdoor
serial_normalization=virtual_sequence"
dbName = casdoor
```

### SQLite3

To configure SQLite3, you should specify `app.conf` like this:

```
driverName = sqlite
dataSourceName = "file:casdoor.db?cache=shared"
dbName = casdoor
```

## 通过 Ini 文件配置

Casdoor can be configured via a single file: [conf/app.conf](#), which by default contains the following content:

```
appname = casdoor
httpport = 8000
runmode = dev
SessionOn = true
```

- `appname` is the application name, which currently has no practical use.
- `httpport` is the port that your backend application is listening on.
- `runmode` can be set to `dev` or `prod`.
- `SessionOn` determines whether to enable session and is enabled by default.
- `driverName`, `dataSourceName`, and `dbName` were introduced earlier. Please see [Configure Database](#) for details.
- `verificationCodeTimeout` sets the expiration time of the verification code. After expiration, the user needs to obtain it again.

As a beginner, you only need to modify two items: `driverName` and `dataSourceName` based on your database. This database will be used by Casdoor to store all data, including users, organizations, and applications.

- `tableNamePrefix` is the prefix of the table when using an adapter.
- `showSql` determines whether to show SQL statements on the logger if the log level is greater than INFO.
- `redisEndpoint` 后填写Redis地址, 被Beego用于session存储 If this parameter is empty, the session data will be stored locally as files in the `./tmp` folder. To use Redis as Beego session storage, the value would be something like: `redis.example.com:6379`. If Redis is deployed locally, you can use `localhost:6379`. If Redis password is enabled, use `redis.example.com:6379, db, password`. See more details at: <https://github.com/beego/beedoc/blob/master/en-US/module/session.md#saving-provider-config>.
- `defaultStorageProvider` 是默认的文件存储服务名称。 如果您需要使用文件存储服务, 例如 `头像上传`, 您需要设置存储提供商, 并在您的 `应用` 中应用它。 详情请参阅 [存储](#)
- `isCloudIntranet` is used to identify whether your provider endpoint is an intranet endpoint.
- `authstate` 是授权应用程序名称。 登录时将检查该参数。

- `socks5Proxy` 是 SOCKS 代理服务器 IP 地址。 Set the proxy port because we have Google-related services or use `Google`, `Github`, `Facebook`, `LinkedIn`, or `Steam` as OAuth Providers, which may be restricted by the network in some areas.
- `initscore` 是每个用户的最初分数。 每个用户都有一个得分属性。 The score is used by `Casnode` and does not control anything in Casdoor.
- `logPostOnly` is used to determine whether only the post method is used to add a record.
- `origin` 是origin形式的后端域名
- `staticBaseUrl` is the address of the static image used when the system initializes the database.
- `enableGzip` will accept and respond with gzip encoding if the request header includes `Accept-Encoding=gzip`.

## 通过环境变量配置

All configuration items defined by Casdoor in the ini file mentioned above can be configured via environment variables, as well as some of the beego configurations items (`httpport`, `appname`).

For example, when you try to start Casdoor, you can use something like this to pass the configuration via environment variables:

```
appname=casbin go run main.go
```

In addition, `export` derivatives are also a possible method. The names of environmental variables should exactly match the names you want to use in the ini file.

Note: configurations in environmental variables can override the configurations in

the ini file.

# 运行

There are currently two methods to start, and you can choose one according to your situation.

## Development Mode

### Backend

Casdoor's Go backend runs on port 8000 by default. You can start the Go backend with the following command:

```
go run main.go
```

After the server is successfully running, you can start the frontend part.

### Frontend

Casdoor's frontend is a very classic [Create-React-App \(CRA\)](#) project. It runs on port `7001` by default. Use the following commands to run the frontend:

```
cd web  
yarn install  
yarn start
```

Visit <http://localhost:7001> in your browser. Log into the Casdoor dashboard with the default global admin account: `built-in/admin`.

```
admin  
123
```

## Production Mode

### Backend

Build the Casdoor Go backend code into an executable and start it.

For Linux:

```
go build  
.casdoor
```

For Windows:

```
go build  
casdoor.exe
```

### Frontend

Build the Casdoor frontend code into static resources (.html, .js, .css files):

```
cd web  
yarn install  
yarn build
```

Visit <http://localhost:8000> in your browser. Log into the Casdoor dashboard with the default global admin account: `built-in/admin`.

admin

123

### 💡 提示

To use another port, please edit `conf/app.conf` and modify `httpport`, then restart the Go backend.

### ❗ CASDOOR PORT DETAILS

In the **dev** environment, the frontend is run by `yarn run` on port 7001, so if you want to go to the Casdoor login page, you need to set the Casdoor link as <http://localhost:7001>.

In the **prod** environment, the frontend files are first built by `yarn build` and served on port 8000, so if you want to go to the Casdoor login page, you need to set the Casdoor link as <https://your-casdoor-url.com:8000> (If you are using a reverse proxy, you need to set the link as your domain).

Take Our Official Forum Casnode as an Example

Casnnode uses Casdoor to handle authentication.

When we are testing Casnode in the **dev** environment, we set the `serverUrl` as <http://localhost:7001>, so when we test the signin and signup functionality using Casdoor, it will go to localhost 7001, which is the Casdoor port.

And when we put Casnode into the **prod** environment, we set the `serverUrl` as <https://door.casdoor.com>, so users can sign in or sign up using Casdoor.

```
|4 import * as ConfBackend from "./backend/ConfBackend.js"
|5
|6 export const AuthConfig = {
|7   // serverUrl: "https://door.casbin.com",
|8   serverUrl: "http://localhost:7001",
|9   clientId: "014ae4bd048734ca2dea",
|10  ...
|11}
```

# (可选) 使用 Docker 运行

## 安装要求

### 硬件

If you want to build the Docker image yourself, please ensure that your machine has at least 2GB of memory. Casdoor's frontend is an NPM project of React. Building the frontend requires at least 2GB of memory. Having less than 2GB of memory may result in a frontend build failure.

If you only need to run the pre-built image, please ensure that your machine has at least 100MB of memory.

### 操作系统

All operating systems (Linux, Windows, and macOS) are supported.

### Docker

You can use Docker (docker-engine version  $\geq$  17.05) in Linux or Docker Desktop in Windows and macOS.

- [Docker](#)

Regardless of the operating system, users must ensure that they have docker-engine version  $\geq$  17.05. This is because we utilize the multi-stage build feature in the docker-compose.yml, which is supported in versions 17.05 and above. For more information, see <https://docs.docker.com/develop/develop-images/>

[multistage-build/](#).

If you are also using docker-compose, please ensure that you have `docker-compose` version  $\geq 2.2$ . For Linux users, you also need to make sure that `docker-compose` is installed, as it is separate from `docker-engine`.

## 获取镜像

我们提供了两个DockerHub 镜像：

名称	描述	建议
<code>casdoor-all-in-one</code>	Both Casdoor and a MySQL database are included in the image	This image already includes a toy database and is only for testing purposes
<code>casdoor</code>	Only Casdoor is included in the image	This image can be connected to your own database and used in production

1. `casbin/casdoor-all-in-one`: This image includes the `casdoor` binary, a MySQL database, and all the necessary configurations. It is designed for new users who want to try Casdoor quickly. With this image, you can start Casdoor immediately with just one or two commands, without any complex configuration. However, please note that we do not recommend using this image in a production environment.

## 选项 1: 使用示例数据库

Run the container with port `8000` exposed to the host. The image will be

automatically pulled if it doesn't exist on the local host.

```
docker run -p 8000:8000 casbin/casdoor-all-in-one
```

Visit <http://localhost:8000> in your browser. Log into the Casdoor dashboard with the default global admin account: `built-in/admin`

```
admin  
123
```

## Option-2: Try directly with the standard image



If it is not convenient to mount the configuration file to a container, using environment variables is also a possible solution.

example

```
docker run \  
-e driverName=mysql \  
-e dataSourceName='user:password@tcp(x.x.x.x:3306)/*' \  
-p 8000:8000 \  
casbin/casdoor:latest
```

Create `conf/app.conf`. You can copy it from `conf/app.conf` in Casdoor. For more details about `app.conf`, you can see [Via Ini file](#).

然后运行

```
docker run -p 8000:8000 -v /folder/of/app.conf:/conf casbin/casdoor:latest
```

Anyway, just mount the app.conf to /conf/app.conf and start the container.

Visit <http://localhost:8000> in your browser. Log into the Casdoor dashboard with the default global admin account: `built-in/admin`

```
admin  
123
```

## Option-3: Try with docker-compose

Create a `conf/app.conf` directory in the same directory level as the `docker-compose.yml` file. Then, copy `app.conf` from Casdoor. For more details about `app.conf`, you can see [Via Ini file](#).

Create a separate database using docker-compose:

```
docker-compose up
```

这样就完成了! 🎉

Visit <http://localhost:8000> in your browser. Log into the Casdoor dashboard with the default global admin account: `built-in/admin`

```
admin  
123
```

### ⓘ 备注

If you dig deeper into the docker-compose.yml file, you may be puzzled by the environment variable we created called "RUNNING\_IN\_DOCKER". When the database 'db' is created via docker-compose, it is available on your PC's localhost but not the localhost of the Casdoor container. To prevent you from running into troubles caused by modifying app.conf, which can be quite difficult for a new user, we provided this environment variable and pre-assigned it in the docker-compose.yml. When this environment variable is set to true, localhost will be replaced with host.docker.internal so that Casdoor can access the database.

# (Optional) Try with K8s Helm

## Introduction

Now we show how to deploy Casdoor on Kubernetes using Helm for easy and scalable management.

## Prerequisites

- A running Kubernetes cluster
- Helm v3 installed

## Installation Steps

### Step 1: Install the Casdoor Chart

Install the Casdoor chart:

```
helm install casdoor oci://registry-1.docker.io/casbin/casdoor-helm-charts --version 1.524.0
```

### Step 2: Accessing Casdoor

Once installed, Casdoor can be accessed at the provided service URL by your Kubernetes cluster.

## Customization and Configuration

Customize your Casdoor installation by modifying the Helm chart values. For detailed options, refer to the [values.yaml](#) file in the chart. The following parameters can be configured.

Parameter	Description	Default Value
<code>replicaCount</code>	Number of replicas of the Casdoor application to run.	<code>1</code>
<code>image.repository</code>	Repository for	<code>casbin</code>

Parameter	Description	Default Value
	the Casdoor Docker image.	
<code>image.name</code>	Name of the Casdoor Docker image.	<code>casdoor</code>
<code>image.pullPolicy</code>	Pull policy for the Casdoor Docker image.	<code>IfNotPresent</code>
<code>image.tag</code>	Tag for the Casdoor Docker image.	<code>""</code>
<code>config</code>	Configuration settings for the Casdoor application.	See <a href="#">config</a> field
<code>database.driver</code>	Database driver to use (supports mysql, postgres, cockroachdb, sqlite3).	<code>sqlite3</code>
<code>database.user</code>	Database username.	<code>""</code>
<code>database.password</code>	Database password.	<code>""</code>
<code>database.host</code>	Database host.	<code>""</code>

Parameter	Description	Default Value
<code>database.port</code>	Database port.	80
<code>database.databaseName</code>	Name of the database used by Casdoor.	casdoor
<code>database.sslMode</code>	SSL mode for the database connection.	disable
<code>service.type</code>	Type of Kubernetes service to create for Casdoor (ClusterIP, NodePort, LoadBalancer, etc.).	ClusterIP
<code>service.port</code>	Port number for the Casdoor service.	8000
<code>ingress.enabled</code>	Whether to enable Ingress for Casdoor.	false
<code>ingress.annotations</code>	Annotations for the Ingress resource.	{}
<code>ingress.hosts</code>	Hostnames for the Ingress resource.	[]

Parameter	Description	Default Value
<code>resources</code>	Resource requests and limits for the Casdoor container.	<code>{}</code>
<code>autoscaling.enabled</code>	Whether to enable Horizontal Pod Autoscaler for Casdoor.	<code>false</code>
<code>autoscaling.minReplicas</code>	Minimum number of replicas for Horizontal Pod Autoscaler.	<code>1</code>
<code>autoscaling.maxReplicas</code>	Maximum number of replicas for Horizontal Pod Autoscaler.	<code>100</code>
<code>autoscaling.targetCPUUtilizationPercentage</code>	Target CPU utilization percentage for Horizontal Pod Autoscaler.	<code>80</code>
<code>nodeSelector</code>	Node labels for pod assignment.	<code>{}</code>
<code>tolerations</code>	Toleration labels for pod assignment.	<code>[]</code>

Parameter	Description	Default Value
<code>affinity</code>	Affinity settings for pod assignment.	<code>{}</code>
<code>extraContainersEnabled</code>	Whether to enable additional sidecar containers.	<code>false</code>
<code>extraContainers</code>	Additional sidecar containers.	<code>[]</code>
<code>extraVolumeMounts</code>	Additional volume mounts for the Casdoor container.	<code>[]</code>
<code>extraVolumes</code>	Additional volumes for the Casdoor container.	<code>[]</code>
<code>envFromSecret</code>	Provide Environment variable from secret.	<code>[{name:"", secretName:"", key:""}]</code>
<code>envFromConfigmap</code>	Provide Environment variable from configmap.	<code>[{name:"", configmapName:"", key:""}]</code>
<code>envFrom</code>	Provide Environment variable from entire secret or configmap.	<code>`[{name:"", type:"configmap \</code>

## Managing the Deployment

To upgrade your Casdoor deployment:

```
helm upgrade casdoor casdoor/casdoor-helm-charts
```

To uninstall Casdoor:

```
helm delete casdoor
```

For further management and customization, refer to the Helm and Kubernetes documentation.

## Conclusion

Using Helm to deploy Casdoor on Kubernetes simplifies the management and scalability of your authentication services within your Kubernetes environment.

# Casdoor公共API

Casdoor 的 Web UI 是由 React 开发的 [SPA \(单页面应用\)](#)。The React frontend consumes the Casdoor RESTful API exposed by the Go backend code. This RESTful API is referred to as the [Casdoor Public API](#). In Another word, with HTTP calls, you can do everything just like how Casdoor web UI itself does. There's no other limitations. The API can be utilized by the following:

- Casdoor前端页面
- Casdoor client SDKs (e.g., casdoor-go-sdk)
- 应用方自定义的任何代码

The full reference for the [Casdoor Public API](#) can be found on Swagger: <https://door.casdoor.com/swagger>. These Swagger docs are automatically generated using Beego's Bee tool. If you want to generate the Swagger docs by yourself, see: [How to generate the swagger file](#)

## How to authenticate with [Casdoor Public API](#)

### 1. 通过 [Access token](#)

We can use the access token granted for an authenticated user to call [Casdoor Public API](#) as the user itself.

How to get the access token?

The application can get the access token for the Casdoor user at the end of OAuth login process (aka get the token by code and state). The permissions for the API calls will be the same as the user.

The below examples shows how to call [GetOAuthToken\(\)](#) function in Go via casdoor-go-sdk.

```
func (c *ApiController) Signin() {
    code := c.Input().Get("code")
    state := c.Input().Get("state")

    token, err := casdoorsdk.GetOAuthToken(code, state)
    if err != nil {
        c.ResponseError(err.Error())
        return
    }

    claims, err := casdoorsdk.ParseJwtToken(token.AccessToken)
    if err != nil {
        c.ResponseError(err.Error())
        return
    }
}
```

All granted access tokens can also be accessed via the web UI by an admin user in the Tokens page. For example, visit: <https://door.casdoor.com/tokens> for the demo site.

#### How to authenticate?

1. HTTP `GET` parameter, the URL format is:

```
/page?access_token=<The access token>
```

Demo site example: `https://door.casdoor.com/api/get-global-providers?access_token=eyJhbGciOiJSUzI1NiIs`

2. HTTP Bearer token, the HTTP header format is:

```
Authorization: Bearer <The access token>
```

## 2. By `Client ID` and `Client secret`

#### How to get the client ID and secret?

The application edit page (e.g., <https://door.casdoor.com/applications/casbin/app-vue-python-example>) will show the client ID and secret for an application. This authentication is useful when you want to call the API as a "machine", "application" or a "service" instead of a user. The permissions for the API calls will be the same as the application (aka the admin of the organization).

The below examples shows how to call `GetOAuthToken()` function in Go via casdoor-go-sdk.

#### How to authenticate?

1. HTTP `GET` parameter, the URL format is:

```
/page?clientId=<The client ID>&clientSecret=<the client secret>
```

Demo site example: `https://door.casdoor.com/api/get-global-providers?clientId=294b09fbc17f95daf2fe&clientSecret=dd8982f7046ccba1bbd7851d5c1ece4e52bf039d`

2. [HTTP Basic Authentication](#), the HTTP header format is:

```
Authorization: Basic <The Base64 encoding of client ID and client secret joined by a single colon ":">
```

If you are not familiar with the Base64 encoding, you can use a library to do that because [HTTP Basic Authentication](#) is a popular standard supported by many places.

### 3. By `username` and `password`

#### ⚠ 注意事项

This authentication method is not safe and kept here only for compatibility or demo purposes. We recommend using the previous two authentication methods instead.

#### What will happen?

The user credential will be exposed as `GET` parameters in the request URL. Moreover, the user credential will be sniffed in plain text by the network if you are using HTTP instead of HTTPS.

We can use the username and password for a Casdoor user to call `Casdoor Public API` as the user itself. The username takes the format of `<The user's organization name>/<The user name>`. The permissions for the API calls will be the same as the user.

#### How to authenticate?

1. HTTP `GET` parameter, the URL format is:

```
/page?username=<The user's organization name>/<The user name>&password=<the user's password>"
```

Demo site example: <https://door.casdoor.com/api/get-global-providers?username=built-in/admin&password=123>

# 教程

## Product Documentation

产品	技术	文档
Dashboard of PingCAP TiDB	React + TypeScript + Go + Gin	Use Casdoor for TiDB Dashboard SSO sign-in (other languages: <a href="#">Chinese</a> , <a href="#">Japanese</a> )
GitLab	Vue + Ruby + Rails	OpenID Connect OmniAuth provider
Apache Shenyu	Java	Casdoor Plugin (other languages: <a href="#">Chinese</a> )
Alist	TypeScript + SolidJS + Go + Gin	Casdoor SSO (other languages: <a href="#">Chinese</a> )
BookStack	jQuery + Bootstrap + Go + Beego	Casdoor integrates registration and login

# 文章

技术	语言	标题
ASP.NET Core 6	英语	<a href="#">ASP.NET Core .NET 6 Demo Authentication Project using local Casdoor Docker Container on Windows Subsystem for Linux</a>
OAuth2 Proxy (Go)	中文	<a href="#">Use Casdoor + OAuth-Proxy to protect web applications on public networks</a>
Casnode (JavaScript + React + Go + Beego)	中文	<a href="#">Use Lighthouse to set up a forum like V2ex</a>
Cloudreve (Go)	中文	<a href="#">Modify Cloudreve to support Casdoor</a>
KodExplorer (PHP)	中文	<a href="#">Modify KodExplorer to support Casdoor</a>



&gt;

Deployment

# Deployment



## Deploying to NGINX

Use Nginx to reverse proxy your backend Go program and quickly start the Casdoor service.



## Deploying to Kubernetes

Learn how to deploy Casdoor in a Kubernetes cluster



## 数据初始化

如何从文件初始化Casdoor 数据



## Hosting Static Files in a CDN

Hosting frontend static files in a CDN

## Hosting Static Files in an Intranet

How to deploy Casdoor static resources

## 数据库迁移

Handling DB Migration in Casdoor

# Deploying to NGINX

Though Casdoor follows a front-end back-end separation architecture, in a production environment, the back-end program still provides static file services for front-end files. Hence, you can employ reverse proxy software like [Nginx](#) to proxy all traffic for the Casdoor domain and redirect it to the port monitored by the backend Go program.

In this chapter, you will learn how to use Nginx to reverse proxy your backend Go program and quickly start the Casdoor service.

## 1. Build front end static files

Assuming you have downloaded Casdoor and completed the necessary configuration (if not, refer to the [Get started](#) section), you only need to build the static files as follows:

[Yarn](#)    [npm](#)

```
yarn install && yarn run build
```

```
npm install && npm run build
```

## 2. Run the back-end program

```
go run main.go
```

Or, build it first:

```
go build && ./main
```

## 3. Configure and run Nginx

```
vim /path/to/nginx/nginx.conf
```

Then, add a server:

```
server {
    listen 80;
    server_name YOUR_DOMAIN_NAME;
    location / {
        proxy_set_header Host $http_host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_redirect off;
        proxy_pass http://127.0.0.1:8000;
    }
}
```

Next, restart your Nginx process. Run:

```
nginx -s reload
```

## 4. Test

Visit `http://YOUR_DOMAIN_NAME` in your favorite browser.

# Deploying to Kubernetes

## Deploy Casdoor in Kubernetes (k8s)

We provide a basic example of deploying Casdoor in a Kubernetes cluster. In the root folder of Casdoor, you will find a file named "k8s.yaml". This file contains an example configuration for deploying Casdoor in Kubernetes, including a deployment and a service.

Before starting the deployment, ensure that you have modified the `conf/app.conf` file so that Casdoor can connect to the database successfully and that the database itself is running. Also, make sure that Kubernetes is able to pull the necessary images.

To deploy Casdoor, run the following command:

```
kubectl apply -f k8s.yaml
```

You can check the deployment status by running the command `kubectl get pods`.

Here is the content of `k8s.yaml`:

```
# this is only an EXAMPLE of deploying casddor in kubernetes
# please modify this file according to your requirements
apiVersion: v1
kind: Service
metadata:
  #EDIT IT: if you don't want to run casdoor in default namespace,
```

Please note that this file is only an example. You can make various modifications as per your requirements, such as using a different namespace, service type, or a ConfigMap to mount the configuration file. Using a ConfigMap is a recommended approach in Kubernetes for mounting configuration files in a production environment.



# 数据初始化

If you are deploying Casdoor with other services as a complete application, you may want to provide an **out-of-the-box** feature for users. This means that users can directly use the application without any configuration.

In such a situation, you can use data initialization to register your service in Casdoor through a configuration file. This file can be pre-defined or dynamically generated by your own service.

## 使用方式

If there is a configuration file named `init_data.json` at the root directory of Casdoor, it will be used to initialize data in Casdoor. All you have to do is place this file in the root directory where Casdoor will run.

If you are using the official Docker image of Casdoor, here are some scripts that can help you to mount `init_data.json` into the container.

### Docker

If you deploy Casdoor with Docker, you can use the `volume` command to mount `init_data.json` into the container.

```
docker run ... -v /path/to/init_data.json:/init_data.json
```

# Kubernetes

If you deploy Casdoor with Kubernetes, you can use the `configmap` to store `init_data.json`.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: casdoor-init-data
data:
  init_data.json:
```

您可以通过挂载 `configmap` 将数据挂载到Casdoor的 `pods` 中。 You can modify your `deployment` as follows:

```
apiVersion: apps/v1
kind: Deployment
...
spec:
  template:
    ...
      spec:
        containers:
          ...
            volumeMounts:
              - mountPath: /init_data.json
                name: casdoor-init-data-volume
                subPath: init_data.json
        volumes:
          - configMap:
              name: casdoor-init-data
              name: casdoor-init-data-volume
```

# 文件内容

There is already a template named `init_data.json.template` in the root directory of Casdoor repository. 您可以参考此文件来自定义您的数据初始化文件。

The following is the Go struct of each part and their documentation:

对象	Go 结构体	文档
组织机构	<a href="#">struct</a>	<a href="#">doc</a>
应用	<a href="#">struct</a>	<a href="#">doc</a>
用户	<a href="#">struct</a>	<a href="#">doc</a>
提供商	<a href="#">struct</a>	<a href="#">doc</a>
证书	<a href="#">struct</a>	
Idaps	<a href="#">struct</a>	<a href="#">doc</a>

If you still feel confused about filling out this template, you can call the RESTful API or use the debug mode of your browser to see the response of `GetXXX` to these objects. The responses are in the same format as `init_data.json`.

# Hosting Static Files in a CDN

Frontend static resources, such as .js and .css files, are located in `web/build/static/`. If you wish to deploy these files in a public cloud's CDN service, Casdoor provides a script that simplifies the deployment process. 请按照以下步骤操作：

① 备注

We assume that you have already built the frontend code of Casdoor. If you have not, please refer to the [documentation](#).

## 准备工作

First, you need to create a valid [Storage Provider](#) in the Casdoor UI. You can refer to the [example](#).

⚠ 注意事项

When filling in the `Domain` field, be sure to end it with a '/'.

Domain [?](#) :

`https://cdn.casbin.com/casdoor/`

# 使用说明

The script can be found at [deployment/deploy\\_test.go](#).

In [deploy\\_test.go](#), you need to modify the `id` parameter in `GetProvider()`. The format of the provider `id` is `<owner>/<name>`.

```
func TestDeployStaticFiles(t *testing.T) {
    provider := object.GetProvider("admin/
provider_storage_aliyun_oss")
    deployStaticFiles(provider)
}
```

After making the necessary modification, use the following commands to run the script:

```
cd deployment
go test
```

If the execution is successful, you will see:

```
PASS
ok      github.com/casdoor/casdoor/deployment  2.951s
```

# 工作原理

脚本的功能：

- Upload all the files in the `css/` and `js/` folders to the CDN service specified

by the storage provider.

- Replace all the URLs of the `.css` and `.js` files in `web/build/index.html` with the URLs hosted in the CDN.

You still need to keep the `index.html` file. After the static files are uploaded to the CDN, `index.html` will still be requested by users through Casdoor's Go backend, and the static files in the CDN will be requested through the URLs provided in `index.html`.

# Hosting Static Files in an Intranet

If you are deploying Casdoor on an [intranet](#), you may not be able to access the static resources directly over the internet. You need to deploy the static resources where you can access them, and then modify the configuration in Casdoor in three places.

## Deploy static resources

All static resources in Casdoor, including images, logos, CSS, etc., are stored in the [casbin/static repository](#).

Clone the repository and deploy it on a web server. Make sure you can access the resources.

## 配置Casdoor

您可以简单地修改配置文件，将静态资源地址设置为您部署的地址。Go to [conf/app.conf](#) and set `staticBaseUrl` to your deployed address.

```
staticBaseUrl = "https://cdn.casbin.org"
```

# 数据库迁移

When upgrading the database, there is a risk of data loss, such as when deleting an old field. Luckily, Casdoor utilizes [xorm](#), which assists with many database migration problems. However, some schema and data migrations must still be handled manually, such as when a field name is changed.

## ① 备注

Refer to the [xorm docs](#) for a better understanding of xorm's schema operations.

## How it Works

As mentioned earlier, xorm is unable to handle field name changes. To address this, xorm provides a [migrate](#) package that can assist with this problem.

To handle field renaming, you can write code like this:

```
migrations := []*migrate.Migration{
{
    ID: "CasbinRule--fill ptype field with p",
    Migrate: func(tx *xorm.Engine) error {
        _, err :=
        tx.Cols("ptype").Update(&xormadapter.CasbinRule{
            Ptype: "p",
        })
        return err
    },
}
```

Our objective is to rename `p_type` to `ptype`. However, since xorm does not support field renaming, we must resort to a more intricate approach: assigning the value of `p_type` to `ptype`, and subsequently deleting the `p_type` field.

The `ID` field uniquely identifies the migration being performed. After `m.Migrate()` runs, the value of `ID` will be added to the migrations table of the database.

Upon starting the project again, the database will check for any existing `ID` field in the table and refrain from performing any operations associated with the same `ID`.



> 如何连接到Casdoor

# 如何连接到Casdoor

## 概览

将您的应用连接到Casdoor

## 标准OIDC 客户端

使用 OIDC 发现迁移到Casdoor

## Casdoor SDKs

Using Casdoor SDKs instead of standard OIDC protocol

## 如何启用单点登录

启用单点登录



## Vue SDK

Casdoor Vue SDK



## 桌面 SDK

4 个项目



## 移动 SDKs

1 个项目



## Casdoor插件

在 Spring Boot, WordPress, Odoo 等其他框架中使用 Casdoor 插件或中间件。



## Next.js

Using Casdoor in a Next.js project



## Nuxt

Using Casdoor in a Nuxt project



## OAuth 2.0

使用AccessToken验证客户端



## Using Casdoor as a CAS Server

How to use Casdoor as a CAS server



## SAML

6 个项目



## WebAuthn

在 Casdoor 中使用webauthn

# 概览

在本节中，我们将向您展示如何将您的应用程序连接到Casdoor。

作为服务提供商(SP)， Casdoor 支持两项认证协议：

- OAuth 2.0 (OIDC)
- SAML

作为身份提供商 (Idp)， Casdoor 支持四项认证协议：

- OAuth 2.0
- OIDC
- SAML
- CAS 1.0, 2.0, 3.0

## OAuth 2.0 (OIDC)

什么是 OAuth 2.0?

OAuth 2 is an authorization framework that enables applications—such as Facebook, GitHub, and Casdoor—to obtain limited access to user accounts on an HTTP service. It works by delegating user authentication to the service that hosts a user account and authorizing third-party applications to access that user account. OAuth 2 provides authorization flows for web and desktop applications, as well as mobile devices.

Casdoor的授权程序基于OAuth 2.0协议。 我们推荐使用 OAuth 2.0 协议，原因如下：

1. 该协议简单易行，能解决多种问题。
2. 该协议成熟度高且社区支持广泛.

如此，您的应用程序将通过 OAuth 2.0 (OIDC) 与 Casdoor 通讯。这里有三种方式连接到 Casdoor:

## 标准 OIDC 客户端

**标准OIDC 客户端:** 使用一个标准的 OIDC 客户端实现，通常在各类编程语言或框架都广泛提供。

什么是OIDC?

OpenID Connect (OIDC) 是一个在OAuth 2.0 框架顶端运行的开放身份验证协议。针对消费者，OIDC允许个人通过单点登录(SSO)访问使用OpenID提供商(OPPs)的依赖方站点，如电子邮件提供商或社交网络平台，以验证其身份。它向应用程序或服务提供用户信息、认证背景，并允许访问用户个人资料。

Casdoor 完全支持 OIDC 协议。 If your application is already using another OAuth 2.0 (OIDC) identity provider via a standard OIDC client library, and you want to migrate to Casdoor, using OIDC discovery will make it very easy to switch to Casdoor.

## Casdoor SDKs

**Casdoor SDKs:** For most programming languages, Casdoor provides easy-to-use SDK libraries on top of OIDC, with extended functionality that is only available in Casdoor.

Compared to the standard OIDC protocol, Casdoor's SDK provides more functionalities, like user management and resource uploading, among others. Connecting to Casdoor via the Casdoor SDK requires more time than using a standard OIDC client library, but it offers the best **flexibility** and the most **powerful API**.

## Casdoor插件

**Casdoor plugin:** If your application is built on top of a popular platform (like Spring Boot, WordPress, etc.) and Casdoor (or a third party) has already provided a plugin or middleware for it, you should use it. Using a plugin is much easier than manually invoking the Casdoor SDK because the former is specially made for the platform.

Plugins:

- [Jenkins 插件](#)
- [APISIX 插件](#)

中间件:

- [Spring Boot插件](#)
- [Django 插件](#)

## SAML

什么是SAML?

安全鉴别标记语言(SAML)是一种开放标准，允许身份提供者(IDP)将授权证书传给

服务提供者。 What this jargon means is that you can use one set of credentials to log into many different websites. It's much simpler to manage one login per user than it is to manage separate logins to email, customer relationship management (CRM) software, Active Directory, etc.

SAML交易使用可扩展标记语言(XML) 在标识提供者和服务提供者之间进行标准化通信。 SAML is the link between the authentication of a user's identity and the authorization to use a service.

Casdoor can be used as an SAML IdP. Currently, Casdoor supports the main features of SAML 2.0. For more details, see [SAML](#).

示例:

[Casdoor as a SAML IdP in Keycloak](#)

Suggestions:

1. The protocol is **powerful** and covers many scenarios, making it one of the most comprehensive SSO protocols.
2. The protocol is **large**, with many optional parameters, so it is difficult to cover all application scenarios 100% in the actual implementation.
3. If the application is **newly** developed, SAML is **not** recommended due to its high technical complexity.

## CAS

什么是CAS?

The Central Authentication Service (CAS) is a single sign-on protocol for the web. Its purpose is to allow a user to access multiple applications while providing their credentials (such as user ID and password) only once. It also allows web applications to authenticate users without gaining access to a user's security credentials, such as a password.

Casdoor has implemented CAS 1.0, 2.0, and 3.0 features. For more details, see [CAS](#).

Suggestions:

1. The protocol itself is relatively lightweight and easy to implement, but it can only solve a single scenario.
2. CAS 客户端和CAS服务器之间的相互信任是通过在没有任何加密或签名机制的情况下使用接口建立的，目的是确保进一步的安全。
3. The CAS protocol has no advantage over other protocols.

## 集成表

一些应用程序已有连接到Casdoor的示例。 您可以按照文档指示操作， 快速连接到 Casdoor。 You can see all applications in the [Integrations table](#).

# 标准OIDC 客户端

## OIDC Discovery

Casdoor has fully implemented the OIDC protocol. If your application is already using a standard OIDC client library to connect to another OAuth 2.0 identity provider, and you want to migrate to Casdoor, using OIDC discovery will make it very easy for you to switch. Casdoor's OIDC discovery URL is:

```
<your-casdoor-backend-host>/.well-known/openid-configuration
```

For example, the OIDC discovery URL for the demo site is:

<https://door.casdoor.com/.well-known/openid-configuration>, and it contains the following information:

```
{
  "issuer": "https://door.casdoor.com",
  "authorization_endpoint": "https://door.casdoor.com/login/oauth/authorize",
  "token_endpoint": "https://door.casdoor.com/api/login/oauth/access_token",
  "userinfo_endpoint": "https://door.casdoor.com/api/userinfo",
  "jwks_uri": "https://door.casdoor.com/.well-known/jwks",
  "introspection_endpoint": "https://door.casdoor.com/api/login/oauth/introspect",
  "response_types_supported": [
    "code",
    "token",
    "id_token",
    "code token",
    "refresh_token"
  ],
  "claims_supported": [
    "sub",
    "name",
    "given_name",
    "family_name",
    "email",
    "email_verified",
    "phone_number",
    "phone_number_verified",
    "profile",
    "picture",
    "updated_at"
  ],
  "grant_types_supported": [
    "authorization_code",
    "refresh_token",
    "client_credentials",
    "password",
    "urn:ietf:params:oauth:grant-type:jwt-bearer"
  ],
  "subject_type_supported": [
    "public",
    "pairwise"
  ],
  "id_token_signed_response_alg": "RS256",
  "id_token_encrypted_response_alg": "RSA1_256",
  "id_token_encrypted_response_enc": "A128CBC-HS256"
}
```

# List of OIDC Client Libraries

Here is a list of some OIDC client libraries for languages like Go and Java:

OIDC 客户端库	语言	链接
go-oidc	Go	<a href="https://github.com/coreos/go-oidc">https://github.com/coreos/go-oidc</a>
pac4j-oidc	Java	<a href="https://www.pac4j.org/docs/clients/openid-connect.html">https://www.pac4j.org/docs/clients/openid-connect.html</a>

Please note that the above table is not exhaustive. For a full list of OIDC client libraries, you can find more details at:

1. <https://oauth.net/code/>
2. <https://openid.net/certified-open-id-developer-tools/>

## OIDC UserInfo Fields

The following table illustrates how OIDC UserInfo fields (via the `/api/userinfo` API) are mapped from properties of Casdoor's User table:

Casdoor User Field	OIDC UserInfo Field
Id	sub

Casdoor User Field	OIDC UserInfo Field
originBackend	iss
Aud	aud
Name	preferred_username
DisplayName	name
Email	email
Avatar	picture
Location	address
Phone	phone

You can see the definition of UserInfo [here](#).



# Casdoor SDKs

## 简介

与标准的OIDC协议相比，Casdoor在SDK中提供了更多的功能，如用户管理、资源上传等。通过Casdoor SDK连接到Casdoor的成本比使用OIDC标准客户端库更低，并将提供灵活性最佳和最强大的API。

Casdoor SDK可分为两类：

1. 前端SDK：用于网站的Javascript SDK和Vue SDK，用于应用的Android或iOS SDK。Casdoor支持为网站和移动应用程序提供身份验证。
2. 后端SDK：Go, Java, Node.js, Python, PHP等后端语言的SDK。



提示

如果您的网站是采用后端分离的方式开发，您可以使用Javascript SDK：[casdoor-js-sdk](#) 或 Vue SDK：[casdoor-vue-sdk](#) 将Casdoor整合到前端。如果您的网页应用程序是由JSP或PHP开发的传统网站，那您就只能使用后端SDK。示例：[casdoor-Python-vue-sdk示例](#)

Mobile SDK	描述	SDK 代码库	示例
Android SDK	For Android apps	<a href="#">casdoor-android-sdk</a>	<a href="#">casdoor-android-example</a>
iOS SDK	For iOS apps	<a href="#">casdoor-ios-sdk</a>	<a href="#">casdoor-ios-example</a>
React Native SDK	For React Native apps	<a href="#">casdoor-react-native-sdk</a>	<a href="#">casdoor-react-native-example</a>
Flutter SDK	For Flutter apps	<a href="#">casdoor-flutter-sdk</a>	<a href="#">casdoor-flutter-example</a>
Firebase SDK	For Google Firebase apps		<a href="#">casdoor-firebase-example</a>
Unity Games SDK	For Unity 2D/3D PC/Mobile games	<a href="#">casdoor-dotnet-sdk</a>	<a href="#">casdoor-unity-example</a>
uni-app SDK	For uni-app apps	<a href="#">casdoor-uniapp-sdk</a>	<a href="#">casdoor-uniapp-example</a>

Desktop SDK	描述	SDK code	示例
Electron SDK	For Electron apps	<a href="#">casdoor-js-sdk</a>	<a href="#">casdoor-electron-example</a>
.NET Desktop SDK	For .NET desktop apps	<a href="#">casdoor-dotnet-sdk</a>	WPF: <a href="#">casdoor-dotnet-desktop-example</a> WinForms: <a href="#">casdoor-dotnet-winform-example</a> Avalonia UI: <a href="#">casdoor-dotnet-avalonia-example</a>
C/C++ SDK	For C/C++ desktop apps	<a href="#">casdoor-cpp-sdk</a>	<a href="#">casdoor-cpp-qt-example</a>

Web frontend SDK	Description	SDK code	Example code
Javascript SDK	For traditional non-SPA websites	<a href="#">casdoor-js-sdk</a>	Nodejs backend: <a href="#">casdoor-raw-js-example</a> Go backend: <a href="#">casdoor-go-react-sdk-example</a>
Frontend-only SPA SDK	For frontend-only SPA websites	<a href="#">casdoor-js-sdk</a>	<a href="#">casdoor-react-only-example</a>

Web frontend SDK	Description	SDK code	Example code
React SDK	For React websites	<a href="#">casdoor-react-sdk</a>	Nodejs backend: <a href="#">casdoor-nodejs-react-example</a> Java backend: <a href="#">casdoor-spring-security-react-example</a>
Next.js SDK	For Next.js websites		<a href="#">nextjs-auth</a>
Nuxt SDK	For Nuxt websites		<a href="#">nuxt-auth</a>

| Vue SDK | For Vue websites | [casdoor-vue-sdk](#) | [casdoor-python-vue-sdk-example](#) || Angular SDK | For Angular websites | [casdoor-angular-sdk](#) | [casdoor-nodejs-angular-example](#) || Flutter SDK | For Flutter Web websites | [casdoor-flutter-sdk](#) | [casdoor-flutter-example](#) || ASP.NET SDK | For ASP.NET Blazor WASM websites | [Blazor.BFF.OpenIDConnect.Template](#) | [casdoor-dotnet-blazorwasm-oidc-example](#) || Firebase SDK | For Google Firebase apps || [casdoor-firebase-example](#) |

接下来，根据您后端的语言，可以选择使用下面的后端SDK之一：

Web backend SDK	Description	Sdk code	Example code
Go SDK	For Go backends	<a href="#">casdoor-go-sdk</a>	<a href="#">casdoor-go-react-sdk-example</a>
Java SDK	For Java backends	<a href="#">casdoor-java-sdk</a>	<a href="#">casdoor-spring-boot-starter</a> , <a href="#">casdoor-spring-boot-example</a> , <a href="#">casdoor-spring-security-react-example</a>
Node.js SDK	For Node.js backends	<a href="#">casdoor-nodejs-sdk</a>	<a href="#">casdoor-nodejs-react-example</a>
Python SDK	For Python backends	<a href="#">casdoor-python-sdk</a>	Flask: <a href="#">casdoor-python-vue-sdk-example</a> Django: <a href="#">casdoor-django-js-sdk-example</a> FastAPI: <a href="#">casdoor-fastapi-js-sdk-example</a>
PHP SDK	For PHP backends	<a href="#">casdoor-php-sdk</a>	<a href="#">wordpress-casdoor-plugin</a>
.NET SDK	For ASP.NET backends	<a href="#">casdoor-dotnet-sdk</a>	<a href="#">casdoor-dotnet-sdk-example</a>
Rust SDK	For Rust backends	<a href="#">casdoor-rust-sdk</a>	<a href="#">casdoor-rust-example</a>
C/C++ SDK	For C/C++ backends	<a href="#">casdoor-cpp-sdk</a>	<a href="#">casdoor-cpp-qt-example</a>
Dart SDK	For Dart backends	<a href="#">casdoor-dart-sdk</a>	
Ruby SDK	For Ruby backends	<a href="#">casdoor-ruby-sdk</a>	

For a full list of the official Casdoor SDKs, please see: <https://github.com/orgs/casdoor/repositories?q=sdk&type=all&language=&sort=>

# 如何使用 Casdoor SDK ?

## 1. 后端 SDK 配置

当您的应用程序启动时，您需要调用 `InitConfig()` 函数来初始化Casdoor SDK 配置。Take casdoor-go-sdk as example: <https://github.com/casbin/casnode/blob/6d4c55f5c9a3c4bd8c85f2493abad3553b9c7ac0/controllers/account.go#L51-L64>

```
var CasdoorEndpoint = "https://door.casdoor.com"
var ClientId = "541738959670d221d59d"
var ClientSecret = "66863369a64a5863827cf949bab70ed560ba24bf"
var CasdoorOrganization = "casbin"
var CasdoorApplication = "app-casnode"

//go:embed token_jwt_key.pem
var JwtPublicKey string

func init() {
    auth.InitConfig(CasdoorEndpoint, ClientId, ClientSecret, JwtPublicKey, CasdoorOrganization, CasdoorApplication)
}
```

`InitConfig()` 的所有参数解释为：

Parameter	Must	Description
endpoint	Yes	Casdoor Server URL, like <code>https://door.casdoor.com</code> or <code>http://localhost:8000</code>
clientId	Yes	Client ID for the Casdoor application
clientSecret	Yes	Client secret for the Casdoor application
jwtPublicKey	Yes	The public key for the Casdoor application's cert
organizationName	Yes	The name for the Casdoor organization
applicationName	No	The name for the Casdoor application



`jwtPublicKey` 可以在 `Certs` 页面中进行管理。

Certs									
Name	Created time	Display name	Scope	Type	Crypto algorithm	Bit size	Expire in years	Action	
cert_rjeegc	2022-02-16 11:04:10	New Cert - rjeegc	JWT	x509	RSA	4096	20	<button>Edit</button>	<button>Delete</button>
cert-built-in	2022-02-15 12:31:46	Built-in Cert	JWT	x509	RSA	4096	20	<button>Edit</button>	<button>Delete</button>

您可以在证书编辑页面中找到公钥，复制或下载它以供 sdk 使用。

Public key [Copy public key](#) [Download public key](#)

```
-----BEGIN CERTIFICATE-----
MIIEvTCzAuGgAwIBAgDAlcAMA0GCSqGSIb3dQEBcwlAMDYrHTAbBgNVBAoT
FENhc2Bvb3lgT3InW5pemf0aW9uMmRUlwIwDVQQDEiwDXNkb29yENlcnQuhNCMjEx
MDxEIMDgxMTUyWhCNDE1MDgxMTUyWjA2MR0wGwYDVQKExeRDYXnk29yE9y
Z2fXpnhGlvbJlVMBMGATUEAxIMQ2FzG9vcBDZX10MlICjAnBqkhkG9w0B
AQEEA0CAGAAU1CgkAgEaslnpb5E1/yml0f1RfSDSSE8IR7y+lw+RJj74e5jr4q
b8zMY
-----END CERTIFICATE-----
```

Private key [Copy private key](#) [Download private key](#)

```
-----BEGIN PRIVATE KEY-----
MIUKQIBAAKCgEAslnpb5E1/yml0f1RfSDSSE8IR7y+lw+RJj74e5jr4q
b8zMY
k7HeHcYzr/hmNewEVnnhXu1P0mBeQ5pp/QGo8vgEmjATNmzklNjOQjCjYwUr
sOf/Mnl1Co13x6mV1kHjZsRksMhYY1vaTEP3+vB8Hjg3MHFWrb07uvfMCj5
W8+OrkTcCKTR8+9V8janebz//zePFvh9bFZate/lhPK0G9P9gOvwloC1A
3sarHTP4Qm/LQR0tHqZfybdySpvAqVnNaDEF7mTrSB0/wjNCUBDPSTLvjC0
4Wl5f6nk0z7kmPstj+btvcqsRAgtwd9bKptjs1Yn/Gau3qt4zo
KbUrYxkjQjXlvwcQsEftUuk5ew5zuPSDRl.oByQTlxojLAfnW3g/pzDjgd/
60d6HtmvbZn45mjdyfHxCdb1Kn7N+xTojnfakwep2REV+RMcf4xGuhrRsnLsmk
mtUDeyZ9aBLs9j1YEQMf2IEq+RVUu+vbaY8K/DT1bcY+lnfs5Bpwlp5262b
oqd4SRsvs327b0w42zx0f/1VLRhPjBL0bhfr/AeZMHPkOXfz4yE+hqz6
8wfDOV9xYcRbsAF7323osjnyjEghnUrohngCpjlk/Mz2Kb48cbvn8CawEA
AQKCAgAHPTxhVNRuydCf21P1Yd+MIMlJmpQH9woRi8604Upulelpbx1CpOYu
npr7x9lTzr0/u6FLDq82tk6GOT7TkFyymNy4+2Wkn/5gTgw5roMqTrbwxxd
Aftxpq4ZVM8tIS3W7zMVhhabHAu50s0RvbVN-znTa7/vM5wXka3h3fLQW
aYEltQVn3Wlk/PzA8WFDF94HkaVaTgSUk40EcelpAcIL6CC01fnnSb6v/BBBG
khaTdAkogVv3cEmldkR2uuax48g8s7dzIkAv/JBwt+K5JrwFpHmy5AYLah
bu9MrrdihEzlxHbmDahoTwEfmc8kbU26gaEhufo4YIMk+84bEGQsnNsNR9
Msau9qkSlprYo6Mjt1Q/y1q3shf8SuZba3Xk0hfbYz297jk+j+Dtqz2akQWzDG
JLEtbGgdeYUMS2yc/C/FUVUN/YPCdn769kw/lmOR2k56wpbFrwR9jYgnQzbj3d
4AOrgs3AdevxDv0t10718cie334WusvxNCVrUzB/YDK/W7ijxjdvevHxGrh
1Gc+FkkBeinTqfDzdlkx6N80nyZuLyymRiavm9bVcrar5h5S1CE0HOw0gH
5GdesqMuqTS0eve1RUnfC1WWMvPeEushW9jbl3BBh4wflsQKCAQEay4x+c+F8
icbaKtsmPRMfYUJwe99tovDbxM/3x60yfEux1LsqagQz7nolJf7xJf0j+
vcG66A0qjwA6j+QdeEf8fe04t56uMac3fWPJA00vHCvrga9gC2h0c940/yn
6EqjWqg2zAx156RA3P/XetgsixVCFCGzPfxYbgzB71Yb9h5c2zz6G3K8+PvZ
dp+DfYfHb0f1.RuWxdkXk/QglXv7fR7zFqvkGhmS7qvcKXfqrWSpx+H71kQAUfQ
cj8lwgqrTouVoiO/D6DB10Pj/1Btg59a+jsnXxcpcj9XyhgjSj1d2sM0dQ
ha/kY4NdHhY20KCAQEa3gkrROpdgcooamednlwplrxryk5MrhEaglBeJidp
98HhNcl0wra5xuMS6ZC7R0xOkISLp4gt22W59lq/nQ7PN5PdN2RzOlrmHcS
-----END PRIVATE KEY-----
```

[Save](#)
[Save & Exit](#)

之后，您可以在应用编辑页面选择证书。

Edit Application [Save](#) [Save & Exit](#)

Name <a href="#">:</a>	app-built-in
Display name <a href="#">:</a>	Casdoor
Logo <a href="#">:</a>	<a href="https://cdn.casbin.com/logo/logo_1024x256.png">https://cdn.casbin.com/logo/logo_1024x256.png</a>
Preview:	
Home <a href="#">:</a>	<a href="https://casdoor.org">https://casdoor.org</a>
Description <a href="#">:</a>	
Organization <a href="#">:</a>	built-in
Client ID <a href="#">:</a>	c2ab05e8460fd3ff9ddee
Client secret <a href="#">:</a>	c9199f102508f089d253638f1a72b4c3e926d05
Cert <a href="#">:</a>	<input style="width: 100%; border: 1px solid #ccc; padding: 2px;" type="text" value="cert-built-in"/> <span style="color: red;">←</span>
Redirect URLs <a href="#">:</a>	<a href="#">cert_rjeegc</a> <a href="#">cert-built-in</a> <a href="#">Redirect URL</a>

Action

## 2. 前端配置

首先，通过 NPM 或 Yarn 安装 `casdoor-js-sdk`

```
npm install casdoor-js-sdk
```

或者：

```
yarn add casdoor-js-sdk
```

然后定义以下实用功能(在全局JS文件中更好，比如 `Setting.js`)：

```

import Sdk from "casdoor-js-sdk";

export function initCasdoorSdk(config) {
  CasdoorSdk = new Sdk(config);
}

export function getSignupUrl() {
  return CasdoorSdk.getSignupUrl();
}

export function getSigninUrl() {
  return CasdoorSdk.getSigninUrl();
}

export function getUserProfileUrl(userName, account) {
  return CasdoorSdk.getUserProfileUrl(userName, account);
}

export function getMyProfileUrl(account) {
  return CasdoorSdk.getMyProfileUrl(account);
}

export function getMyResourcesUrl(account) {
  return CasdoorSdk.getMyProfileUrl(account).replace("/account?", "/resources?");
}

export function signin() {
  return CasdoorSdk.signin(ServerUrl);
}

export function showMessage(type, text) {
  if (type === "") {
    return;
  } else if (type === "success") {
    message.success(text);
  } else if (type === "error") {
    message.error(text);
  }
}

export function goToLink(link) {
  window.location.href = link;
}

```

在您前端代码的入口文件(如 `index.js` 或 `app.js` 在React中), 您需要通过调用 `InitConfig()` 函数来初始化 `casdoor-js-sdk` 前4个参数应该使用与 Casdoor 后端SDK 相同的值。最后一个参数 `重定向路径` 是从Casdoor的登录页面返回的重定向URL的相对路径。

```

const config = {
  serverUrl: "https://door.casdoor.com",
  clientId: "014ae4bd048734ca2dea",
  organizationName: "casbin",
  appName: "app-casnnode",
  redirectPath: "/callback",
};

xxx.initCasdoorSdk(config);

```

(可选) 因为我们正在使用React作为示例, 我们的 `/callback` 路径正在撞击React路由。我们使用以下React组件接收 `/回调` 调用并发送到后端。如果您直接重定向到后端(如JSP 或 PHP), 您可以忽略此步骤。

```

import React from "react";
import {Button, Result, Spin} from "antd";
import {withRouter} from "react-router-dom";

```

### 3. 获取登录 URL

接下来，您可以显示“注册”和“登录”按钮或链接到您的用户。可以在前端或后端检索URL。详细信息见：[/docs/basic/core-concepts#login-urls](#)

### 4. 获取并验证token

步骤如下：

1. 用户点击登录URL并重定向到Casdoor的登录页面，如 `https://door.casbin.com/login/oauth/authorize?client_id=014ae4bd048734ca2dea&response_type=code&redirect_uri=https%3A%2F%2Fforum.casbin.com%2Fcallback&scope=read&state=app-casnode`
2. 用户输入用户名和密码，并点击登录（或者选择第三方登录，例如通过GitHub进行登录）
3. 该用户被重定向到您的应用，使用Casto发行的授权码(例如： `https://forum.casbin.com?code=xxx&state=yyy`)，您的应用程序的后端需要将授权码与访问令牌交换，并验证访问令牌是否有效和由Casdoor签发。函数`GetOAuthToken()` 和`ParseJwtToken()` 由Casdoor后端SDK提供。

以下代码显示如何获取并验证访问令牌。For a real example of Casnode (a forum website written in Go), see: <https://github.com/casbin/casnode/blob/6d4c55f5c9a3c4bd8c85f2493abad3553b9c7ac0/controllers/account.go#L51-L64>

```
// 从重定向 URL 的 GET 参数中获取代码和状态
code := c.Input().Get("code")
state := c.Input().Get("state")

// 用代码和状态交换token
token, err := auth.GetOAuthToken(code, state)
if err != nil {
    panic(err)
}

// 验证访问令牌
claims, err := auth.ParseJwtToken(token.AccessToken)
if err != nil {
    panic(err)
}
```

如果`ParseJwtToken()`结束时没有错误，那么用户已成功登录到应用程序。返回的`claims`可以稍后用来识别用户。

### 4. 用token识别用户



信息

这一部分实际上是您的应用程序本身的业务逻辑，而不是OIDC、OAuth 或Casdoor的一部分。我们只是提供正确做法，因为许多人不知道该怎么

在Casdoor中，访问令牌通常与ID令牌相同。他们是一样的。因此，访问令牌包含登录用户的所有信息。

由`ParseJwtToken()`返回的变量`claims`被定义为：

```
Type Claims struct {
    User
    AccessToken string `json:"accessToken"`
    jwt.RegisteredClaims
}
```

1. `User`: User 对象，包含登录用户的所有信息，请参见定义：[/docs/basic/core-concepts#user](#)
2. `AccessToken`: token信息
3. `jwt.RegisteredClaim`: JWT需要一些其他值。

这时，应用程序通常有两种方法记住用户会话： `session` and `JWT`。

## Session

设置Session的方法因语言和框架而大不相同。例如，Casnode 使用 [Beego web 框架](#) 并通过调用设置会话：`c.SetSessionUser()`。

```
token, err := auth.GetOAuthToken(code, state)
if err != nil {
    panic(err)
}

claims, err := auth.ParseJwtToken(token.AccessToken)
if err != nil {
    panic(err)
}

claims.AccessToken = token.AccessToken
c.SessionUser(claims) // 设置会话
```

## JWT

从 Casdoor 返回的 `accessToken` 实际上是一个 JWT。因此，如果您的应用程序使用 JWT 来保持用户 `session`，只需直接为它使用访问令牌：

1. 将访问令牌发送到前端，在本地存储浏览器等地方保存。
2. 让浏览器为每一个请求发送访问令牌到后端。
3. 调用 `ParseJwtToken()` 或使用您自己的函数来验证 `token`，通过后端提供的已登录用户信息。

## 5. (可选) 与用户表的互动



信息

这一部分是由 [Castor Public API](#) 提供的，而不是OIDC 或 OAuth 的一部分。

Casdoor Backend SDK 提供了许多辅助功能，不仅限于：

- `GetUser(name string)`: 通过用户名获取用户。
- `GetUsers()`: 获取所有用户。
- `AddUser()`: 添加一个用户。
- `UpdateUser()`: 更新一个用户。
- `DeleteUser()`: 删除一个用户。
- `CheckUserPassword(auth.User)`: 检查用户的密码。

这些函数是通过对 [Castor Public API](#) 调用 RESTful API 实现的。如果 Casdoor Backend SDK 中没有提供功能，您可以自己调用 RESTful API。

# 如何启用单点登录

## 简介

您已连接了 Casdoor，并在组织中配置了多个应用程序。 You want users to sign in once to any app in the organization and then be able to sign in when they go to another app without any extra clicks.

We offer this single sign-on feature. To enable it, you just need to:

- Enable the Auto Sign-In button.
- Fill in the URL for the home page.
- Add a Silent Sign-In function to the application home page.

### 备注

The basic sign-in process provided by Casdoor allows users to log in to other applications in the organization by selecting the user who is currently logged in or using another account.

After enabling auto sign-in, the selection box will not be displayed, and the logged-in user will log in directly.

## Configuration

1. Fill in the "home" field. 它可以是应用程序的主页或登录页面。

**Casdoor** Home Organizations Users Roles Permissions Models Adapters Providers Applications

Edit Application

Name [?](#) : app-casbin-  
Display name [?](#) : Casbin OA  
Logo [?](#) : URL [https://cdn.casbin.org/img/casbin\\_logo\\_1024x256.png](https://cdn.casbin.org/img/casbin_logo_1024x256.png)

Preview: 

Home [?](#) : <https://oa.casbin.com>

Description [?](#) : OA system for Casbin

## 2. Enable the Auto Sign-In button.

Password ON [?](#) :

Enable signup [?](#) :

Signin session [?](#) :

Auto signin [?](#) :

Enable code signin [?](#) :

Enable WebAuthn signin [?](#) :

# Add Silent Sign-In

In fact, we implement auto login by carrying parameters in the URL. Therefore, your applications need to have a method to trigger the login after jumping to the URL. We provide the [casdoor-react-sdk](#) to help you quickly implement this feature. You can see the details in [use-in-react](#).

## ① 信息

### 工作原理

1. 在应用程序主页的 URL 中，我们将携带 `silentSignin` 参数。
2. In your home page, determine whether you need to log in silently (automatically) by checking the `silentSignin` parameter. If `silentSignin === 1`, the function should return the `SilentSignin` component, which will help you initiate a login request. Since you have auto-login enabled, users will log in automatically without clicking.

# Add Popup Sign-In

The "popup sign-in" feature will open a small window. After logging in to Casdoor in the child window, it will send authentication information to the main window and then close automatically. We implement this feature by carrying parameters in the URL.

## ① 信息

### How to use

Use the `popupSignin()` method in the [casdoor-js-sdk](#) to quickly implement this feature. You can see a demo in [casdoor-nodejs-react-example](#).

### How it works

1. In the URL to the application home page, we will carry the `popup` parameter.
2. When `popup=1` is in the login parameters, Casdoor will send `code` and `state` as a message to the main window and finish getting the `token` in the main window using the SDK.

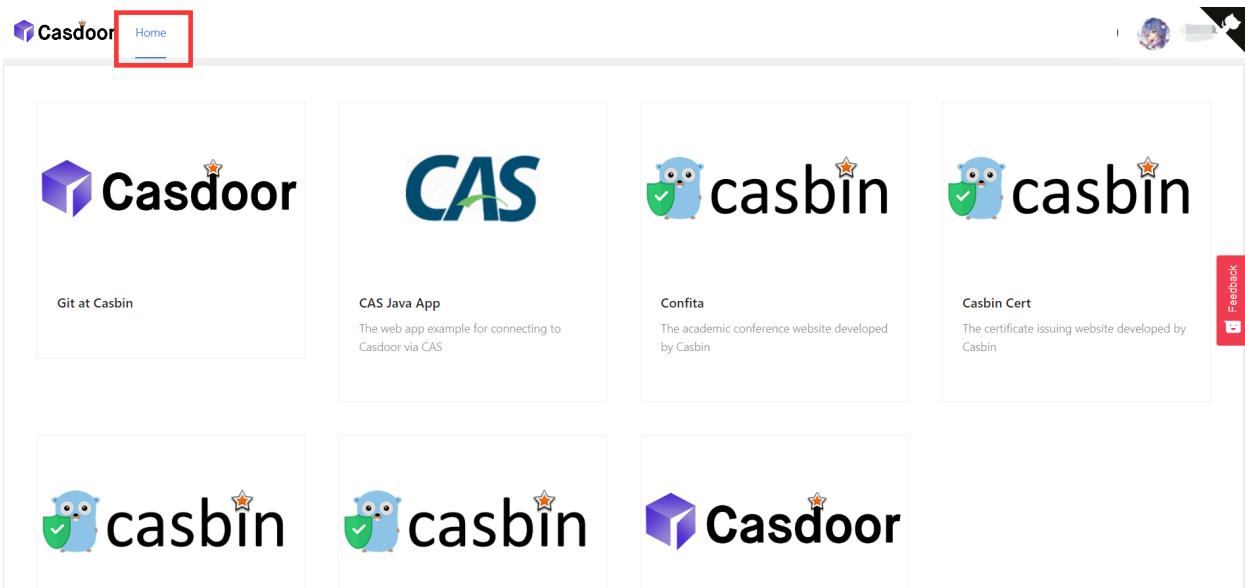
## 使用 SSO

The configuration is complete. Below, we will show you how to use auto login.

### ① 信息

Make sure your application can redirect to the user's profile page. The [getMyProfileUrl\(account, returnUrl\)](#) API is provided in our SDK for each language.

Open the profile page and go to the "Home" page (`/` URL path). You will see the application list provided by the organization. It's worth noting that only users in organizations other than "built-in" can see the application list on the "Home" page. All the global administrators (those in the "built-in" organization) cannot see it.



Click on a tile in the application list, and it will jump to the homepage URL of that application with the GET parameter `?silentSignin=1`. It will automatically log into the application if the application has integrated with Casdoor SSO (so it will recognize the `?silentSignin=1` parameter and perform a silent login in the background).

# Vue SDK

The Casdoor Vue SDK is designed for Vue 2 and Vue 3, making it very convenient to use.

The Vue SDK is based on casdoor-js-sdk. You can also use the casdoor-js-sdk directly, which will allow for more customization.

Please note that this plugin is still in development. If you have any questions or suggestions, please feel free to contact us by opening an [issue](#).

We will now show you the necessary steps below.

If you are still unsure how to use it after reading the README.md, you can refer to the example: [casdoor-python-vue-sdk-example](#) for more details.

The example's front-end is built with casdoor-vue-sdk, while the back-end is built with casdoor-python-sdk. You can view the source code in the example.

## 安装

```
# NPM  
npm install casdoor-vue-sdk  
  
# Yarn  
yarn add casdoor-vue-sdk
```

# Initializing the SDK

To initialize the SDK, you will need to provide 5 string parameters in the following order:

Name	Required	描述信息
服务器Url	是	The URL of your Casdoor server.
客户端Id:	是	The Client ID of your Casdoor application.
应用程序名称	是	The name of your Casdoor application.
组织名称	是	The name of the Casdoor organization linked to your Casdoor application.
重定向路径	否	The path of the redirect URL for your Casdoor application. If not provided, it will default to <code>/callback</code> .

For Vue 3:

```
// in main.js
```

For Vue 2:

```
// in main.js
import Casdoor from 'casdoor-vue-sdk'
import VueCompositionAPI from '@vue/composition-api'

const config = {
  serverUrl: "http://localhost:8000",
  clientId: "4262bea2b293539fe45e",
  organizationName: "casbin",
  appName: "app-casnnode",
  redirectPath: "/callback",
};

Vue.use(VueCompositionAPI)
Vue.use(Casdoor, config)

new Vue({
  render: h => h(App),
}).$mount('#app')
```

## Example

```
// in app.vue
<script>
export default {
  name: 'App',
  methods: {
    login() {
      window.location.href = this.getSigninUrl();
    },
    signup() {
      window.location.href = this.getSignupUrl();
    }
}
```

## Auto Fix

If the `postinstall` hook does not get triggered or if you have updated the Vue version, try running the following command to resolve the redirecting issue:

```
npx vue-demi-fix
```

For more information about switching Vue versions, please refer to the [vue-demi docs](#).

# 桌面 SDK

## Electron App Example for Casdoor

This is an Electron app example that demonstrates Casdoor's integration capabilities.

## dotNET Desktop App

A dotNET desktop app example for Casdoor

## Mobile SDKs .NET MAUI App

A .NET MAUI App example for Casdoor

## Qt 桌面应用程序

A Qt desktop app example for Casdoor

# Electron App Example for Casdoor

An [Electron app example](#) that demonstrates Casdoor's integration capabilities.

## How to Run the Example

### 初始化

You need to initialize 6 parameters, all of which are string type:

名称	描述	路径
serverUrl	Your Casdoor server URL	<code>src/App.js</code>
clientId	The Client ID of your Casdoor application	<code>src/App.js</code>
appName	The name of your Casdoor application	<code>src/App.js</code>
redirectPath	The path of the redirect URL for your Casdoor application, will be <code>/callback</code> if not provided	<code>src/App.js</code>
clientSecret	The Client Secret of your Casdoor application	<code>src/App.js</code>
casdoorServiceDomain	Your Casdoor server URL	<code>public/electron.js</code>

如果您没有设置这些参数，此项目将使用 [Casdoor 在线演示](#) 作为默认的Casdoor 服务器，并使用 [Casnode](#) 作为默认的 Casdoor 应用程序。

### Available Commands

在项目目录中，您可以运行：

`npm run dev` 或者 `yarn dev`

Builds the electron app and runs this app.

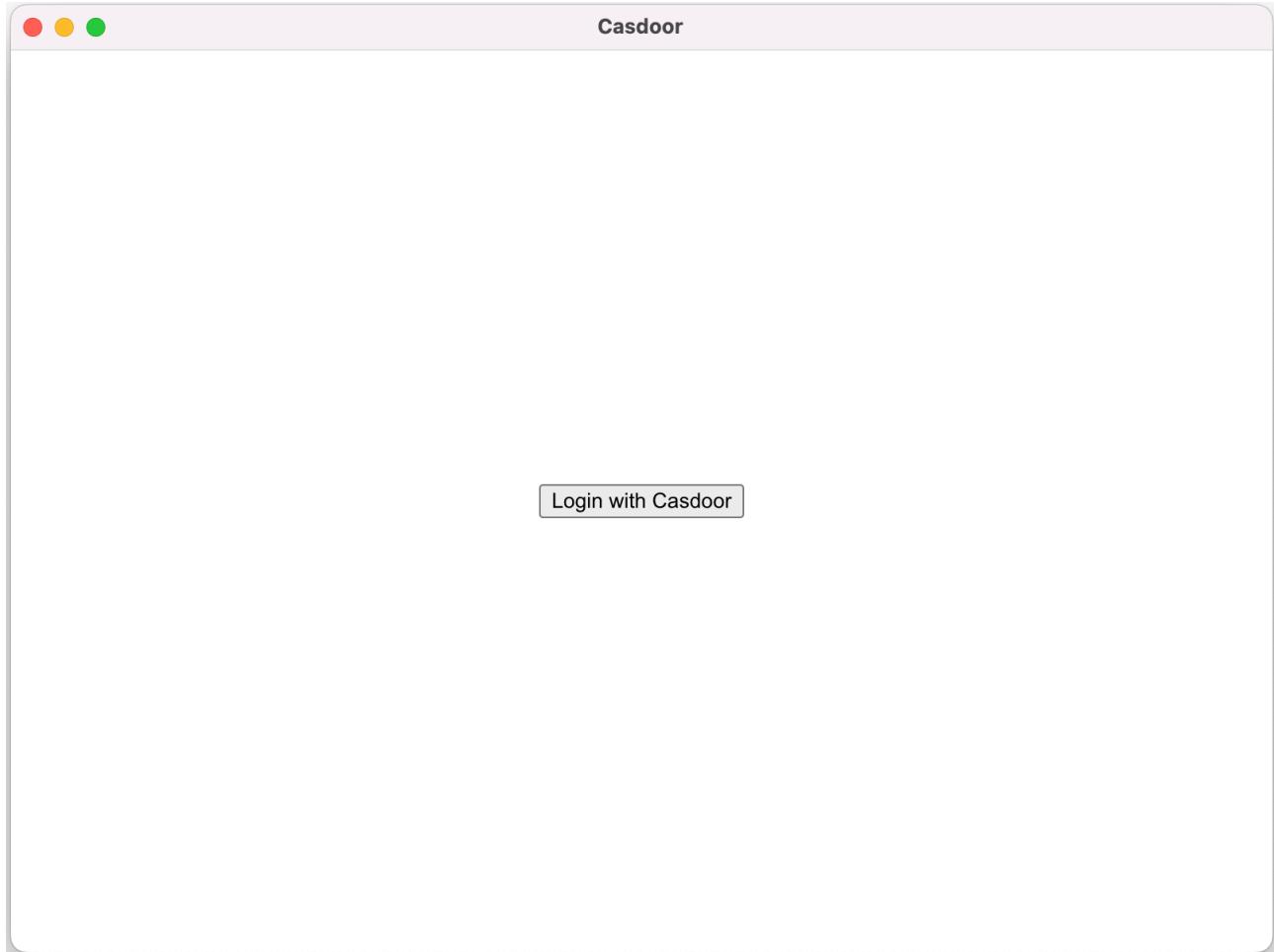
`npm run make` 或者 `yarn make`

Packages and distributes your application. 它将创建 `out` 文件夹，您的软件包将被定位于：

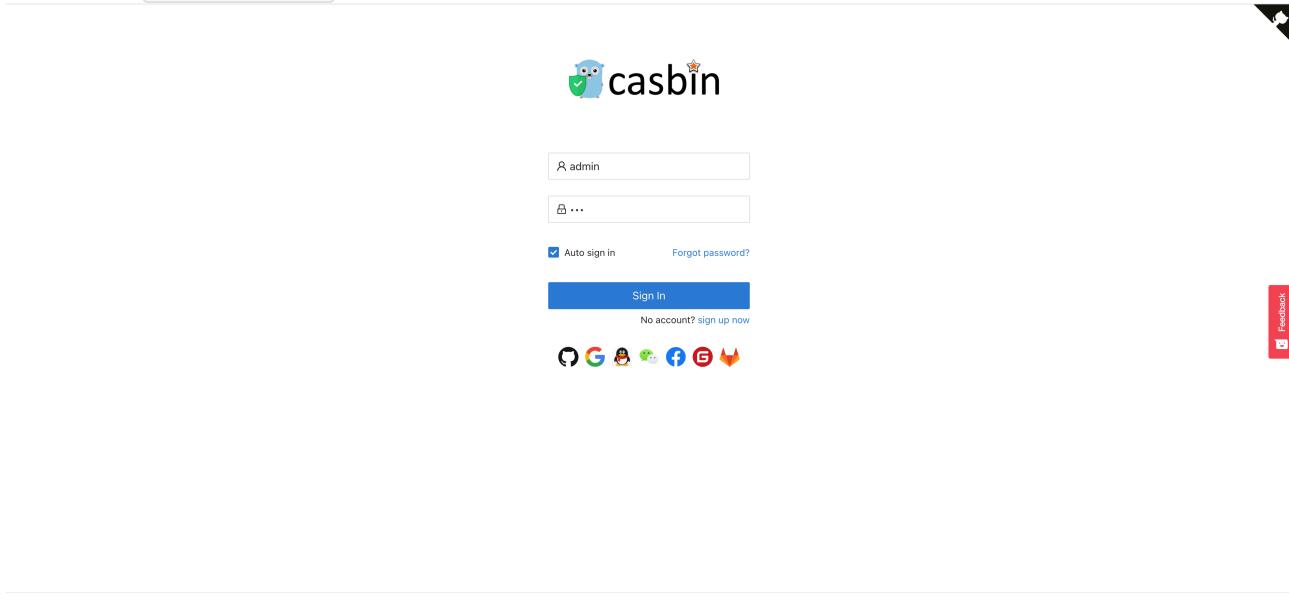
```
// macOS下的out/示例
├── out/make/zip/darwin/x64/casdoor-electron-example-darwin-x64-1.0.0.zip
└── ...
  └── out/casdoor-electron-example-darwin-x64/casdoor-electron-example.app/Contents/MacOS/casdoor-electron-example
```

### Preview

Once you run this Electron application, a new window will appear on your desktop.



If you click the `Login with Casdoor` button, your default browser will automatically open and display the login page.

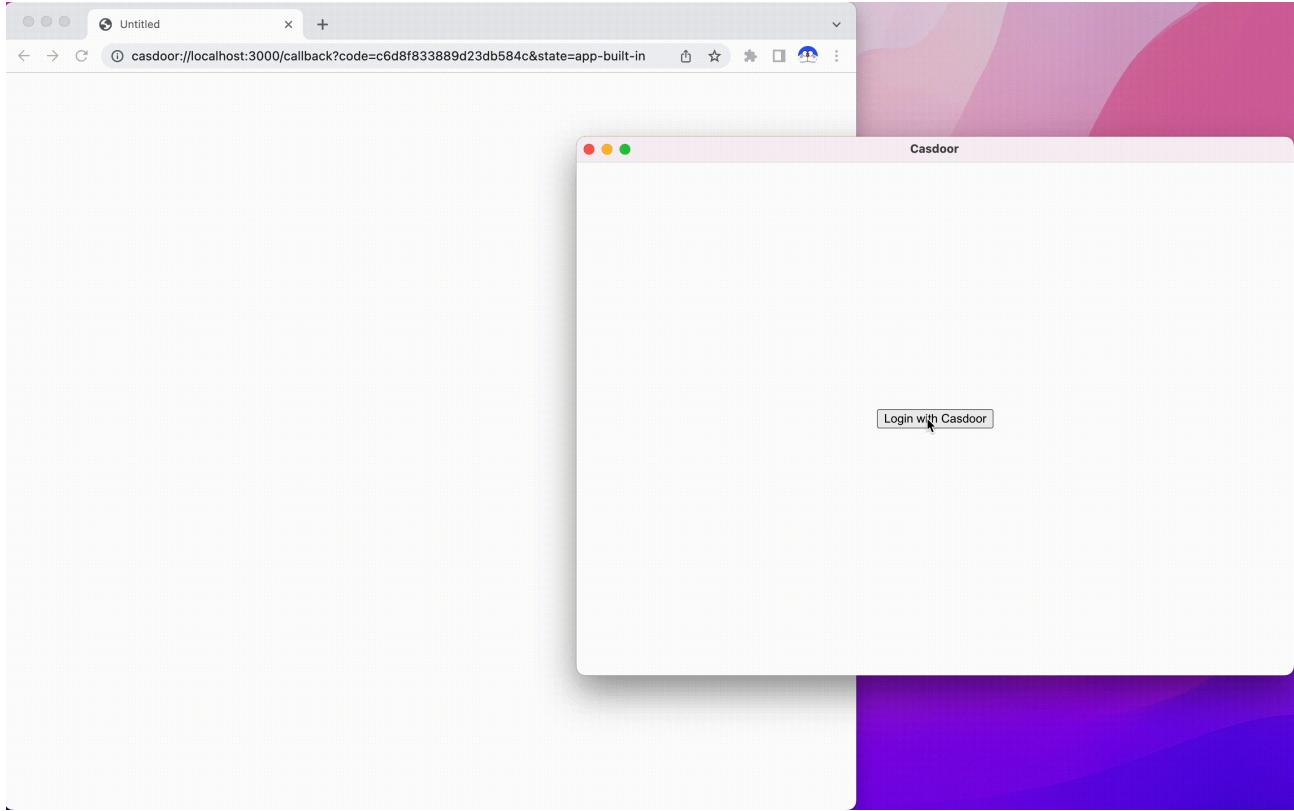


Made with ❤ by Casdoor

Following a successful login, your Electron application will open, and your user name will be displayed on your application.



You can preview the entire process in the gif image below.



## Integration Steps

### 设置自定义协议

首先，您需要设置自定义协议名为 `casdoor`。

```
const protocol = "casdoor";

if (process.defaultApp) {
  if (process.argv.length >= 2) {
    app.setAsDefaultProtocolClient(protocol, process.execPath, [
      path.resolve(process.argv[1]),
    ]);
  }
} else {
  app.setAsDefaultProtocolClient(protocol);
}
```

This will allow the browser to open your electron application and send the login info to the electron application.

### Open the login URL in the browser

```
const serverUrl = "https://door.casdoor.com";
const appName = "app-casnode";
const redirectPath = "/callback";
const clientId = "014ae4bd048734ca2dea";
const clientSecret = "f26a4115725867b7bb7b668c81e1f8f7fae1544d";

const redirectUrl = "casdoor://localhost:3000" + redirectPath;
```

You can change the first five parameters.

## 监听开启的应用程序事件

Once you successfully log in through the browser, the browser will open your Electron application. Therefore, you must listen to the open application event.

```
const gotTheLock = app.requestSingleInstanceLock();
const ProtocolRegExp = new RegExp(`^${protocol}://`);

if (!gotTheLock) {
  app.quit();
} else {
  app.on("second-instance", (event, commandLine, workingDirectory) => {
    if (mainWindow) {
      if (mainWindow.isMinimized()) mainWindow.restore();
      mainWindow.focus();
      commandLine.forEach((str) => {
        if (ProtocolRegExp.test(str)) {
          const params = url.parse(str, true).query;
          if (params && params.code) {
            store.set("casdoor_code", params.code);
            mainWindow.webContents.send("receiveCode", params.code);
          }
        }
      });
    }
  });
app.whenReady().then(createWindow);

app.on("open-url", (event, openUrl) => {
  const isProtocol = ProtocolRegExp.test(openUrl);
  if (isProtocol) {
    const params = url.parse(openUrl, true).query;
    if (params && params.code) {
      store.set("casdoor_code", params.code);
      mainWindow.webContents.send("receiveCode", params.code);
    }
  }
});
}
```

您可以从broswer获取代码，即`casdoor_code`或`params.code`。

## 解析代码并获取用户信息

```
async function getUserInfo(clientId, clientSecret, code) {
  const { data } = await axios({
    method: "post",
    url: authCodeUrl,
    headers: {
      "content-type": "application/json",
    },
    data: JSON.stringify({
      grant_type: "authorization_code",
      client_id: clientId,
      client_secret: clientSecret,
      code: code,
    }),
  });
  const resp = await axios({
    method: "get",
    url: `${getUserInfoUrl}?accessToken=${data.access_token}`,
  });
  return resp.data;
}
```

Finally, you can parse the code and get the user info following the [OAuth docs page](#).

# dotNET Desktop App

A [Dotnet desktop app example](#) for Casdoor.

## How to Run the Example

### Prerequisites

- [dotNET 6 SDK](#)
- [WebView2 Runtime](#) (It is usually preinstalled on Windows)

### Initialization

The initialization requires 5 parameters, all of which are of type string:

名称	描述	文件
Domain	The host/domain of your Casdoor server	<a href="#">CasdoorVariables.cs</a>
Clientid	您的 Casdoor 应用程序的客户端 ID	<a href="#">CasdoorVariables.cs</a>
AppName	您的Casdoor应用程序的名称	<a href="#">CasdoorVariables.cs</a>
CallbackURL	The path of the callback URL for your Casdoor application. If not	<a href="#">CasdoorVariables.cs</a>

名称	描述	文件
	provided, it will be <code>casdoor://callback</code>	
ClientSecret	您的Casdoor应用程序的客户端密钥	<code>CasdoorVariables.cs</code>

If you do not set these parameters, the project will default to using the [Casdoor online demo](#) as the Casdoor server and the [Casnode](#) as the Casdoor application.

## Running

### Visual Studio

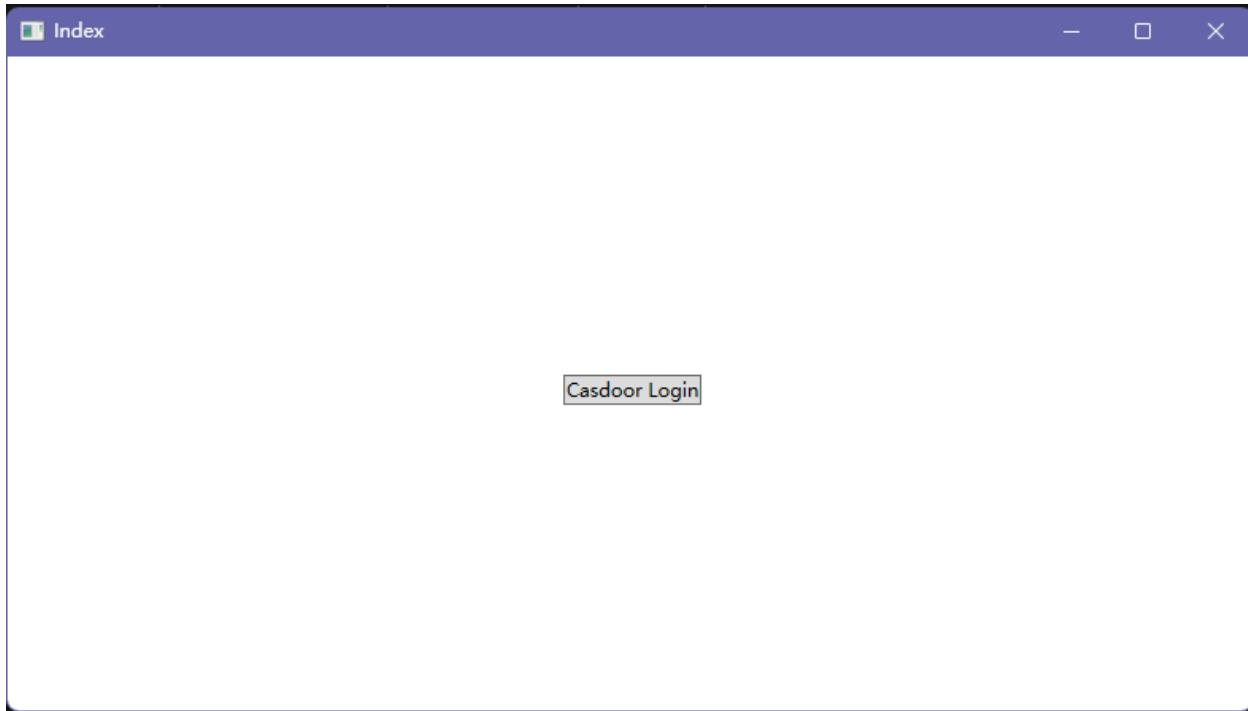
1. Open `casdoor-dotnet-desktop-example.sln`
2. Press `Ctrl + F5` to start

### Command Line

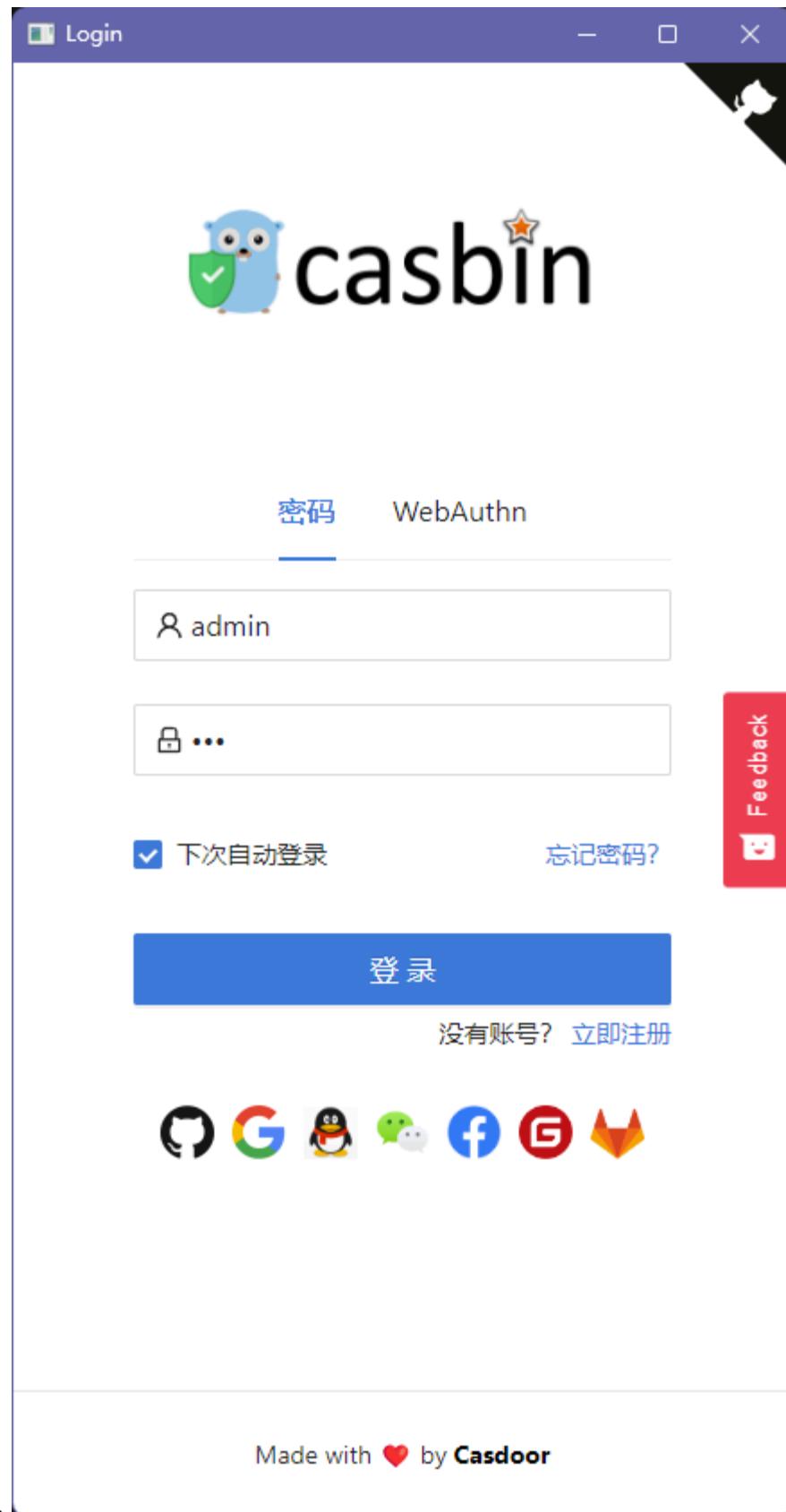
1. `cd src/DesktopApp`
2. `dotnet run`

## Preview

After running the dotNET desktop application, a new window will appear on your desktop.

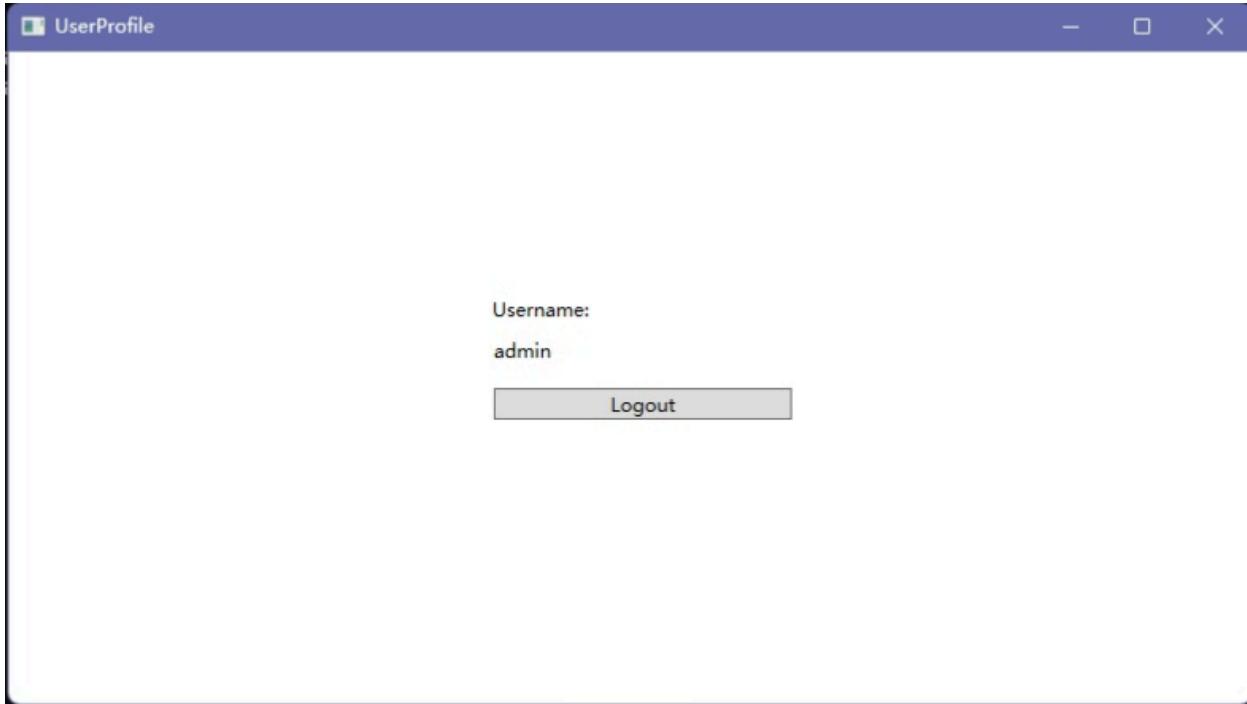


If you click the `Casdoor Login` button, a login window will appear on your



desktop.

After successfully logging in, a user profile window will appear on your desktop, displaying your username.



You can preview the entire process in the GIF image below.



# How to Integrate

## Opening the Login Window

```
var login = new Login();
// Triggered when login succeeds, you will receive an auth code in
// the event handler
login.CodeReceived += Login_CodeReceived;
login.ShowDialog();
```

## Using the Auth Code to Get User Info

```
public async Task<string?> RequestToken(string clientId, string
clientSecret, string code)
{
    var body = new
    {
        grant_type = "authorization_code",
        client_id = clientId,
        client_secret = clientSecret,
        code
    };

    var req = new RestRequest(_requestTokenUrl).AddJsonBody(body);
    var token = await _client.PostAsync<TokenDto>(req);

    return token?.AccessToken;
}

public async Task<UserDto?> GetUserInfo(string token)
{
    var req = new
    RestRequest(_getUserInfoUrl).AddQueryParameter("accessToken",
```





> 如何连接到Casdoor > 桌面 SDK > Mobile SDKs .NET MAUI App

# Mobile SDKs .NET MAUI App

This repository contains a .NET MAUI app and .NET MAUI library for demonstrating Casdoor authentication by OpenID Connect.

# Demonstration

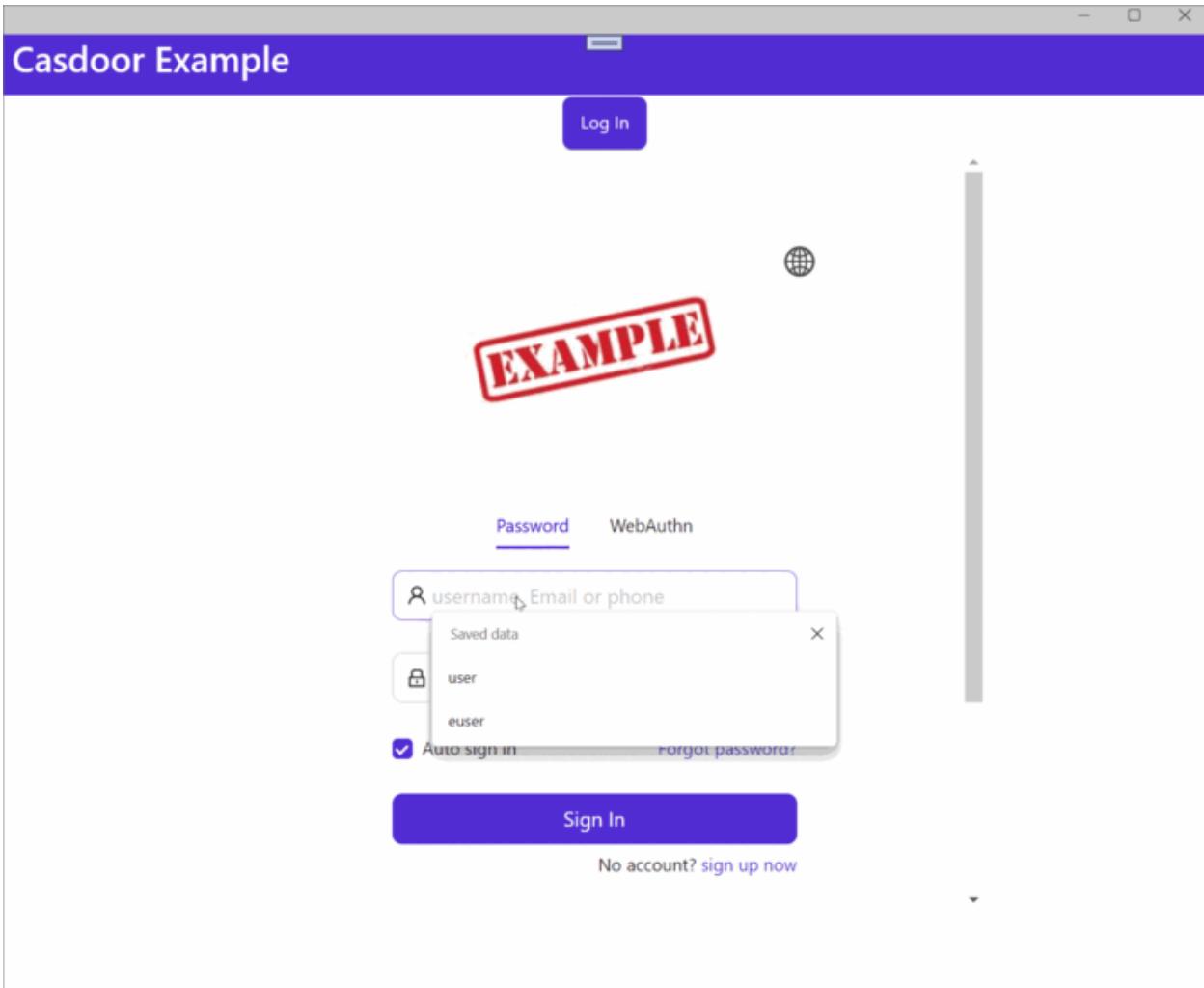
Android

21:06



.NET

# Windows



# Requirements

- [.NET 7 SDK](#) installed on your machine
- The required assets needed for your target platform(s), as described [here](#)
- Visual Studio 2022 for Windows 17.3 or Visual Studio 2022 for Mac 17.4 (optional)

# Getting Started

## Step 1: Create a MAUI Application

Create your [MAUI Application](#).

## Step 2: Add a Reference

Add a reference to the `Casdoor.MauiOidcClient` in your project.

## Step 3: Add the Casdoor Client

Add `CasdoorClient` as a singleton in the services.

```
builder.Services.AddSingleton(new CasdoorClient(new()
{
    Domain = "<your domain>",
    ClientId = "<your client>",
    Scope = "openid profile email",

#if WINDOWS
    RedirectUri = "http://localhost/callback"
#else
    RedirectUri = "casdoor://callback"
#endif
}));
```

## Step 4: Design the UI

Add code to the `MainPage` file.

## MainPage.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://schemas.microsoft.com/dotnet/2021/maui"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="Casdoor.MauiOidcClient.Example.MainPage">

    <ScrollView>
        <VerticalStackLayout>

            <StackLayout
                x:Name="LoginView">
                <Button
                    x:Name="LoginBtn"
                    Text="Log In"
                    SemanticProperties.Hint="Click to log in"
                    Clicked="OnLoginClicked"
                    HorizontalOptions="Center" />

                <WebView x:Name="WebViewInstance" />
            </StackLayout>

            <StackLayout
                x:Name="HomeView"
                IsVisible="false">

                <Label
                    Text="Welcome to .NET Multi-platform App UI"
                    SemanticProperties.HeadingLevel="Level2"
                    SemanticProperties.Description="Welcome to dot net
Multi-platform App UI"
                    FontSize="18"
                    HorizontalOptions="Center" />

                <Button
                    x:Name="CounterBtn"
                    Text="Click me"
```

## MainPage.cs

```
namespace Casdoor.MauiOidcClient.Example
{
    public partial class MainPage : ContentPage
    {
        int count = 0;
        private readonly CasdoorClient client;
        private string accessToken;
        public MainPage(CasdoorClient client)
        {
            InitializeComponent();
            this.client = client;

#if WINDOWS
            client.Browser = new
WebViewBrowserAuthenticator(WebViewInstance);
#endif
        }

        private void OnCounterClicked(object sender, EventArgs e)
        {
            count++;

            if (count == 1)
                CounterBtn.Text = $"Clicked {count} time";
            else
                CounterBtn.Text = $"Clicked {count} times";

            SemanticScreenReader.Announce(CounterBtn.Text);
        }

        private async void OnLoginClicked(object sender, EventArgs e)
        {
            var loginResult = await client.LoginAsync();
            accessToken = loginResult.AccessToken;
```

## Step 5: Support the Android Platform

Modify the `AndroidManifest.xml` file.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/
    android">
    <application android:allowBackup="true" android:icon="@mipmap/
        appicon" android:roundIcon="@mipmap/appicon_round"
        android:supportsRtl="true"></application>
    <uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <queries>
        <intent>
            <action
                android:name="android.support.customtabs.action.CustomTabsService"
            />
            </intent>
        </queries>
    </manifest>
```

## Step 6: Launch the Application

Visual Studio: Press Ctrl + F5 to start.

# Qt 桌面应用程序

一个为 Casdoor 的 [Qt桌面应用程序示例](#)。

## How to Run the Example

### 前置要求

- [Qt6 SDK](#)
- [OpenSSL toolkit](#)

### 初始化

You need to initialize 7 string parameters:

名称	描述	文件
endpoint	您的 Casdoor 服务器主机/域	<a href="#">mainwindow.h</a>
client_id	您的 Casdoor 应用程序的客户端 ID	<a href="#">mainwindow.h</a>
client_secret	The Client Secret of your Casdoor application	<a href="#">mainwindow.h</a>
certificate	Casdoor 应用程序证书的公钥	<a href="#">mainwindow.h</a>

名称	描述	文件
org_name	您的Casdoor应用程序的名称	mainwindow.h
app_name	您的Casdoor应用程序的名称	mainwindow.h
redirect_url	您的Casdoor 应用程序的回调URL路径将是 casdoor://callback 如果没有提供	mainwindow.h

If you don't set the `endpoint` parameter, this project will use <http://localhost:8000> as the default Casdoor server.

## Running the Application

### Using Qt Creator

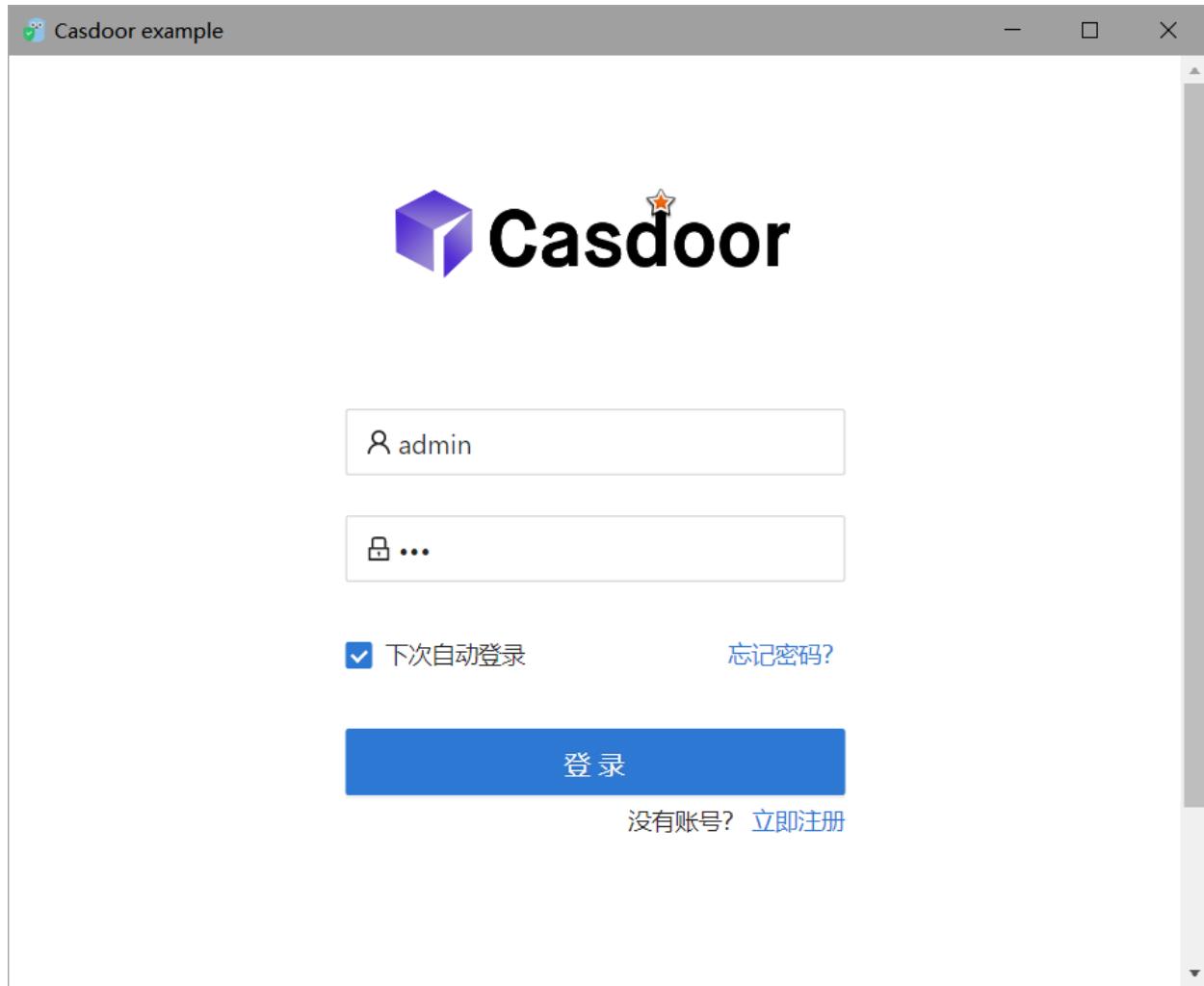
1. Open `casdoor-cpp-qt-example.pro`
2. Set the `INCLUDEPATH` of OpenSSL in `casdoor-cpp-qt-example.pro`
3. Press `Ctrl + R` to start

## 效果预览

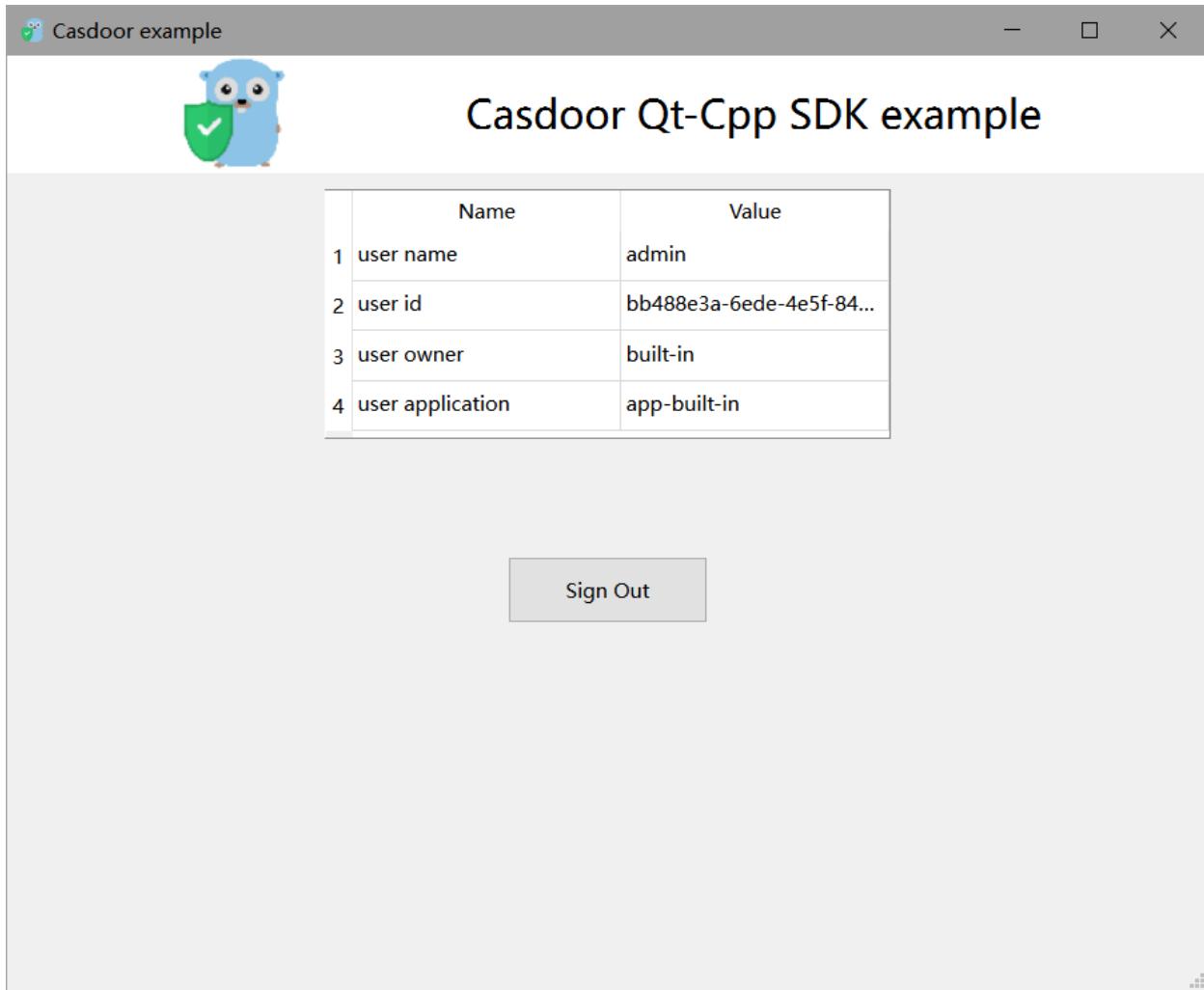
After running this Qt desktop application, a new window will be shown on your desktop.



If you click the `Sign In` button, a login window will be shown on your desktop.



After a successful login, a user profile window will be shown on your desktop, displaying your user information.



You can preview the entire process in the following GIF image.



# How to Integrate

## Opening the Login Window

```
// Load and display the login page of Casdoor
m_webview->page()->load(*m_signin_url);
m_webview->show();
```

## Listening to the Open Application Event

```
// Initialize the TcpServer object and listen on port 8080
m_tcpserver = new QTcpServer(this);
if (!m_tcpserver->listen(QHostAddress::LocalHost, 8080)) {
    qDebug() << m_tcpserver->errorString();
    close();
}
connect(m_tcpserver, SIGNAL(newConnection()), this,
SLOT(on_tcp_connected()));
```

## Using Auth Code to Get the User Info

```
// Get the token and parse it with the JWT library
std::string token = m_casdoor->GetOAuthToken(code.toStdString());
auto decoded = m_casdoor->ParseJwtToken(token);
```

# 移动 SDKs

## React Native App

A React Native mobile app example for Casdoor

# React Native App

There is a [Casdoor React Native mobile app example](#) to get you up to speed on how to use Casdoor in React Native.

## How to Run the Example

### Quick Start

- download the code

```
git clone git@github.com:casdoor/casdoor-react-native-example.git
```

- install dependencies

```
cd casdoor-react-native-example
yarn install
cd ios/
pod install
```

- run on ios

```
cd casdoor-react-native-example
react-native start
react-native run-ios
```

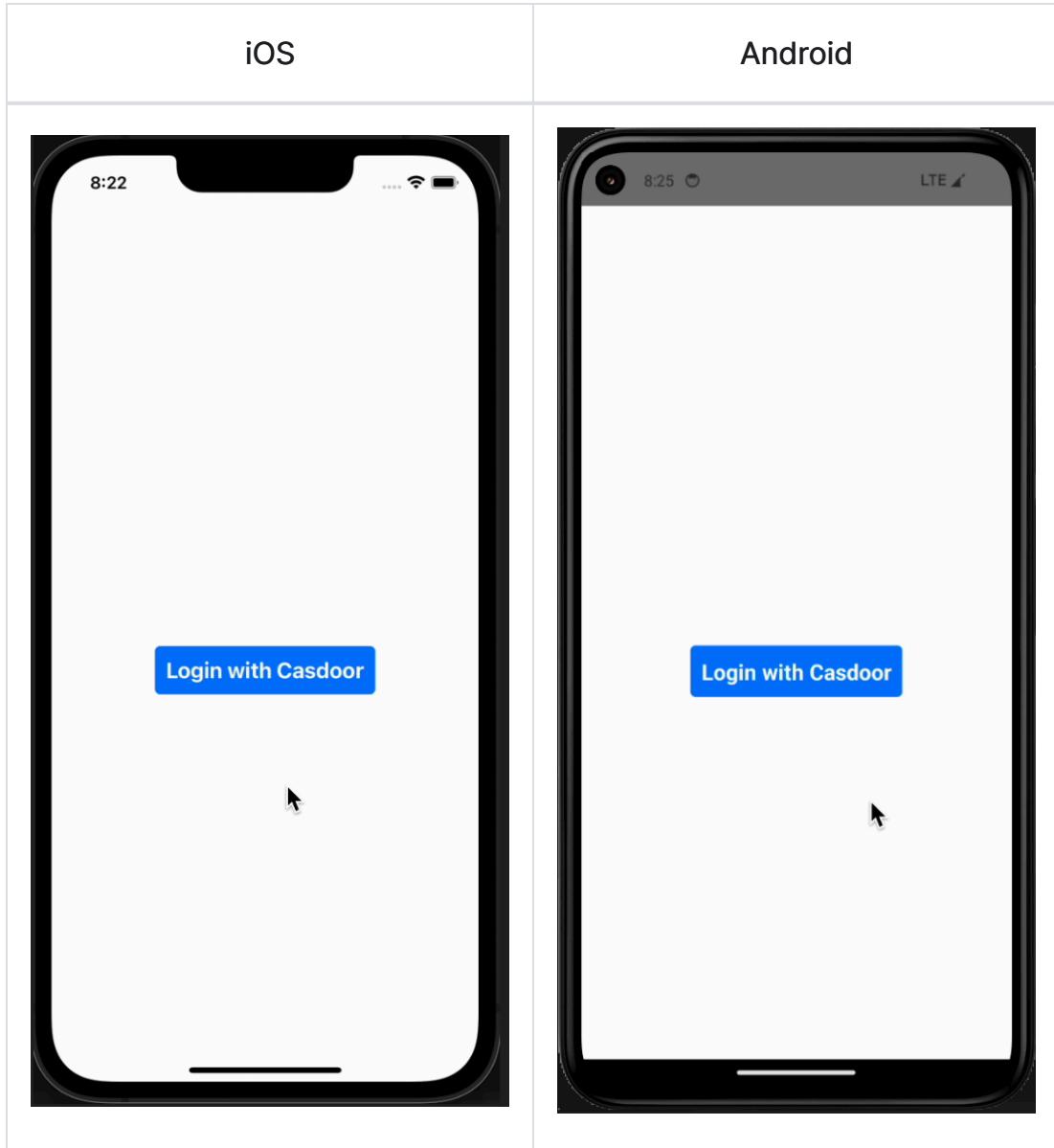
- run on android

```
cd casdoor-react-native-example  
react-native start  
react-native run-android
```

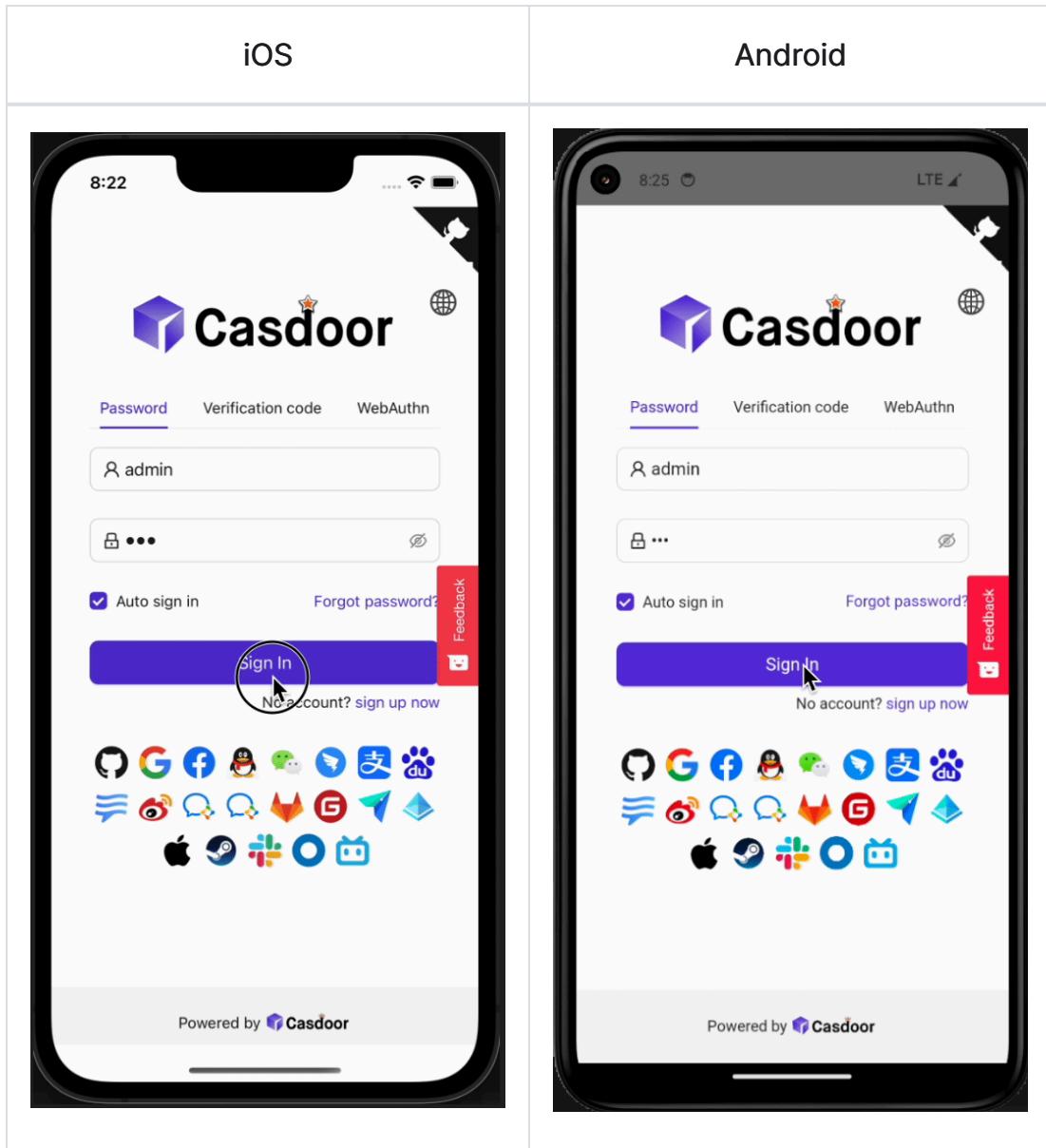
Make sure to turn on the emulator or real device before running.

## Preview

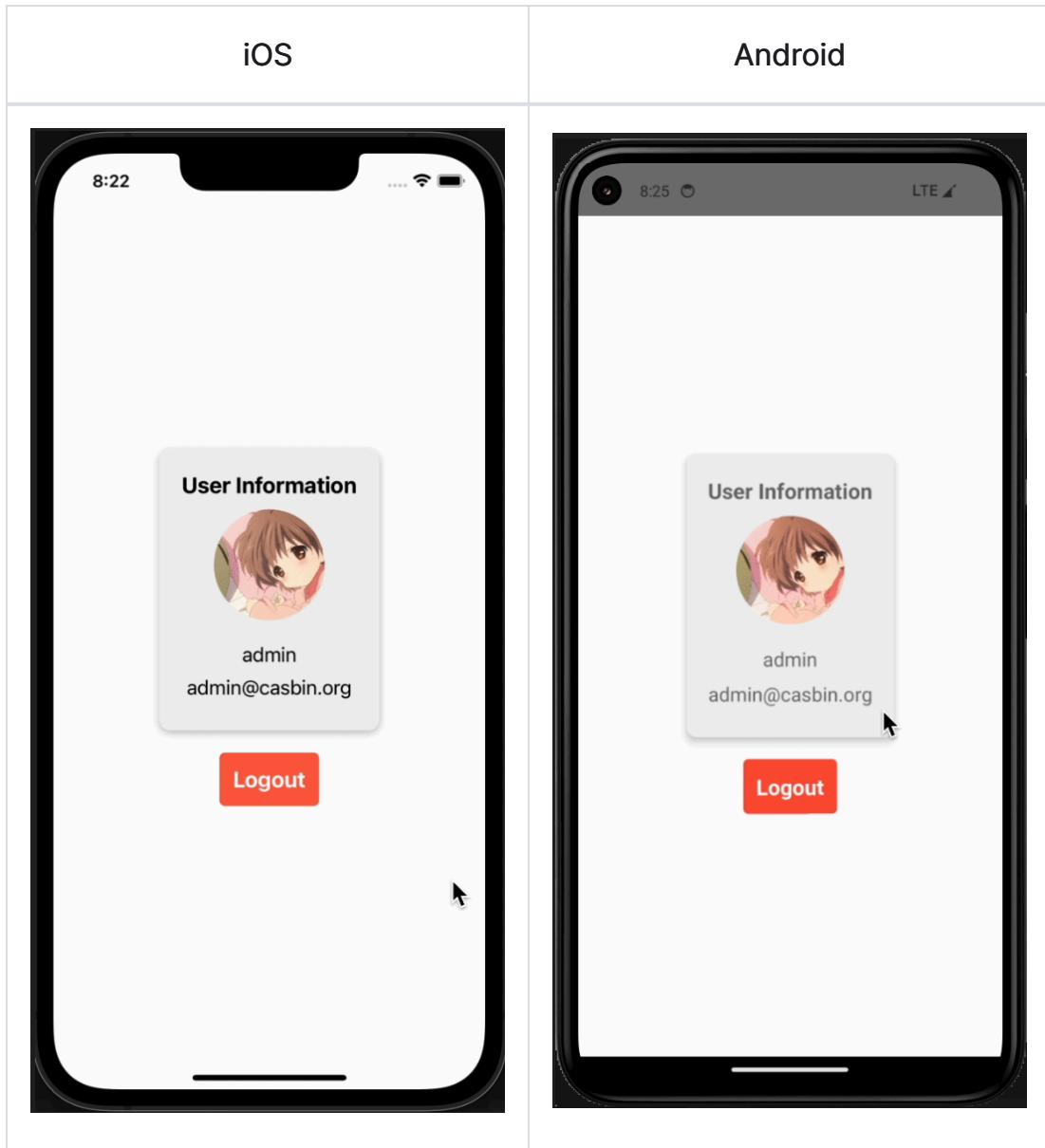
After running this react-native-example mobile application, the following window will be displayed on the emulator or real device.



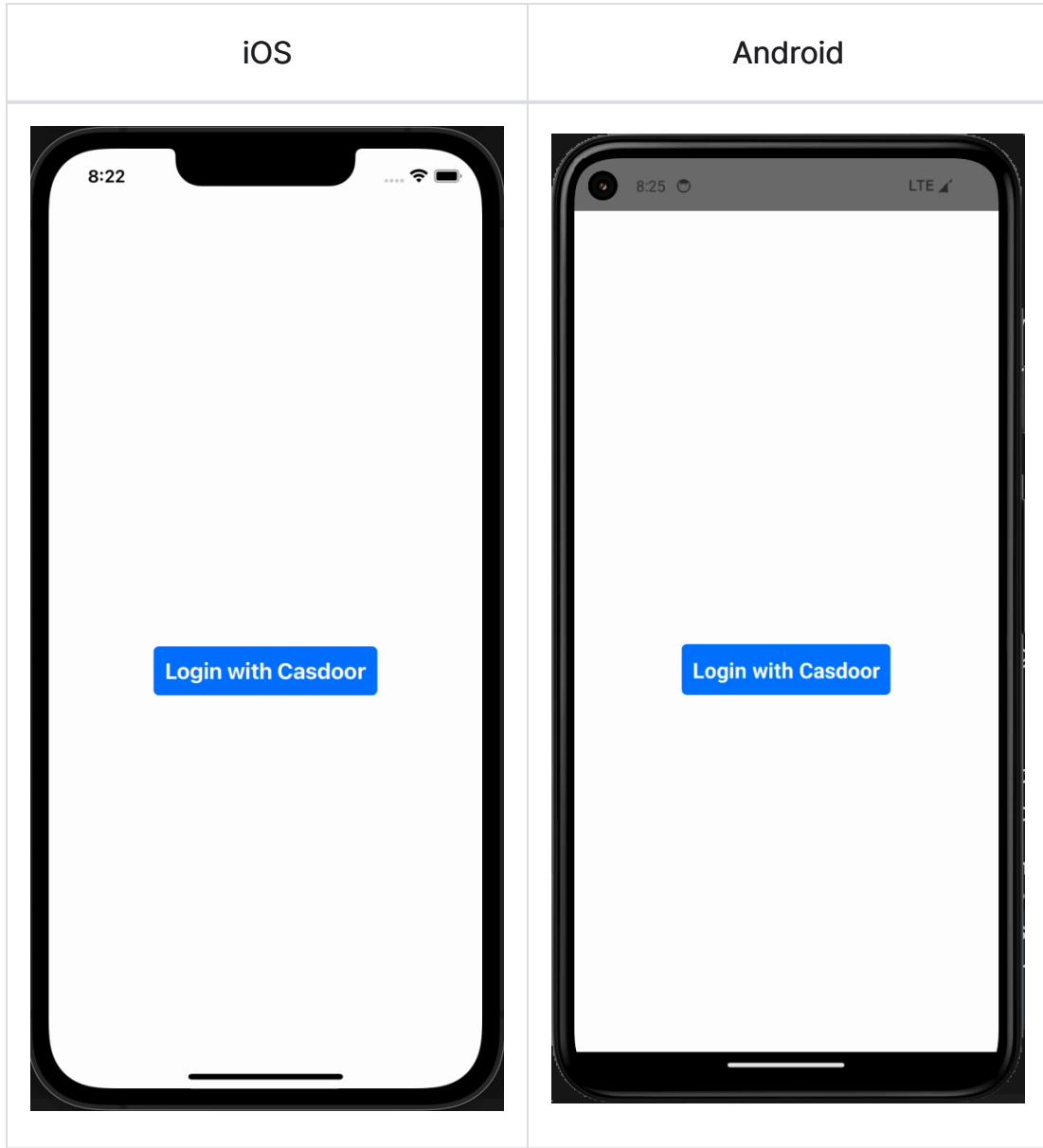
If you click the `Login with Casdoor` button, the Casdoor login window will appear on the screen.



After a successful login, a user profile window will appear on your screen displaying your user information.



You can preview the entire process in the following GIF image.



## How to Integrate

The above example uses [casdoor-react-native-sdk](#), you can also integrate this sdk in your own project.

The integration and use of the sdk is very simple, the following steps will show

you how to integrate and use:

## Step 1: Import SDK

```
# NPM  
npm i casdoor-react-native-sdk  
  
# Yarn  
yarn add casdoor-react-native-sdk
```

## Step 2: Initialize SDK

Initialization requires 7 parameters, which are all string type:

Name (in order)	Must	Description
serverUrl	Yes	your Casdoor server URL
clientId	Yes	the Client ID of your Casdoor application
appName	Yes	the name of your Casdoor application
organizationName	Yes	the name of the Casdoor organization connected with your Casdoor application
redirectPath	No	the path of the redirect URL for your Casdoor application, will be <code>/callback</code> if not provided
signinPath	No	the path of the signin URL for your Casdoor application

```
import SDK from 'casdoor-react-native-sdk'

const sdkConfig = {
  serverUrl: 'https://door.casdoor.com',
  clientId: 'b800a86702dd4d29ec4d',
  appName: 'app-example',
  organizationName: 'casbin',
  redirectPath: 'http://localhost:5000/callback',
  signinPath: '/api/signin',
};
const sdk = new SDK(sdkConfig)
```

## Step 3: Use SDK

Use the corresponding API interface of the sdk at the appropriate place.

The simplest casdoor authorization and authentication process can be realized by using the following three APIs:

```
// get the signin url
getSigninUrl()

// get Access Token
getAccessToken(redirectUrl); // http://localhost:5000/
callback?code=b75bc5c5ac65ffa516e5&state=gjmfqdgqf498

// decode jwt token to get user info
JwtDecode(jwtToken)
```

If you want to use other interfaces, please check [casdoor-react-native-sdk](#) for more help.



如何连接到Casdoor

Casdoor插件

# Casdoor插件

Casdoor also provides plugins or middlewares for some very popular platforms, such as Java's Spring Boot, PHP's WordPress, and Python's Odoo, among others.

Casdoor 插件	语言	源代码
Spring Boot插件	Java	<a href="https://github.com/casdoor/casdoor-spring-boot-starter">https://github.com/casdoor/casdoor-spring-boot-starter</a>
Spring Boot示例	Java	<a href="https://github.com/casdoor/casdoor-spring-boot-example">https://github.com/casdoor/casdoor-spring-boot-example</a>
WordPress 插件	PHP	<a href="https://github.com/casdoor/wordpress-casdoor-plugin">https://github.com/casdoor/wordpress-casdoor-plugin</a>
Odoo 插件	Python	<a href="https://github.com/casdoor/odoo-casdoor-oauth">https://github.com/casdoor/odoo-casdoor-oauth</a>
Django 插件	Python	<a href="https://github.com/casdoor/django-casdoor-auth">https://github.com/casdoor/django-casdoor-auth</a>

For a complete list of the official Casdoor plugins, please visit the [Casdoor repositories](#).

# Next.js

[nextjs-auth](#) is an example of how to integrate casdoor in a next-js project. We will guide you through the steps below.

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed.

You can refer to the Casdoor official documentation for the [Server Installation](#). Please deploy your Casdoor instance in production mode.

After a successful deployment, make sure the following:

- Open your favorite browser and visit `http://localhost:8000`. You will see the login page of Casdoor.
- Test the login functionality by entering `admin` as the username and `123` as the password.

After that, you can quickly implement a Casdoor-based login page in your own app using the following steps.

## Step 2: Add Middleware

Middleware allows you to run code before a request is completed. Then, based on the incoming request, you can modify the response by rewriting, redirecting, modifying the request or response headers, or responding directly.

Use the file `middleware.ts` (or `.js`) in the root of your project to define Middleware. For example, at the same level as `pages` or `app`, or inside `src` if applicable.

## Example

```
//define which paths Middleware will run on
const protectedRoutes = ["/profile"];

export default function middleware(req) {
  if (protectedRoutes.includes(req.nextUrl.pathname)) {
    //redirect the incoming request to a different URL
    return NextResponse.redirect(new URL("/login", req.url));
  }
}
```

See next.js official documentation [middleware](#) for more details.

## Step 3: Use Casdoor SDK

### 1. Install the SDK

First, install `casdoor-js-sdk` via NPM or Yarn:

```
npm install casdoor-js-sdk
```

Or:

```
yarn add casdoor-js-sdk
```

## 2.Initializing the SDK

Then initialization 6 string-type parameters in the following order:

Name	Required	Description
serverUrl	Yes	Casdoor Server URL, such as <code>http://localhost:8000</code>
clientId	Yes	Application client ID
clientSecret	Yes	Application client secret
organizationName	Yes	Application organization
appName	Yes	Application name
redirectPath	Yes	redirected URL

## Example

```
const sdkConfig = {
  serverUrl: "https://door.casdoor.com",
  clientId: "294b09fbc17f95daf2fe",
  clientSecret: "dd8982f7046ccba1bbd7851d5c1ece4e52bf039d",
  organizationName: "casbin",
  appName: "app-vue-python-example",
  redirectPath: "/callback",
};
```

### ⚠ 注意事项

Replace the configuration values with your own Casdoor instance, especially the `clientId`, `clientSecret`, and `serverUrl`.

## 3.Redirect to the Login Page

When you need to authenticate users who access your app, you can send the target URL and redirect to the login page provided by Casdoor.

Make sure you have added the callback URL (e.g. `http://localhost:8080/callback`) in the application configuration beforehand.

```
const CasdoorSDK = new Sdk(sdkConfig);
CasdoorSDK.signin_redirect();
```

## 4.Get Token and Storage

After the Casdoor verification is passed, it will redirect back to your application with token.

You can opt in to use cookie to storage the token.

```
CasdoorSDK.exchangeForAccessToken()
  .then((res) => {
    if (res && res.access_token) {
      //Get Token
      return CasdoorSDK.getUserInfo(res.access_token);
    }
  })
  .then((res) => {
    // Storage Token
  })
```

You can refer to the Casdoor official documentation for the [How to use Casdoor SDK](#).

## Step 4: Add Middleware Authentication Function

when users attempt to access a protected route, Middleware Authentication function verifies their identity. If the user is not authenticated, they are redirected to a login page or denied access.

### Example

```
//protected route
const protectedRoutes = ["/profile"];
const casdoorUserCookie = req.cookies.get("casdoorUser");
const isAuthenticated = casdoorUserCookie ? true : false;

//Authentication Function
if (!isAuthenticated &&
protectedRoutes.includes(req.nextUrl.pathname)) {
  return NextResponse.redirect(new URL("/login", req.url));
}
```

# Nuxt

[nuxt-auth](#) is an example of how to integrate casdoor in a nuxt project. We will guide you through the steps below. Many steps are similar to [nextjs-auth](#).

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed.

You can refer to the Casdoor official documentation for the [Server Installation](#). Please deploy your Casdoor instance in production mode.

After a successful deployment, make sure the following:

- Open your favorite browser and visit `http://localhost:8000`. You will see the login page of Casdoor.
- Test the login functionality by entering `admin` as the username and `123` as the password.

After that, you can quickly implement a Casdoor-based login page in your own app using the following steps.

## Step 2: Add Middleware

Middleware allows you to run code before a request is completed. Then, based on the incoming request, you can modify the response by rewriting, redirecting, modifying the request or response headers, or responding directly.

Create `.js` or `.ts` files in `middleware` directory in the root of your project to define Middleware. And the filenames are identified as the names of middleware. For example, in `nuxt-auth`, we create a file named `myMiddleware.js` in `middleware` directory, which can be referenced as `myMiddleware` in other places like `nuxt.config.js`.

## Example

```
//define which paths Middleware will run on
const protectedRoutes = ["/profile"];

export default function ({route, redirect}) {

  if (protectedRoutes.includes(route.path)) {
    //redirect the incoming request to a different URL
    redirect('/login');
  }
}
```

To make middleware work, you should add router in `nuxt.config.js`, like that:

```
export default {
  // other configurations

  // what to add
  router: {
    middleware: ['myMiddleware'] // replace to your middleware name
  },
}
```

See nuxt official documentation [middleware](#) for more details.

# Step 3: Use Casdoor SDK

## 1. Install the SDK

First, install `casdoor-js-sdk` via NPM or Yarn:

```
npm install casdoor-js-sdk
```

Or:

```
yarn add casdoor-js-sdk
```

## 2. Initializing the SDK

Then initialization 6 string-type parameters in the following order:

Name	Required	Description
serverUrl	Yes	Casdoor Server URL, such as <code>http://localhost:8000</code>
clientId	Yes	Application client ID
clientSecret	Yes	Application client secret
organizationName	Yes	Application organization

Name	Required	Description
appName	Yes	Application name
redirectPath	Yes	redirected URL

## Example

```
const sdkConfig = {  
    serverUrl: "https://door.casdoor.com",  
    clientId: "294b09fbc17f95daf2fe",  
    clientSecret: "dd8982f7046ccba1bbd7851d5c1ece4e52bf039d",  
    organizationName: "casbin",  
    appName: "app-vue-python-example",  
    redirectPath: "/callback",  
};
```

### ⚠ 注意事项

Replace the configuration values with your own Casdoor instance, especially the `clientId`, `clientSecret`, and `serverUrl`.

## 3.Redirect to the Login Page

When you need to authenticate users who access your app, you can send the target URL and redirect to the login page provided by Casdoor.

Make sure you have added the callback URL (e.g. `http://localhost:8080/callback`) in the application configuration beforehand.

```
const CasdoorSDK = new Sdk(sdkConfig);
CasdoorSDK.signin_redirect();
```

## 4.Get Token and Storage

After the Casdoor verification is passed, it will redirect back to your application with token.

You can opt in to use cookie to storage the token.

```
CasdoorSDK.exchangeForAccessToken()
  .then((res) => {
    if (res && res.access_token) {
      //Get Token
      return CasdoorSDK.getUserInfo(res.access_token);
    }
  })
  .then((res) => {
    // Storage Token
    Cookies.set("casdoorUser", JSON.stringify(res));
  });
}
```

You can refer to the Casdoor official documentation for the [How to use Casdoor SDK](#).

## Step 4: Add Middleware Authentication Function

when users attempt to access a protected route, Middleware Authentication function verifies their identity. If the user is not authenticated, they are redirected to a login page or denied access.

## Example

```
import Cookies from "js-cookie";

const protectedRoutes = ["/profile"];

export default function ({route, redirect}) {
  const casdoorUserCookie = Cookies.get('casdoorUser');
  const isAuthenticated = !!casdoorUserCookie;

  if (!isAuthenticated && protectedRoutes.includes(route.path)) {
    redirect('/login');
  }
}
```

# OAuth 2.0

## 介绍

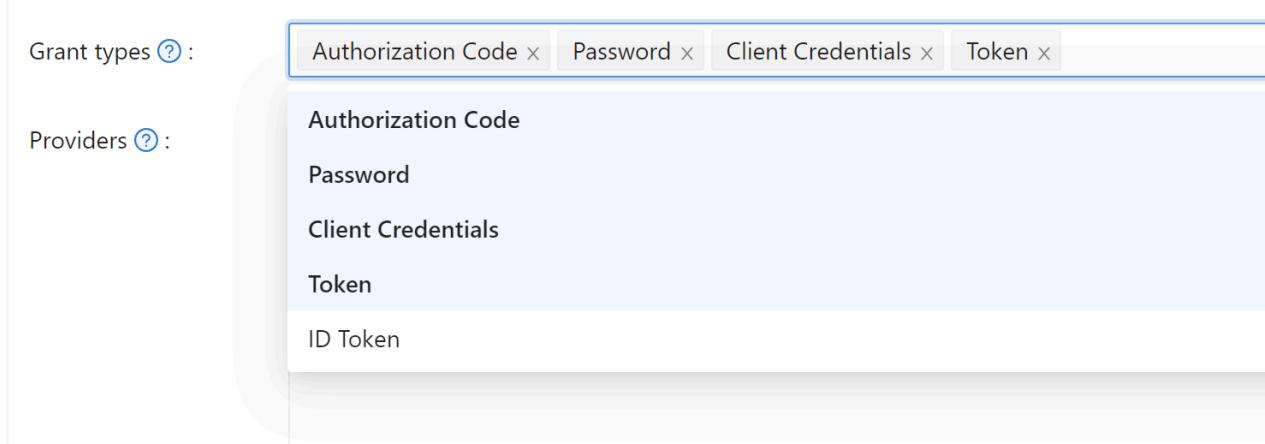
Casdoor 支持使用 AccessToken 验证客户端。在本节中，我们将向您展示如何获取 AccessToken，如何验证 AccessToken，以及如何使用 AccessToken。

## 如何获取AccessToken

获取访问令牌有两种方式：您可以使用 [Casdoor SDK](#) 详情请参阅SDK文档。这里我们将主要向您展示如何使用 API 来获取访问令牌。

Casdoor支持四种OAuth 授予类型: [Authorization Code Grant](#), [Implicit Grant](#), [Resource Owner Password Credentials Grant](#), 和 [Client Credentials Grant](#).

出于安全考虑，Casto应用默认已开启授权码模式。如果您需要使用其他模式，请前往相应的应用程序来设置它。



## 获取授权码

首先，重定向您的用户到：

```
https://<CASDOOR_HOST>/login/oauth/authorize?  
client_id=CLIENT_ID&  
redirect_uri=REDIRECT_URI&  
response_type=code&  
scope=openid&  
state=STATE
```

## 可用的作用域 (scope)

名称	描述
openid (no scope)	sub (用户ID), iss (发行人) 和 aud (受众)
profile	用户资料信息，包括名称、显示名称、头像
email	用户的电子邮件地址
address	用户地址
phone	用户的电话号码

### ① 信息

您的 OAuth 应用程序可以在首次重定向时带上请求的作用域。您可以指定多个作用域并使用空格（转义后为%20）分隔：

```
https://<CASDOOR_HOST>/login/oauth/authorize?  
client_id=...&  
scope=openid%20email
```

更多详情，请参阅 [OIDC 标准](#)

当您的用户通过 Casdoor 身份验证后，他会被 Casdoor 重定向到：

```
https://REDIRECT_URI?code=CODE&state=STATE
```

现在您已经获得授权码，在你的后端应用发送 POST 请求：

```
https://<CASDOOR_HOST>/api/login/oauth/access_token
```

在你的后端应用

```
{  
  "grant_type": "authorization_code",  
  "client_id": ClientId,  
  "client_secret": ClientSecret,  
  "code": Code,  
}
```

您将得到以下响应：

```
{  
    "access_token": "eyJhb...","  
    "id_token": "eyJhb...","  
    "refresh_token": "eyJhb...","  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

#### ⓘ 备注

Casdoor 也支持 PKCE 功能。当获取验证码时，您可以添加两个参数来启用PKCE:

```
&code_challenge_method=S256&code_challenge=YOUR_CHANNELLENGE
```

获取令牌时，您需要传递 `code_verifier` 参数来验证 PKCE。值得一提的是，启用 PKCE 后，`Client_Secret` 并不是必需的，但如果想要传递这个参数，它的值就必须是正确的。

## 隐式授权

如果您的应用程序没有后端，您需要使用隐式授权。首先，您需要确保您启用了隐式授权，然后将您的用户请求重定向到:

```
https://<CASDOOR_HOST>/login/oauth/  
authorize?client_id=CLIENT_ID&redirect_uri=REDIRECT_URI&response_type=token&scope=openid&state=STATE
```

After your user has authenticated with Casdoor, Casdoor will redirect them to:

```
https://REDIRECT_URI/#access_token=ACCESS_TOKEN
```

Casdoor also supports the `id_token` as `response_type`, which is a feature of OpenID.

## 使用资源拥有者的密码凭据授权

如果您的应用程序没有前端来重定向用户到Casdoor，那么您可能需要这个功能。

First, you need to make sure you have Password Credentials Grant enabled and send a POST request to:

```
https://<CASDOOR_HOST>/api/login/oauth/access_token
```

```
{  
    "grant_type": "password",  
    "client_id": ClientId,  
    "client_secret": ClientSecret,  
    "username": Username,  
    "password": Password,
```

您将得到以下响应：

```
{  
    "access_token": "eyJhb... ",  
    "id_token": "eyJhb... ",  
    "refresh_token": "eyJhb... ",  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

## 使用客户端凭据授权

当应用程序没有前端时，您也可以使用客户端凭据授权。

First, you need to make sure you have Client Credentials Grant enabled and send a POST request to

[https://<CASDOOR\\_HOST>/api/login/oauth/access\\_token](https://<CASDOOR_HOST>/api/login/oauth/access_token):

```
{  
    "grant_type": "client_credentials",  
    "client_id": ClientId,  
    "client_secret": ClientSecret,  
}
```

您将得到以下响应：

```
{  
    "access_token": "eyJhb... ",  
    "id_token": "eyJhb... ",  
    "refresh_token": "eyJhb... ",  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

必须指出，以这种方式获得的AccessToken 不同于前三个，因为它与应用程序相对应，而不是与用户相对应。

## Refresh Token

Maybe you want to update your Access Token, then you can use the `refreshToken` you obtained above.

First, you need to set the expiration time of the Refresh Token in the application (default is 0 hours), and send a POST request to [https://<CASDOOR\\_HOST>/api/login/oauth/refresh\\_token](https://<CASDOOR_HOST>/api/login/oauth/refresh_token)

```
{  
    "grant_type": "refresh_token",  
    "refresh_token": REFRESH_TOKEN,  
    "scope": SCOPE,
```

You will get a response like this:

```
{  
    "access_token": "eyJhb... ",  
    "id_token": "eyJhb... ",  
    "refresh_token": "eyJhb... ",  
    "token_type": "Bearer",  
    "expires_in": 10080,  
    "scope": "openid"  
}
```

## How to Verify Access Token

Casdoor currently supports the [token introspection](#) endpoint. This endpoint is protected by Basic Authentication (ClientId:ClientSecret).

```
POST /api/login/oauth/introspect HTTP/1.1  
Host: CASDOOR_HOST  
Accept: application/json  
Content-Type: application/x-www-form-urlencoded  
Authorization: Basic Y2xpZW50X2lkOmNsawVudF9zZWNyZXQ=  
  
token=ACCESS_TOKEN&token_type_hint=access_token
```

You will receive the following response:

```
{  
    "active": true,  
    "client_id": "c58c... ",  
    "username": "admin",  
    "token_type": "Bearer",  
    "exp": 1647138242,  
    "iat": 1646533442,  
    "nbf": 1646533442,  
    "sub": "7a6b4a8a-b731-48da-bc44-36ae27338817",  
    "aud": [  
        "c58c... "  
    ],  
    "iss": "http://localhost:8000"  
}
```

## How to Use AccessToken

您可以使用AccessToken访问需要认证的 Casdoor API。

For example, there are two different ways to request [/api/userinfo](#).

Type 1: Query parameter

```
https://<CASDOOR\_HOST>/api/userinfo?accessToken=<your\_access\_token>
```

Type 2: HTTP Bearer token

```
https://<CASDOOR\_HOST>/api/userinfo with the header: "Authorization: Bearer <your_access_token>"
```

Casdoor will parse the access\_token and return corresponding user information according to the scope. You will receive the same response, which looks like this:

```
{  
  "sub": "7a6b4a8a-b731-48da-bc44-36ae27338817",  
  "iss": "http://localhost:8000",  
  "aud": "c58c..."  
}
```

If you expect more user information, add scope when obtaining the AccessToken in step [Authorization Code Grant](#).

## Differences between the userinfo and get-account APIs

- </api/userinfo>: This API returns user information as part of the OIDC protocol. It provides limited information, including only the basic information defined in OIDC standards. For a list of available scopes supported by Casdoor, please refer to the [available scopes](#) section.
- </api/get-account>: This API retrieves the user object for the currently logged-in account. It is a Casdoor-specific API that allows you to obtain all the information of the user in Casdoor.

# Using Casdoor as a CAS Server

## Using Casdoor as a CAS Server

Casdoor can now be used as a CAS server. It currently supports CAS 3.0.

### 简介

The CAS endpoint prefix in Casdoor is `<Casdoor endpoint>/cas/<organization name>/<application name>`. Here is an example using the endpoint `https://door.casdoor.com` with an application named `cas-java-app` under the organization `casbin`:

- `/login` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/login`
- `/logout` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/logout`
- `/serviceValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/serviceValidate`
- `/proxyValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/proxyValidate`
- `/proxy` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/proxy`
- `/validate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/validate`
- `/p3/serviceValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/p3/serviceValidate`
- `/p3/proxyValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/p3/proxyValidate`
- `/samlValidate` endpoint: `https://door.casdoor.com/cas/casbin/cas-java-app/samlValidate`

For more information about CAS, its different versions, and parameters for these endpoints, refer to the [CAS Protocol Specification](#).

### An Example

Here is an official example [GitHub Repository](#) that contains a web app and utilizes the official CAS Java client [GitHub Repository](#). By going through this example, you can learn how to connect to Casdoor via CAS.



备注

Note: Currently, Casdoor only supports all three versions of CAS: CAS 1.0, 2.0, and 3.0.

The CAS configuration is located in `src/main/webapp/WEB-INF/web.yml`.

By default, this app uses CAS 3.0, which is specified by the following configurations:

```
<filter-name>CAS Validation Filter</filter-name>
<filter-
class>org.jasig.cas.client.validation.Cas30ProxyReceivingTicketValidationFilter</filter-
class>
```

If you want to protect this web app using CAS 2.0, change the CAS Validation Filter to the following:

```
<filter-name>CAS Validation Filter</filter-name>
<filter-
class>org.jasig.cas.client.validation.Cas20ProxyReceivingTicketValidationFilter</filter-
class>
```

For CAS 1.0, use the following:

```
<filter-name>CAS Validation Filter</filter-name>
<filter-class>org.jasig.cas.client.validation.Cas10TicketValidationFilter</filter-class>
```

For all instances of the `casServerUrlPrefix` parameter, change them to:

```
<param-name>casServerUrlPrefix</param-name>
<param-value>http://door.casdoor.com/cas/casbin/cas-java-app</param-value>
```

For all instances of the `casServerLoginUrl` parameter, change them to:

```
<param-name>casServerLoginUrl</param-name>
<param-value>http://door.casdoor.com/cas/casbin/cas-java-app/login</param-value>
```

If you need to customize more configurations, see the [Java CAS client GitHub Repository](#) for detailed information.

# SAML

## Overview

Using Casdoor as SAML IdP

## AWS Client VPN

Using Casdoor as a SAML IdP

## Keycloak

Using Casdoor as a SAML IdP

## Google Workspace

Using Casdoor as a SAML IdP

 **Appgate (POST)**

How to Use Casdoor as SAML IdP for Appgate

 **Tencent Cloud**

Using Casdoor as a SAML IdP

# Overview

Casdoor can now be used as a SAML IdP. Up to this point, Casdoor has supported the main features of SAML 2.0.

## Configuration in SP

In general, the SP requires three required fields: `Single Sign-On`, `Issuer`, and `Public Certificate`. Most SPs can obtain these fields by uploading the XML Metadata file or the XML Metadata URL for autocompletion.

The metadata of the SAML endpoint in Casdoor is `<Endpoint of casdoor>/api/saml/metadata?application=admin/<application name>`. Suppose the endpoint of Casdoor is `https://door.casdoor.com`, and it contains an application called `app-built-in`. The XML Metadata endpoint will be:

```
https://door.casdoor.com/api/saml/metadata?application=admin/app-built-in
```

You can also find the metadata in the application edit page. Click the button to copy the URL and paste it into the browser to download the XML Metadata.

```
SAML metadata ⓘ
<EntityDescriptor xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" entityID="https://door.casdoor.com">
    <IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
        <KeyDescriptor use="signing">
            <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
                    <X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TCCAUgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDIxNkb...
                </X509Data>
            </KeyInfo>
        </KeyDescriptor>
        <NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat>
        <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat>
        <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
        <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0-bindings:HTTP-Redirect" Location="https://door.casdoor.com/login/saml/authorize/admin/app-built-in"></SingleSignOnService>
    
```

[Copy SAML metadata URL](#)

# Configuration in Casdoor IdP

Casdoor supports both GET and POST `SAMLResponse`. Casdoor needs to know what types of requests the SP supports when Casdoor sends the `SAMLResponse` to the SP. You need to configure the application in Casdoor based on the `SAMLResponse` type supported by your SP.

## ⓘ 信息

If you fill in the `Reply URL`, Casdoor will send the `SAMLResponse` by POST Request. If the Reply URL is empty, Casdoor will use GET request. You might wonder how Casdoor knows the `Reply URL` of the SP if the `Reply URL` is empty. Actually, Casdoor can get the URL called `AssertionConsumerServiceURL` by parsing the `SAMLRequest` and send the request with `SAMLResponse` to `AssertionConsumerServiceURL`. The `Reply URL` will overwrite the `AssertionConsumerServiceURL` in `SAMLRequest`.

- **Reply URL:** Type in the URL of the ACS verifying the SAML response.

The screenshot shows the Casdoor configuration interface for a specific application. At the top, under 'Grant types', 'Authorization Code' and 'Password' are selected. Below that, the 'SAML Reply URL' field contains the value `https://mycontroller.mycompany.com/admin/saml`, which is highlighted with a red border. Underneath, there is a toggle switch labeled 'Enable SAML compress' which is turned on (blue). The entire configuration section is enclosed in a light blue rounded rectangle.

Grant types	Authorization Code	Password
SAML Reply URL	🔗 https://mycontroller.mycompany.com/admin/saml	
Enable SAML compress	<input checked="" type="checkbox"/>	

- **Redirect URL:** Type in a unique name. This may be called `Audience` or `Entity ID` in your SP. Make sure you fill the same `Redirect URL` here as in your SP.

Redirect URLs [?](#) :

The screenshot shows a list of redirect URLs. At the top is a header with 'Redirect URLs' and an 'Add' button. Below is a table with three rows. The first row has a 'Redirect URL' column containing 'appgate'. The second row has a 'Redirect URL' column containing 'https://git.casbin.com/user/oauth2/casdoor/callback'. The third row has a 'Redirect URL' column containing 'http://localhost:3000/callback'.

Redirect URL
appgate
https://git.casbin.com/user/oauth2/casdoor/callback
http://localhost:3000/callback

## User profile

After successfully logging in, the user profile in the returned [SAMLResponse](#) from Casdoor has three fields. The attributes in the XML and the attributes of the user in Casdoor are mapped as follows:

XML Attribute Name	User field
Email	email
DisplayName	displayName
Name	name

See [https://en.wikipedia.org/wiki/SAML\\_2.0](https://en.wikipedia.org/wiki/SAML_2.0) for more information about SAML and its different versions.

## An example

[gosaml2](#) is a SAML 2.0 implementation for Service Providers based on etree and goxmlsig, a pure Go implementation of XML digital signatures. We use this library to test the SAML 2.0 in Casdoor as shown below.

Suppose you can access Casdoor through `http://localhost:7001/`, and your Casdoor contains an application called `app-built-in`, which belongs to an organization called `built-in`. The URLs, `http://localhost:6900/acs/example` and `http://localhost:6900/saml/acs/example`, should be added to the Redirect URLs in `app-built-in`.

```
import (
    "crypto/x509"
    "fmt"
    "net/http"

    "io/ioutil"

    "encoding/base64"
    "encoding/xml"

    saml2 "github.com/russellhaering/gosaml2"
    "github.com/russellhaering/gosaml2/types"
    dsig "github.com/russellhaering/goxmldsig"
)

func main() {
    res, err := http.Get("http://localhost:7001/api/saml/
metadata?application=admin/app-built-in")
    if err != nil {
        panic(err)
    }

    rawMetadata, err := ioutil.ReadAll(res.Body)
    if err != nil {
        panic(err)
    }

    metadata := &types.EntityDescriptor{}
    err = xml.Unmarshal(rawMetadata, metadata)
    if err != nil {
```

Run the above code, and the console will display the following message.

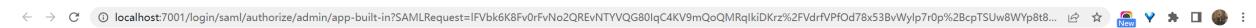
Visit this URL To Authenticate:

<http://localhost:7001/login/saml/authorize/admin/app-built-in?SAMLRequest=lFVbk6K8Fv0rFvNo2QR...>

Supply:

SP ACS URL : [http://localhost:6900/v1/\\_saml\\_callback](http://localhost:6900/v1/_saml_callback)

Click the URL to authenticate, and the login page of Casdoor will be displayed.





Continue with :



Or sign in with another account :

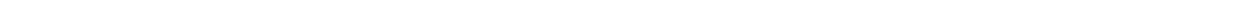
Auto sign in [Forgot password?](#)

[Sign In](#)

[Sign in with code](#) [No account? sign up now](#)



After authenticating, you will receive the response messages as shown below.



```
NameID: admin@example.com
Assertion:
  Email: [{XMLName: 'Space:urn:oasis:names:tc:SAML:2.0:assertion Local:Attribute} FriendlyName: Name>Email NameFormat:urn:oasis:names:tc:SAML:2.0:attributeName-format:basic Values:[{XMLName: 'Space:urn:oasis:names:tc:SAML:2.0:assertion Local:AttributeValue} Type: Value:admin@example.com}]
  Name: [{XMLName: 'Space:urn:oasis:names:tc:SAML:2.0:assertion Local:Attribute} FriendlyName: Name>Name NameFormat:urn:oasis:names:tc:SAML:2.0:attributeName-format:basic Values:[{XMLName: 'Space:urn:oasis:names:tc:SAML:2.0:assertion Local:AttributeValue} Type: Value:admin]]
  DisplayName: [{XMLName: 'Space:urn:oasis:names:tc:SAML:2.0:assertion Local:Attribute} FriendlyName: Name>DisplayName NameFormat:urn:oasis:names:tc:SAML:2.0:attributeName-format:basic Values:[{XMLName: 'Space:urn:oasis:names:tc:SAML:2.0:assertion Local:AttributeValue} Type: Value:Admin]}
Warnings:
&{OneTimeUse:false ProxyRestriction:<nil> NotInAudience:false InvalidTime:false}
```







# AWS Client VPN

## Casdoor as a SAML IdP in AWS Client VPN

This guide will show you how to configure Casdoor and AWS Client VPN to add Casdoor as a SAML IdP in AWS Client VPN.

## Prerequisites

To complete this setup, you will need:

- An AWS Account with administrative rights to access configuration settings of the service provider.
- An Amazon VPC with an EC2 instance
  - [Setting up the VPC](#)
  - [Launching an EC2 instance](#)
    - In the instance Security Group, allow ICMP traffic from the VPC CIDR range - this is needed for testing.
- A private certificate imported into [AWS Certificate Manager \(ACM\)](#)
  - [Generating and importing a certificate to ACM](#)
- A Windows or Mac system running the latest AWS Client VPN software.
  - [Download the software](#)

# Configure SAML Application

- In the Casdoor Application, set the `Redirect URL` to `urn:amazon:webservices:clientvpn`.

Tags [?](#) :

Client ID [?](#) : 235aca38d69a868ae432

Client secret [?](#) : d8942f2181908041106f3b2b56c2f91fd2ad13de

Cert [?](#) : cert-built-in

Redirect URLs [?](#) :

Redirect URLs	Add
Redirect URL	
<code>urn:amazon:webservices:clientvpn</code>	

Token format [?](#) :

Token expire [?](#) : 168 Hours

Refresh token expire [?](#) : 0 Hours

Enable password  [?](#):

- Set the `SAML reply URL` to `http://127.0.0.1:35001`.

Signup HTML [?](#) :

Signin HTML [?](#) :

Grant types [?](#) : Authorization Code x

SAML reply URL [?](#) : `http://127.0.0.1:35001`

Enable SAML compression [?](#) :

SAML metadata [?](#) :

```
<EntityDescriptor xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
  <IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocol="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect">
    <KeyDescriptor use="signing">
      <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
        <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
```

- Save the content in the `SAML metadata` as an XML file.

SAML metadata [?](#) :

```
<EntityDescriptor xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
  <IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocol="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect">
    <KeyDescriptor use="signing">
      <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
        <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
          <X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TC CAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW50QData</X509Certificate>
        </X509Data>
      </KeyInfo>
    </KeyDescriptor>
    <NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat>
    <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat>
    <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
    <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="https://test.v2tl.com/login/saml/authorize/admin/app">
```

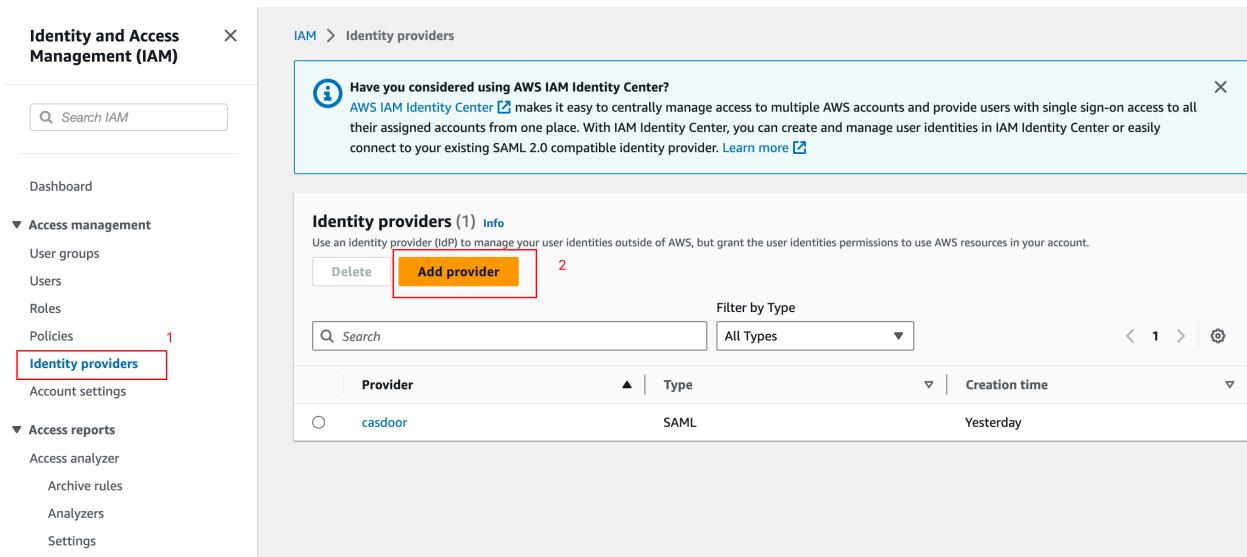
[Copy SAML metadata URL](#)

## Configure AWS

### Configure Casdoor as an AWS Identity Provider

1. Open the IAM console and select Identity providers from the navigation bar.
2. Click Create a Provider.

3. Specify SAML for the Provider Type, add a unique name for this provider, and upload the metadata document - the same file you saved from the Casdoor Application in the previous section.
  
4. Click Next Step. On the next screen, click Create.



The screenshot shows the AWS IAM Identity providers page. On the left, there's a navigation sidebar with options like Dashboard, Access management (with Identity providers selected), and Access reports. The main area shows a single identity provider named 'casdoor' (Type: SAML, Created Yesterday). A prominent orange 'Add provider' button is highlighted with a red box and the number '2'. A tooltip at the top right encourages using AWS IAM Identity Center.

Provider	Type	Creation time
casdoor	SAML	Yesterday

## Add an Identity provider Info

### Configure provider

Provider type Info

**SAML**  
Establish trust between your AWS account and a SAML 2.0 compatible Identity Provider such as Shibboleth or Active Directory Federation Services.

**OpenID Connect**  
Establish trust between your AWS account and Identity Provider services, such as Google or Salesforce.

Provider name  
Enter a meaningful name to identify this provider  
  
Maximum 128 characters. Use alphanumeric or '-' characters.

Metadata document Info

3  
File needs to be a valid UTF-8 XML document.

## Create an AWS Client VPN Endpoint

1. Open the Amazon VPC console in an AWS Region of your choice.
2. On the left-hand side navigation, select Client VPN Endpoints under Virtual Private Network (VPN).
3. Click Create Client VPN Endpoint.
4. Enter the IP range for your remote users in the Client IPv4 CIDR field to allocate an IP range.
5. For Server Certificate ARN, select the certificate you created.
6. For Authentication Options, select Use user-based authentication, then Federated authentication.

7. For SAML provider ARN, select the identity provider you created.

8. Click Create Client VPN Endpoint.

The screenshot shows the AWS Client VPN endpoint creation process across two pages. The first page (steps 2-3) lists existing endpoints and allows creating a new one. The second page (steps 4-8) configures the new endpoint.

**Client VPN endpoints (1/1)**

- Client IPv4 CIDR**: 172.31.32.0/20
- Authentication information**
  - Server certificate ARN**: arn:aws:acm:ap-southeast-1:580652580210:certificate/f028f870-16ee-41b7-8...
  - Authentication options**:
    - Use mutual authentication
    - Use user-based authentication
  - User-based authentication options**:
    - Active directory authentication
    - Federated authentication
- SAML provider ARN**: arn:aws:iam::580652580210:saml-provider/casdoor
- Self-service SAML provider ARN - optional**: Select self-service SAML provider ARN

# Associate a Client VPN with a Target VPC

1. Select Target network associations in the Client VPN options, then click Associate target network.
2. From the drop-down menu, select the target VPC and subnet you want to associate your endpoint with.

The screenshot shows the AWS CloudFormation console interface. On the left, there's a navigation sidebar with sections like Virtual private network (VPN), Customer gateways, Virtual private gateways, Site-to-Site VPN connections, Client VPN endpoints (which is selected and highlighted in blue), Transit gateways, Traffic Mirroring, and others. The main area displays the 'Client VPN endpoints' page for a single endpoint named 'cvpn-endpoint-06e947f15ddf5687c'. The 'Target network associations' tab is selected, and a red box highlights the 'Associate target network' button. Below it, a table lists one association: 'cvpn-assoc-0bf639762212d5a04' (Associated), 'subnet-0596ebfd975cdd125', 'sg-09d2a80e3c2795429', and 'cvpn-endpoint-06e947f15d'. Other tabs include Details, Security groups, Authorization rules, Route table, Connections, and Tags.

# Configure SAML Group-Specific Authorization

1. Choose the Authorization rules tab in your Client VPN options and click Add Authorize rule.
2. For Destination network to enable, specify the IP address of your EC2 instance created in the prerequisites. For example, `172.31.16.0/20`.
3. Under Grant access to, select Allow access to users in a specific access group. For example, `casdoor`.
4. Provide an optional description and click Add authorization rule.

## Add authorization rule Info

Add authorization rules to grant clients access to the networks.

Details	
Client VPN endpoint ID	<input type="checkbox"/> cvpn-endpoint-06e947f15ddf5687c
Destination network to enable access	The IP address, in CIDR notation, of the destination network. <input type="text" value="172.31.16.0/20"/> 2 <input type="button" value="X"/>
Grant access to:	<input type="radio"/> Allow access to all users <input checked="" type="radio"/> Allow access to users in a specific access group
Access group ID	Unique group identifier. It can be active directory SID or group name in IDP. <input type="text" value="casdoor"/> 3
Description - optional	A brief description of the authorization rule. <input type="text" value="description"/> 4
<input type="button" value="Cancel"/> <input type="button" value="Add authorization rule"/>	

# Connect to Client VPN

1. Select the Client VPN endpoint you just created. It should now be in the Available state.
2. Click Download Client Configuration to download the configuration profile to your desktop.
3. Open the AWS Client VPN desktop app on your machine.
4. In the top menu, select File and Manage Profiles.
5. Click Add Profile and point to the recently downloaded file.

6. You should now see the profile in the list on the AWS Client VPN software.

Select it and click Connect.

The screenshot shows the AWS VPC console interface. On the left, there's a sidebar with navigation links like 'VPC dashboard', 'EC2 Global View', 'Filter by VPC', 'Virtual private cloud' (with 'Your VPCs' and 'Subnets' options), 'Endpoint services', 'NAT gateways', 'Peerings connections', and 'Security'. The main area is titled 'Client VPN endpoints (1/1)' and shows a single endpoint named 'cvpn-endpoint-06e947f15ddf5687c'. The endpoint is listed as 'Available' with a CIDR range of '172.31.32.0/20'. A red box highlights the 'Download client configuration' button. Below the list is a detailed view of the endpoint, with tabs for 'Details', 'Target network associations', 'Security groups', 'Authorization rules', 'Route table', 'Connections', and 'Tags'. The 'Details' tab is selected, showing fields like Client VPN endpoint ID ('cvpn-endpoint-06e947f15ddf5687c'), Server certificate ARN ('arn:aws:acm:ap-southeast-1:580652580210:certificate/f028f870-16ee-41b7-8b4e-66a2a0ebfe33'), Connection log ('false'), Transport protocol ('udp'), and Cloudwatch log group ('-').

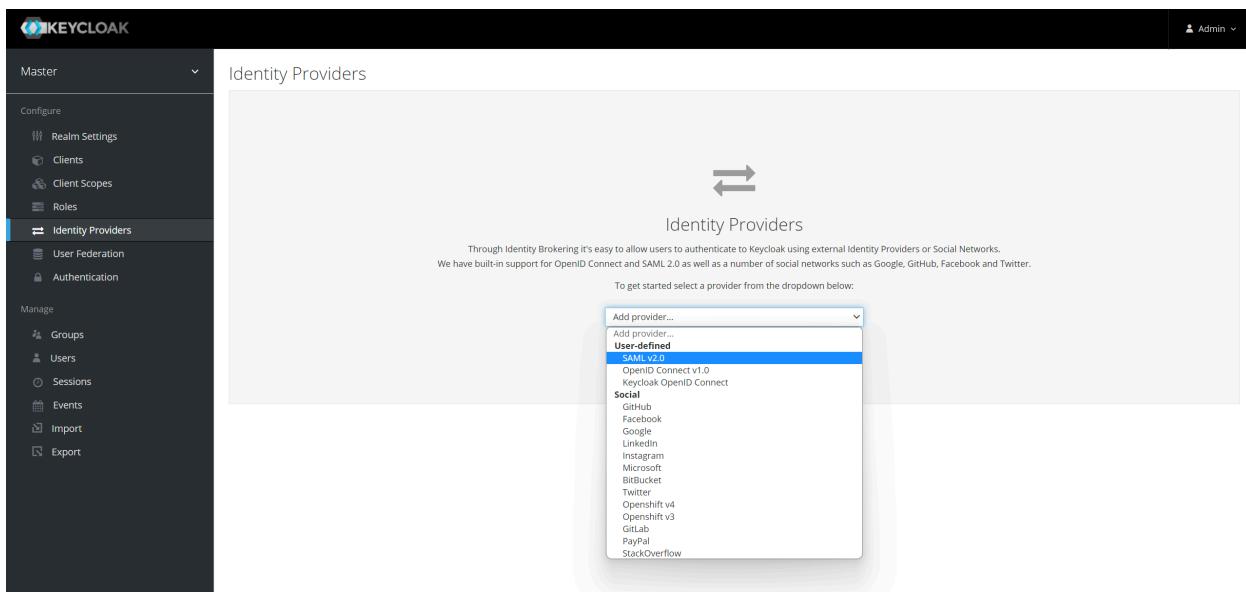
# Keycloak

## Casdoor as a SAML IdP in Keycloak

This guide will show you how to configure Casdoor and Keycloak to add Casdoor as a SAML IdP in Keycloak.

### Adding SAML IdP in Keycloak

Open the Keycloak admin page, click on Identity Providers, and select SAML v2.0 from the list of providers.



#### ⓘ 信息

You can visit the Keycloak SAML Identity Providers [documentation](#) to get more detailed information.

Enter the Alias and the Import from URL in the Keycloak IdP edit page. The content of the Import from URL can be found on the Casdoor application edit page. Click Import and the SAML config will be filled automatically.

The screenshot shows the 'Import External IDP Config' section of the Keycloak IdP edit page. It includes fields for 'Import from URL' (containing 'http://localhost:7001/api/saml/metadata?application=admin/app-built-in'), 'Import' (a blue button), 'Import from file' (with a 'Select file' button), and 'Save' (a blue button) and 'Cancel' (a grey button) buttons at the bottom.

Remember the Service Provider Entity ID and save the configuration.

## Configuring the SAML application in Casdoor

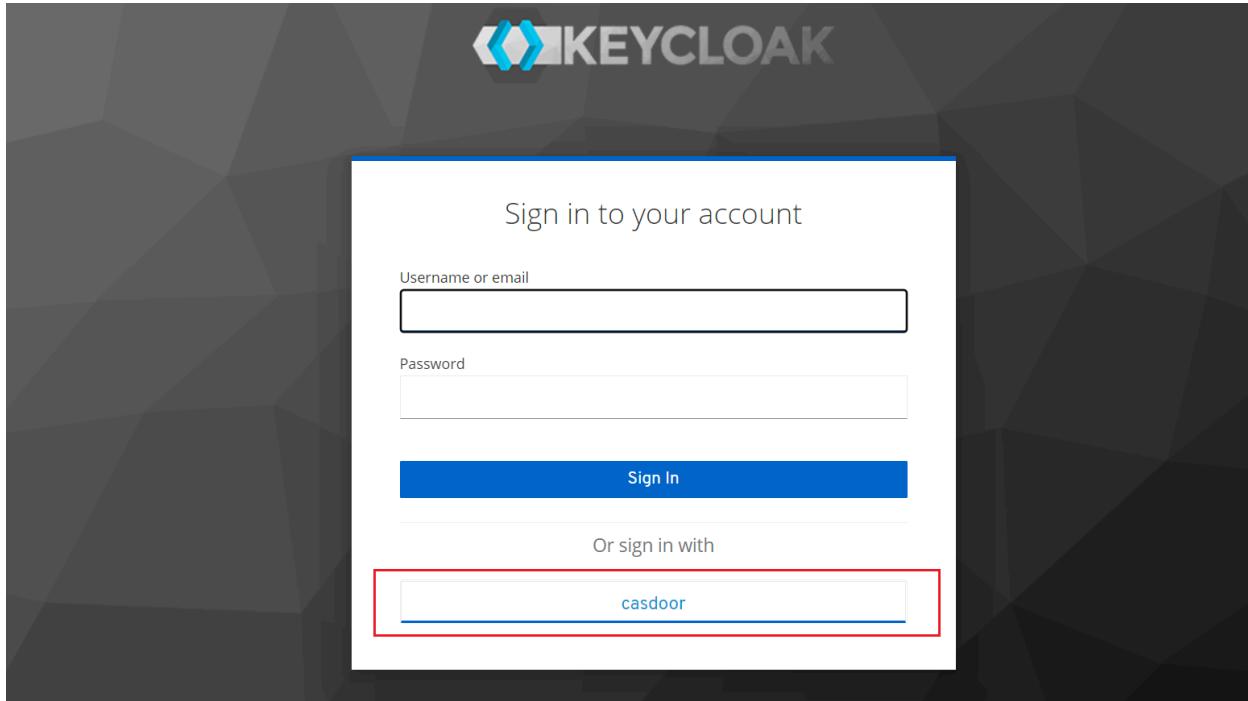
In the application edit page, add a redirect URL which contains the Service Provider Entity ID from Keycloak. Also, make sure to enable SAML compress for Keycloak.

The screenshot shows the Casdoor application edit page with the SAML metadata configuration. It includes a 'Enable SAML compress' toggle switch (which is checked), a large text area containing XML SAML metadata, and a 'Copy SAML metadata URL' button at the bottom.

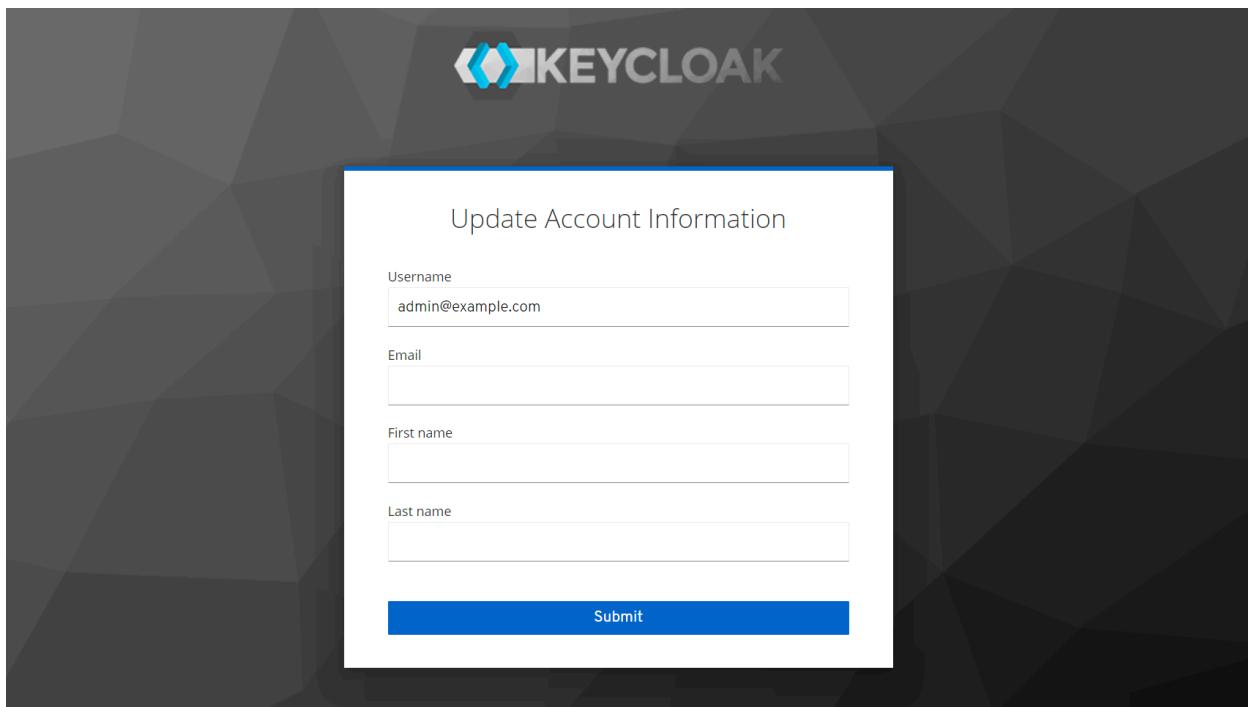
```
<EntityDescriptor xmlns="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns="urn:oasis:names:tc:SAML:2.0:metadata" entityID="http://localhost:8000">
<IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
<KeyDescriptor use="signing">
<KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
<X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
<X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TCIAuGgAwIBAgIDAeJAMAOGCSqGSIb3DQEBCwUAMDVshTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDVNNkb29yIENlcnQwHhcNMjExMDE1MDgxM
</X509Data>
</KeyInfo>
</KeyDescriptor>
<NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat>
<NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat>
<NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
<SingleSignOnService Bindings="urn:oasis:names:tc:SAML:2.0-bindings:HTTP-Redirect" Location="http://localhost:7001/login/saml/authorize/admin/app-built-in"></SingleSignOnService>
```

## Logging in using Casdoor SAML

Open the Keycloak login page and you will find an additional button that allows you to log in to Keycloak using the Casdoor SAML provider.



Click on the button and you will be redirected to the Casdoor SAML provider for authentication. After successful authentication, you will be redirected back to Keycloak. Then you need to assign users to the application.



We also provide a demo video that demonstrates the entire process, which we hope will be helpful to you.

# Google Workspace

## Casdoor as a SAML IdP in Google Workspace

This guide will show you how to configure Casdoor and Google Workspace to add Casdoor as a SAML IdP in Google Workspace.

### Add Certificate

In Casdoor, add a certificate of type X.509 with RSA crypto algorithm and download it.

Edit Cert Save Save & Exit

Organization : admin (Shared)

Name : cert-built-in

Display name : Built-in Cert

Scope : JWT

Type : x509

Crypto algorithm : RS256

Bit size : 4096

Expire in years : 20

Certificate : Copy certificate Download certificate

Private key : Copy private key Download private key

-----BEGIN CERTIFICATE-----  
MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgN  
VBAoTFENh

-----BEGIN PRIVATE KEY-----  
MIJKQIBAAKCAgEAslnpb5E1ym0fRfSDSSE8R7y+lw+RJj74e5ejrq4b8zM  
Y

### Configure SAML Application

On the application edit page, select the certificate you just created. Add the

domain name of the Google application you will use in the Redirect URLs, such as google.com.

Cert ⓘ :	cert-built-in	▼
Redirect URLs ⓘ :	Redirect URLs <a href="#">Add</a>	Action <a href="#">↑</a> <a href="#">↓</a> <a href="#">trash</a>

Redirect URL

<a href="http://google.com">🔗 google.com</a>	<a href="#">↑</a> <a href="#">↓</a> <a href="#">trash</a>
<a href="http://gmail.com">🔗 gmail.com</a>	<a href="#">↑</a> <a href="#">↓</a> <a href="#">trash</a>

In the SAML reply URL field, enter `https://www.google.com/a/<your domain>/acs`, which is the ACS URL. You can find relevant information about ACS URL here: [SSO assertion requirements](#).

SAML reply URL	<a href="https://www.google.com/a/casbin.com/acs">https://www.google.com/a/casbin.com/acs</a>
Enable SAML compression	<input checked="" type="checkbox"/>
SAML metadata	<pre>&lt;EntityDescriptor xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"&gt;     &lt;IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"&gt;         &lt;KeyDescriptor use="signing"&gt;             &lt;KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#"&gt;                 &lt;X509Data xmlns="http://www.w3.org/2000/09/xmldsig#"&gt;                     &lt;X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#"&gt;MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTabBgNVBAoTFENhc2Rvb3IgT3JnYWYwDQYJKoZIhvcNAQELBQADggEAMHk...&lt;/X509Certificate&gt;                 &lt;/X509Data&gt;             &lt;/KeyInfo&gt;         &lt;/KeyDescriptor&gt;     &lt;NameIDFormat&gt;urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress&lt;/NameIDFormat&gt;     &lt;NameIDFormat&gt;urn:oasis:names:tc:SAML:2.0:nameid-format:persistent&lt;/NameIDFormat&gt;     &lt;NameIDFormat&gt;urn:oasis:names:tc:SAML:2.0:nameid-format:transient&lt;/NameIDFormat&gt;     &lt;SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="http://localhost:7001/login/saml/authorize/admin/wor...&lt;/SingleSignOnService&gt; &lt;/IDPSSODescriptor&gt; &lt;/EntityDescriptor&gt;</pre>
	<a href="#">Copy SAML metadata URL</a>

Copy the sign-in page URL. This will be used in the next step.

Providers [?](#)

Name	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action

No data

Preview [?](#)

## Add Third-Party SAML IdP for Google Workspace

In the Google Workspace Admin console, navigate to **Security** and then **Overview**. Look for the **SSO with third-party IdP** section. Click on "Add SSO profile" to access the editing page. Check the "Set up SSO with third-party identity provider" checkbox. Paste the copied sign-in page URL into the **Sign-in page URL** and **Sign-out page URL** fields. Upload the certificate downloaded in the previous step. Click "Save" to save the changes.

The screenshot shows the Google Workspace Admin console interface. On the left, there's a navigation sidebar with various options like Home, Dashboard, Directory, Devices, Apps, Security, Overview (which is selected), Alert centre, Authentication, 2-step verification, Account recovery, Advanced Protection Programme, Login challenges, Passwordless (BETA), and Password management. The main content area has a header "Search for users, groups or settings" and a breadcrumb path "Security > SSO with third-party IDPs > Third-party SSO profile for your organisation". The main content is titled "Single Sign-On (SSO) with third-party Identity Providers (IDPs)". It includes a section for "Third-party identity provider" with a checked checkbox for "Set up SSO with third-party identity provider". Below this, there are fields for "Sign-in page URL" containing "https://localhost/login/oauth/authorize?client\_id=12" and "Sign-out page URL" containing "https://localhost/login/oauth/authorize?client\_id=12". A "Verification certificate" section shows a message "A certificate file has been uploaded" and a "REPLACE CERTIFICATE" button. A note at the bottom says "The certificate file must contain the public key for Google to verify sign-in requests." with a "Learn more" link.

## Add Users for Testing

In Google Workspace, create a user with the username "test" (you can customize the username, this is just an example).

Your new user can start using Google Workspace within 24 hours. In most cases, it should just take a few minutes.



test test

Username: test@casbin.com

[COPY PASSWORD](#) [PRINT](#)

## Send sign-in instructions

The email will provide a link to set the password and sign in to Google Workspace

## PREVIEW AND SEND



The user will be assigned licences based on your current subscriptions. [View billing](#)

## ADD ANOTHER USER

DONE

In Casdoor, add a user with the same username as set in Google Workspace. Make sure to select the appropriate organization and enter the user's email address.

Organization [?](#) : built-in

ID [?](#) : 4899cef3-8eeb-485a-8f6d-12b41df0d8d2

Name [?](#) : test

Display name [?](#) : test

Avatar [?](#) :

Preview:



Upload a photo...

User type [?](#) : normal-user

Password [?](#) : [Modify password...](#)

Email [?](#) : test@casbin.com

Phone [?](#) : +1 34086653696

As an example using "google.com," follow these steps:

1. Click on the login button on the Google.com page. Enter the user's email address to initiate the login process.
2. You will be redirected to the Casdoor page. On the Casdoor page, enter the corresponding email address and password.
3. If the login is successful, you will be redirected back to google.com.

Gmail Images ☰ Sign in



Google Search I'm Feeling Lucky

Google offered in: 日本語

Japan

About Advertising Business How Search works

Privacy Terms Settings

# Appgate (POST)

## Casdoor as a SAML IdP in Appgate

Appgate accepts the `SAMLResponse` sent by a POST request. If you use another Service Provider (SP) that also supports a POST request, you can refer to this document.

### Casdoor Configuration

Go to your Casdoor account and add a new application.

Enter basic SAML configuration in the application:

- Redirect URLs – Type in a unique name. This may be called `Audience` or `Entity ID` in your SP. See the table below.

Redirect URLs [?](#) :

Redirect URLs	Add
Redirect URL	
<a href="#">appgate</a>	
<a href="https://git.casbin.com/user/oauth2/casdoor/callback">https://git.casbin.com/user/oauth2/casdoor/callback</a>	
<a href="http://localhost:3000/callback">http://localhost:3000/callback</a>	

- Reply URL – Type in the URL of the ACS (Assertion Consumer Service) that verifies the SAML response. Refer to the table below.

Grant types [?](#) :  Authorization Code  Password

SAML Reply URL [?](#) : <https://mycontroller.mycompany.com/admin/saml>

Enable SAML compress [?](#) :

Administrator Authentication	User Authentication
Redirect URL = "AppGate"	Redirect URL = "AppGate Client"
SAML Reply URL = <a href="https://mycontroller.your-site-url.com/admin/saml">https://mycontroller.your-site-url.com/admin/saml</a>	SAML Reply URL = <a href="https://redirectserver.your-site-url.com/saml">https://redirectserver.your-site-url.com/saml</a>

## Download the XML metadata file

You can copy the URL of the metadata and download the file from your browser.

Enable SAML compress [?](#) :

SAML metadata [?](#) :

```
<KeyDescriptor use="signing">
<KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#>
<X509Data xmlns="http://www.w3.org/2000/09/xmldsig#>
<X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE+TCCAuGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxSYXnkba...</X509Data>
</X509Data>
</KeyInfo>
</KeyDescriptor>
<NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat>
<NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat>
<NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
<SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="https://door.casdoor.com/login/saml/authorize/admin/app-gitea"><SingleSignOnService>
<Attribute Name="Email" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" FriendlyName="E-Mail" xmlns="urn:oasis:names:tc:SAML:2.0:assertion"></Attribute>
<Attribute Name="DisplayName" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:basic" FriendlyName="displayName" xmlns="urn:oasis:names:tc:SAML:2.0:assertion"></Attribute>
```

# Add SAML IdP in Appgate

In your AppGate SDP console:

- Select System > Identity Providers.
- Create a new Identity Provider.
- Choose the type SAML.
- Start configuring your identity provider following the details in the tables below.

Administrator Authentication	
Name	Enter a unique name, e.g. "Casdoor SAML Admin".
Single Sign-on URL	See below
Issuer	See below
Audience	Type in the Redirect URL from the Casdoor application
Public Certificate	See below

- Upload the XML Metadata file to autocomplete the Single Sign-On, Issuer, and Public Certificate fields.
- Click Choose a file and select the metadata file that you previously downloaded - this will autocomplete the relevant fields.

## Map Attributes

Map the Name to username. Your completed form should look something like this:

The screenshot shows a user interface titled "Map Attributes to User Claims". At the top right is a blue button labeled "Add New" with a plus sign icon. Below the title, there is a single row in a table-like structure. The first column contains the text "Name mapped to claim username". The second column is empty. The entire row is highlighted with a light gray background.

## Test Integration

On your AppGate SDP Controller console:

- Log out of the admin UI.
- Log in using the following information:
  - Identity Provider – choose your Azure IdP from the drop-down list.
  - Click **Sign in with browser** to connect to your authenticator.
- You may see the following message: "You don't have any administration rights" – this confirms that the test user credentials have been successfully authenticated by your Identity Provider.

## Access Policy

You need to modify the access policy to allow administrators to log in to Appgate using the SAML IdP. Enter the Builtin Administrator Policy:

Your completed form should look something like this:

## Editing Policy - Admin

- Enabled  
 Disabled

Assignment - Active when custom logic is met ▾

 Add New

Custom Logic (1 OR 3) AND 2

- 1 Identity Provider is local
- 2 user.username is admin
- 3 Identity Provider is Casdoor SAML Admin



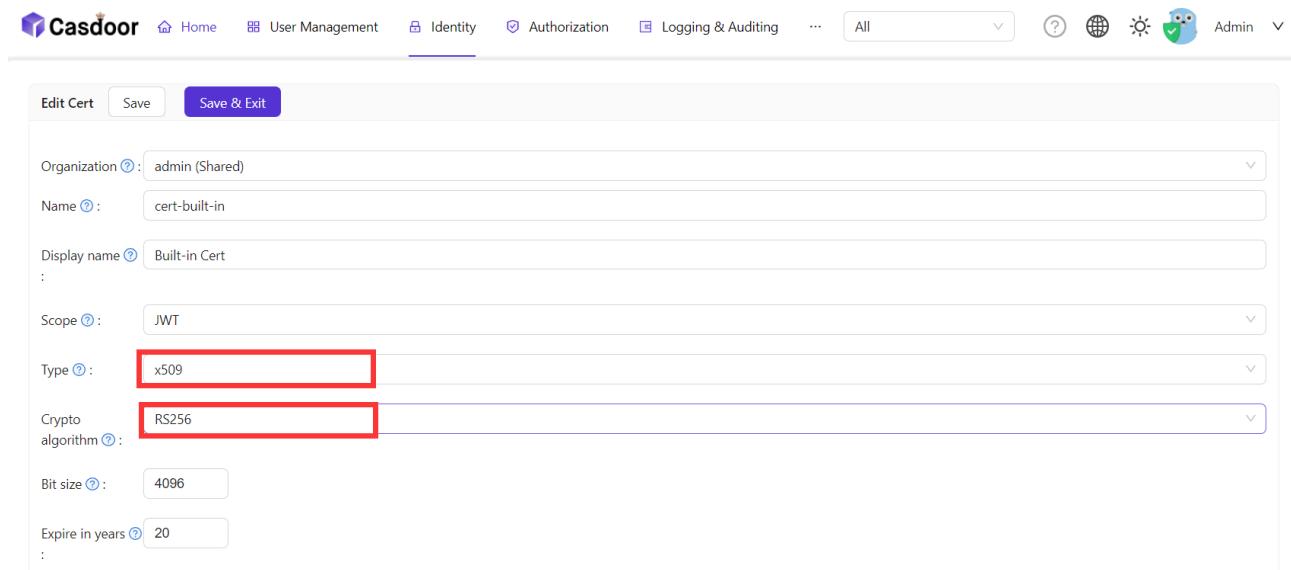
# Tencent Cloud

## Casdoor as a SAML IdP in Tencent Cloud

This guide will show you how to configure Casdoor and Tencent Cloud to add Casdoor as a SAML IdP in Tencent Cloud.

### Copy Saml MetaData

In Casdoor, add a certificate of type X.509 with RSA crypto algorithm.



The screenshot shows the 'Edit Cert' page in Casdoor. The 'Type' field is set to 'x509' and the 'Crypto algorithm' field is set to 'RS256'. Both fields are highlighted with red boxes. The rest of the fields (Organization, Name, Display name, Scope, Bit size, Expire in years) have their default values.

Then copy the SamlMetadata in Casdoor.



The screenshot shows the 'Edit Cert' page with the SamlMetadata code highlighted by a red box. The code is a XML snippet for an EntityDescriptor, including sections for KeyDescriptor, NameIDFormat, and SingleSignOnService.

```
<EntityDescriptor xmlns="http://www.w3.org/2000/09/xmldsig#> <urn: oasis:names:tc:SAML:2.0:metadata> <urn: oasis:names:tc:SAML:2.0:metadata entityID="http://localhost:7001/login/saml/authorize/admin/application_tencent_cloud">
    <IDPSSODescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
        <KeyDescriptor use="signing">
            <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#>
                <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#>MIIE+TCCAUggAwIBAgIDAeJAMAOCGSeqGSib3DQEBCwUAMDYxHTAbBzNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDVXNkb25...
            </X509Data>
        </KeyInfo>
    </KeyDescriptor>
    <NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</NameIDFormat>
    <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</NameIDFormat>
    <NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-format:transient</NameIDFormat>
    <SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect" Location="http://localhost:7001/login/saml/authorize/admin/application_tencent_cloud"></SingleS...
```

[Copy SAML metadata URL](#)

Providers: Providers Add

### Adding SAML IdP in Tencent Cloud

Log in to Tencent Cloud and enter the access management interface.

The screenshot shows the Tencent Cloud Account Center interface. On the left sidebar, under '账号中心' (Account Center), the '访问管理' (Access Management) option is highlighted with a red box. The main content area displays account information for a '微信用户' (WeChat User). It includes fields for '账号昵称' (Account Nickname), '认证状态' (Authentication Status), '所属行业' (Industry), and '注册时间' (Registration Time). Below this is a section titled '登录方式' (Login Methods) which lists various authentication methods: WeChat (支持微信扫码授权登录), QQ (支持QQ授权登录), 企业微信 (支持企微扫码授权登录), 邮箱 (支持账号密码登录), and 微信公众号 (支持小程序、公众号授权登录). A note at the top of this section states: '登录方式为登录腾讯云账号的途径, 请确认您所绑定的登录方式的使用主体与当前腾讯云账号实名主体的相关性。'

Create a new Identity Providers and upload the previously copied saml metadata to Tencent Cloud.

The screenshot shows the '访问管理' (Access Management) section of the Tencent Cloud console. On the left sidebar, the '角色SSO' (Role SSO) option is highlighted with a red box. The main page title is '角色SSO'. A blue button labeled '新建提供商' (Create Provider) is visible. The page contains a table with columns: '提供商名称' (Provider Name), '提供商类型' (Provider Type), '创建时间' (Creation Time), '最后更新时间' (Last Update Time), and '操作' (Operations). The table currently shows '暂无数据' (No data available). At the bottom right, there are pagination controls: '10 条 / 页' (10 items per page), page numbers (1, 2, 3, 4, 5), and navigation icons.

Then Create a new ROLE and select the previously Identity Providers as idp provider.

## Configuring the SAML application in Casdoor

On the application edit page, select the certificate you just created. Add the domain name of the Tencent Cloud application you will use in the Redirect URLs.

In the application edit page, enter the ACS URL and configure the Saml Attribute.

The configuration information for Saml Attribute is as follows:

Name	Name Format	Value
<code>https://cloud.tencent.com/SAML/Attributes/Role</code>	Unspecified	<code>qcs::cam::uin/{AccountID}:roleName/{RoleName1};qcs::cam::uin/{AccountID}:roleName/{RoleName2},qcs::cam::uin/provider/{ProviderName}</code>
<code>https://cloud.tencent.com/SAML/Attributes/RoleSessionName</code>	Unspecified	<code>casdoor</code>

### ① 信息

- In the Role source attribute, replace {AccountID}, {RoleName}, and {ProviderName} with the following content:
- Replace {AccountID} with your Tencent Cloud account ID, which can be viewed in the [Account Information - Console](#).
- Replace {RoleName} with the role name you created in Tencent Cloud, which can be viewed in the [Roles - Console](#).
- Replace {ProviderName} with the name of the SAML identity provider you created in Tencent Cloud, which can be viewed in the [Identity Providers - Console](#).

You can visit the Tencent Cloud SAML Identity Providers [documentation](#) to get more detailed information.

## Logging in using Casdoor SAML

The general login steps for SAML are as follows: User → Tencent Cloud (not logged in) → Redirect to Casdoor for login → Tencent Cloud (logged in). Now, use code externally to simulate the first two steps and generate a URL that redirects to Casdoor. Sample code:

```
func main() {
    res, err := http.Get("your casdoor application saml metadata url")
    if err != nil {
        panic(err)
    }

    rawMetadata, err := ioutil.ReadAll(res.Body)
    if err != nil {
        panic(err)
    }

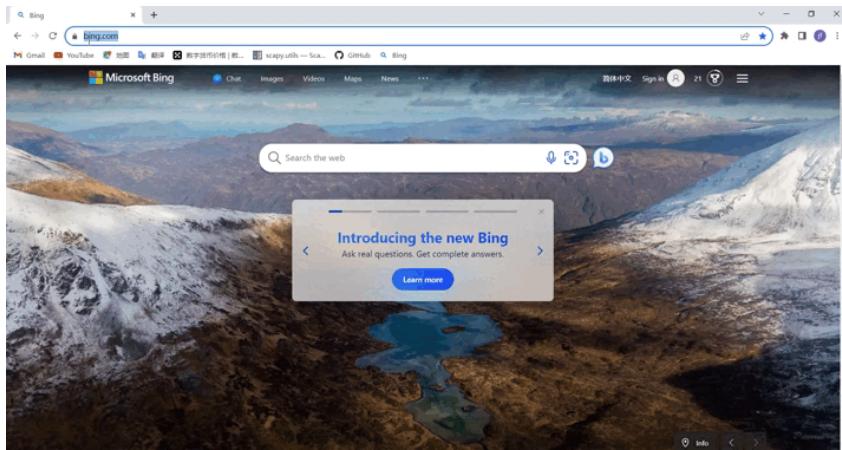
    metadata := &types.EntityDescriptor{}
    err = xml.Unmarshal(rawMetadata, metadata)
    if err != nil {
        panic(err)
    }

    certStore := dsig.MemoryX509CertificateStore{
        Roots: []*x509.Certificate{},
    }

    for _, kd := range metadata.IDPSSODescriptor.KeyDescriptors {
        for idx, xcert := range kd.KeyInfo.X509Data.X509Certificates {
            if xcert.Data == "" {
                panic(fmt.Errorf("metadata certificate(%d) must not be empty", idx))
            }
            certData, err := base64.StdEncoding.DecodeString(xcert.Data)
            if err != nil {
                panic(err)
            }

            idpCert, err := x509.ParseCertificate(certData)
            if err != nil {
                panic(err)
            }
        }
    }
}
```

Once we run the code and obtain the auth URL, clicking on the URL will allow us to test the login. we provide a demo this process.



# WebAuthn

## 概述

We are delighted to inform Casdoor's customers that Casdoor now supports logging in with WebAuthn. This means that you can log in using your biological identifications such as fingerprints or facial recognition, or even U-disks, provided that your device supports these cool authorization methods and WebAuthn.

## 什么是 WebAuthn?

WebAuthn is the Web Authentication API, a specification written by the W3C and FIDO in collaboration with Google, Mozilla, Microsoft, Yubico, and others. This API allows servers to register and authenticate users using public key cryptography instead of a password. It enables servers to integrate with strong authenticators built into devices, such as Windows Hello or Apple's Touch ID.

To put it simply, WebAuthn requires users to generate a public key-private key pair and provide the public key to the website. When a user wants to log in to a website, the web generates a random number and asks the user to encrypt it with their private key and send the result back. Upon receiving the result, the website uses the public key to decrypt it. If the decrypted number matches the random number generated earlier, the user is considered a legitimate user and is granted access to log in. The combination of the public key and necessary information, like the username or information about the user's authorizer, is called the WebAuthn Credential, which is stored by the website.

The public key-private key pair is exclusively and uniquely associated with three pieces of information: the user's username, the user's authorizer, and the

website's URL. This means that if the combination of (user's username, user's authorizer, and the website's URL) is the same, the key pair should be identical, and vice versa.

For more detailed information about WebAuthn technology, you can visit <https://webauthn.guide/>.

## How to use WebAuthn in Casdoor?

On the login page, you may have already noticed the option to log in using WebAuthn. However, if you don't have a WebAuthn credential yet (which can be likened to a WebAuth password), this tutorial will show you how to create and manage a credential and then log in using it.

### Step 0: Modify the configurations and enable WebAuthn authentication

In the `conf/app.conf` file, you can find the following configuration:

```
origin = "http://localhost:8000"
```

Please ensure that this configuration exactly matches the URL of your website.

**Note:** Only HTTPS is supported for WebAuthn, unless you are using localhost.

Next, log in as the administrator and go to the edit page of your application. Turn on the "Enable WebAuthn signin" switch. 默认情况下，此功能未启用。

### Step 1: Go to "My Account" page

Navigate to the account page. On this page, you should see the "Add WebAuthn Credential" button and a list displaying all the WebAuthn credentials you have previously registered.

The screenshot shows the Casdoor application configuration interface. At the top, there's a field for "Signup application" with a placeholder "(empty)". Below it, a section for "3rd-party logins" shows a GitHub icon with the text "GitHub: (empty)" and a blue "Link" button. Under "WebAuthn credentials", there's a table with one row. The row has columns for "WebAuthn credentials" (containing "Add" and "WebAuthn credentials"), "Action" (containing a red "Delete" button), and the credential value "OAUzbyDy1SCxNyW3vkNJP1feXhwm/pHBDmMOszRRNvg=". Below the table, sections for "Roles" and "Permissions" are shown, each with several toggle switches. At the bottom are two buttons: "Save" and "Save & Exit".

Click the button and follow the instructions of your device to register a new credential in Casdoor. You can remove any credentials using the "delete" button in the list.

## Step 2: Log in using WebAuthn

Before starting this step, make sure you have logged out of Casdoor.

Go to the login page, select the WebAuthn login method, enter your username, and click the login button. Follow the instructions of your device.

(For example, if you are using fingerprint and Windows Hello, you should see something like this)

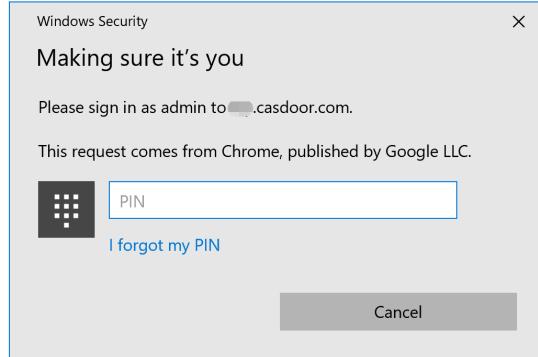


Windows Security

>Password [WebAuthn](#)

Auto sign in [Forgot password?](#)

[Sign in with WebAuthn](#)



You will then be logged in successfully.



&gt;

开发指南

# 开发指南

## 前端

Casdoor Frontend Development Guide

## 生成 Swagger 文件

生成 Swagger 文件

# 前端

The source code for Casdoor's frontend is located inside the `/web` folder:  
<https://github.com/casdoor/casdoor/tree/master/web>

It is a [Create-React-App \(CRA\)](#) project, which follows the classic CRA folder structure as outlined below:

文件/目录	描述
public	The root HTML file for React
src	Source code
craco.config.js	The Craco configuration file. You can change the theme color (blue by default) here
crowdin.yml	Crowdin i18n configuration file
package.json	NPM/Yarn 依赖文件
yarn.lock	Yarn lockfile

Inside the `/src` directory, you will find several important files and folders:

文件/目录	描述
account	The "My profile" page for logged-in users

文件/目录	描述
auth	All code related to authentication, such as OAuth, SAML, sign up page, sign in page, forget password page, etc.
backend	The SDK for calling the Go backend API. It contains all the <code>fetch()</code> calls
basic	The homepage (dashboard page) for Casdoor, which contains several card widgets
common	Shared UI widgets
locales	i18n translation files in JSON, synced with our Crowdin project: <a href="https://crowdin.com/project/casdoor-site">https://crowdin.com/project/casdoor-site</a>
App.js	The entry JS file containing all the routes
Setting.js	Utility functions used by other code
OrganizationListPage.js	The page for the organization list, similar to all other <code>XXXListPage.js</code> files
OrganizationEditPage.js	The page for editing one organization, similar to all other <code>XXXEditPage.js</code> files

# 生成 Swagger 文件

## 概述

As we know, the beego framework provides support for generating swagger files to clarify the API via the command line tool called "bee". Casdoor is also built based on beego. However, we found that the swagger files generated by bee failed to categorize the APIs with the "@Tag" label. So, we modified the original bee to implement this function.

## 如何写comment

Most rules are exactly identical to the original bee comment formats. The only discrepancy is that the API shall be divided into different groups according to the "@Tag" label. Therefore, developers are obliged to ensure that this tag is correctly added. Here is an example:

```
// @Title Login
// @Tag Login API
// @Description login
// @Param oAuthParams     query    string  true      "oAuth
parameters"
// @Param body    body    RequestForm  true      "Login
information"
// @Success 200 {object} controllers.api_controller.Response The
Response object
// @router /login [post]
func (c *ApiController) Login() {
```

APIs with the same "@Tag" labels will be put into the same group.

## How to generate the swagger file

0. Write comments for the API in the correct format.
1. Fetch this repository: <https://github.com/casbin/bee>.
2. Build the modified bee. For example, in the root directory of casbin/bee, run the following command:

```
go build -o mybee .
```

3. Copy mybee to the base directory of casdoor.
4. In that directory, run the following command:

```
mybee generate docs
```

5. (Optional) If you want to generate swagger document for specific tags or apis, here are some example commands:

```
mybee generate docs --tags "Adapter API"  
mybee generate docs --tags "Adapter API,Login API"  
mybee generate docs --apis "add-adapter"  
mybee generate docs --apis "add-adapter,delete-adapter"
```

Notably: We only accept a comma  as the separator when multiple tags/apis provided.

Then you will find that the new swagger files are generated.





&gt;

组织

# 组织

## 概述

Casdoor的基本单位 — 组织

## 组织树

组织内的用户群组(group)

## Password Complexity

Supporting different password complexity options.

## 账户自定义

自定义用户帐户项目



## Customizing Themes

Learn how to customize themes for organizations and applications within an organization



## Manage Multi-Factor Authentication Items

Configure Multi-Factor Authentication Items in Organization

# 概述

组织是Casdoor的基本单位，它管理着用户和应用。如果一个用户登录到一个组织，他可以在无需再次登录的情况下访问属于该组织的所有应用。

在[应用](#)和[提供商](#)配置中，选择一个组织是很重要的一项，它决定了一个用户是否能够使用特定的提供商访问该应用。

我们还可以在Casdoor建立LDAP。更多详细信息，请参阅[LDAP 文档](#)。

Casdoor提供了多种密码存储算法，可在组织编辑页面中选择。

名称	算法	描述	场景
plain	-	密码将以明文形式存储。(默认)	-
salt	SHA-256	SHA-256 是一个获得专利的加密哈希函数，输出256位长的值。	-
md5-salt	MD5	MD5 message-digest algorithm是一种加密中断但仍广泛使用的哈希函数，可产生128位哈希值。	Discuz!
bcrypt	bcrypt	bcrypt 是一个密码哈希函数，用于安全地对密码进行哈希和加密。	Spring Boot, WordPress
pbkdf2-salt	SHA256	PBKDF2 是一个简单的派生函数，能够	Keycloak

名称	算法	描述	场景
	与 PBKDF2	抵制字典攻击和彩虹表攻击. 它是 Cassdoor 为 Keycloak 同步器而采用的原生实现。如果您通过 Keycloak 同步器导入用户，请选择此选项。	

### 💡 提示

除了通过应用程序登录到 Casdoor 之外(重定向到Casdoor以便SSO)， Casdoor 用户也可以选择通过该组织的登录页面直接登录到 Casdoor:

`/login/<organization_name>`，例如示例站点中的  
<https://door.casdoor.com/login/casbin>

# 组织树

群组是组织内用户的一个集合 一个用户可以属于多个群组

## Group properties

- **Owner**: 群组所属的组织
- **Name**: 群组的唯一名称
- ``
- ``
- ``
- **Type**: 群组可以被区分为 **Physical** 或 **Virtual** 一个用户只能存在在一个 **Physical** 组中, 但可以存在在多个 **Virtual** 组中.
- **ParentGroup**: 群组的父级群组.(最顶级群组的父组是其组织)

## 管理群组

有两种方式管理群组

1. 在组织列表页, 你可以预览所有组织下的群组

Name	Organization	Created time	Updated time	Display name	Type	Parent group	Action
casdoor_virtual	built-in	2023-06-12 12:37:44	2023-06-12 12:37:51	Casdoor Project Virtual Team	Virtual		<button>Edit</button> <button>Delete</button>
casbin_virtual	built-in	2023-06-12 12:37:18	2023-06-12 12:37:36	Casbin Project Virtual Team	Virtual		<button>Edit</button> <button>Delete</button>
dev_frontend	built-in	2023-06-12 09:43:18	2023-06-12 12:35:51	Dev (Frontend)	Physical		<button>Edit</button> <button>Delete</button>
dev_backend	built-in	2023-06-12 09:20:28	2023-06-12 12:35:58	Dev (Backend)	Physical		<button>Edit</button> <button>Delete</button>
dev	built-in	2023-06-09 18:19:06	2023-06-12 12:36:08	R & D	Physical		<button>Edit</button> <button>Delete</button>
sales	built-in	2023-06-09 01:27:19	2023-06-12 12:36:27	Sales	Physical		<button>Edit</button> <button>Delete</button>
marketing	built-in	2023-06-09 01:26:16	2023-06-12 12:36:32	Marketing	Physical		<button>Edit</button> <button>Delete</button>
hr	built-in	2023-06-09 01:25:46	2023-06-12 12:36:43	HR	Physical		<button>Edit</button> <button>Delete</button>
sales_and_marketing	built-in	2023-06-09 01:23:35	2023-06-12 12:36:57	Sales & Marketing	Physical		<button>Edit</button> <button>Delete</button>

9 in total < 1 > 10 / page

## 2. 在组织的列表页中单击 群组 按钮

Name	Created time	Display name	Favicon	Website URL	Password type	Password salt	Action
saas	2023-05-31 00:05:42	SaaS Users		<a href="https://saas.casbin.com">https://saas.casbin.com</a>	plain		<button>Edit</button> <button>Delete</button>
gsoc	2021-02-11 23:26:20	GSoC Community		<a href="https://gsoc.com.cn">https://gsoc.com.cn</a>	plain		<button>Edit</button> <button>Delete</button>
casbin	2021-02-11 23:26:20	Casbin Organization		<a href="https://forum.casbin.com">https://forum.casbin.com</a>	plain		<button>Edit</button> <button>Delete</button>
built-in	2021-02-10 00:37:06	Built-in Organization		<a href="https://door.casdoor.com">https://door.casdoor.com</a>	plain		<button>Edit</button> <button>Delete</button>

4 in total < 1 > 10 / page

这将展示组织下群组的树形结构

The screenshot shows the Casdoor interface for managing organizations. On the left, there is a sidebar with a tree view of groups:

- Root node: Casdoor Project Virtual Team
- Child node: Casbin Project Virtual Team
- Child node: R & D
  - Child node: Dev (Frontend)
  - Child node: Dev (Backend)
- Child node: HR
- Child node: Sales & Marketing
  - Child node: Sales
  - Child node: Marketing

To the right of the tree view is a table listing users:

Organization	Application	Name	Created time	Display name
built-in	app-built-in	牛头	2023-06-16 16:16:35	牛头
built-in	app-built-in	喝咖啡就大蒜	2023-06-15 10:58:48	喝咖啡就大蒜
built-in	app-built-in	danceshow	2023-06-15 10:53:48	街舞show
built-in	app-built-in	TT珍惜	2023-06-15 10:36:46	TT珍惜
built-in	app-built-in	hashjoin	2023-06-15 01:01:17	hashjoin
built-in	app-built-in	zhangyaphet@gmail.com	2023-06-14 15:50:23	pengwei zhang

这里有一个视频展示了如何管理群组

The screenshot shows the Casdoor interface for managing groups. The table lists the following groups:

Name	Organization	Created time	Updated time	Display name	Type	Parent group	Action
group_smf8bi	built-in	2023-06-13 23:25:31	2023-06-13 23:25:36	总部	Virtual	built-in	<button>Edit</button> <button>Delete</button>
group_zxak7d	built-in	2023-06-11 23:28:45	2023-06-13 23:24:36	New Group - zxak7d	Virtual	New Group - nahuap	<button>Edit</button> <button>Delete</button>
group_1t8lrr	built-in	2023-06-09 09:39:44	2023-06-13 23:24:46	美工	Virtual	研发子部门	<button>Edit</button> <button>Delete</button>
group_nahuap	built-in	2023-06-09 09:27:47	2023-06-12 09:36:45	New Group - nahuap	Virtual		<button>Edit</button> <button>Delete</button>
group_38ii7o	built-in	2023-06-07 21:48:49	2023-06-11 09:49:13	研发子部门	Virtual		<button>Edit</button> <button>Delete</button>
group_gnrtip9	built-in	2023-06-07 20:59:10	2023-06-13 23:24:51	实体组3	Physical	built-in	<button>Edit</button> <button>Delete</button>
group_5aca0x	forum	2023-06-06 08:24:33	2023-06-13 23:25:12	实体组2	Physical	forum	<button>Edit</button> <button>Delete</button>
group_3tt9wf	forum	2023-06-06 08:20:14	2023-06-13 23:25:06	顶级2	Virtual	forum	<button>Edit</button> <button>Delete</button>
group_azpdif	forum	2023-06-06 08:19:32	2023-06-09 10:50:25	顶级1	Virtual		<button>Edit</button> <button>Delete</button>
group_r89hga	built-in	2023-06-05 14:41:41	2023-06-12 09:38:28	研发部1	Virtual		<button>Edit</button> <button>Delete</button>

群组也可以在用户的资料页被编辑。

Title <a href="#">?</a> :	1122
Homepage <a href="#">?</a> :	
Bio <a href="#">?</a> :	
Tag <a href="#">?</a> :	222
Karma <a href="#">?</a> :	333
Signup application <a href="#">?</a> :	app-built-in
Groups <a href="#">?</a> :	 Dev (Frontend)   Casdoor Project Virtual Team 
Roles <a href="#">?</a> :	
Permissions <a href="#">?</a> :	

# Password Complexity

Casdoor supports customizing password complexity options for user passwords in each organization.

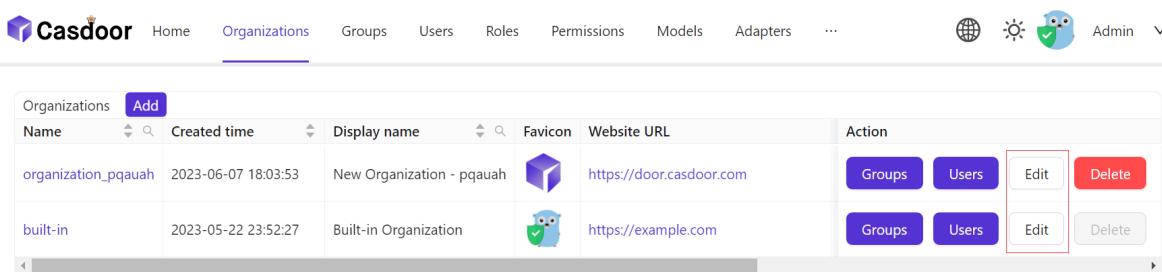
## Supported Complexity Options

We currently support five options:

- `AtLeast6`: The password must have at least six characters.
- `AtLeast8`: The password must have at least eight characters.
- `Aa123`: The password must contain at least one uppercase letter, one lowercase letter, and one digit.
- `SpecialChar`: The password must contain at least one special character.
- `NoRepeat`: The password must not contain any repeated characters.

If you want to use multiple options, you can select them on the organization edit page:

1. Click the Edit button on the organization list page.



Name	Created time	Display name	Favicon	Website URL	Action
organization_pquaah	2023-06-07 18:03:53	New Organization - pquaah		<a href="https://door.casdoor.com">https://door.casdoor.com</a>	<a href="#">Groups</a> <a href="#">Users</a> <a href="#">Edit</a> <a href="#">Delete</a>
built-in	2023-05-22 23:52:27	Built-in Organization		<a href="https://example.com">https://example.com</a>	<a href="#">Groups</a> <a href="#">Users</a> <a href="#">Edit</a> <a href="#">Delete</a>

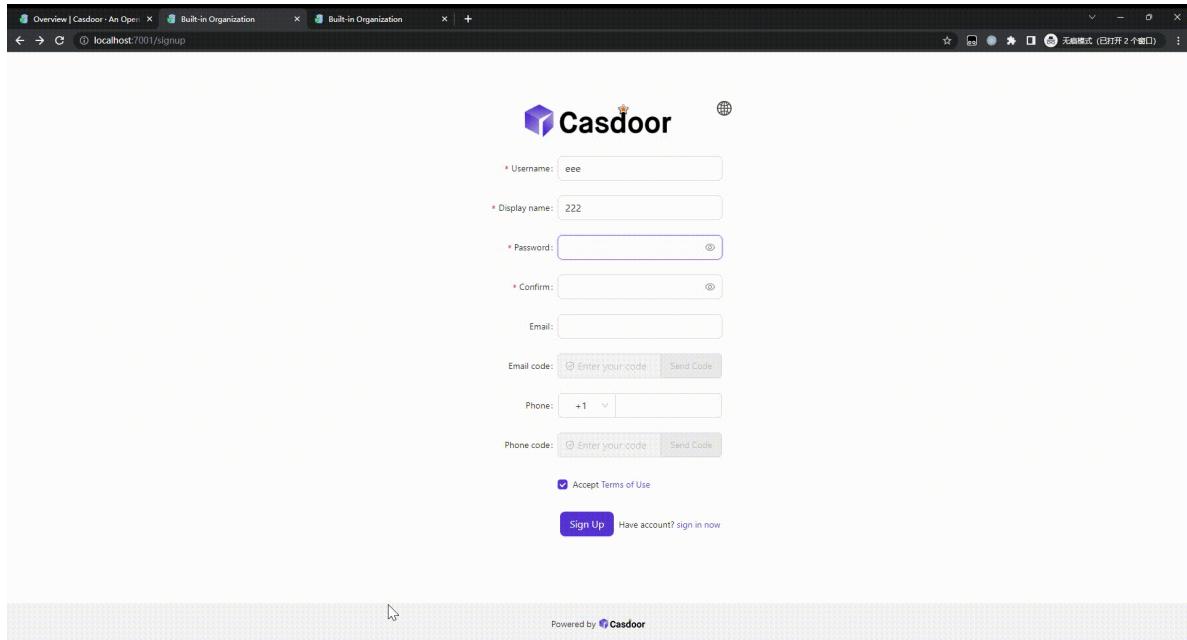
2. Then select the option you need in the **Password complexity options** column.

The screenshot shows the Casdoor web interface for managing organizations. The top navigation bar includes links for Home, Organizations, Groups, Users, Roles, Permissions, Models, Adapters, Applications, Providers, and Admin. The 'Organizations' tab is active. Below the navigation is a form titled 'Edit Organization' with fields for Name (built-in), Display name (Built-in Organization), Favicon (https://cdn.casbin.org/img/casbin/favicon.ico), URL (https://example.com), Preview (an owl icon), Password type (plain), Password salt, and Password complexity options. The 'Password complexity options' section contains four rules: 'The password must have at least 8 characters', 'The password must contain at least one special character', 'The password must not contain any repeated characters', and 'The password must contain at least one uppercase letter, one lowercase letter and one digit'. A dropdown menu for Supported country codes lists Germany +49, United Kingdom +44, India +91, and another entry with a delete icon. A dropdown menu for Languages lists German, English, and Indian.

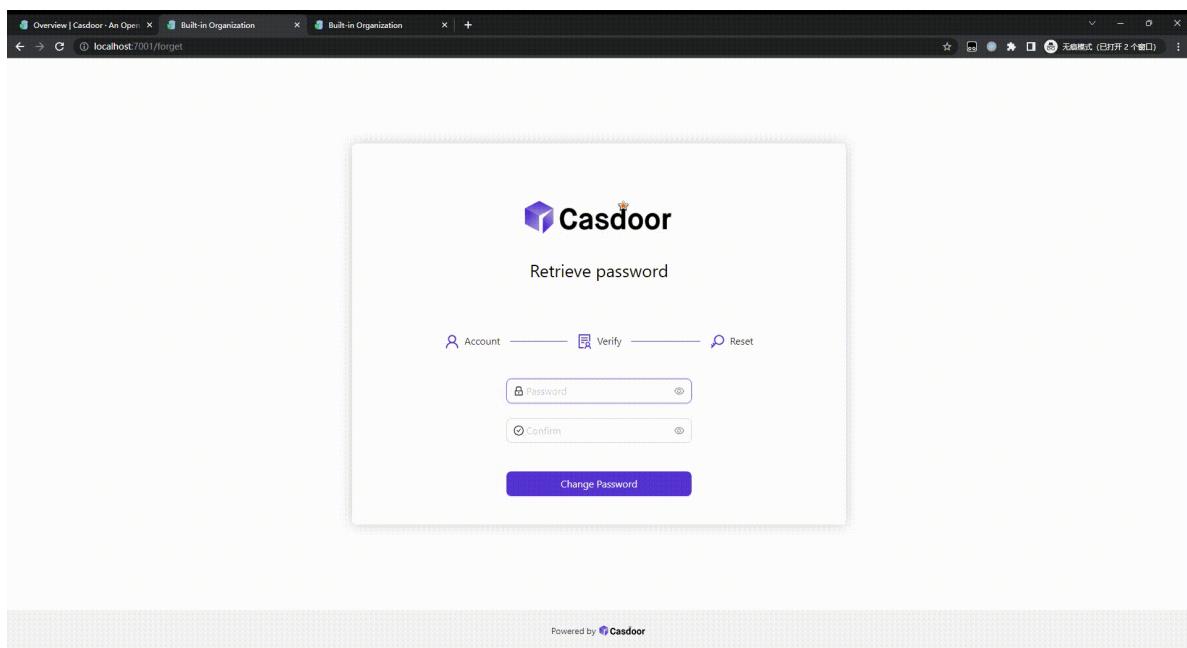
# Password Complexity Validation

We support password complexity validation on the following pages:

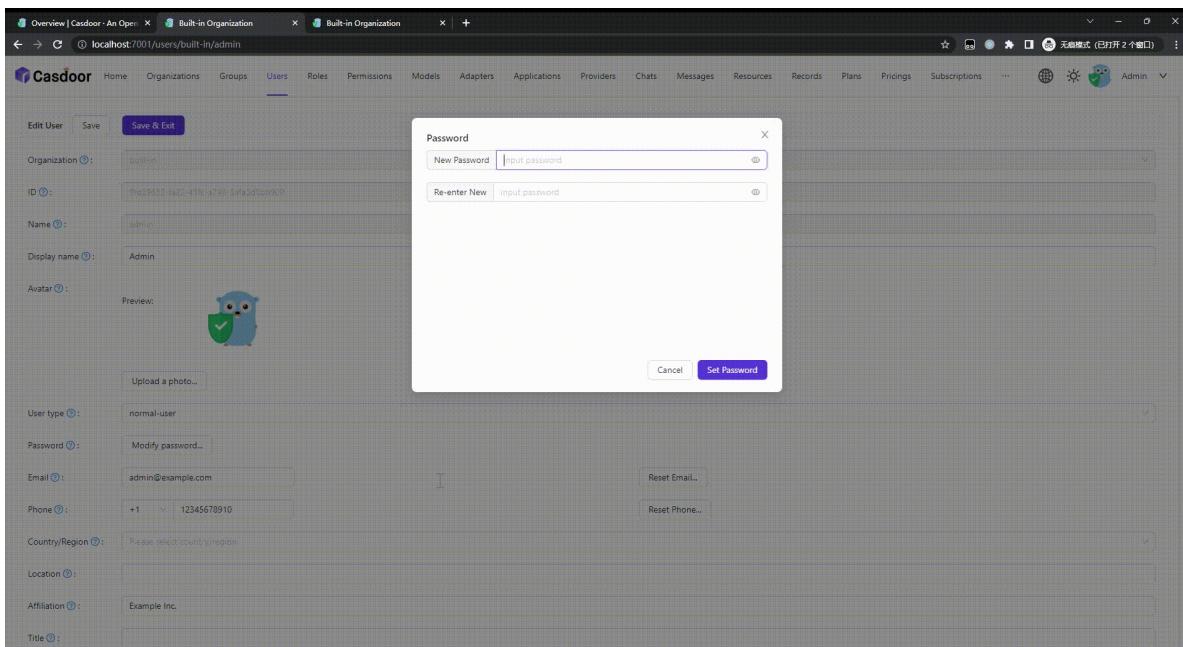
1. Sign up page.



## 2. Forget password page.



## 3. User edit page.



# 账户自定义

## 介绍

在组织中，您可以自定义用户的 账户项。这包括每个项目是否是 可见 及其 视图规则 和 修改规则。

当您自定义一个组织中的账户项目时，此配置将在该组织所有成员的主页上生效。

## 如何定制？

账户项目有四个属性：

属性	可选值	描述
Name	-	账户名称。
Visible	<input checked="" type="checkbox"/> 是 / <input type="checkbox"/> 否	选择此账户项是否在用户主页上可见。
ViewRule	规则项	选择用于查看帐户项目的规则。
ModifyRule	规则项	选择用于修改帐户项目的规则。

要自定义账户项目，请遵循以下步骤：

1. 转到组织编辑页面。

## 2. 您将找到以下选项:

Account items <small>(?)</small>		visible	viewRule	modifyRule	Action
Name	Add				
Organization	<input checked="" type="checkbox"/>	Public	Admin		
ID	<input checked="" type="checkbox"/>	Public	Immutable		
Name	<input checked="" type="checkbox"/>	Public	Admin		
Display name	<input checked="" type="checkbox"/>	Public	Self		
Avatar	<input checked="" type="checkbox"/>	Public	Self		
User type	<input checked="" type="checkbox"/>	Public	Admin		
Password	<input checked="" type="checkbox"/>	Self	Self		
Email	<input checked="" type="checkbox"/>	Public	Self		
Phone	<input checked="" type="checkbox"/>	Public	Self		
Country/Région	<input checked="" type="checkbox"/>	Public	Self		
Location	<input checked="" type="checkbox"/>	Public	Self		
Affiliation	<input checked="" type="checkbox"/>	Public	Self		
Title	<input checked="" type="checkbox"/>	Public	Self		
Homepage	<input checked="" type="checkbox"/>	Public	Self		
Bio	<input checked="" type="checkbox"/>	Public	Self		
Tag	<input checked="" type="checkbox"/>	Public	Admin		
Signup application	<input checked="" type="checkbox"/>	Public	Admin		
3rd-party logins	<input checked="" type="checkbox"/>	Self	Self		

## 3. Casdoor 提供简单的操作来配置帐户项目:

- 将项目设置为可见或不可见.

Account items		Add	visible	viewRule	modifyRule	Action
Name						
Organization	<input checked="" type="checkbox"/>	Public	Admin			
ID	<input type="checkbox"/>					
Name	<input checked="" type="checkbox"/>	Public	Admin			
Display name	<input checked="" type="checkbox"/>	Public	Self			
Avatar	<input checked="" type="checkbox"/>	Public	Self			
User type	<input checked="" type="checkbox"/>	Public	Admin			

- 设置查看和修改规则.

visible	viewRule	modifyRule	Action
<input checked="" type="checkbox"/>	Public	Admin	<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>
<input type="checkbox"/>	Public		<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>
<input checked="" type="checkbox"/>	Self	Admin	<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>
<input checked="" type="checkbox"/>	Admin	Self	<input type="button" value="^"/> <input type="button" value="v"/> <input type="button" value="Delete"/>

有 3 条规则可用：

- Public: 所有人都有权限。
- Self: 用户有自己的权限。
- Admin: 管理员拥有权限。

## 账户列表

以下是帐户项目中的所有字段。 关于描述详情，您可以查看 [用户](#)。

- 组织
- ID
- 姓名
- 显示名称
- 头像
- 用户类型
- 密码
- 邮件
- 手机号
- 国家/地区
- 城市

- 附属机构
- 职务
- 个人主页
- 自我介绍
- 标签
- 注册应用
- 第三方登录
- 属性
- 是否为管理员
- 是否为全局管理员
- 是否被禁用
- 是否已删除

# Customizing Themes

Casdoor allows you to customize themes to meet the UI diversity requirements of businesses or brands, including primary color and border radius.

Within Casdoor, themes can be customized at the global, organization, and application levels.

1. Global scope: This is the default theme of Casdoor and is applied to any organization that chooses to follow the global theme. Modifications can only be made in the Casdoor source code and cannot be modified in the web UI.
2. Organization scope: The theme for an organization can be customized on the organization edit page. This theme applies to all Casdoor after-login pages for users within the organization, as well as the entry pages (signup, signin, forget password, etc.) of applications that follow the organization theme.
3. Application scope: The theme for an application can be customized on the application edit page. This theme applies to the entry pages (signup, signin, forget password, etc.) of the specific application.

## Customizing the Organization Theme

We provide a demo to demonstrate how to configure the theme for an organization:

The screenshot shows the Casdoor application's configuration interface. On the left, there is a sidebar with a tree view of configuration sections: Roles, Permissions, 3rd-party logins, Properties, Is admin, Is global admin, Is forbidden, Is deleted, WebAuthn credentials, and Managed accounts. To the right of the sidebar, there are two main panels. The top panel displays a grid of permissions for various roles. The bottom panel shows the LDAP configuration section, listing a single server named 'BuildIn LDAP Server' with the URL 'example.com:389' and base DN 'ou=BuildIn,dc=example,dc=com'. There are buttons for 'Save' and 'Save & Exit' at the bottom.

### ① 信息

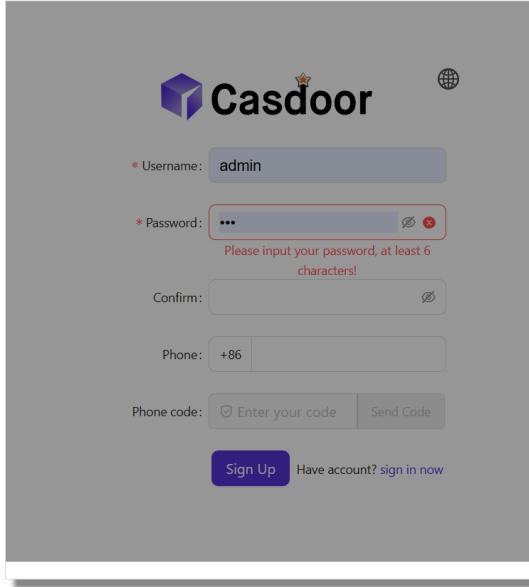
If your account organization is the same as the organization you are editing, the configuration changes will take effect immediately as shown in the video above. However, if they are different, you will need to log in to the organization to see the changes.

## Customizing the Application Theme

Applications can customize themes using the same theme editor as the organization. Additionally, you can preview the theme conveniently in the preview panel.

Preview ⓘ

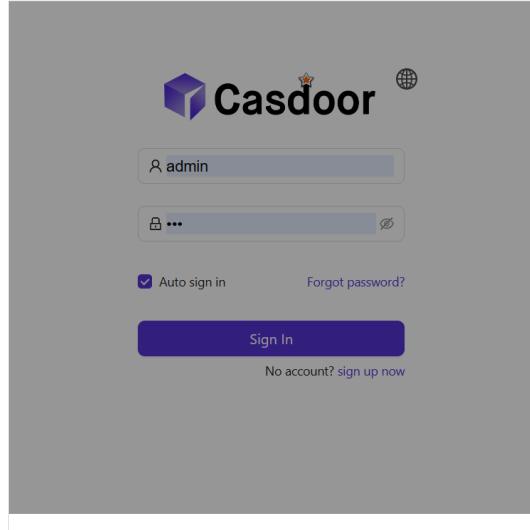
 Copy signup page URL



The screenshot shows the Casdoor Signup page. At the top is the Casdoor logo. Below it is a form with the following fields:

- \* Username: admin
- \* Password:  (The field is highlighted with a red border.)  
Please input your password, at least 6 characters!
- Confirm:
- Phone: +86
- Phone code:  Enter your code
- Have account? sign in now

 Copy signin page URL



The screenshot shows the Casdoor Signin page. At the top is the Casdoor logo. Below it is a form with the following fields:

- admin
- 

Below the form are two links: "Auto sign in" (with a checked checkbox) and "Forgot password?". At the bottom is a large blue "Sign In" button and a link "No account? sign up now".

Background URL

# Manage Multi-Factor Authentication Items

## Add Multi-Factor Authentication Item in Organization

In the organization, admins can add Multi-Factor Authentication items to the account settings. This allows users to configure Multi-Factor Authentication on their own profile pages.

The screenshot shows the Casdoor web interface for managing organization settings. At the top, there are fields for 'Master password' (with placeholder '\*\*\*'), 'Languages' (English, 中文, Español, Français, Deutsch, Indonesia, 日本語, 한국어, Русский, Tiếng Việt), 'Init score' (0), 'Soft deletion' (disabled), and 'Is profile public' (disabled). Below these are sections for 'Account items' and 'Profile items'. The 'Account items' section has a table with columns: Name, Visible, View rule, Modify rule, and Action. A new row for 'Multi-factor authentication' is being added, indicated by a red border around its row. The table rows are as follows:

Name	Visible	View rule	Modify rule	Action
Organization	Visible	Public	Admin	[Edit]
ID	Visible	Public	Immutable	[Edit]
Name	Visible	Public	Admin	[Edit]
Display name	Visible	Public	Self	[Edit]
Avatar	Visible	Public	Self	[Edit]
User type	Visible	Public	Admin	[Edit]
Password	Visible	Self	Self	[Edit]
Multi-factor authentication	Visible	Self	Self	[Edit]
Email	Visible	Public	Self	[Edit]
Phone	Visible	Public	Self	[Edit]

# Manage Multi-Factor Authentication Items

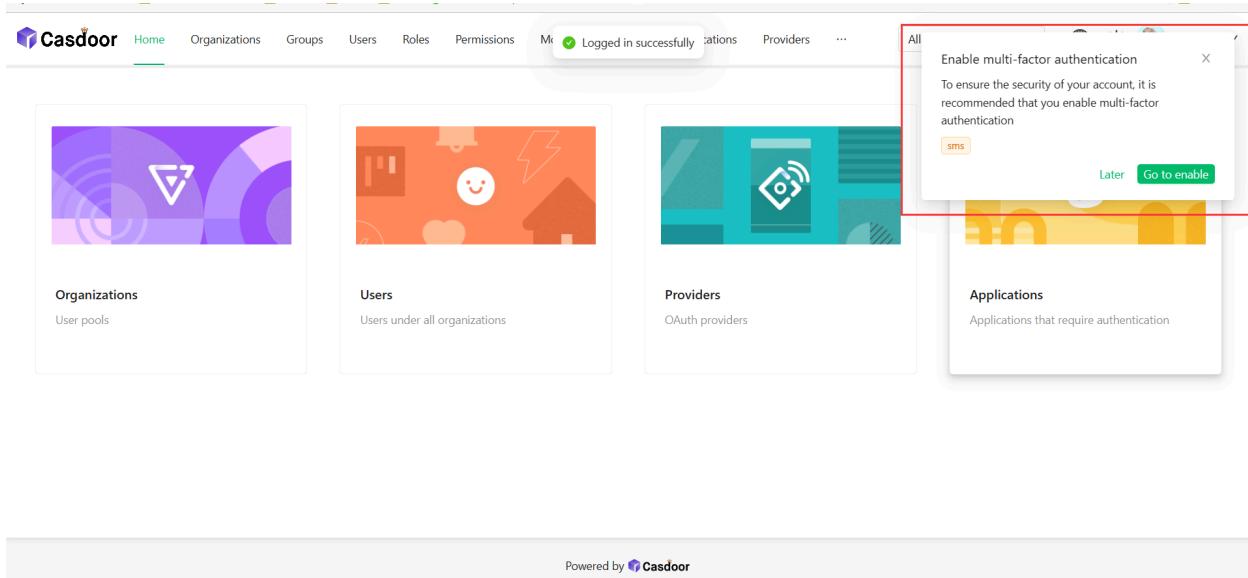
You can manage Multi-Factor Authentication to determine which methods are available to users.

There are two rules for managing Multi-Factor Authentication items:

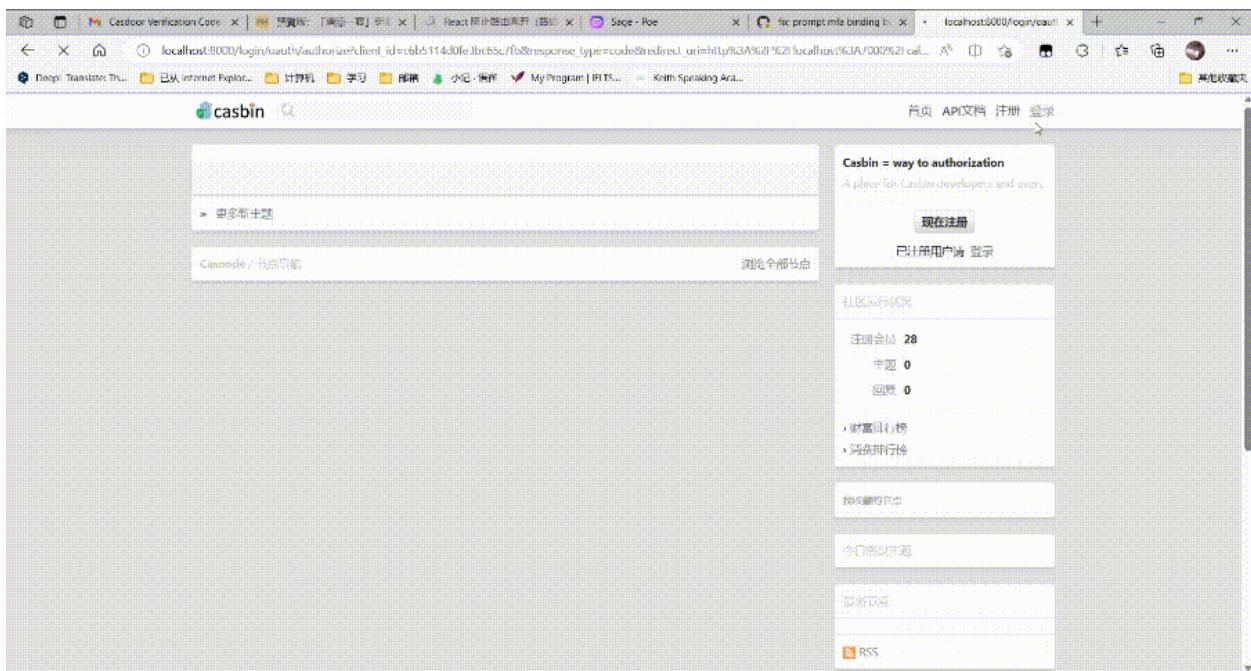
- Optional: Users can choose whether to enable this type of Multi-Factor Authentication.
- Prompt: If the user does not enable this Multi-Factor Authentication mode, they will be prompted to enable it after logging in to Casdoor.
- Required: Users must enable this Multi-Factor Authentication method.

MFA items		Add	
Name	Rule	Action	
Phone	Prompt	  	
Email	Optional	  	
App	Required	  	

The image below shows the notification that prompts users to enable Multi-Factor Authentication.



This video demonstrates that when the Multi-Factor Authentication method is set to required, users need to enable Multi-Factor Authentication before they can complete the login process.





&gt;

应用

# 应用

## 概览

Casdoor应用概述

## Terminology Reference

术语参考

## 应用程序配置

配置应用程序的身份验证

## Signin Methods

Configure the login method and the display order of the login methods

## Signup Items Table

Configure the signup items table to create a custom registration page

## 自定义登录界面

自定义您的应用程序登录页面

## Specify Login Organization

Specify the login organization on the login page

## Tags

Configure your application tags

## 应用邀请码

使用邀请码限制用户通过应用注册

# 概览

Casdoor 中的每个应用都被称为“application”。它们之间没有关联，不会相互影响，这意味着只要你愿意，你可以单独部署或停止任何应用程序。

如果您想要使用Casdoor为您的网站提供登录服务，您可以将其添加为Casdoor应用。

这样用户在访问组织中的所有应用程序时就无需重复登录。

应用程序配置非常灵活和简单。您可以设置是否允许密码登录或第三方登录，配置您想要用户登录的第三方应用程序。您甚至可以自定义应用程序的注册项等。

在本章中，您将学习如何从头开始启动您自己的应用程序。

让我们一起探索吧！

# Terminology Reference

- `Name`: The name of the created app.
- `CreatedTime`: The time when the application is created.
- `DisplayName`: The name which the application displays to the public.
- `Logo`: Application logos will be displayed on the login and sign up pages.
- `HomepageUrl`: The URL of the application's homepage.
- `Description`: Describes the application.
- `Tags`: Only users with tags listed in the application tags can login.
- `Organization`: The organization that the app belongs to.
- `EnableSignUp`: If users can sign up. If not, accounts of the application.
- `SigninMethods`: Configuration of Sign-in Methods
- `SignupItems`: Fields that need to be filled in when users register.
- `Providers`: Provide all kinds of services for the applications (such as OAuth, Email, SMS service).
- `ClientId`: OAuth client ID.
- `ClientSecret`: OAuth client secret.
- `RedirectUris`: Casdoor will navigate to one of the URIs if the user logged in successfully.
- `TokenFormat`: The format of the generated token. It can be in the following formats: `JWT` (containing all `User` fields), `JWT-Empty` (containing all non-empty values) or `JWT-Custom` customizing `User` fields inside access token.
- `ExpireInHours`: Login will expire after hours.
- `SigninUrl`:
- `SignupUrl`: If you provide a sign-up service independently outside of Casdoor, please fill in the URL here.

- `ForgotUrl`: Same as `SignupUrl`.
- `AffiliationUrl`:

# 应用程序配置

当你在你的服务器上部署你的casdoor并设置你的组织后，你可以现在就部署你的应用程序！

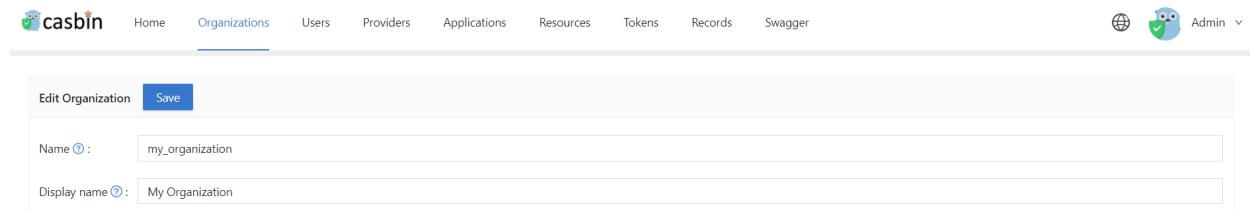
让我们看看如何使用Casdoor配置您的应用程序的身份验证！

## ① 备注

例如，我想要使用 Casnnode 设置我的论坛

我创建了我的应用程序并填写了必要的配置。

选择我创建的组织，以便组织中的用户也可以使用该应用程序。



The screenshot shows the Casdoor web interface. At the top, there is a navigation bar with links: Home, Organizations (which is underlined), Users, Providers, Applications, Resources, Tokens, Records, and Swagger. On the far right, there is a user profile icon labeled "Admin". Below the navigation bar, the main content area has a title "Edit Organization" and a "Save" button. There are two input fields: "Name" with the value "my\_organization" and "Display name" with the value "My Organization".

这个组织命名为 `my_organization`，我从下拉菜单中选择到它。

Edit Application Save

Name ? : my\_forum

Display name ? : My Forum

Logo ? : URL: [https://cdn.casbin.com/logo/logo\\_1024x256.png](https://cdn.casbin.com/logo/logo_1024x256.png)

Preview: 

Home ? :

Description ? :

Organization ? : built-in

Client ID ? : my\_organization  
built-in

接下来，我希望我的用户在注册时能够使用 Casdoor 作为认证。因此，我在这里填写重定向URL为 <https://your-site-url.com/callback>。

#### :::注意事项

请注意，在提供商应用程序中的 `callback URL` 应该是Casdoor的回调 URL，而 Casdoor 中的 `Redirect URL` 应该是您网站的回调 URL 。

#### 进一步了解

为了使认证过程发挥作用，详细步骤如下：

1. 用户将请求发送到Casdoor。

2. Casdoor 使用 `Client ID` (客户端ID) 和 `Client Secret` (客户端密钥) 获取 GitHub、谷歌或其他供应商的身份验证。
3. 如果认证成功, GitHub 会调用回到 Casdoor, 通知 Casdoor 已成功验证。因此, GitHub 认证回调 URL 应该是您的 Casdoor 的回调URL, 即 <http://your-casdoor-url.com/callback>
4. 接着, Casdoor 将认证成功通知应用程序。这意味着 Casdoor 回调 URL 应该是您的应用程序的回调 URL , 即 <http://your-site-url.com/callback>。

:::

你也可以通过添加提供商和设置其属性来添加第三方应用程序注册。

Name	canSignUp	canSignIn	canUnlink	prompted	Action
provider_casbin_email					▲ ▼ 🗑
provider_casbin_sms					▲ ▼ 🗑
provider_storage_aliyun_oss					▲ ▼ 🗑
provider_casdoor_github_localhost	toggle	toggle	toggle	toggle	▲ ▼ 🗑
provider_casdoor_github	toggle	toggle	toggle	toggle	▲ ▼ 🗑
provider_casdoor_google	toggle	toggle	toggle	toggle	▲ ▼ 🗑
provider_casdoor_qq	toggle	toggle	toggle	toggle	▲ ▼ 🗑
provider_casdoor_wechat	toggle	toggle	toggle	toggle	▲ ▼ 🗑
provider_casdoor_facebook	toggle	toggle	toggle	toggle	▲ ▼ 🗑
provider_casdoor_gitee	toggle	toggle	toggle	toggle	▲ ▼ 🗑
provider_casdoor_gitlab	toggle	toggle	toggle	toggle	▲ ▼ 🗑

You need to enable JavaScript to run this app.

 提示

If you want to do more personalized configuration of the application's sign-in methods, such as disabling a certain sign-in method or turning off a certain sign-in method, you can refer to the [Signin Methods](#)

# Signin Methods

On the Application Configuration page, we can configure the sign-in item table. We can add and remove sign-in items from the table.

Signin methods		Add			
Name	Display name	Rule	Action		
Password	Password	All			
Verification code	Email	All			
LDAP	LDAP				
WebAuthn	WebAuthn				

For a detailed explanation of each sign-in item, please refer to the table below. Currently, only Password, verification code, WebAuthn and LDAP login methods are available.

Column Name	Selectable Value	Description
Name	-	The name of the sign-in method.
DisplayName	*	The name which the sign-in method displays to the public.
Rule		Select a rule to customize this sign-in method. Detailed rules are described in the table below.
Action	-	Users can perform actions such as moving this sign-in method up, moving it down, or deleting

Column Name	Selectable Value	Description
	it.	

At present, configuration rules are only supported for the `Password` and `Verification code` sign-in methods.

Sign-in Method Name	Selectable Rules	Description
Password	<code>All(default)</code> / <code>Non-LDAP</code>	Select the sign-in methods available to the user. Choosing <code>All</code> , then LDAP users can also sign-in. Choosing <code>Non-LDAP</code> , then LDAP users are prohibited from sign-in.
Verification code	<code>All(default)</code> / <code>Email only</code> / <code>Phone only</code>	Select the sign-in methods available to the user. Choosing <code>All</code> , then both email and phone numbers can be verified for sign-in. Choosing <code>Email only</code> , then only email login is allowed. Choosing <code>Phone only</code> , then only the phone number is allowed to authenticate the login.

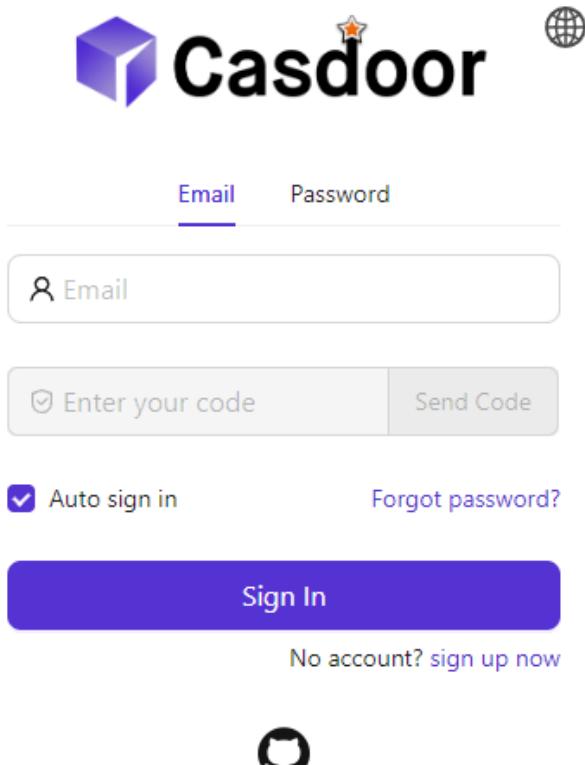
### ① 备注

For example, we want users to prioritize logging in with their email, and then consider logging in with a password if they can't use their email.

First, we configure two login options, `Verification Code` and `Password`, and `Verification Code` is the first login option. Then we change the `verification code` rule to `Email only`, so that the user can only receive the login verification code by email.

Signin methods		Add		
Name	Display name	Rule	Action	
Verification code	Email	Email only		
Password	Password			

To make it easier for users to understand, we can change the display name of the `Verification code` login method so that users can easily understand that it is an email login.



 提示

All login options, except for LDAP, are enabled by default. And it is required that at least one sign-in method be added.

Here is a video of how the sign-in method works:

# Signup Items Table

On the application configuration page, we can configure the signup items table to create a customized registration page. We can add or delete any signup item on this signup items table.

Signup items :

Name	Visible	Required	Prompted	Rule	Action
ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Random	
Username	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
Display name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		None	
Password	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
Confirm	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
Email	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Normal	
Phone	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
Agreement	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		None	

For a detailed explanation of each signup item, please refer to the table below.

Column Name	Selectable Value	Description
Name	-	The name of the signup item.
Visible	True / False	Select whether this signup item is visible on the registration page.
Required	True / False	Select whether this signup item is mandatory.
Prompted	True / False	Select whether to prompt the user when they forget to fill in this signup item.

Column Name	Selectable Value	Description
Rule	Rule Items	Select a rule to customize this signup item. Detailed rules are described in the table below.
Action	-	Users can perform actions such as moving this signup item up, moving it down, or deleting it.

Currently, the signup items that support configuration rules include ID, Display name, Email, and Agreement.

Item Name	Selectable Rules	Description
ID	Random / Incremental	Select whether the user ID should be randomly generated or incremented.
Display name	None / Real name / First, last	Choose how the display name should be presented. Choosing None will display Display name. Choosing Real name will display the user's actual name. Choosing First, last will display the first and last name separately.
Email	Normal / No verification	Select whether to verify the email address with a verification code. Choosing Normal will require email verification. Choosing No verification will allow signup without email

Item Name	Selectable Rules	Description
		verification.
Agreement	<input type="checkbox"/> None / <input type="checkbox"/> Signin / <input type="checkbox"/> Signin <input type="checkbox"/> (Default True)	Select whether the user needs to confirm the terms of use when logging in. Choosing <input type="checkbox"/> None will not display any terms of use, allowing users to log in directly. Choosing <input type="checkbox"/> Signin will require users to confirm the terms before logging in. Choosing <input type="checkbox"/> Signin (Default True) will set the terms as confirmed by default, allowing users to log in directly.

### ⓘ 备注

For example, let's say I want to set up my registration page to include an email field, but without requiring email verification.

Firstly, I added some signup items necessary for registration, such as ID, Username, Password, and Email.

Name	visible	required	prompted	rule	Action
ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Incremental	<input type="button"/> <input type="button"/> <input type="button"/>
Username	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="button"/> <input type="button"/> <input type="button"/>	<input type="button"/> <input type="button"/> <input type="button"/>
Password	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="button"/> <input type="button"/> <input type="button"/>	<input type="button"/> <input type="button"/> <input type="button"/>
Email	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	No verification	<input type="button"/> <input type="button"/> <input type="button"/>

Then, I selected the email row's rule item as  No verification. As a result, the generated preview registration page will have the desired effect.



\* Username:

\* Password:

\* Email:

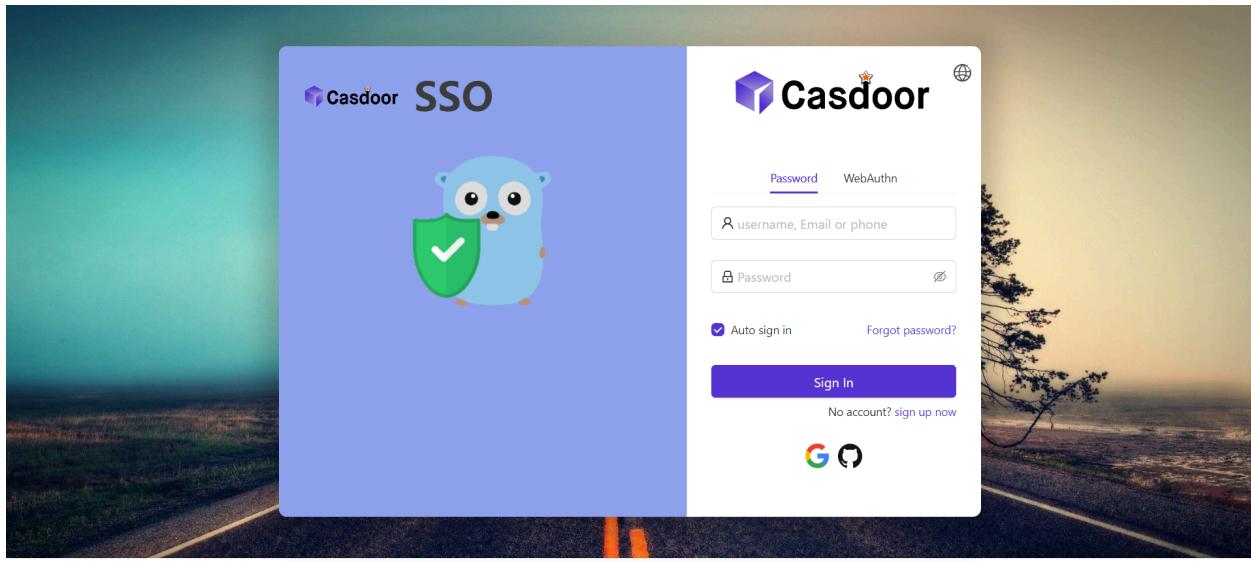
Sign Up

Have account? [sign in](#)

now

# 自定义登录界面

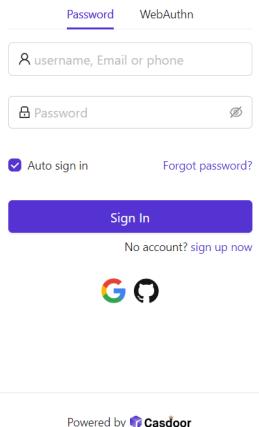
您已创建应用程序。现在，让我为你展示如何自定义应用的登录页面的UI。在这个指南中，我们将创建一个应用的自定义登录页面。



让我们开始吧

## Part 1: 增加背景图片

首先，我们要添加背景图像。默认背景是白色的，看起来非常简单。



若要添加背景图像，请使用您喜欢的图像的 URL，填写 **背景 URL**。如果您填写了有效的URL，预览区域将会显示您所选择的图像。

Background URL ② :	URL ② : <input type="text" value=""/>
Preview:	
Form CSS ② : <input type="text"/>	
Form position ② : <input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right <input type="radio"/> Enable side panel	

## Part 2: 自定义登录面板

现在我们在第一步操作结束的页面：



Powered by Casdoor

为了让面板看起来很好，您需要添加一些CSS 代码。 复制下面的代码并粘贴到 **表单 CSS** 字段。

```
<style>
.login-panel{
    padding: 40px 30px 0 30px;
    border-radius: 10px;
    background-color: #ffffff;
    box-shadow: 0 0 30px 20px rgba(0, 0, 0, 0.20);
}
</style>
```

Background URL  
URL : <https://static.runoob.com/images/demo/demo2.jpg>

Preview:



Form CSS :

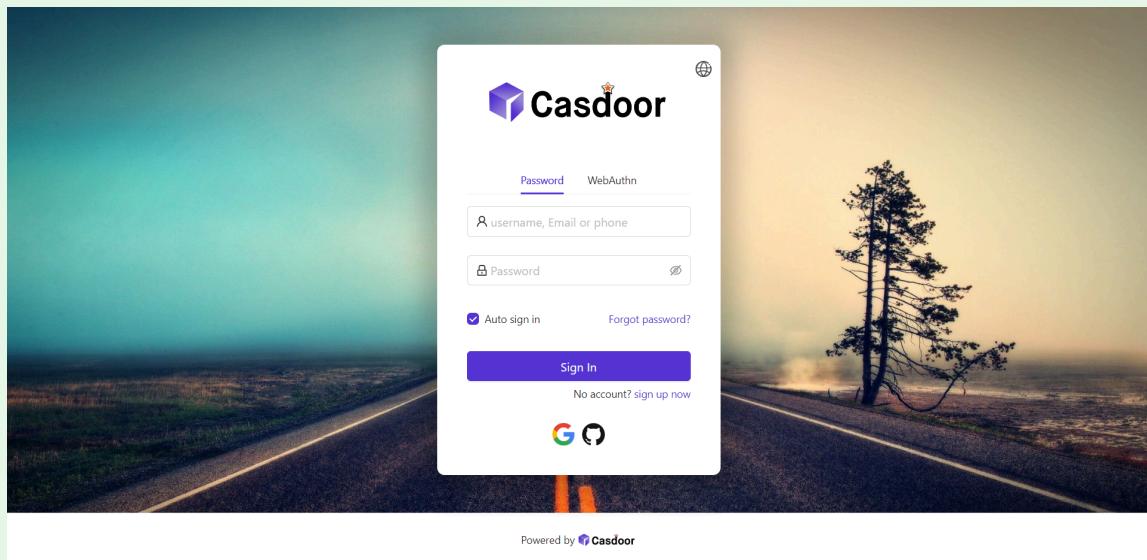
Form position :

### 💡 提示

当您编辑 `form CSS` 时，如果值为空，编辑器将显示默认值。然而，您仍然需要复制内容并将其粘贴到字段。

...

填写 `表单 CSS` 后，不要忘记在底部保存配置。好，让我们看看效果。



## Part 3: 选择面板位置

现在登录页面就比刚开始的页面更为美观。 我们还为您提供了三个按钮，可以决定面板的位置。

Background URL  
URL : <https://static.runoob.com/images/demo/demo2.jpg>

Preview:

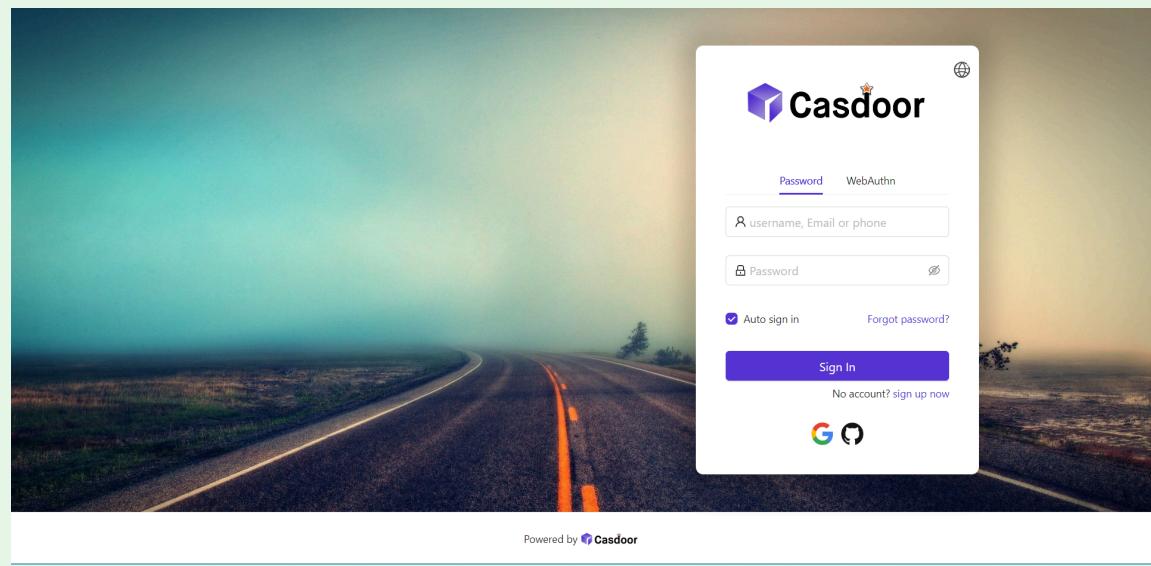


Form CSS :

```
<style>.login-panel{ padding: 40px 30px 0 30px; border-radius: 10px; background-color: #fff; }</style>
```

Form position :

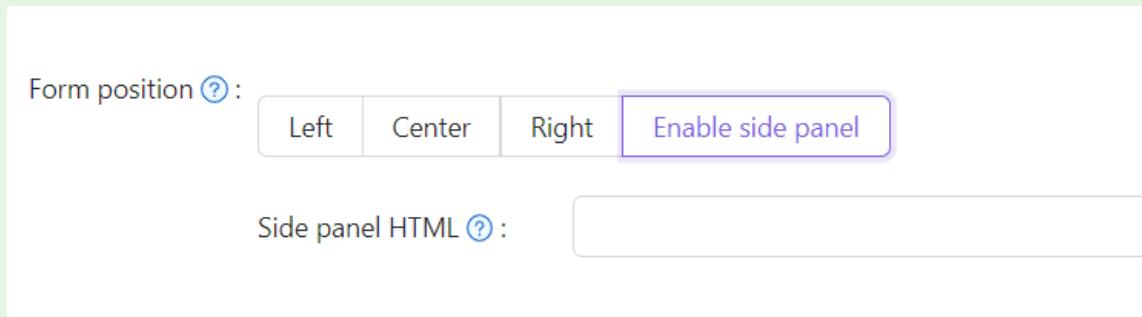
例如，选择 Right 按钮：



## Part 4: 启用侧边面板

接下来，让我们看看如何启用侧边面板并自定义其样式。

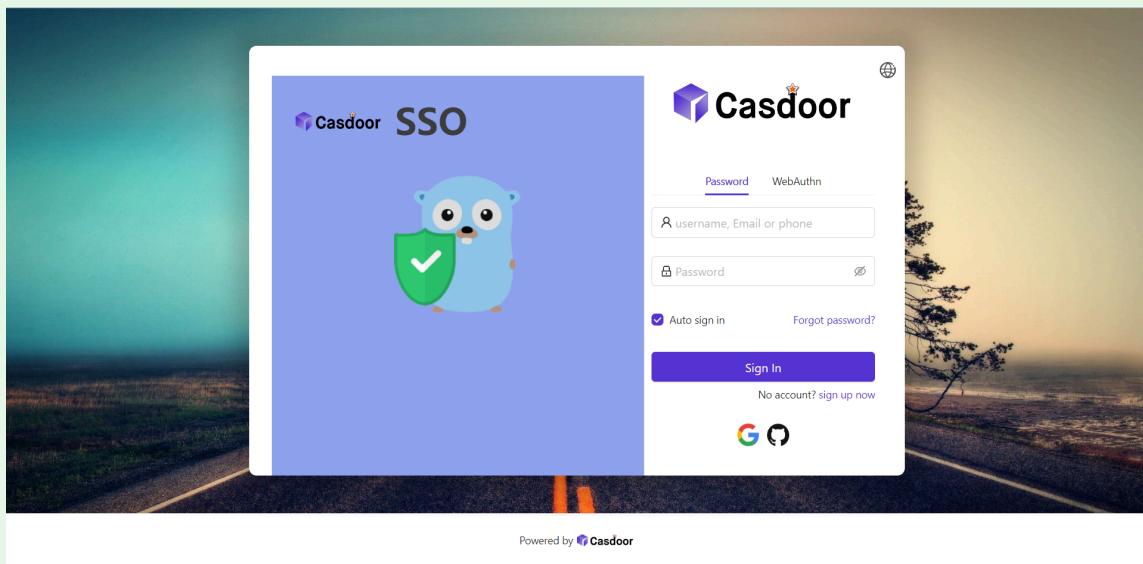
首先选择按钮。在 **启用侧边面板** 模式时，面板会在中间位置。



然后编辑 **侧边板 HTML**，它决定了将显示在侧边板中的内容。我们提供了一个默认模板，所以您可以简单地复制并粘贴它。

```
<style>
  .left-model{
    text-align: center;
    padding: 30px;
    background-color: #8ca0ed;
    position: absolute;
    transform: none;
    width: 100%;
    height: 100%;
  }
  .side-logo{
    display: flex;
    align-items: center;
  }
  .side-logo span {
    font-family: Montserrat, sans-serif;
    font-weight: 900;
```

好，让我们看看效果。 显示带有标志和图像的侧面板，但结果不能令人满意。



为了改进外观，您需要在 [表单 CSS](#) 中修改和添加一些CSS。

Background URL  
② : URL ② : <https://static.runoob.com/images/demo/demo2.jpg>

Preview:

Form CSS ② : 

```
<style> .login-panel{ padding: 40px 30px 0 30px; border-radius: 10px; background-color: #ffffff; box-shadow: 0 0 30px 20px rg
```

Form position ② : [Left](#) [Center](#) [Right](#) [Enable side panel](#)

Side panel HTML ② : 

```
<style> .left-model{ text-align: center; padding: 30px; background-color: #8ca0ed; position: absolute
```

Signup items ② : [Signup items](#) [Add](#)

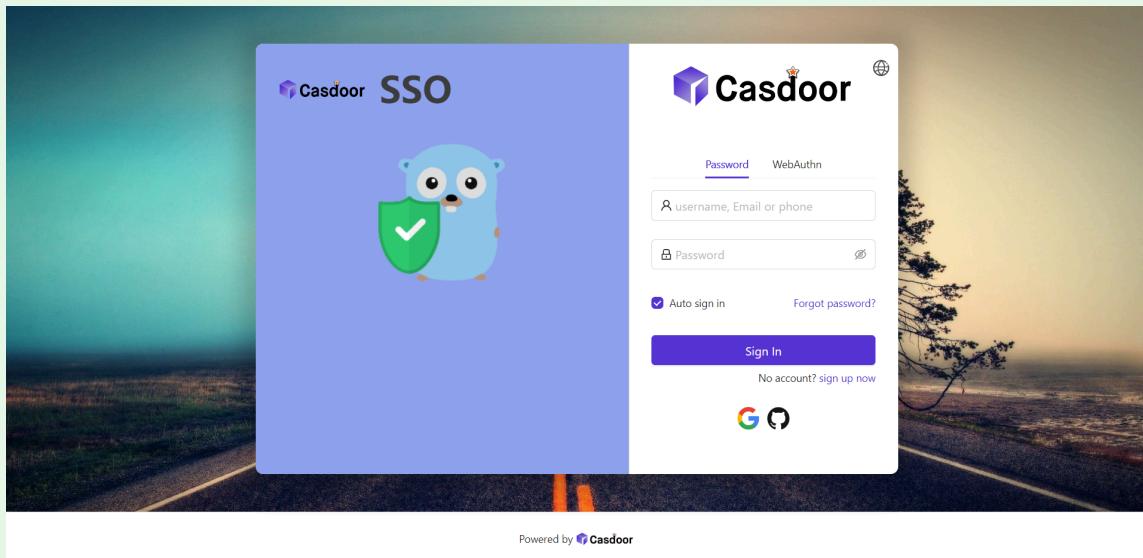
最后代码如下。

```
<style>
.login-panel{
```

## ① 信息

.login-panel 和 .login-form 是div的类名。它们对应于页面的不同区域。如果你想要进一步自定义登录页面，你可以在这里写CSS代码，目标是这些类名称。

最后，我们有了一个美丽的登录页面！



## 总结

我们来总结一下：我们已添加背景图像，自定义了登录面板的风格，并且启用了侧边板。

下面是一些关于定制Casdoor应用程序的额外资源：

- [自定义主题](#): 自定义主题，包括主颜色和边框半径。
- [注册项目表](#)
- [应用程序配置](#)

感谢你的阅读！

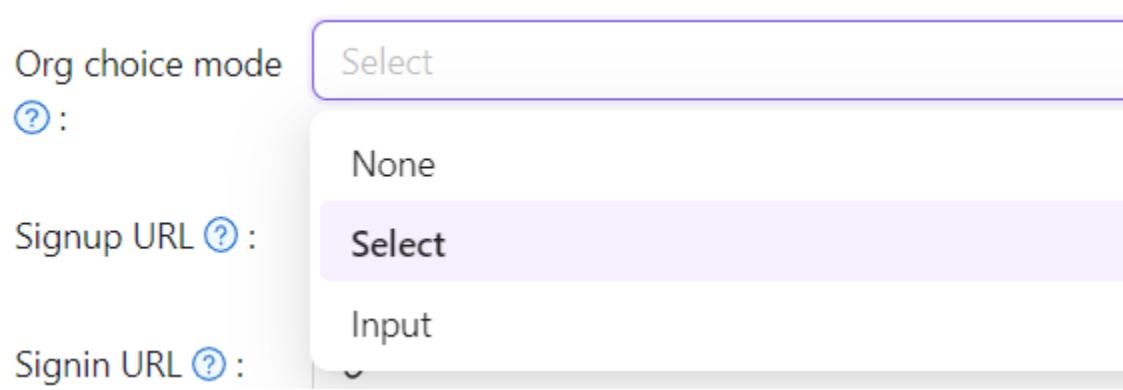
# Specify Login Organization

Here, we will show you how to enable the option to specify the login organization for the application.

For example, the endpoint `/login` is the default sign-in page for accounts belonging to the **built-in** organization. However, you can enable the option to specify the login organization on the **app-built-in** application that belongs to the **built-in** organization. This allows the user to select an organization when logging in. After the user selects the organization, they will be redirected to `/login/<organization>`.

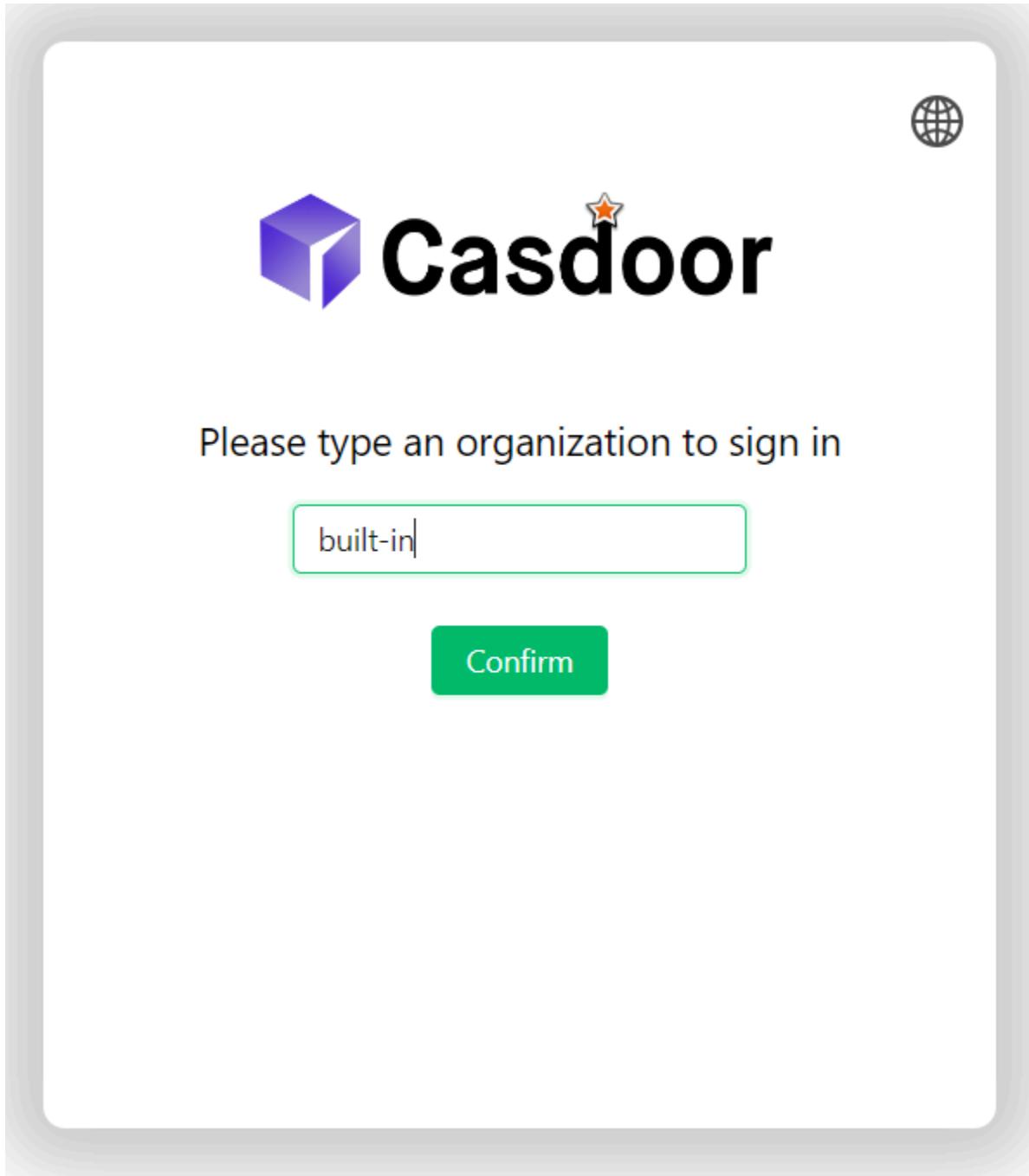
## Configuration

On the application edit page, you can find the `Org select mode` configuration option. You can select the mode from the dropdown list.



- None: The organization select page will not be shown.
- Input: The user can input the organization name in the input box.

- Select: The user can select the organization from the dropdown list.





Please select an organization to sign  
in



built-in

forum

test

Star

### ① 信息

The organization select page will only be shown when the route is `/login` or `<organization>/login`. This means that the application should be set as the default application in the organization or the app-built-in.



# Tags

The application tags are used to restrict user access to the application.

Specifically, only users with tags listed in the application tags are allowed to log in.

For example, the application `dev_app` has tags `dev`, `prd`. Only users with the tag `dev` or `prd` can log in to `dev_app`. Please note that admin and global admin users are not affected by application tags.

On the application edit page, you can find the `Tags` configuration section where you can add tags.

The screenshot shows the Casdoor application edit page. The top navigation bar includes links for Home, Organizations, Groups, Users, Roles, Permissions, Models, Adapters, Applications (which is highlighted), Providers, Chats, Messages, and Resources. Below the navigation is a form for editing an application. The form fields include:

- Name**: only\_tag23
- Display name**: New Application - 907akg
- Logo**: URL: https://cdn.casbin.org/img/casdoor-logo\_1185x256.png
- Preview**: Shows the Casdoor logo with a star icon.
- Home**: (empty)
- Description**: (empty)
- Organization**: built-in
- Tags**: tag2 × tag3 × (This field is highlighted with a red border.)
- Client ID**: 6bed0d62b1e3f21be758
- Client secret**: 7e03320d65163f3ae12fb1d0bd1720eca8f60a14
- Cert**: cert-built-in

Here is a video demonstrating how application tags work:



# 应用邀请码

## 简介

如果你想要限制用户通过应用注册，你可以使用邀请码。 邀请码是能够用来允许用户通过应用注册的字符串。 它由管理员生成并可重复使用。 一个应用可以有多个邀请码。

## 配置

- 首先，将“邀请码”添加到应用的注册项中。
- 然后在应用的设置界面添加邀请码。

The screenshot shows the 'Signup items' configuration screen. At the top, there is a 'Signup items' section with a 'Add' button. Below it is a table with columns: Name, Visible, Required, Prompted, Rule, and Action. The table lists various fields like ID, Username, Display name, Password, etc., with their respective settings. A new row for 'Invitation code' has been added, highlighted with a red box. This row also has 'Visible', 'Required', and 'Prompted' checkboxes, all of which are checked. At the bottom of the table, there is a 'Copy' and a 'Delete' button.

…提示 一旦应用拥有了邀请码，用户只能使用合法的邀请码通过应用进行注册。无论“邀请码”在注册项中是否可见，用户都必须在注册时提供邀请码。因此，如果您想要使用邀请码，您需要将“邀请码”添加到注册项表中。…

以下是一个演示如何配置和使用邀请码的视频：

localhost:7001/applications/built-in/app-built-in

DeepL Translate This... 巴比 Internet Explorer 计算机 学习 网页 小说·古文 My Program | IDTS... Keith Speaking Acc... 480

is compact:

Signup items (1):

Signup items Add

Name	Visible	Required	Prompted	Rule	Action
ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Random	
Username	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	None	
Display name	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	None	
Password	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	
ID card	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	None	
Email	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Normal	
Phone	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	None	
Affiliation	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	None	
Country/Region	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	None	

Invitation code (0):

Add

No data



&gt;

权限

# 权限

## 概述

Using Casbin to manage user access rights in organizations

## 权限配置

Using exposed Casbin APIs to manage users' access rights in an organization

## 开放的 Casbin API

Using exposed Casbin APIs to manage user access rights in organizations

## Adapter

Configure adapter and perform basic CRUD operations on policy

# 概述

## Introduction

All users associated with a single Casdoor organization share access to the organization's applications. However, there may be instances where you want to restrict user access to certain applications or specific resources within an application. In such cases, you can utilize the [Permission](#) feature provided by [Casbin](#).

Before delving deeper into the topic, it is important to have a basic understanding of how Casbin works and its related concepts, such as Models, Policies, and Adapters. In a nutshell, a Model defines the structure of your permission policies and the criteria for matching requests against these policies and their outcomes. A Policy, on the other hand, describes the specific permission rules. Once Casbin has the necessary Model and Policy information, it can enforce permission control on incoming requests. Acting as an abstraction layer, an Adapter shields Casbin's executor from the source of the Policy, allowing the storage of Policies in various locations like files or databases.

Returning to the subject of permission configuration in Casdoor, you can add a Model for your organization in the [Model](#) configuration item within the Casdoor Web UI, and a Policy for your organization in the [Permission](#) configuration item. The [Casbin Online Editor](#) can provide you with Model and Policy files tailored to your specific usage scenarios. You can effortlessly import the Model file into Casdoor through its Web UI for use by the built-in Casbin. However, for the Policy (i.e., the [Permission](#) configuration item in the Casdoor Web UI), further instructions are necessary, which will be discussed later.

Just as your application needs to enforce permission control through Casdoor's built-in Casbin, Casdoor itself utilizes its own Model and Policy to regulate access permissions for the API interfaces through Casbin. Though Casdoor can call Casbin from internal code, external applications cannot. As a solution, Casdoor exposes an API for external applications to call the built-in Casbin. We will provide definitions of these API interfaces and instructions on how to use them shortly.

Towards the end of this chapter, we will showcase a practical example to demonstrate how Casdoor works in collaboration with external applications for permission control.

Let's get started!

# 权限配置

Let's explain each item in the Permission Configuration page.

- **Organization**: 该策略所属的组织名，一个组织可以拥有多个权限策略文件。
- **Name**: The globally unique name of the permission policy in the organization. It is used to identify the policy file.
- **Display name**: Not important.
- **Model**: The name of the model file that describes the structure and matching patterns of the permission policy.
- **Adapter**: Attention! In the current version, this field describes the name of the database table that stores the permission policy, rather than the name of the adapter configured in the Adapter menu item in the Casdoor Web UI. Casdoor uses its own database to store configured permission policies. If this field is empty, the permission policy will be stored in the `permission_rule` table. Otherwise, it will be stored in the specified database table. 如果指定的表名在 Casdoor 所使用的数据库中不存在，将会自动创建。我们强烈建议为不同模型指定不同的适配器，因为将所有的策略保存在同一个表格中可能导致冲突。
- **Sub users**: Which users will the permission policy be applied to.
- **Sub roles**: 如果使用了RBAC模型，哪些角色将被应用该权限策略。这将为该角色中的每一个用户添加诸如 `g user role` 的权限策略。
- **Sub domains**: Which domains will the permission policy be applied to.
- **Resource type**: In the current version, Casdoor does not use this field for external applications that want to authenticate. 你可以暂且忽略它。
- **Resources**: 这个字段描述了你希望强制实施权限控制的资源。但请注意，这里的资源并非在 Casdoor Web UI 的 Resources 菜单项中所配置的资源。你可以在此处添加任何你所希望的字符串，例如一个 URL 或者一个文件名。

- **Actions**: This field describes the actions to operate on resources. Similar to resources, it can be any string you want, such as an HTTP method or other natural language. 但是请注意，Casdoor 会将这些字符串全部转化成为小写再存储。另外，Casdoor 将会为每一个资源应用所有的动作。 You cannot specify that an action only takes effect on certain resources.
- **Effect**: 这个选项对于 Casdoor 自身控制应用访问权限生效。如果你希望外部应用使用 Casdoor 所暴露的接口来强制实施权限控制，它不会起到任何作用。你应该在 Model 文件中描述模式匹配后的作用。

你可以看到，这个配置页面几乎就是为 **(sub, obj, act)** 模型量身定制的。

# 开放的 Casbin API

## 介绍

Let's assume that your application's front-end has obtained the `access_token` of the logged-in user and now wants to authenticate the user for some access. You cannot simply place the `access_token` into the HTTP request header to use these APIs because Casdoor uses the `Authorization` field to check the access permission. Like any other APIs provided by Casdoor, the `Authorization` field consists of the application client id and secret, using the [Basic HTTP Authentication Scheme](#). It looks like `Authorization: Basic <Your_Application_ClientId> <Your_Application_ClientSecret>`. 因此, Casbin API 应当被应用的后端服务器调用。Here are the steps on how to do it.

Take the [app-vue-python-example](#) application in the demo site for example, the authorization header should be: `Authorization: Basic 294b09fbc17f95daf2fe dd8982f7046ccba1bbd7851d5c1ece4e52bf039d`.

1. The front-end passes the `access_token` to the backend server through the HTTP request header.
2. The backend server retrieves the user id from the `access_token`.

As a note in advance, these interfaces are also designed (for now) for the `(sub, obj, act)` model. The `permissionId` in the URL parameters is the identity of the applied permission policy, which consists of the organization name and the permission policy name (i.e., `organization name/permission name`). The body is the request format defined by the Casbin model of the permission, usually representing `sub`, `obj` and `act` respectively.

除了请求强制执行权限控制的 API 接口以外，Casdoor 也提供了其它一些有助于外部应用获取权限策略信息的接口，也一并列在此处。

## Enforce

请求：

```
curl --location --request POST 'http://localhost:8000/api/enforce?permissionId=example-org/example-permission' \
--header 'Content-Type: application/json' \
--header 'Authorization: Basic <Your_Application_ClientId><Your_Application_ClientSecret>' \
--data-raw '["example-org/example-user", "example-resource", "example-action"]'
```

响应：

```
{
  "status": "ok",
  "msg": "",
  "sub": "",
  "name": "",
  "data": [
    true
  ],
  "data2": null
}
```

## BatchEnforce

请求：

```
curl --location --request POST 'http://localhost:8000/api/batch-enforce?permissionId=example-org/example-permission' \
--header 'Content-Type: application/json' \
--header 'Authorization: Basic <Your_Application_ClientId><Your_Application_ClientSecret>' \
--data-raw '[["example-org/example-user", "example-resource", "example-action"], ["example-org/example-user2", "example-resource", "example-action"], ["example-org/example-user3", "example-resource", "example-action"]]'
```

响应:

```
{  
    "status": "ok",  
    "msg": "",  
    "sub": "",  
    "name": "",  
    "data": [  
        [  
            true,  
            true,  
            false  
        ]  
    ],  
    "data2": null  
}
```

## GetAllObjects

请求:

```
curl --location --request GET 'http://localhost:8000/api/get-all-objects' \
--header 'Authorization: Basic <Your_Application_ClientId>
```

响应:

```
[  
    "app-built-in"  
]
```

## GetAllActions

请求:

```
curl --location --request GET 'http://localhost:8000/api/get-all-  
actions' \  
--header 'Authorization: Basic <Your_Application_ClientId>  
<Your_Application_ClientSecret>'
```

响应:

```
[  
    "read",  
    "write",  
    "admin"  
]
```

## GetAllRoles

请求:

```
curl --location --request GET 'http://localhost:8000/api/get-all-  
roles' \  
--header 'Authorization: Basic <Your_Application_ClientId>  
<Your_Application_ClientSecret>'
```

响应：

```
[  
    "role_kcx661"  
]
```

# Adapter

Casdoor supports using the UI to connect the adapter and manage policy rules. In Casbin, the storage of policy rules is implemented as an adapter, which acts as middleware for Casbin. A Casbin user can use an adapter to load policy rules from a storage or save policy rules to it.

## Adapter

- `type`: Adapter type. Currently supports database adapter.
- `Host`
- `Port`
- `User`
- `Password`
- `Database type`: Currently supports MySQL, PostgreSQL, SQL Server, Oracle, SQLite 3.
- `Database`: The name of the database.
- `Table`: The name of the table. If the table does not exist, it will be created.

Edit Adapter Save Save & Exit

Organization <small>②</small> :	built-in
Name <small>②</small> :	casdoor_adapter
Type <small>②</small> :	Database
Host <small>②</small> :	localhost
Port <small>②</small> :	3306
User <small>②</small> :	root
Password <small>②</small> :	123456
Database type <small>②</small> :	MySQL
Database <small>②</small> :	casdoor
Table <small>②</small> :	casbin_rule

## ① 信息

After filling in all the fields, please remember to **save** the configuration.  
 Then click the **sync** button to load the policy rules. The policy rules will be displayed in the table below.

Policies ②:

Rule Type	V0	V1	V2	V3	V4	V5	Option	
p	built-in	*	*	*	*	*		
p	app	*	*	*	*	*		
p	*	*	POST	/api/signup	*	*		
p	*	*	POST	/api/get-email-and-phone	*	*		
p	*	*	POST	/api/login	*	*		
p	*	*	GET	/api/get-app-login	*	*		
p	*	*	POST	/api/logout	*	*		
p	*	*	GET	/api/logout	*	*		
p	*	*	GET	/api/get-account	*	*		
p	*	*	GET	/api/userinfo	*	*		

< 1 2 3 4 5 >

Is enabled ②:

# Basic CRUD Operations

If you have successfully connected the adapter, you can perform basic CRUD operations on the policy rules.

- Add

Policies ⓘ	Sync	Add	Rule Type	V0	V1	V2	V3	V4	V5	Option
p	built-in	↳	p	*	*	*	*	*	*	edit delete
p	*	*	p	*	*	POST	/api/signup	*	*	edit delete
p	*	*	p	*	*	POST	/api/get-email-and-phone	*	*	edit delete
p	*	*	p	*	*	POST	/api/login	*	*	edit delete
p	*	*	p	*	*	GET	/api/get-app-login	*	*	edit delete
p	*	*	p	*	*	POST	/api/logout	*	*	edit delete
p	*	*	p	*	*	GET	/api/logout	*	*	edit delete
p	*	*	p	*	*	GET	/api/get-account	*	*	edit delete
p	*	*	p	*	*	GET	/api/userinfo	*	*	edit delete
p	*	*	p	*	*	POST	/api/webhook	*	*	edit delete



You can only add one policy at a time. The newly added policy will appear as the first row in the table, but it will actually be saved in the last row. So, when you sync the policies next time, they will appear in the last row of the table.

- Edit

casbin_rule								
Model	casbin_rule							
Policies		Sync	Add					
Rule Type	V0	V1	V2	V3	V4	V5	Option	
p	built-in	*	POST	*	*	*	 	
p	app	*	*	/api/signup	*	*	 	
p	*	*	POST	/api/get-email-and-phone	*	*	 	
p	*	*	POST	/api/login	*	*	 	
p	*	*	GET	/api/get-app-login	*	*	 	
p	*	*	POST	/api/logout	*	*	 	
p	*	*	GET	/api/logout	*	*	 	
p	*	*	GET	/api/get-account	*	*	 	
p	*	*	GET	/api/userinfo	*	*	 	

< 1 2 3 4 5 >

- Delete

User	root	Password	123456	Database type	MySQL	Database	casdoor	Table	casbin_rule	Model	casbin_rule	Policies
Rule Type	V0	V1	V2	V3	V4	V5	Option					
p	*	*	GET	/api/get-default-application	*	*	 					
p	test	*	*	*	*	*	 					

< 1 2 3 4 5 >



&gt;

提供商

# 提供商

## 概述

添加第三方服务到您的应用程序

## OAuth

22 个项目

## 电子邮箱

5 个项目

## 短信

5 个项目



## 通知

7 个项目



## 存储

9 个项目



## SAML

4 个项目



## 支付

5 个项目



## 验证码

7 个项目



Web3

2 个项目

# 概述

Casdoor utilizes providers to offer third-party services for the platform. In this chapter, you will learn how to add providers to Casdoor.

## What We Have

Currently, we have six types of providers:

- OAuth Providers

允许用户通过其他 OAuth 应用程序登录。 You can add GitHub, Google, QQ, and many other OAuth applications to Casdoor. For more details, please refer to the [OAuth](#) section.

- 短信提供商

当用户想要验证他们的电话号码时，Casdoor将发送短信给他们。 短信提供者被用来发送Casdoor短信。

- 电子邮件提供商

电子邮件提供者与短信提供者类似。

- 存储提供商

Casdoor allows users to store files using the local file system or cloud OSS services.

- Payment Providers

Casdoor 可以添加付款提供者，用于在产品页面上添加付款方法。 Currently, the supported payment providers include Alipay, WeChat Pay, PayPal, and GC.

- **Captcha Providers**

Casdoor 支持在用户流程中配置验证码。 Currently, the supported captcha providers include Default Captcha, reCAPTCHA, hCaptcha, Alibaba Cloud Captcha, and Cloudflare Turnstile.

## How to Configure and Use

### 范围

Providers have different scopes, which are determined by the creator. Only Administrators have the permission to add and configure providers. There are two types of Administrators in Casdoor:

- **Global Administrator:** All users under the `built-in` organization and the users who enable `IsGlobalAdmin`. The providers created by the Global Administrator can be used by all applications.
- **Organization Administrator:** Users who enable `IsAdmin`. The providers created by the Organization Administrator can **only** be used by the applications under the organization (under development...).

### Add to Application

Follow the steps below to add providers to your application. Note that you cannot use the provider in your application until you have added it.

1. 转到应用程序编辑页面并添加一个新的提供商。

Providers [?](#) :

Name	Category	Type
provider_storage_aliyun_oss	Storage	
provider_casdoor_github	OAuth	
provider_casdoor_google	OAuth	
provider_casdoor_qq	OAuth	
provider_casdoor_wechat	OAuth	
Please select a provider		

2. Select a provider that you want to add to the application. You will see all the providers that the application can use.

Providers [?](#) :

Name	Category	Type	canSignUp
provider_storage_aliyun_oss	Storage		
provider_casdoor_github	OAuth		<input checked="" type="checkbox"/>
provider_casdoor_google	OAuth		<input checked="" type="checkbox"/>
provider_casdoor_qq	OAuth		<input checked="" type="checkbox"/>
provider_casdoor_wechat	OAuth		<input checked="" type="checkbox"/>
Please select a provider			

Preview [?](#) :

- provider\_email\_submail
- provider\_4olfdm
- provider\_casdoor\_bilibili
- provider\_casdoor\_okta
- provider\_casdoor\_alipay
- provider\_casdoor\_slack
- provider\_casdoor\_steam
- provider\_casdoor\_infoflow

[Copy](#)

3. For OAuth and Captcha providers, you can configure their usage. See [OAuth](#)

and [Captcha](#) for more information.

Type	canSignUp	canSignIn	canUnlink	prompted	Rule
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Always ▾
	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Finally, [save](#) the configuration. You can now try using the provider in your application.

# OAuth

## 概述

将 OAuth 提供商添加到您的应用程序

## 谷歌

Add the Google OAuth provider to your application

## Google One Tap

Learn how to add Google One Tap support to your application

## GitHub

Add GitHub OAuth provider to your application

 **LinkedIn**

Add LinkedIn OAuth provider to your application

 **Facebook**

Add the Facebook OAuth provider to your application.

 **AD FS**

Add AD FS as a third-party service to complete authentication.

 **Azure AD**

Add Azure AD as a third-party service to complete authentication

 **Azure AD B2C**

Add Azure AD B2C as a third-party service to complete authentication

 **Custom OAuth**

Add your custom OAuth provider to Casdoor

 **Okta**

将 Okta OAuth 提供商添加到您的应用程序

 **Twitter**

添加Twitter OAuth 提供商到您的应用程序

 **微博**

将 Weibo OAuth 提供商添加到您的应用程序

 **微信**

将微信 OAuth 提供商添加到您的应用程序

 **企业微信**

将 WeCom OAuth 提供商添加到您的应用程序

 **腾讯 QQ**

添加腾讯QQ OAuth 提供商到您的应用程序。

 **钉钉**

添加钉钉 OAuth 到您的应用程序

 **Steam**

Add the Steam OAuth provider to your application

 **Gitee**

添加Gitee OAuth 提供商到您的应用程序

 百度

向您的应用程序添加 Baidu OAuth 提供商

 Infoflow

在应用程序中添加 Infoflow OAuth 提供商

 Lark

Add Lark OAuth provider to your application

# 概述

Casdoor allows for the use of other OAuth applications as a sign-in method.

Currently, Casdoor supports multiple OAuth application providers. The icons of these providers will be displayed on the login and signup pages once they have been added to Casdoor. The following are the providers that Casdoor supports:

Provider	Logo	Provider	Logo	Provider	Logo	Provider	Logo
ADFS		Alipay		Amazon		Apple	
Auth0		Azure AD		Azure AD B2C		Baidu	
Bilibili		Bitbucket		Box		Casdoor	
Cloud Foundry		Dailymotion		Deezer		DigitalOcean	
DingTalk		Discord		Tiktok		Dropbox	
Eve Online		Facebook		Fitbit		Gitea	
Gitee		GitHub		GitLab		Google	
Heroku		InfluxCloud		Infoflow		Instagram	

Provider	Logo	Provider	Logo	Provider	Logo	Provider	Logo
Intercom		Kakao		Lark		Lastfm	
Line		LinkedIn		Mailru		Meetup	
Microsoft		Naver		Nextcloud		Okta	
OneDrive		Oura		Patreon		PayPal	
QQ		Salesforce		Shopify		Slack	
SoundCloud		Spotify		Steam		Strava	
Stripe		TikTok		Tumblr		Twitch	
Twitter		TypeTalk		Uber		VK	
WeChat		WeCom		Weibo		WePay	
Xero		Yahoo		Yammer		Yandex	
Zoom		Email		SMS		Battle.net	

我们将向您展示如何申请第三方服务并将其添加到Casdoor。

# 申请成为开发者

在此之前，你需要理解一些概念。

- **RedirectUrl**, 认证后重定向地址, 填写您的应用程序地址, 例如 <https://forum.casbin.com/>
- **Scope**, 用户授予您的权限, 如基本个人资料, 电子邮件地址和帖子及其他。
- **ClientId/AppId, ClientKey/AppSecret**, 这是最重要的信息 而且这是您在申请开发者帐户后需要得到的信息。您无法与任何人共享 的密钥。

## 添加 OAuth 提供商

1. Go to your Casdoor index page.
2. Click on `Providers` in the top bar.
3. Click on `Add`, and you will see a new provider added to the list at the top.
4. Click on the new provider to make changes to it.
5. In the `Category` section, select `OAuth`.
6. Choose the specific OAuth provider that you require from the `Type` dropdown.
7. Fill in the necessary information, such as `Client ID` and `Client Secret`.

## Application Setup

1. Click on `Application` in the top bar and select the desired application to edit.
2. Click on the provider add button and choose the newly added provider.
3. Modify the provider's permissions, such as enabling registration, login, and unbinding.
4. You're all set!

# 谷歌

To set up the Google OAuth provider, please go to the [Google API console](#) and log in using your Google account.

Project name \*

My Casdoor



Project ID: my-casdoor. It cannot be changed later. [EDIT](#)

Location \*

 No organization

[BROWSE](#)

Parent organization or folder

[CREATE](#)

CANCEL

Next, navigate to the OAuth consent screen tab to configure the OAuth consent screen.

**API** APIs & Services

## OAuth consent screen

 Dashboard

Choose how you want to configure and register your app, including your target users. You can only associate one app with your project.

 Library Credentials OAuth consent screen Domain verification Page usage agreements

## User Type

 Internal 

Only available to users within your organization. You will not need to submit your app for verification. [Learn more about user type](#)

 External 

Available to any test user with a Google Account. Your app will start in testing mode and will only be available to users you add to the list of test users. Once your app is ready to push to production, you may need to verify your app. [Learn more about user type](#)

**CREATE**

Register your Google app by following these steps:

## Edit app registration

1 OAuth consent screen — 2 Scopes — 3 Test users — 4 Summary

### App information

This shows in the consent screen, and helps end users know who you are and contact you

App name \*

The name of the app asking for consent

User support email \*

For users to contact you with questions about their consent

App logo

BROWSE

Upload an image, not larger than 1MB on the consent screen that will help users recognize your app. Allowed image formats are JPG, PNG, and BMP. Logos should be square and 120px by 120px for the best results.

### App domain

To protect you and your users, Google only allows apps using OAuth to use Authorized Domains. The following information will be shown to your users on the consent screen.

Application home page

Afterward, go to the Credential tab.

## Credentials

[+ CREATE CREDENTIALS](#) [DELETE](#)

Create credentials to access your enabled APIs. [Learn more](#)

### API Keys

<input type="checkbox"/>	Name	Creation date
No API keys to display		

### OAuth 2.0 Client IDs

<input type="checkbox"/>	Name	Creation date
No OAuth clients to display		

### Service Accounts

<input type="checkbox"/>	Email	Name
No service accounts to display		

Create a credential for your app:

[Create OAuth client ID](#)

A client ID is used to identify a single app to Google's OAuth servers. If your app runs on multiple platforms, each will need its own client ID. See [Setting up OAuth 2.0](#) for more information. [Learn more](#) about OAuth client types.

Application type \*



### **!** ENSURE THAT YOU SET THE AUTHORIZED REDIRECT URIS CORRECTLY

In the Google OAuth configuration, the `Authorized redirect URIs` must be set to your Casdoor's callback URL, while the `Redirect URL` in Casdoor should be set to your application's callback URL.

For more details, please refer to the [App configuration](#).

After creating the Client ID, you will obtain the `Client ID` and `Client Secret`.

## OAuth client created

The client ID and secret can always be accessed from Credentials in APIs & Services



OAuth access is restricted to the [test users](#) listed on your [OAuth consent screen](#)

Your Client ID

487708653175-11oih9gfqb2u3tvfp6684qaes5ujjdca.apps.googleusercontent.com



Your Client Secret

HbxoqxxkGSs1lCVRuMTVvK57



DOWNLOAD JSON

OK

Add the Google OAuth provider and enter the `Client ID` and `Client Secret` in your Casdoor.

[Edit Provider](#) [Save](#)

---

Name [?](#) :

Display name [?](#) :

Category [?](#) :

Type [?](#) :

Client ID [?](#) :

Client secret [?](#) :

Provider URL [?](#) : <https://console.cloud.google.com/apis/credentials/oauthclient/498643462012-46>

You can now use Google as a third-party service to complete authentication.

# Google One Tap

## Step 1: Configure Your Application

If you want to allow login through Google One Tap, you'll need to add Google OAuth Provider to your application. For information on how to do this, please refer to [Google's documentation](#).

Once you've added the Google OAuth Provider, navigate to the application edit page, add the Google OAuth Provider, and switch the **Rule** from **Default** to **One Tap**.

The screenshot shows the application edit page with the following details:

- SAML metadata:** A large block of XML code representing the SAML metadata for the application.
- Providers:** A table listing available providers:

Name	Category	Type	Can signup	Can signin	Can unlink	Rule	Action
provider_storage_minio_s3	Storage	None	On	On	On	Off	Up/Down
provider_oauth_fark	OAuth	Icon	On	On	On	Off	Up/Down
provider_email_qq	Email	Icon	On	On	On	Off	Up/Down
provider_web3_metamask	Web3	Icon	On	On	On	Off	Up/Down
provider_google_oauth	OAuth	Icon	On	On	On	On	Up/Down
- Rule:** A dropdown menu currently set to "Default". A red arrow points to the "One Tap" option in the dropdown menu.
- Preview:** Buttons to "Copy SAML metadata URL", "Copy sign up page URL", and "Copy sign in page URL".

## Step 2: Logging In with Google One Tap

With the setup completed, users can now log in with Google One Tap.

# GitHub

GitHub OAuth supports both the web application flow and device flow. Please continue reading to obtain OAuth credentials.

First, please visit the [GitHub developer settings](#) to register a new GitHub App.

## 注意事项

**Tricks:** We recommend that you use GitHub Apps to replace OAuth Apps because GitHub Apps can add multiple redirect URLs, which can bring convenience when deploying test and production environments. The [GitHub](#) official also recommends using GitHub Apps instead of OAuth Apps.

## Settings / Developer settings

 GitHub Apps

 OAuth Apps

 Personal access tokens

Then fill in the GitHub App name, Homepage URL, description, and Callback URL.

GitHub App name \*

The name of your GitHub App.

Write

Preview

Markdown supported

A UI-first centralized authentication / Single-Sign-On (SSO) platform supporting OAuth 2.0, OIDC and SAML, integrated with Casbin RBAC and ABAC permission management

Homepage URL \*

The full URL to your GitHub App's website.

Add Callback URL

Callback URL

Delete

Callback URL

Delete

## ⚠ SET THE AUTHORIZATION CALLBACK URL CORRECTLY

In the GitHub App config, the `Callback URL` must be **your Casdoor's callback URL**, and the `Redirect URL` in Casdoor should be **your application's callback URL**.

For more details, please read [App config](#).

After registering your GitHub App, you can now generate your `Client Secret`!

## About

Owned by: [REDACTED]

App ID: [REDACTED]

Client ID: lv1 [REDACTED] d2e

[Revoke all user tokens](#)

GitHub Apps can use OAuth credentials to identify users. Learn more about identifying users by reading our [integration developer documentation](#).

## Client secrets

[Generate a new client secret](#)



\*\*\*\*\*dba81954

Added 5 minutes ago by [REDACTED]

[Client secret](#)

[Delete](#)

Last used within the last week



\*\*\*\*\*15822f89

Added on 15 Feb by [REDACTED]

[Client secret](#)

[Delete](#)

Add a GitHub OAuth provider and fill in the `Client ID` and `Client Secret` in your Casdoor.

Edit Provider

[Save](#) [Save & Exit](#)

Name <a href="#">?</a> :	provider_github_localhost
Display name <a href="#">?</a> :	provider_github_localhost
Category <a href="#">?</a> :	OAuth
Type <a href="#">?</a> :	GitHub
Client ID <a href="#">?</a> :	Iv` [REDACTED] .2e
Client secret <a href="#">?</a> :	***
Provider URL <a href="#">?</a> :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>

[Save](#)

[Save & Exit](#)

Now you can use GitHub as a third-party service to complete authentication.

# LinkedIn

To set up the LinkedIn OAuth provider, please go to the [LinkedIn Developer](#) page to create a new app.

 DEVELOPERS    Products    Docs and tools ▾    Resources ▾    My apps ▾

## Create an app

\* indicates required

**App name\***

**LinkedIn Page\***  
ⓘ This action can't be undone once the app is saved.  
  
The LinkedIn Company Page you select will be associated with your app. Verification can be done by a Page Admin. Please note this cannot be a member profile page. [Learn more](#)

[+ Create a new LinkedIn Page ↗](#)

**Privacy policy URL**

**App logo\***  
This is the logo displayed to users when they authorize with your app  
  
 [Upload a logo](#)

After filling in the form above and creating your app, you'll need to verify the LinkedIn page associated with the app.



## Identity Cloud Login

Client ID: 860t47n8b4jh7w | Created: Sep 4, 2020

**Settings**

Auth

Products

Analytics

Team members

### App settings

[Delete app](#)

Company:



**Identity Cloud Documentation**

Computer Software; 1-10 employees

[Verify](#)



This app is not verified as being associated with this company.

[Learn more](#)

ⓘ 备注

Only the company page administrator can verify your app and grant permission to it.

Once your app is verified, you can continue:

 Identity Cloud Login  
Client ID: 860t47n8b4jh7w | Created: Sep 4, 2020

Settings Auth **Products** Analytics Team members

**Products**

Additional available products

 **Marketing Developer Platform**  
Build marketing experiences to reach the right audiences **Select**  
[View docs ↗](#)

 **Share on LinkedIn**  
Amplify your content by sharing it on LinkedIn **Select**  
[View docs ↗](#)

 **Sign In with LinkedIn**  
Let users easily sign in with their professional identity **Select**  
[View docs ↗](#)

Add authorized redirect URLs for your app as your Casdoor callback URL.

**Authorized redirect URLs for your app**

*No redirect URLs added*

**+ Add redirect URL**

 正确设置授权的重定向 URL

In the LinkedIn OAuth configuration, the `authorized redirect URLs` must be your Casdoor's callback URL, and the `Redirect URL` in Casdoor should be your application's callback URL.

For more details, please read the [App Config](#) section.

You can then obtain your `Client ID` and `Client Secret`.

## Application credentials

### Authentication keys

#### Client ID:

860t47n8b4jh7w

#### Client Secret:

.....



Add a LinkedIn OAuth provider and fill in the `Client ID` and `Client Secret` in your Casdoor.

Edit Provider

Save

Name [?](#) : my\_linkedin\_provider

Display name [?](#) : Linkedin provider

Category [?](#) : OAuth

Type [?](#) : LinkedIn

Client ID [?](#) : 860t47n8b4jh7w

Client secret [?](#) : \*\*\*\*

Now you can use LinkedIn as a third-party service to complete authentication!

# Facebook

To set up the Facebook OAuth provider, please go to the [Facebook Developer](#) website and create a new app.

Select the type of app you are going to create.

## Select an app type

X

The app type can't be changed after your app is created.



Create or manage business assets like Pages, Events, Groups, Ads, Messenger and Instagram Graph API using the available business permissions, features and products.



### Consumer

Connect consumer products, and permissions, like Facebook Login and Instagram Basic Display to your app.



### Instant Games

Create an HTML5 game hosted on Facebook.



### Gaming

Connect an off-platform game to Facebook Login.



### Workplace

Create enterprise tools for Workplace from Facebook.



### None

Create an app with combinations of consumer and business permissions and products.

[Learn More About App Types](#)

Cancel

Continue

After entering your name and contact email, you will be taken to the Facebook Developer dashboard.

**FACEBOOK for Developers**

Docs Tools Support My Apps  ?

Casdoor App ID: 1231340483981478 In development

Dashboard Settings Roles Alerts App Review Products Add Product

Add a Product

**Facebook Login**  
The world's number one social login product.  
[Read Docs](#) [Set Up](#)

**Audience Network**  
Monetize your app and grow revenue with ads from Facebook advertisers.  
[Read Docs](#) [Set Up](#)

**App Events**  
Understand how people engage with your business across apps, devices, platforms and websites.  
[Read Docs](#) [Set Up](#)

**Messenger**  
Customize the way you interact with people on  
[Read Docs](#)

**Webhooks**  
Subscribe to changes and receive updates in real time.  
[Read Docs](#)

**Instant Games**  
Create a cross-platform HTML 5 game hosted on  
[Read Docs](#)

Next, set up Facebook login:



## Facebook Login

The world's number one social login product.

[Read Docs](#)

[Set Up](#)

Choose the Web platform for this app:

Use the Quickstart to add Facebook Login to your app. To get started, select the platform for this app.



iOS



Android



Web



Other

After filling in the website URL, go to **Facebook Login > Settings** and enter valid OAuth Redirect URIs.

#### Client OAuth Settings

Yes

##### Client OAuth Login

Enables the standard OAuth client token flow. Secure your application and prevent abuse by locking down which token redirect URIs are allowed with the options below. Disable globally if not used. [?]

Yes

##### Web OAuth Login

Enables web-based Client OAuth Login. [?]

Yes

##### Enforce HTTPS

Enforce the use of HTTPS for Redirect URIs and the JavaScript SDK. Strongly recommended. [?]

No

##### Force Web OAuth Reauthentication

When on, prompts people to enter their Facebook password in order to log in on the web. [?]

No

##### Embedded Browser OAuth Login

Enable webview Redirect URIs for Client OAuth Login. [?]

Yes

##### Use Strict Mode for Redirect URIs

Only allow redirects that exactly match the Valid OAuth Redirect URIs. Strongly recommended. [?]

#### Valid OAuth Redirect URIs

A manually specified redirect\_uri used with Login on the web must exactly match one of the URIs listed here. This list is also used by the JavaScript SDK for in-app browsers that suppress popups. [?]

Valid OAuth redirect URIs.

### ① 正确设置授权的 REDIRECT URL

In the Facebook OAuth configuration, the `Valid OAuth Redirect URIs` must be your Casdoor's callback URL, and the `Redirect URL` in Casdoor should be your application's callback URL.

For more details, please read the [App Configuration](#) section.

The basic app configuration is almost complete!

Switch the mode from **In development** to **Live** in the top bar of the dashboard.



Now you can use your **App ID** and **App Secret** in Casdoor.

App ID	App Secret
<input type="text" value="1231340483981478"/>	<input type="text" value="*****"/> <a href="#">Show</a>

Add the Facebook OAuth provider and fill in the **Client ID** and **Client Secret** with the **App ID** and **App Secret** from your Casdoor.

修改提供商保存

名称 <small>②</small> :	<input type="text" value="my_facebook_provider"/>
显示名称 <small>②</small> :	<input type="text" value="Facebook provider"/>
分类 <small>②</small> :	<input type="text" value="OAuth"/>
类型 <small>②</small> :	<input type="text" value="Facebook"/>
Client ID <small>②</small>	<input type="text" value="1231340483981478"/>
Client secret <small>②</small>	<input type="text" value="*****"/>

You can now use Facebook as a third-party service for authentication!

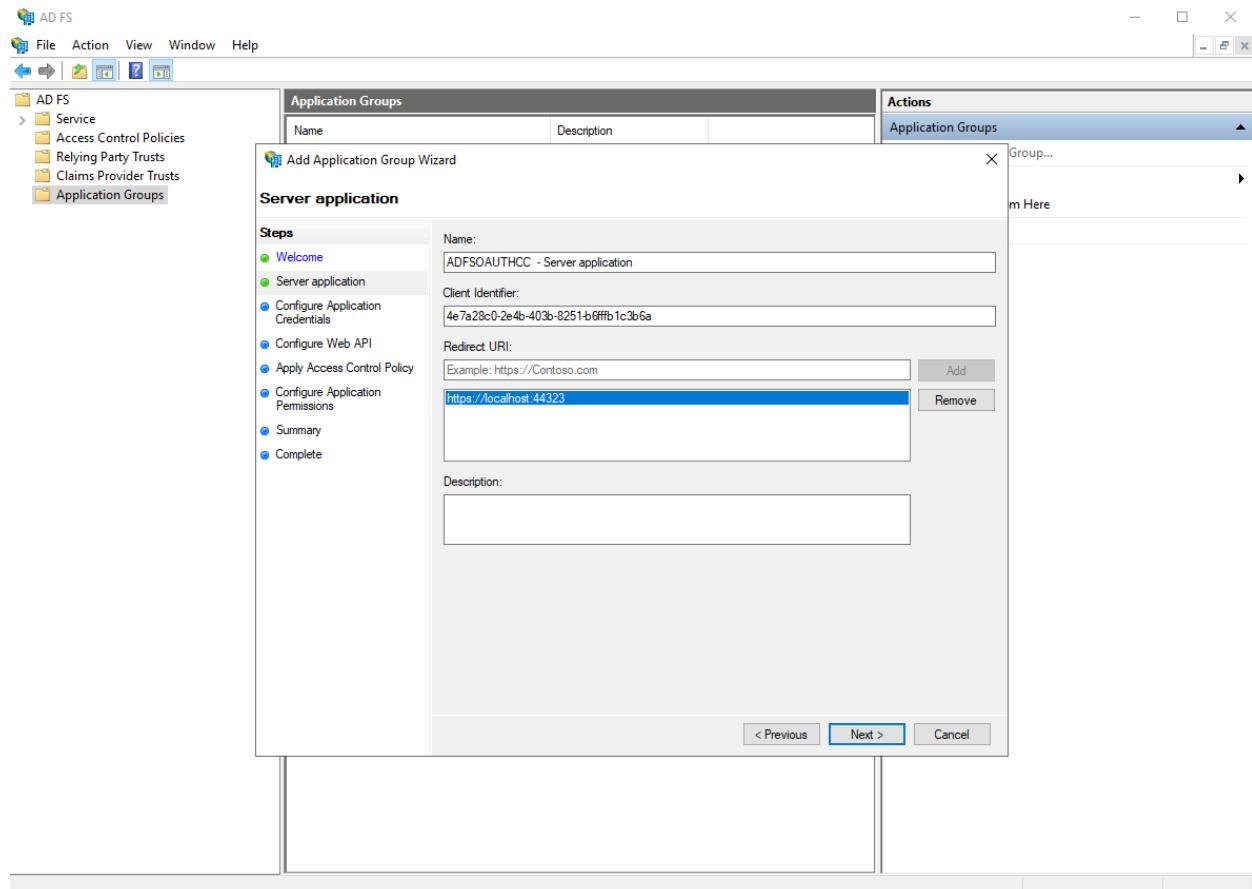
# AD FS

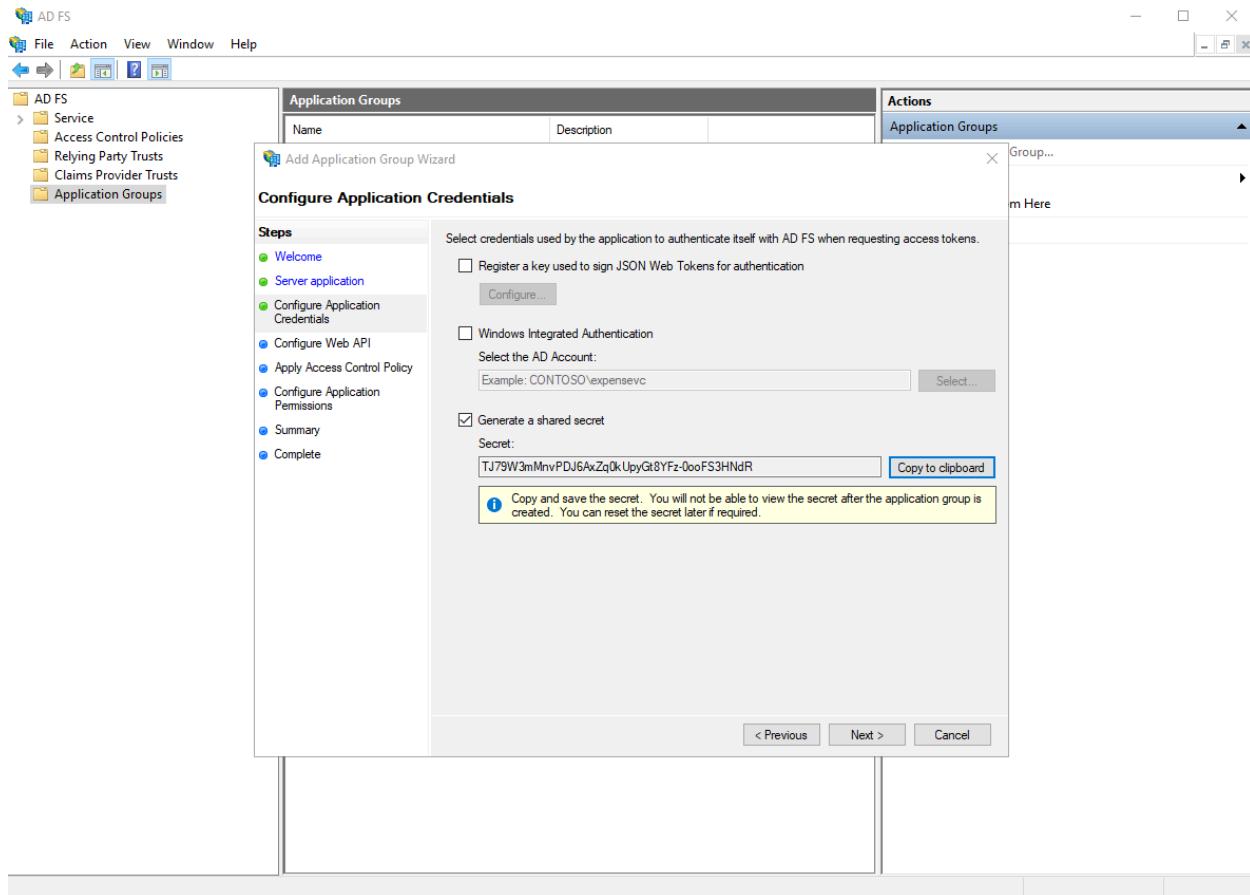
To set up Active Directory Federation Service, please refer to the [AD FS documentation](#) for a basic understanding of ADFS, and consult the [AD FS Deployment Guide](#) for instructions on setting up an AD FS server. Ensure that you have a fully operational AD FS server before proceeding to the next steps.

## Step 1: Enabling OAuth via AD FS

For detailed instructions on creating an app step by step, refer to the [Enabling OAuth Confidential Clients with AD FS](#) guide.

By the end of this step, you should have obtained a client ID and client secret as shown in the following screenshots:





The client identifier in the first picture and the secret in the second picture should be used as the client ID and client secret in the OAuth setup.

## Enabling Casdoor AD FS Provider

Add an AD FS provider and enter the "Client ID" and "Client Secret" in your Casdoor settings.

Edit Provider Save Save & Exit

Name ? :

Display name ? :

Category ? : OAuth

Type ? : Adfs ←

Client ID ? :

Client secret ? :   

Domain ? :

Provider URL ? : <https://openhome.alipay.com/platform/appManage.htm#/app/2021003111697088/overview>

Save Save & Exit

# Azure AD

## 介绍

Azure Active Directory (Azure AD) simplifies application management by providing a single identity system for cloud and on-premises applications. Software as a Service (SaaS) applications, on-premises applications, and Line of Business (LOB) applications can be added to Azure AD. 然后用户可以一次登录来安全和无缝地访问这些应用程序。 以及微软公司提供的办公室365项和其他商业应用程序。

## 如何使用？

注册应用程序的步骤如下所示。

### Step 1: Register an application

First, [register](#) an application and choose the account type as needed. 演示站使用下面显示的类型。

[Home](#) >

## Register an application

\* Name

The user-facing display name for this application (this can be changed later).

### Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (Default only - Single tenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- Personal Microsoft accounts only

[Help me choose...](#)

---

### Redirect URI (optional)

By proceeding, you agree to the [Microsoft Platform Policies](#) 

[Register](#)

## Step 2: Create a client secret

Create a `client secret` and save the value because it will be used later.

The screenshot shows the 'Certificates & secrets' section of the Casdoor application in the Azure portal. The left sidebar has 'Certificates & secrets' selected. The main area displays a table with one entry:

Description	Expires	Value	Secret ID
casdoor	1/8/2023	3Xr8Q~dFau2Hwyhg6y8Upb53PCFbuF... (redacted)	f3c7d37c-1def-4e29-b75f-457fa7c081e8

## Step 3: Add redirect URIs

Follow the example in the picture to add the redirect URIs for Casdoor.

The screenshot shows the 'Authentication' configuration page for the Casdoor application. The left sidebar has 'Authentication' selected. The right pane shows the 'Platform configurations' section, specifically the 'Redirect URLs' configuration. A red box highlights the 'Add a platform' button. The 'Supported account types' section indicates that accounts in any organizational directory (Any Azure AD directory - Multitenant) and accounts (e.g. Skype, Xbox) are supported. A warning message states: 'To change the supported accounts for an existing registration, use the manifest editor. Take properties may cause errors for personal accounts.' The 'Configure' button at the bottom right is highlighted with a red box.

## Step 4: Grant admin consent

`user.read` API 默认是打开的。 You can add more scopes according to your needs. 最后，记得 给予管理员权限。

The screenshot shows the Casdoor API permissions page. On the left, there's a sidebar with links like Overview, Quickstart, Integration assistant, Manage (with Branding & properties, Authentication, Certificates & secrets, Token configuration, and API permissions), Expose an API, App roles, Owners, Roles and administrators, Manifest, Support + Troubleshooting, Troubleshooting, and New support request. The 'API permissions' link is highlighted with a red box. The main content area has a success message: 'Successfully granted admin consent for the requested permissions.' It also includes a warning about the end of user consent for unverified publishers on November 9th, 2020, and a note about the 'Admin consent required' column. Below these are sections for 'Configured permissions' and 'Granted permissions'. The 'Configured permissions' section lists Microsoft Graph permissions: email, offline\_access, openid, profile, and User.Read. The 'Granted permissions' section shows five entries, each with a green checkmark and the text 'Granted for Default Dire...'. A red box highlights the 'Grant admin consent for Default Directory' button in the 'Configured permissions' section.

API / Permissions name	Type	Description	Admin consent requ...	Status
email	Delegated	View users' email address	No	Granted for Default Dire...
offline_access	Delegated	Maintain access to data you have given it access to	No	Granted for Default Dire...
openid	Delegated	Sign users in	No	Granted for Default Dire...
profile	Delegated	View users' basic profile	No	Granted for Default Dire...
User.Read	Delegated	Sign in and read user profile	No	Granted for Default Dire...

## Step 5: Create AzureAD provider in Casdoor

The last step is to add an AzureAD OAuth provider and fill in the `Client ID` and `Client Secret` in your Casdoor.

Edit Provider

Save

Save & Exit

Name ? : provider\_casdoor\_azuread

Display name ? : Casdoor AzureAD

Category ? : OAuth

Type ? : AzureAD

Client ID ? : 621cc0f0-055f-433f-9894-bfa1bfde169d

Client secret ? : \*\*\*

Provider URL ? : [https://portal.azure.com/#view/Microsoft\\_AAD\\_RegisteredApps/Applications列表](https://portal.azure.com/#view/Microsoft_AAD_RegisteredApps/Applications列表)

Save

Save & Exit

# Azure AD B2C

## Introduction

Azure AD B2C is a customer identity access management solution, supporting standards like OpenID Connect, OAuth 2.0, and SAML. It allows the integration of consumer-facing applications with a scalable and customizable identity management solution.

## How to use?

The steps to set up Azure AD B2C for authentication are shown below.

### Step 1: Create a B2C Tenant

First, create a B2C Tenant in your Azure portal.

### Step 2: Register an application

Register an application within your B2C tenant.

[Home](#) >

## Register an application

\* Name

The user-facing display name for this application (this can be changed later).

### Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (Default only - Single tenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- Personal Microsoft accounts only

[Help me choose...](#)

---

### Redirect URI (optional)

By proceeding, you agree to the [Microsoft Platform Policies](#) 

[Register](#)

## Step 3: Create a client secret

Create a `client secret` for your application and save the value as it will be used later.

casdoor | Certificates & secrets

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable location (using an HTTPS scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Application registration certificates, secrets and federated credentials can be found in the tabs below.

Description	Expires	Value	Secret ID
casdoor	1/8/2023	3Xr8Q~dFau2Hwyhg6y8Upb53PCFbuF... (red box)	f3c7d37c-1def-4e29-b75f-457fa7c081e8

## Step 4: Add redirect URIs

Add the redirect URIs for your application in the Azure AD B2C settings.

casdoor | Authentication

Platform configurations

The URLs we will accept as destinations when returning from the sign-in or sign-up flow after successfully authenticating or signing out us. The URL entered in the 'Redirect URIs' field must match the URL requested by the login server.

\* Redirect URIs

http://localhost:8000/callback

Front-channel logout URL

This is where we send a request to have the application log the user out. It is required for single sign-out to work correctly.

e.g. https://example.com/logout

Implicit grant and hybrid flows

Request a token directly from the authorization endpoint or from a single-page application (SPA) and doesn't use the auth code grant. If your application uses a SPA, select both access tokens and ID tokens. Other web apps that use hybrid authentication must obtain tokens.

Configure Cancel

## Step 5: Define User Flows

Define user flows in Azure AD B2C to manage how users sign up, sign in, and manage their profiles.

## Step 6: Create Azure AD B2C provider in Casdoor

Finally, add an Azure AD B2C OAuth provider in Casdoor, using the `Client ID` and `Client Secret` from your B2C tenant.

Edit Provider Save Save & Exit

Name ?: provider\_casdoor\_azuread

Display name ?: Casdoor AzureAD

Category ?: OAuth

Type ?: AzureAD

Client ID ?: 621cc0f0-055f-433f-9894-bfa1bfde169d

Client secret ?: \*\*\*

Provider URL ?: [https://portal.azure.com/#view/Microsoft\\_AAD\\_RegisteredApps/Applications](https://portal.azure.com/#view/Microsoft_AAD_RegisteredApps/Applications)

Save Save & Exit

# Custom OAuth

**i** 备注

Casdoor supports custom providers. However, the custom providers must follow the standard process of 3-legged OAuth, and the return values of `Token URL` and `Userinfo URL` must conform to the format specified by Casdoor.

To create a new custom provider, navigate to the provider page of Casdoor, and select “Custom” in the Type field. You will then need to fill in `Client ID`, `Client Secret`, `Auth URL`, `Scope`, `Token URL`, `Userinfo URL`, and `Favicon`.

Type [?](#) :

Custom

Auth URL [?](#)

<https://door.casdoor.com/login/oauth/authorize>

Scope [?](#)

openid profile email

Token URL [?](#)

[https://door.casdoor.com/api/login/oauth/access\\_token](https://door.casdoor.com/api/login/oauth/access_token)

Userinfo URL [?](#)

<https://door.casdoor.com/api/userinfo>

Favicon [?](#) :

URL [?](#) :



Preview:

Client ID [?](#)



Client secret [?](#)



- `Auth URL` 是自定义提供商的 OAuth 登录页面地址。

If you fill in `https://door.casdoor.com/login/oauth/authorize` as the `Auth URL`, then, when a user logs in with this custom provider, the browser will first redirect to

```
https://door.casdoor.com/login/oauth/  
authorize?client_id={ClientID}&redirect_uri=https://{{your-casdoor-  
hostname}}/callback&state={State_generated_by_Casdoor}&response_type=code&scope={Scope}`
```

授权完成后，自定义提供商应该重定向到

```
https://{{your-casdoor-hostname}}/callback?code={code}
```

After this step, Casdoor will recognize the code parameter in the URL.

- `Scope` is the scope parameter carried when accessing the `Auth URL`, and you should fill it in as per the custom provider's requirements.
- `Token URL` is the API endpoint for obtaining the accessToken.

Once you obtain the code in the previous step, Casdoor should use it to get the accessToken.

If you fill in `https://door.casdoor.com/api/login/oauth/access_token` as the `Token URL`, then Casdoor will access it using the following command

```
curl -X POST -u "{ClientID}:{ClientSecret}" --data-binary  
"code={code}&grant_type=authoritiation_code&redirect_uri=https://{{your-casdoor-  
hostname}}/callback" https://door.casdoor.com/api/login/oauth/access_token
```

The custom provider should return at least the following information:

```
{  
  "access_token": "eyJhbGciOiJSUzI1NiIsImtpZCI6IxXXXXXXXXXXXXXX",  
  "refresh_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IxXXXXXXXXXXXXXX",  
  "token_type": "Bearer",  
  "expires_in": 10080,  
  "scope": "openid profile email"  
}
```

- `UserInfo URL` is the API endpoint for obtaining user information via the accessToken.

If you fill in `https://door.casdoor.com/api/userinfo` as the `Userinfo URL`, then Casdoor will access it using the following command

```
curl -X GET -H "Authorization: Bearer {accessToken}" https://door.casdoor.com/api/userinfo
```

The custom provider should return at least the following information:

```
{  
  "name": "admin",  
  "preferred_username": "Admin",  
  "email": "admin@example.com",  
  "picture": "https://casbin.org/img/casbin.svg"  
}
```

- **Favicon** 是自定义提供商的标识URL。

This logo will be displayed on Casdoor's login page together with other third-party login providers.

# Okta

To set up the Okta OIDC provider, first visit [Okta Developer](#) and sign up to get a developer account.

Navigate to the Applications > Applications tab, click Create App Integration, select a Sign-in method of OIDC - OpenID Connect, and choose an Application type of Web Application, then click Next.

## Create a new app integration

X

### Sign-in method

[Learn More](#)

**OIDC - OpenID Connect**

Token-based OAuth 2.0 authentication for Single Sign-On (SSO) through API endpoints. Recommended if you intend to build a custom app integration with the Okta Sign-In Widget.

**SAML 2.0**

XML-based open standard for SSO. Use if the Identity Provider for your application only supports SAML.

**SWA - Secure Web Authentication**

Okta-specific SSO method. Use if your application doesn't support OIDC or SAML.

**API Services**

Interact with Okta APIs using the scoped OAuth 2.0 access tokens for machine-to-machine authentication.

### Application type

What kind of application are you trying to integrate with Okta?

Specifying an application type customizes your experience and provides the best configuration, SDK, and sample recommendations.

**Web Application**

Server-side applications where authentication and tokens are handled on the server (for example, Go, Java, ASP.Net, Node.js, PHP)

**Single-Page Application**

Single-page web applications that run in the browser where the client receives tokens (for example, Javascript, Angular, React, Vue)

**Native Application**

Desktop or mobile applications that run natively on a device and redirect users to a non-HTTP callback (for example, iOS, Android, React Native)

[Cancel](#)

[Next](#)

Enter the Sign-in redirect URIs, such as <https://door.casdoor.com/callback>.

### Sign-in redirect URIs

Okta sends the authentication response and ID token for the user's sign-in request to these URIs

[Learn More](#)

Allow wildcard \* in sign-in URI redirect.

X

[+ Add URI](#)

In the **Assignments** section, define the type of **Controlled access** for your app and then click **Save** to create the app integration.

Now you will have the Client ID, Client secret, and Okta domain.

### Client Credentials

[Edit](#)

Client ID	Ooa4we8u8iivyscpb5d7	<a href="#">Edit</a>
	Public identifier for the client that is required for all OAuth flows.	
Client secret	.....	<a href="#">Edit</a>
	Secret used by the client to exchange an authorization code for a token. This must be kept confidential! Do not include it in apps which cannot keep it secret, such as those running on a client.	

### General Settings

[Edit](#)

Okta domain	dev-53555475.okta.com	<a href="#">Edit</a>
-------------	-----------------------	----------------------

Add an Okta OAuth provider in the Casdoor dashboard by entering your Client ID, Client secret, and Domain.

Edit Provider Save Save & Exit

Name ? : provider\_casdoor\_okta

Display name ? : Casdoor Okta

Category ? : OAuth

Type ? : Okta

Client ID ? : Ooa4we8u8iivyscpb5d7

Client secret ? : \*\*\*

Domain ? : <https://dev-53555475.okta.com/oauth2/default>

Provider URL ? : <https://dev-53555475.okta.com>

Save Save & Exit

**!** 正确设置域名

Note that the `Domain` is not just the `Okta domain`; `/oauth2/default` should be appended to it.

在授权服务器上访问 [Okta 文档](#) 获取更多详情。

Now you can use Okta as a third-party service to complete authentication.

# Twitter

## Twitter (Work in Progress

Applying for a developer account on Twitter can be a bit cumbersome due to the strict official restrictions. It may be more challenging compared to other third-party platforms.

To get started, visit the [Developer Portal](#) and create an account if you don't have one. Twitter requires you to provide detailed information about your application for a developer account. Make sure to fill in the information accurately to avoid any issues during the review process.

Once your application is approved, you can proceed to create an application. You need to complete two important tasks in the **Authentication settings** section:

1. Manually enable 3-legged OAuth. This is necessary for features such as "Sign in with Twitter" and posting Tweets on behalf of other accounts.
2. Enable Request email address from users to obtain the user's email address.

Make sure to carefully fill in the callback address and other required information for your application.

# 微博

## Weibo ✓

Applying for a developer account with Weibo is not difficult, but the process can be slow, taking about 2-3 days.

To get started, visit the [Developer Website](#) and fill in the required basic information. Then, you will need to wait for a thorough review...

Once your application is approved, you will receive the Client Id and Client Secret.

# 微信

## 微信 ✓

若要将微信OAuth提供商添加到你的应用，请遵循以下步骤

1. 访问 [微信开放平台](#) 并注册成为开发者。
2. 在你的网站应用或移动应用获得批准后，您将得到您的 App ID 和 App Secret。

Name ⓘ: provider\_casdoor\_wechat

Display name ⓘ: Casdoor WeChat

Organization ⓘ: admin (Shared)

Category ⓘ: OAuth

Type ⓘ:  WeChat

Client ID ⓘ: wx049c70e6c2027b0b

Client secret ⓘ: \*\*\*

Client ID 2 ⓘ: wxe933a9cd81c396d1

Client secret 2 ⓘ: \*\*\*

服务器配置(已启用)

Use WeChat

Media Platform in PC ⓘ:  令牌(Token) 123

Access token ⓘ: 

Follow-up action: Use WeChat Open Platform to login | Use WeChat Media Platform to login

Provider URL ⓘ:  <https://open.weixin.qq.com/>

WeChat 提供商提供两套不同的密钥:

- 第一组密钥对 (`Client ID`, `Client Secret`) 是 `WeChat Open Platform` (微信开放平台) 的, 并且适用于PC登录场景。它允许你在PC浏览器显示二维码, 让用户能够使用微信APP扫描此二维码进行登录。
- 第二组密钥对(`Client ID 2`, `Client Secret 2`) 以及 `Access Token` 是 `WeChat Media Platform` (微信公众平台) 的, 目的是让用户能在微信APP内进行

登录。Access Token 是你在 WeChat Media Platform (微信公众平台) 的服务器配置中填写的 Token。它允许用户使用微信内部的浏览器进行登录，他会将用户重定向到你的 WeChat Official Account (微信公众号) 以实现登录。请注意：微信不支持在除微信APP之外的任何移动浏览器或者APP进行登录。这一限制是由 WeChat 而不是 Casdoor 施加的。

If you fill in the second keypair (Client ID 2, Client Secret 2), fill the Access Token field and enable the Enable QR code switch, then you can choose to login directly using the information from the WeChat Media Platform (微信公众平台) after scanning the QR code, or use the information from the WeChat Open Platform (微信开放平台) to login, if you choose use Wechat Open Platform to login, after user follow the WeChat official account (微信公众号), users will be required to scan the QR code of Wechat Open Platform (微信开放平台) to login. Casdoor will ask the user to follow the WeChat official account (微信公众号) before proceeding with the login process when the user clicks on the WeChat button to login. 值得注意的是，这只能在PC登录场景中使用，因为手机无法自行扫描二维码。当在移动场景中使用时(即WeChat App的内置浏览器)，Casdoor 将自动跳过这一步。

::: 提示

我们建议同时设置两套密钥对用来在 WeChat Open Platform (微信开放平台) 内部连接你的 WeChat Open Platform (微信开放平台) 账号与 WeChat Media Platform (微信公众平台) 账号。从而让 Casdoor 将通过 PC 和手机登录的用户视为同一用户。

:::

::: 注意

由于 WeChat OAuth 的限制，目前没有任何方法通过微信登录除微信APP以外的任何第三方手机APP或移动浏览器。目前移动端登录只能在微信APP中进行。

:::

更多细节请浏览 [微信开放平台](#)。

# 企业微信

## 介绍

WeCom provides an authorized login method using OAuth, which allows you to obtain members' identity information directly from the webpage opened by the WeCom terminal, eliminating the need for a login process.

There are two types of applications: internal applications and third-party applications.

## Basic Configuration

To configure a WeCom provider, you need to provide the following parameters:

参数描述:

参数	描述
Sub type	内部或第三方
Method	静默或正常模式
Client ID	The enterprise CorpID
Client secret	The enterprise CorpSecret

参数	描述
Agent ID	应用程序Agentid

### ① 信息

WeCom supports two authorization methods: Silent authorization and normal authorization.

**Silent authorization:** After the user clicks the link, the page is redirected to  
`redirect_URI? code=CODE&state=STATE`

**Normal authorization:** After the user clicks the link, a middle page is displayed for the user to choose whether to authorize or not. After the user confirms the authorization, they are redirected to

`redirect_uri?code=CODE&state=STATE`

For more details, please refer to the [official documentation](#).

## More Information

For more information about internal applications, please refer to the [Internal Application](#) documentation.

For information about third-party applications, please refer to the [Third-Party Application](#) documentation.

# 腾讯 QQ

## Tencent QQ ✓

To add Tencent QQ OAuth provider to your application, visit the authentication platform of QQ - [Connect QQ](#).

First, you need to apply to [become a developer](#). After your application is approved, follow the instructions of the platform to obtain your Client Id and Client Secret.

# 钉钉

## DingTalk ✓

### Configuring DingTalk

To configure DingTalk, visit the [DingTalk developer platform](#) and log in using your DingTalk account. Once you're on the platform, follow the instructions provided to obtain your `Client ID` and `Client Secret`. The corresponding terms in DingTalk are as follows:

Term	DingTalk Name
Client ID	AppKey
Client secret	AppSecret

In DingTalk, you can find the `Appkey` and `AppSecret` in the App Info.

基础信息

应用信息

开发管理

权限管理

应用功能

机器人与消息推送

事件与回调

登录与分享

酷应用

安全与监控

监控中心

部署与发布

版本管理与发布

## 应用信息



casdoor

document

## 应用凭证

AgentId

2687194261

AppKey

ding6dposo0nm8u4t2g5

AppSecret

hE4cwQ4PjKDSp\_uCHTBTqjAAfZfsNGkxwNg1q1FCiiTRW7apxJhzjFOjw46NfFWn

## 删除应用

删除操作不可逆，该应用所有信息将被删除，请谨慎操作。

删除

Make sure to add the **Redirect Domain**, which should be your Casdoor domain.

基础信息

应用信息

开发管理

权限管理

应用功能

机器人与消息推送

事件与回调

登录与分享

酷应用

安全与监控

监控中心

部署与发布

## 接入登录

添加重定向 URL 作为免登授权码跳转地址。[了解更多](#)

\* 回调域名

请填写 HTTP/HTTPS 开头的 URL

添加

微应用回调的URL

http://localhost:7001

生成

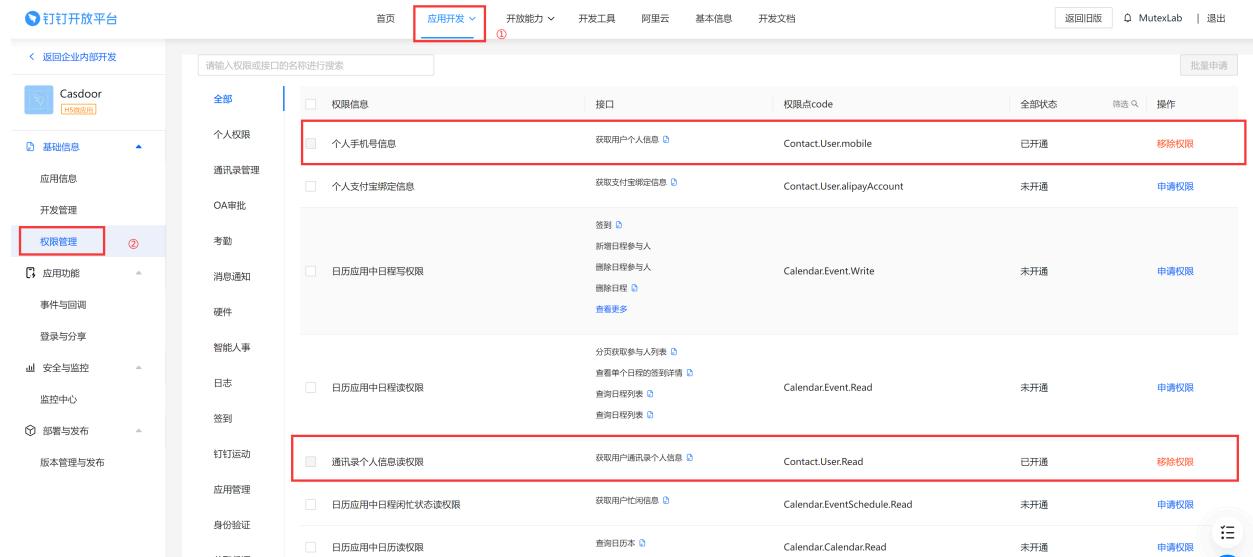
## 接入分享

嵌入分享SDK，实现一键登录后内容分享。[了解更多](#)

iOS 分享

For more detailed information, please refer to the [DingTalk developer docs](#).

Additionally, you need to add the following permissions to the DingTalk application:



The screenshot shows the DingTalk Open Platform interface with the '应用开发' (Application Development) tab selected. In the '权限管理' (Permission Management) section, two specific permissions are highlighted with red boxes:

- 个人手机号信息**: Contact.User.mobile (已开通, 移除权限)
- 通讯录个人信息读权限**: Contact.User.Read (已开通, 移除权限)

## Configuring Casdoor

Here's the final configuration for DingTalk:

Name <a href="#">?</a> :	dingding
Display name <a href="#">?</a> :	dingding
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	OAuth
Type <a href="#">?</a> :	 DingTalk
Client ID <a href="#">?</a> :	ding6dposoonm8u4t2g5
Client secret <a href="#">?</a> :	***
Provider URL <a href="#">?</a> :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>

# Steam

## Steam ✓

To add the Steam OAuth provider to your application, follow these steps:

1. Visit the [Steam WebAPI platform](#) and log in using your Steam account.
2. Apply for an API Key for your Casdoor domain or IP.
3. Fill in your API Key as the Client Secret into Casdoor. The ClientID does not need to be filled.
4. Make sure that your Steam account has games in order to apply for the API.

For more detailed information, please visit the [Steam WebAPI documentation](#).

# Gitee

To set up the Gitee OAuth provider, please go to the [Gitee developer website](#). If you haven't created applications before, the Gitee workbench will look like this:

The screenshot shows the Gitee developer workbench interface. At the top, there is a navigation bar with links for '企业版' (Enterprise), '高校版' (University Edition), '私有云' (Private Cloud), '博客' (Blog), and '我的' (My). On the right side of the header is a search bar labeled '搜开源' (Search Open Source) and a series of icons for notifications, location, and account management. Below the header, there are two tabs: '我的应用' (My Applications) which is selected and highlighted in orange, and '已授权应用' (Authorized Applications). Under the '我的应用' tab, there is a message '无数据' (No data) with a small circular icon containing a minus sign.

You can then create your Gitee app.

## 创建第三方应用

应用名称 \*

应用名称

应用描述

应用描述

应用主页 \*

你的应用主页

应用回调地址 \* +

用户授权后，重定向的地址，例如: <https://gitee.com/login>

Enter the name, description, homepage, and callback URL, and carefully choose the permissions.

### ⚠ SET THE AUTHORIZATION CALLBACK URL CORRECTLY

In the Gitee OAuth config, the `authorization callback URL` must be your Casdoor's callback URL, and the `Redirect URL` in Casdoor should be your application's callback URL.

For more details, please read the [App config](#) guide.

After creating the Gitee app, you can obtain the `Client ID` and `Client Secrets`!

### Casdoor (今日请求次数: 0 次)

#### 应用名称 \*

Casdoor

#### Client ID

300ff94d994a7597850bbafb2d5dc67929676dd8e7176b029e067dc6966ef9c4

#### Client Secret

60be2e4e0f3fb8286cfe9f129ab0c3d6b40718a964dade150a8095eb2748730c

[重置 Client Secret](#)

[移除已授权用户的有效 Token](#)

Add a Gitee OAuth provider and enter the `Client ID` and `Client Secrets` in your Casdoor.

Edit Provider

Save

Name ③ :

my\_gitee\_provider

Display name ③ :

Gitee provider

Category ③ :

OAuth

Type ③ :

Gitee

Client ID ③ :

300ff94d994a7597850bbafb2d5dc67929676dd8e7176b029e067dc6966ef9c4

Client secret ③ :

\*\*\*\*\*

Now you can use Gitee as a third-party service to complete authentication!

### ⚠ 注意事项

Since Casdoor needs to obtain the user's email, the email option must be checked; otherwise, it will cause scope authorization errors.

Permissions (Be careful to select scopes, users might deny authorization when there are too many scopes.)

All

- |   |  |
|---|--|
| <input checked="" type="checkbox"/> user_info | Access and update user data, activities, etc |
| <input type="checkbox"/> projects             | Full control of user projects                |
| <input type="checkbox"/> pull_requests        | Full control of user pull requests           |
| <input type="checkbox"/> issues               | Full control of user issues                  |
| <input type="checkbox"/> notes                | Access, create and edit user comments        |
| <input type="checkbox"/> keys                 | Full control of user public keys             |
| <input type="checkbox"/> hook                 | Full control of user webhook                 |
| <input type="checkbox"/> groups               | Full control of user orgs and teams          |
| <input type="checkbox"/> gists                | Access, create and update user gists         |
| <input type="checkbox"/> enterprises          | Full control of user enterprises and teams   |
| <input checked="" type="checkbox"/> emails    | Access user emails data                      |

Submit

Delete

# 百度

To set up the Baidu OAuth provider, please read the [Baidu documentation](#) and follow their steps to complete the [application creation](#).

**开发者服务管理**

📍 提示:  
轻应用平台不再支持创建直达号，如需开通直达号请登录<http://zhida.baidu.com>

**创建工作**

\* 应用名称: CasdoorTest 11/32

传统接入扩展:  合作网站

解决方案:  使用BAE

**创建**

After creating your app, the redirect URL should be set in the following position:

**Casdoor**

**基本信息**

接入类型 —————

其他应用

开发者服务 ————— ✖

Oauth2.0

**安全设置**

**基本信息**

	名称: Casdoor
Icon:	
ID:	25547043
API Key:	Hn'██████████yQmAp61

在以下位置添加您的 Casdoor 域名：

The screenshot shows the Casdoor web interface. On the left sidebar, there are several tabs: '基本信息' (Basic Information), '接入类型' (Access Type), '其他应用' (Other Applications), '开发者服务' (Developer Services), 'Oauth2.0', and '安全设置' (Security Settings). The '安全设置' tab is selected and shows the 'Implicit Grant 授权方式' (Authorization Grant Type) set to '禁用' (Disabled). In the 'Root Domain Binding' section, the domain 'door.casbin.com' is listed with a note below it: '限制访问OpenAPI的Referer' (Restrict OpenAPI Referer access). A red box highlights this section. Below it is a section for '应用服务器IP地址' (Application Server IP Address) with a note: '可以同时将应用访问OpenAPI（如 Passport、翻译等API）的IP限制在所填的“应用服务器IP地址”设置项中' (You can simultaneously restrict the IP of application OpenAPI access (such as Passport, Translation API) in the "Application Server IP Address" setting item). At the bottom are '确定' (Confirm) and '取消' (Cancel) buttons.

### ⚠ 注意事项

This part is very different from the information provided in the Baidu documentation:

1. Adding the URL to the callback URL setting will most likely fail to validate the URL and cause the login to fail, so we add our domain name to the domain setting.
2. Only one URL or domain name can be added, which is very different from the documentation.

Then you can obtain the `Client ID` and `Client Secrets`.

# Casdoor

- 基本信息
- 接入类型
- 其他应用
- 开发者服务
- Oauth2.0
- 安全设置

## 基本信息

名称: Casdoor

Icon:

ID:

Client ID API Key: HnhK7...QmAp61

Client Secret Secret Key: DTgBZ...ls1bLm1Gha

创建时间: 2022-01-22 16:20:05

更新时间: 2022-01-23 15:45:06

Add a Baidu OAuth provider and fill in the `Client ID` and `Client Secrets` in your Casdoor.

casbin

- Home
- Organizations
- Users
- Roles
- Permissions
- Providers
- Applications
- Resources
- Tokens

Edit Provider Save Save & Exit

Name ②: Baidu

Display name ②: Baidu

Category ②: OAuth

Type ②: Baidu

Client ID ② HsM...nWT

Client secret ② \*\*\*

Provider URL ②: <https://github.com/organizations/xxx/settings/applications/1234567>

Save Save & Exit

Now you can use Baidu as a third-party service to complete authentication!

### ① 一般故障排除

If you encounter a Baidu prompt that states your redirect URL is incorrect, here are some ways you might be able to fix it:

1. Add your domain name to the appropriate location and then reset the Secret (Baidu reset Secret has a bug, it will prompt you an error, but after refreshing the page the Secret has been refreshed).
2. 如果以上方法都不能解决问题，我们建议您删除应用程序并创建一个新的应用程序，并先设置您的域名。

Another problem is that the user name returned by Baidu is masked, unlike their documentation which shows the user name and displayed name. Therefore, we can currently only use the masked name as the user name.

# Infoflow

To set up the Infoflow OAuth provider, please follow these steps:

1. Go to [Infoflow{:target="\\_blank"} and log in using your Infoflow account.](#)
2. Visit the [Infoflow Application{:target="\\_blank"} page.](#)

如流 Infoflow

首页 通讯录 应用中心 数据统计 设置

应用(5)

新建应用 应用分组/排序 应用宣传栏

3. Register your Infoflow app.

返回 Casdoor 保存 取消

基本信息

应用logo:

建议使用640\*640, 2M以内的jpg、png图片

应用名称: Casdoor

应用介绍: Casdoor单点登录系统

功能:  应用 (在客户端应用面板中, 为用户提供访问内部系统的入口, [查看客户端示例](#))  
 机器人 (在企业群聊中, 为用户提供机器人服务, [查看客户端示例](#))  
 服务号 (以双人会话方式, 为用户提供交流服务, [查看客户端示例](#))

4. Obtain the [AgentID](#).

< 返回

Casdoor

I 基本信息

应用logo:



应用名称: Casdoor

应用介绍: Casdoor单点登录系统

功能: 应用

AgentID

应用ID: 55

5. Navigate to the Setting tab and create a new management group.

如流 Infoway

首页 通讯录 应用中心 数据统计 设置 ①

基本信息 成员加入 权限设置 ② 通讯录设置 安全设置 客户端启动页

系统管理组 ③ 管理员 新建下级管理组 ④

普通管理组 暂未设置管理员

通讯录权限 组织架构

6. Add your structure to the address book permissions and give it the necessary permissions. Also, add the application you just created to the specified location.

### 通讯录权限

[修改](#)

组织架构	查看	管理
	<input checked="" type="checkbox"/>	<input type="checkbox"/>

对部门仅有查看权限时，只可查看被授权的成员资料信息；对部门有管理权限时，可查看成员的所有资料信息

<input checked="" type="checkbox"/> 成员ID	<input checked="" type="checkbox"/> 姓名	<input checked="" type="checkbox"/> 部门	<input checked="" type="checkbox"/> 头像
<input checked="" type="checkbox"/> 手机号	<input checked="" type="checkbox"/> 邮箱	<input checked="" type="checkbox"/> 登录帐号	

### 应用权限

[修改](#)

应用权限	发消息	配置应用
Casdoor	<input checked="" type="radio"/>	<input type="radio"/>

7. Add the sensitive interface permissions as shown.

## 敏感接口权限

修改

接口名称	权限开放
获取部门成员	<input checked="" type="checkbox"/>
获取部门列表	<input type="checkbox"/>
获取成员信息	<input checked="" type="checkbox"/>
获取标签成员	<input type="checkbox"/>
维护通信录	<input type="checkbox"/>
获取成员群组列表	<input type="checkbox"/>
获取群组成员列表	<input type="checkbox"/>
维护群组成员	<input type="checkbox"/>
发送群组消息	<input type="checkbox"/>
维护群组话题	<input type="checkbox"/>
维护勋章	<input type="checkbox"/>
通讯录搜索	<input type="checkbox"/>

8. On the same page, you will find the `CorpID` and `Secret`.

## 开发者凭据

### Client ID

CorpID	hir...1
Secret	HgH...NB
Client Secret	<a href="#">重置</a>

9. Add an Infoflow OAuth provider to Casdoor and fill in the `Client ID`, `Client Secret`, and `Agent ID`.

Edit Provider [Save](#) [Save & Exit](#)

Name ② :	Infoflow
Display name ② :	Infoflow
Category ② :	OAuth
Type ② :	Infoflow
Sub type ② :	Internal
Client ID ②	<input type="text"/> CorpID
Client secret ②	<input type="text"/> Secret
Agent ID ②	<input type="text"/> AgentID

You can now use Infoflow as a third-party service for authentication.

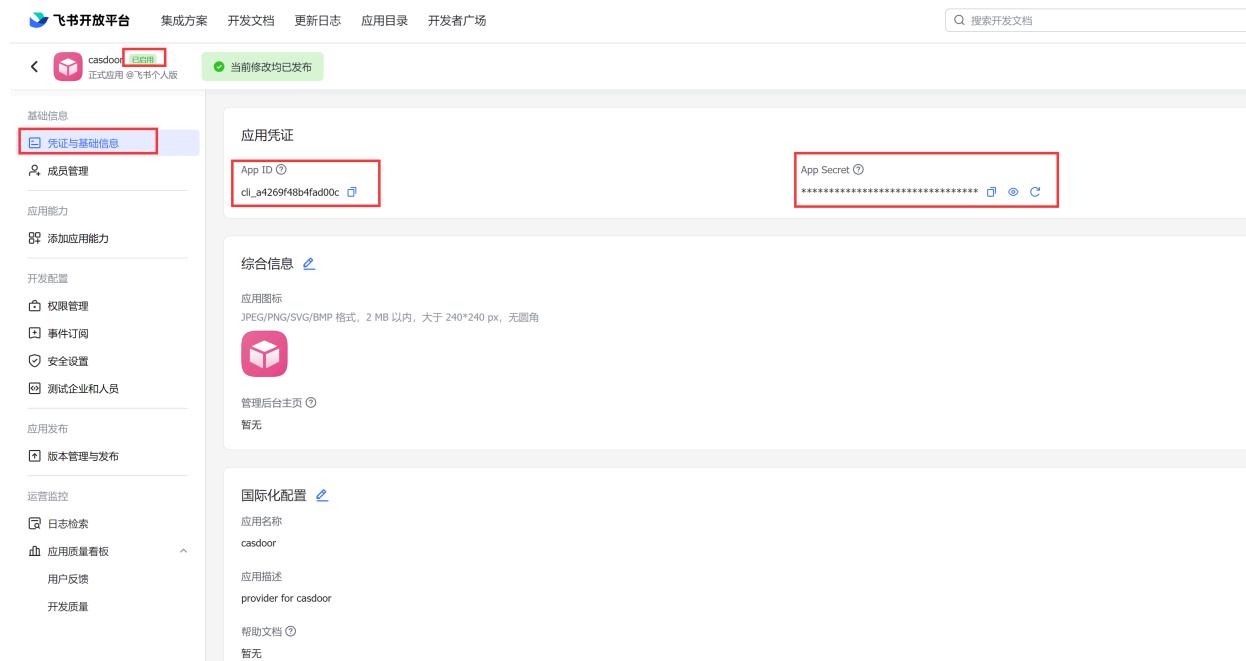
# Lark

## ⓘ 备注

This is an example of how to configure a Lark OAuth provider.

## Step 1: Create a Lark application

First, you need to create a new application on the [Lark Open Platform](#) and enable it. You can find the `App ID` and `App Secret` in the basic information of your application.



The screenshot shows the Lark Open Platform application configuration interface. On the left is a sidebar with various settings like Basic Information, Application Abilities, Development Configuration, and Operation Monitoring. The main area shows the application details for 'casdoor'. The 'Basic Information' tab is selected, displaying the '凭证与基础信息' section. Within this section, the '应用凭证' (Application Credentials) part is highlighted, showing the 'App ID' field containing 'cli\_a4269f48b4fad00c' and the 'App Secret' field containing a redacted string. Below this, the '综合信息' (Comprehensive Information) and '国际化配置' (Internationalization Configuration) sections are partially visible.

Next, add the redirect URL `<your-casdoor-domain>/callback` (e.g., <http://localhost:7001/callback>) in the security settings of your application.

The screenshot shows the Casdoor application settings interface. On the left sidebar, under the '安全设置' (Security Settings) section, there is a red box highlighting the '重定向 URL' (Redirection URL) and 'IP 白名单' (IP Whitelist) sections. The '重定向 URL' section contains a single entry: 'http://localhost:7001/callback'. The 'IP 白名单' section is currently empty. At the bottom right of the main content area, there is a blue button labeled '技术支持' (Technical Support) with a gear icon.

## Step 2: Create a Lark OAuth provider

Now create a Lark OAuth provider in Casdoor. Fill in the necessary information.

Name	Name in Lark
Category	Choose <button>OAuth</button>
Type	Choose <button>Lark</button>
Client ID	<input type="text"/> App ID obtained from Step 1
Client secret	<input type="text"/> App Secret obtained from Step 1

The image displays two side-by-side screenshots from the Feishu Open Platform developer console.

**Left Screenshot (Provider Configuration):**

- Name: lark
- Display name: lark-display
- Organization: admin (Shared)
- Category: OAuth
- Type: Lark
- Client ID: cli\_a4269f48b4fad00c (highlighted with a red box)
- Client secret: \*\*\* (highlighted with a red box)
- Provider URL: [View](#)

**Right Screenshot (Application Settings):**

- 基础信息:
  - 应用凭证:
    - App ID: cli\_a4269f48b4fad00c (highlighted with a red box)
    - App Secret: (redacted) (highlighted with a red box)
- 综合信息:
  - 应用图标: (Image placeholder)
  - 管理后台主页: 新建
- 国际化配置:
  - 应用名称: casdoor
  - 应用描述: provider for casdoor
  - 帮助文档: 新建

Now you can use Lark as the third-party service to complete authentication.

# 电子邮箱

## Overview

Using Email for authentication

## SendGrid

Using SendGrid as the SMTP server

## Azure ACS

Using Azure ACS as the email provider

## Brevo

使用 Brevo 作为SMTP 服务器

 MailHog

Using MailHog as the SMTP server

# Overview

## Adding an Email provider

1. Click on [Add](#) to add a new provider.
2. Select [Email](#) under the [Category](#) section.

Name <a href="#">?</a> :	<input type="text" value="email provider"/>
Display name <a href="#">?</a> :	<input type="text" value="My Email"/>
Category <a href="#">?</a> :	<input type="text" value="Email"/>
Type <a href="#">?</a> :	<input type="text" value="Default"/>

3. Fill in the fields for [Username](#), [Password](#), [Host](#), and [Port](#) for your SMTP service.

Username <a href="#">?</a>	no-reply@casbin.com
Password <a href="#">?</a>	***
Host <a href="#">?</a> :	smtp.qiye.aliyun.com
Port <a href="#">?</a> :	465

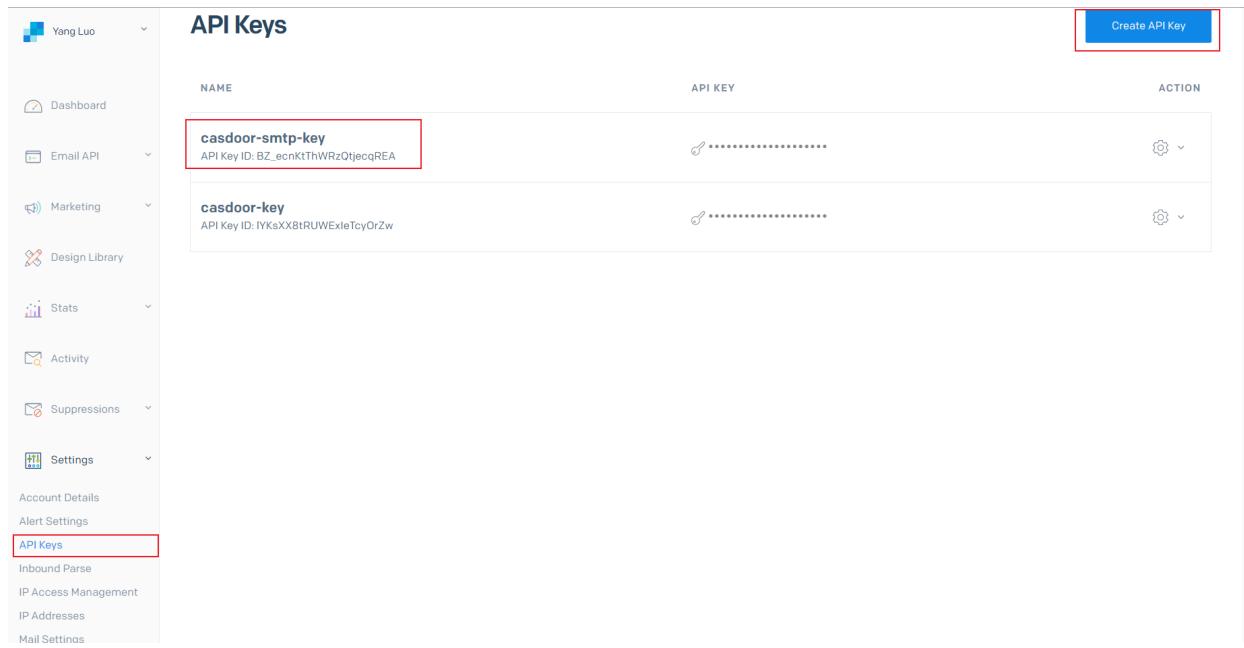
4. Customize the [Email Title](#) and [Email Content](#), then save the changes.

# SendGrid

In this guide, we will be using SendGrid as the SMTP server.

## Step 1: Create API key for your SendGrid account

Expand the **Settings** from the left navigation bar, click on the **API Keys** option from this list. Here, you will see all of your API keys if you have generated any in the past. To generate a new one, you need to click on **Create API Key** and pay attention to the permissions.



The screenshot shows the SendGrid dashboard with the 'API Keys' section selected. On the left, the 'Settings' menu is expanded, and 'API Keys' is highlighted with a red box. The main table lists two API keys:

NAME	API KEY	ACTION
casdoor-smtp-key API Key ID: BZ_ecnKThWRzQtjecqREA	copy ······	⚙️
casdoor-key API Key ID: IYksXX8tRUWExleTcyOrZw	copy ······	⚙️

A red box highlights the 'Create API Key' button in the top right corner of the table area.

## Step 2: Sender Verification

Refer to the document to verify your email sender, you can choose **Single Sender Verification** or **Domain Authentication**: [Sender Identity](#)

## Step 3: Configure Casdoor email Provider

Now create an email provider in Casdoor. Fill in the required fields below:

Required fields	Remark
Username	Enter "apikey"
Password	Your SendGrid's API key
From Address	Your verified sender
Host	Enter "smtp.sendgrid.net"
Port	Default is 465

Name ⓘ :	sendgrid
Display name ⓘ :	sendgrid
Organization ⓘ :	admin (Shared)
Category ⓘ :	Email
Type ⓘ :	 Default
Username ⓘ :	apikey
Password ⓘ :	***
From address ⓘ :	notifications@casbin.com
From name ⓘ :	casdoor
Host ⓘ :	 smtp.sendgrid.net
Port ⓘ :	465
Disable SSL ⓘ :	<input checked="" type="checkbox"/>
Email title ⓘ :	Casdoor Verification Code
Email content ⓘ :	You have requested a verification code at Casdoor. Here is your code: %s, please enter in 5 minutes.
Test Email ⓘ :	1270329076@qq.com
	<button>Test SMTP Connection</button>
	<button>Send Testing Email</button>

Click on the **Test SMTP Connection** button. If you see **provider: SMTP connected successfully**, it means that your Casdoor service can access the SendGrid service.

Next, click on the **Send Testing Email** button. If you see **Email sent successfully**, it means that the test email has been sent successfully from the **From** address to the **Test Email**.

# Azure ACS

In this guide, we will be using ACS as the Email Provider.

## Step 1: Config ACS

Follow the documentation below, complete configuration.

- [Create and manage Email Communication Service](#)
- [Get a free Azure managed domain](#) or [Add a custom domain](#)
- [Connect domain](#)

Copy your `Endpoint` and `Private Key` for usage

Home > casdoor

**casdoor | Keys** ⋮

Communication Service

Search

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Sample applications

Events

Settings

Keys

Identities & User Access Tokens

Push notifications

Identity

Properties

Locks

Telephony and SMS

Endpoint: https://casdoor.unitedstates.communication.azure.com/

Primary key

Regenerate primary key Show values

Key: [REDACTED]

Connection string: [REDACTED]

Secondary key

Regenerate secondary key Show values

Key: [REDACTED]

Connection string: [REDACTED]

Use access keys to authorize your API calls when you use the Communication Services SDKs. Store your access keys securely – for example, using Azure Key Vault – and don't share them. We recommend regenerating your access keys regularly. You are provided two access keys so that you can maintain connects using one key while regenerating the other.

## Step 2: Configure Casdoor email Provider

Now create an email provider in Casdoor, fill in the necessary information. The

relationship between the fields and Azure ACS is as follows:

ⓘ 备注

From Address must be a verified email domain.

Name	Name in Azure ACS
From Address	
Secret key	Private Key
Host	Endpoint

Name ⓘ : acs

Display name ⓘ : acs

Organization ⓘ : admin (Shared)

Category ⓘ : Email

Type ⓘ :  Azure ACS

Secret key ⓘ : \*\*\*

From address ⓘ : donotreply@59e2329a-a0c8-491f-b470-6d3c044fd8cf.azurecomm.net

Host ⓘ :  https://casdoor.unitedstates.communication.azure.com

Email title ⓘ : Casdoor Verification Code

Email content ⓘ : You have requested a verification code at Casdoor. Here is your code: %s, please enter in 5 minutes.

Test Email ⓘ : 1270329076@qq.com

Provider URL ⓘ :  https://github.com/organizations/xxx/settings/applications/1234567

# Brevo

在本指南中，我们将使用 Brevo 作为SMTP 服务器。

## 第 1 步：请求激活您的 Brevo SMTP 帐户

请参阅文档以激活 Brevo SMTP: [发送交易电子邮件使用Brevo SMTP](#)

在我的案例下，当我创建一个工单来激活我的smtp帐户时，我得到了这个答复：



Rafael Guimaraes

Last Response: 2 days ago



Hi there,

Thank you for reaching out!

I've activated your account's SMTP/Transactional capabilities.

You can find your account's SMTP credentials by clicking [here](#).

To get you started, I'll include some useful links about our SMTP/Transactional services:

- [Our complete library of help articles related to SMTP](#)
- [Troubleshooting common issues with SMTP](#)

The SMTP port listed by default, Port 587, will be used without a secure connection. If you want to use a secure connection (SSL or TLS), please use Port 465.

Please be sure to let me know if you have more questions or if you need any help.

Best regards,

## 步骤 2: 获取 Brevo SMTP 配置

在你的 Brevo 看板中, 找到 SMTP&API, 查看 **SMTP 服务器**, **端口**, **登录**, **SMTP 验证 Key** 信息

## 第 3 步：配置 Casdoor 邮件提供商

在 Casdoor 中创建一个新的提供商。 填写必要的信息.

单击 **测试SMTP连接** 按钮。 如果您看到 **提供商：SMTP连接成功**, 这意味着您的 Casdoor 服务可以访问 Brevo 服务。

接下来，点击 **发送测试电子邮件** 按钮。 如果你看到**邮件发送成功**，意味着测试邮件已经成功的从**发件地址** 发送至 **测试接收邮箱**

# MailHog

In this guide, we will be using MailHog as the SMTP server. [MailHog](#) is an email-testing tool that operates with a fake SMTP server.

## Step 1: Deploy the MailHog service

The IP address for the MailHog service is `192.168.24.128`, and the SMTP service port is `1025`.

```
[HTTP] Binding to address: 0.0.0.0:8025
2023/07/13 03:06:43 Serving under http://0.0.0.0:8025/
Creating API v1 with WebPath:
Creating API v2 with WebPath:
[APIv1] KEEPALIVE /api/v1/events
[HTTP] Binding to address: 0.0.0.0:8025
Creating API v1 with WebPath:
Creating API v2 with WebPath:
2023/07/13 03:10:36 Using maildir message storage
2023/07/13 03:10:36 Maildir path is /tmp/mailhog641072855
2023/07/13 03:10:36 [SMTP] Binding to address: 0.0.0.0:1025
2023/07/13 03:10:36 Serving under http://0.0.0.0:8025/
[APIv2] GET /api/v2/jim
[APIv2] GET /api/v2/messages
2023/07/13 03:10:50 [HTTP] Connection closed by client: 127.0.0.1:641072855
```

## Step 2: Create an email provider

Provide the necessary information and save the settings.

Category <a href="#">?</a> :	Email
Type <a href="#">?</a> :	Default
Username <a href="#">?</a> :	
Password <a href="#">?</a> :	
From address <a href="#">?</a> :	notification@casdoor.com
From name <a href="#">?</a> :	Casdoor Notification
Host <a href="#">?</a> :	192.168.24.128
Port <a href="#">?</a> :	1025
Disable SSL <a href="#">?</a> :	<input checked="" type="checkbox"/>
Email title <a href="#">?</a> :	Casdoor Verification Code (Test)
Email content <a href="#">?</a> :	You have requested a verification code at <a href="#">Casdoor (Test)</a> . Here is your code: <u>%s</u> , please enter in 5 minutes.
Test Email <a href="#">?</a> :	admin@example.com
	<a href="#">Test SMTP Connection</a>
	<a href="#">Send Testing Email</a>
<a href="#">Provider URL <a href="#">?</a>:</a>	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>
<a href="#">Save</a>	<a href="#">Save &amp; Exit</a>

## Step 3: Send a test email

First, click on the [Test SMTP Connection](#) button. If you see [provider: SMTP connected successfully](#), it means that your Casdoor service can access the MailHog service.

Next, click on the [Send Testing Email](#) button. If you see [Email sent successfully](#), it means that the test email has been sent successfully from the [From](#) address to the [Test Email](#).

Name ?: email\_provider provider:SMTP connected successfully

Display name ?: Email Provider Email sent successfully

Organization ?: admin (Shared)

Category ?: Email

Type ?: Default

Username ?:

Password ?:

From address ?: notification@casdoor.com

From name ?: Casdoor Notification

Host ?: 192.168.24.128

Port ?: 1025

Disable SSL ?:

Email title ?: Casdoor Verification Code (Test)

Email content ?: You have requested a verification code at Casdoor (Test). Here is your code: 123456, please enter in 5 minutes.

Test Email ?: admin@example.com **Test SMTP Connection** **Send Testing Email**

Provider URL ?: <https://github.com/organizations/xxx/settings/applications/1234567>

 MailHog

Connected

Inbox (4)

Delete all messages

**Jim**  
Jim is a chaos monkey.  
[Find out more at GitHub.](#)

Enable Jim

**Message Preview:**

From: "Casdoor Notification" <notification@casdoor.com>  
 Subject: Casdoor Verification Code (Test)  
 To: admin@example.com

HTML Plain text Source

You have requested a verification code at Casdoor (Test). Here is your code: 123456, please enter in 5 minutes.

# 短信

## Overview

Using SMS for authentication

## Twilio

Using Twilio as an SMS provider for Casdoor

## Amazon SNS

Using Amazon SNS as an SMS provider for Casdoor

## Azure ACS

Using ACS as an SMS provider for Casdoor

 Alibaba Cloud

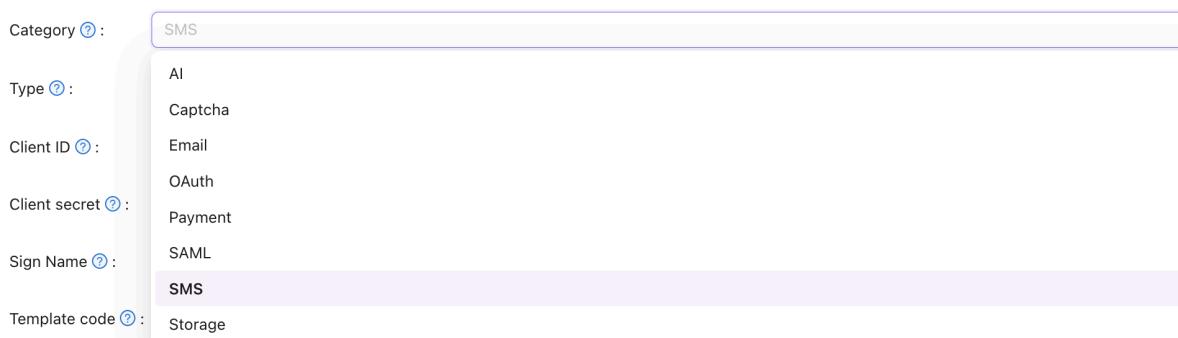
Using Alibaba Cloud as an SMS provider for Casdoor

# Overview

We use [casdoor/go-sms-sender](#) to send SMS for Casdoor. The [go-sms-sender](#) library currently supports Twilio, Submail, SmsBao, Alibaba Cloud, Tencent Cloud, Huawei Cloud, and Volc SMS APIs. If you want to add support for other SMS providers, you can either raise an issue or submit a pull request.

## Adding an SMS provider

1. Click on [Add](#) to add a new provider.
2. Select [SMS](#) in the [Category](#) section.



3. Choose the type of your provider.

Category  : SMS

Type  : Aliyun SMS

Client ID  : Aliyun SMS  
Huawei Cloud SMS

Client secret  : SmsBao SMS  
SUBMAIL SMS

Sign Name  : Tencent Cloud SMS  
Twilio SMS

Template code  : Volc Engine SMS

4. Retrieve the necessary information from your SMS provider and fill out the corresponding fields.

# Twilio

## Fill in the necessary information in Casdoor

There are four required fields: `Client ID`, `Client secret`, `Sender number`, and `Template code`. The corresponding relationship to the Twilio account is as follows:

Name	Name in Twilio	Required
Client ID	Account SID	Required
Client secret	Auth Token	Required
Sender number	Twilio phone number	Required
Template code		Required

## Twilio information

- Account SID, Auth Token, and Twilio phone number

**Step 4: invite and upgrade**

**Develop Monitor**

**Invite teammates**  
Invite developers to your Twilio account to start building! [Learn more about user access management](#)

**Upgrade your account**  
Upgrade your account to send to any number, buy local numbers, and more. [Learn more about trial account limitations](#)

**Account Info**

- Account SID**: AC06b73d65c8ee67ce8e448edcc64b6ec6
- Auth Token**: ..... [Show](#)
- My Twilio phone number**: +12186751069

**Helpful links**

- [How does Twilio work?](#)
- [SMS Quickstart guides](#)
- [Support help center](#)

## Configure Casdoor provider

You can configure the `template code` to suit your requirements, and then enter your phone number in `SMS Test` to test.

Name <a href="#">?</a> :	twilio
Display name <a href="#">?</a> :	twilio
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	SMS
Type <a href="#">?</a> :	Twilio SMS
Client ID <a href="#">?</a> :	AC06b73d65c8ee67ce8e448edcc64b6ec6
Client secret <a href="#">?</a> :	***
Sender number <a href="#">?</a> :	+12186751069
Template code <a href="#">?</a> :	get the message
SMS Test <a href="#">?</a> :	+1 <a href="#">▼</a> Input your phone num... <a href="#">Send Testing SMS</a>
Provider URL <a href="#">?</a> :	<a href="#">🔗</a>



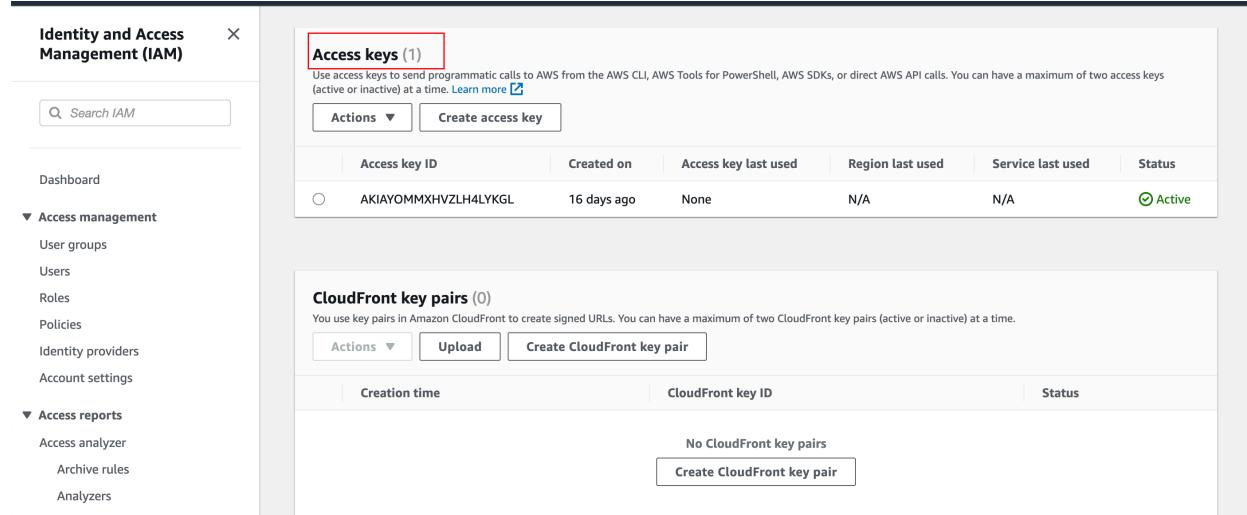
# Amazon SNS

## Obtaining the necessary information in Amazon

There are four required fields: Access Key, Secret Access Key, Region, and Template code. I will show you how to obtain this information from Amazon SNS.

- Access Key and Secret Access Key

In Identity and Access Management (IAM), you can create an Access Key and Secret Access Key.



The screenshot shows the AWS Identity and Access Management (IAM) console. On the left, there's a navigation sidebar with options like Dashboard, Access management, Access reports, and Analytics. The main area is titled "Access keys (1)". It contains a table with one row of data:

Access key ID	Created on	Access key last used	Region last used	Service last used	Status
AKIAVOMMXHVZLH4LYKGL	16 days ago	None	N/A	N/A	Active

Below this, there's a section titled "CloudFront key pairs (0)" with a "Create CloudFront key pair" button.

- Region

The Region is related to the topic you created.

The screenshot shows the AWS Amazon SNS Dashboard. On the left, there's a sidebar with links for 'Dashboard', 'Topics', 'Subscriptions', and sections for 'Mobile' (Push notifications, Text messaging (SMS), Origination numbers). The main area is titled 'Dashboard' and shows a summary: 'Resources for ap-southeast-1'. It lists 'Topics' (1), 'Platform applications' (0), and 'Subscriptions' (0). Below this, there's a section titled 'Overview of Amazon SNS' with a sub-section 'Application-to-application (A2A)' which describes Amazon SNS as a managed messaging service for decoupling publishers from subscribers.

## Configuring the Casdoor provider

The `Template code` is the message you want to send. Enter your phone number in the `SMS Test` to test.

Name [?](#) :

Display name [?](#) :

Organization [?](#) :

Category [?](#) :

Type [?](#) :

Access key [?](#) :

Secret access key [?](#) :

Region [?](#) :

Template code [?](#) :

SMS Test [?](#) :

Provider URL [?](#) :

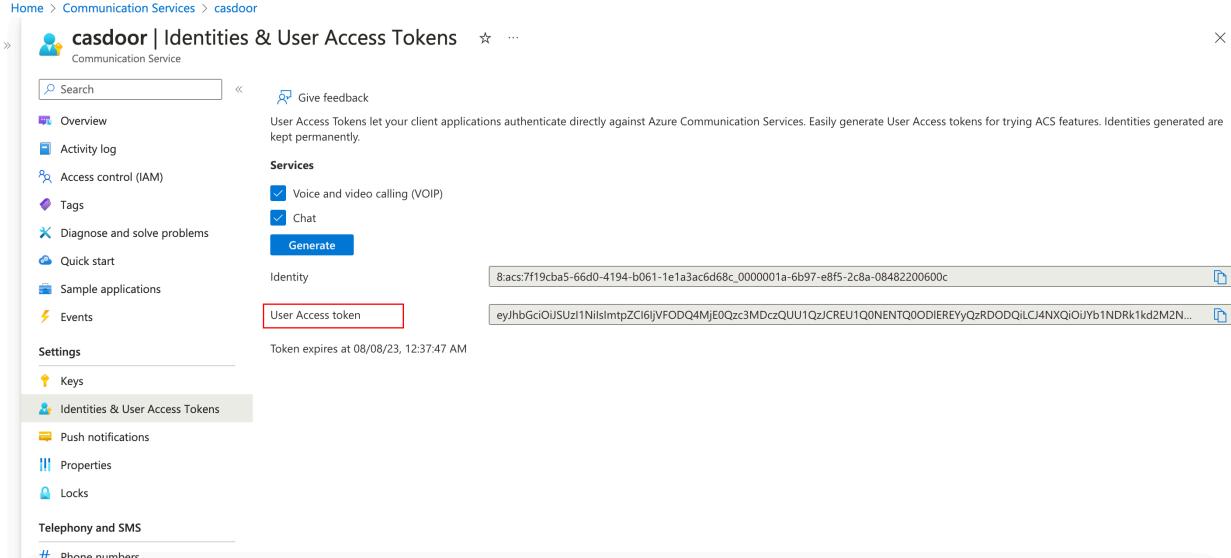
# Azure ACS

## Obtaining the necessary information in Azure

There are four required fields: `Client secret`, `Sender number`, `Template code`, and `Provider Url`. I will show you how to obtain this information from Azure ACS.

- `Client secret`

In Communication Service, you can create a User Access Token, which is the `Client secret` in Casdoor.



The screenshot shows the Azure Communication Service Identity & User Access Tokens page for the 'casdoor' service principal. The left sidebar lists various service principals under 'Communication Service'. The main area shows a 'User Access token' section with a generated token value: 8.acs:7f19cba5-66d0-4194-b061-1e1a3ac6d68c\_0000001a-6b97-e8f5-2c8a-08482200600c eyJhbGciOiJSUzI1NlsltmpZCI6IjVFQDQ4MjE0Qzc3MDczQUU1QzJCREU1Q0NENTQ0ODIERYyQzRDODQjLCj4NXQjOjYb1NDRk1kd2M2N... . The token has a short expiration time of 08/08/23, 12:37:47 AM.

- `Sender number`

The `Sender number` is the phone number you create in Communication Service.

Communication Service

Search (Cmd+/) Get Port Release Give feedback

Tools

- Keys
- Identities & User Access Tokens
- Push notifications

Voice Calling - PSTN

- # Phone numbers
- Direct routing (Preview)

SMS

- Short Codes (Preview)

Monitoring

- Insights (preview)
- Metrics
- Diagnostic settings
- Logs

	Number	Status	Cost (monthly)
<input checked="" type="checkbox"/>	1-833-920-3625	Active	\$2

- Provider Url

The **Provider Url** is the endpoint in Communication Service.

Communication Service

Search Move Delete Give feedback

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Sample applications

Events

Settings

Keys

Identities & User Access Tokens

Endpoint : https://casdoor.unitedstates.communication.azure.com

Resource group (move) : casdoor

Status : Active

Location : Global

Subscription (move) : 免费试用

Subscription ID : e054e27a-96e8-4cca-a1cf-32717dcd303c

Tags (edit) : Add tags

Build engaging communication experiences at scale

Azure Communication Services brings rich communication APIs to all of your apps across any device, on any platform, using the same reliable and secure infrastructure that powers Microsoft Teams.

Learn more

# Configure Casdoor provider

The `template code` is the message you want to send. Enter your phone number in `SMS Test` to test.

---

Name [?](#) :

Display name [?](#) :

Organization [?](#) :

Category [?](#) :

Type [?](#) :  

Client ID [?](#) :

Client secret [?](#) :

Sender number [?](#) :

Template code [?](#) :

SMS Test [?](#) :

Provider URL [?](#) :

---

# Alibaba Cloud

## Fill in the necessary information in Casdoor

There are four required fields: `Client ID`, `Client secret`, `Sign Name`, and `Template code`. The corresponding relationship with the Alibaba Cloud account is as follows:

Name	Name in Alibaba	is required
Client ID	AccessKey ID	required
Client secret	AccessKey Secret	required
Sign Name	Signature	required
Template code	Template code	required

## Alibaba information

- AccessKey ID and AccessKey Secret

After logging into my Alibaba Cloud workbench, I click on "AccessKey" to create an ID and Secret.

The screenshot shows the Alibaba Cloud SMS service interface. At the top, there's a search bar and navigation links like '费用' (Cost), '工单' (Work Orders), 'ICP 备案' (ICP Registration), '企业' (Enterprise), '支持' (Support), 'App', and '帮助中心'. Below the header, a banner reads '【有奖调研】阿里云短信服务易用性有奖调研' (SMS Service Usability Survey) with a '点击进入' (Click to Enter) button. The main area has tabs for '新手引导' (Newbie Guide), 'OpenAPI 开发者门户' (OpenAPI Developer Portal), '开发者指南' (Developer Guide), and 'AccessKey' (highlighted with a red circle). On the left, there's a section for '发送量数据' (Delivery Volume Data) and a progress bar indicating 20% completion. The right side includes sections for '用户监控信息' (User Monitoring Information), '快捷操作入口' (Quick Operation Entry), and '国内消息' (Domestic Messages). A central call-to-action button says '快速学习短信服务' (Quickly Learn SMS Service).

By creating an AccessKey, I obtain my AccessKey ID and AccessKey Secret:

This screenshot shows the '安全信息管理' (Security Information Management) section of the Alibaba Cloud interface. It displays a table for 'User AccessKey' with columns: AccessKey ID, AccessKey Secret, Status, Last Used Time, and Creation Time. One row is shown with the AccessKey ID 'LTAI4Fy4mVoMjAzC95rt5Wh7', AccessKey Secret '显示' (Visible), Status '启用' (Enabled), Last Used Time '2020年7月19日 20:24:58', and Creation Time '2020年7月11日 17:52:50'. There are '禁用' (Disable) and '删除' (Delete) buttons for this entry. A note at the top states: '① AccessKey ID和AccessKey Secret是您访问阿里云API的密钥，具有该账户完全的权限，请您妥善保管。' (① AccessKey ID and AccessKey Secret are your keys to access Alibaba Cloud API, giving you full control over the account. Please keep them safe.)

- Signature

This screenshot shows the 'Signature' management interface. The left sidebar includes 'Go China', 'Go Globe', 'Analytics', 'Dashboard', 'Delivery Report', 'Messaging Logs', 'Bills', 'Resource Plan Usage', 'Short URL Statistics', 'System Configurations', and 'General Settings'. The main area features a QR code with the text 'Join the group chat to try new features.' and a large blue feather icon. A note below the QR code states: 'Alibaba Cloud SMS prohibits illegal content such as illegal financial marketing, gambling, fraud, obscenity, pornography, and violence in a message. If you send a message that contains illegal content, Alibaba Cloud will suspend your service and account according to Short Message Service Terms of Service. If your message is against the law, Alibaba Cloud will transfer evidences to the police, including but not limited to the personal information authenticated in Alibaba Cloud.' Below this is a tab bar with 'Signatures' (selected), 'Message Templates', and 'Mass Messaging'. A note says: 'Generally, signatures are reviewed within 2 hours. Enterprise or institution signatures are reviewed within 2 business days. Recently, a review process took about 1 hour on average. More time may be required due to system upgrades or workload spikes. Business hours: Signatures are reviewed from 9:00 to 21:00 (UTC+8) every day, excluding statutory holidays. Thanks for your cooperation.' A table lists signatures with columns: Create Signature, Select a review status, Search by signature, Actions, Signature, Ticket ID, Scenario, Review Status, and Created At. One row is highlighted with a red box around the 'casdoor' signature name.

- Template code

**Short Message Service**

- Overview
- Quick Start & Delivery Test
- Go China**
- Go Globe
- Analytics
- Dashboard
- Delivery Report
- Messaging Logs
- Bills
- Resource Plan Usage
- Short URL Statistics
- System Configurations
- General Settings
- Domestic SMS Settings

Create Dedicated DingTalk Group Chat

Scan the QR code to create your dedicated DingTalk group chat.

You can use the group chat to submit and modify signatures and message templates, and check whether the signatures and message templates are valid.

Join the group chat to try new features.

Alibaba Cloud SMS prohibits illegal content such as illegal financial marketing, gambling, fraud, obscenity, pornography, and violence in a message. If you send illegal content, Alibaba Cloud will suspend your service and account according to Short Message Service Terms of Service. If your message is against the law, Alibaba Cloud will not be liable for any personal information authenticated in Alibaba Cloud.

	Signatures	Message Templates	Mass Messaging			
<b>1.</b>	Generally, signatures are reviewed within 2 hours. Recently, a review process took about <b>1 hour</b> on average. More time may be required due to system up hours: Signatures are reviewed from 9:00 to 21:00 (UTC+8) every day, excluding statutory holidays. Thanks for your cooperation.					
<b>2.</b>	Message templates provided by Alibaba Cloud SMS do not require submission for approval. However, you are still charged for message delivery.					
<a href="#">Create Message Template</a>		Select a template type <input type="button" value="▼"/>	Select a review status <input type="button" value="▼"/>			
		Search by message template name <input type="text"/>	<input type="button" value="Q"/>			
		<input type="button" value="Tag Filter"/>				
	Template Name	Tag	Ticket ID	Template Code	Template Type	Created At
	cassandra	测	20020389144	SMS_462155126	Verification Code Message	2023-07-26 17:54:00

## Configure Casdoor provider

Enter your phone number in the **SMS Test** field to test.

Name <small>②</small> :	alibaba
Display name <small>②</small> :	alibaba
Organization <small>②</small> :	admin (Shared)
Category <small>②</small> :	SMS
Type <small>②</small> :	Aliyun SMS
Client ID <small>②</small> :	LTAI5tFwxoA51CnSiQFyyPU5
Client secret <small>②</small> :	***
Sign Name <small>②</small> :	casdoor
Template code <small>②</small> :	SMS_462155126
SMS Test <small>②</small> :	+86 <input type="button" value="▼"/> Input your phone number... <input type="button" value="Send Testing SMS"/>
Provider URL <small>②</small> :	<input type="text"/>

# 通知

## Overview

Add Notification providers to your application

## Telegram

Using Telegram as a notification provider for Casdoor

## Custom HTTP

Using Custom HTTP as a notification provider for Casdoor

## Slack

Using Slack as a notification provider for Casdoor

 **Google Chat**

Using Google Chat as a notification provider for Casdoor

 **Twitter**

Using Twitter as a notification provider for Casdoor

 **Discord**

Using Discord as a notification provider for Casdoor

# Overview

Casdoor can be configured to send notification messages using various Notification providers.

Currently, Casdoor supports multiple Notification providers. Here are the providers that Casdoor supports:

Provider	Logo
Telegram	
Custom HTTP	
Slack	
Google Chat	
Twitter	
Discord	
Bark	

Provider	Logo
DingTalk	
Lark	
Line	
Matrix	[matrix]
Microsoft Teams	
Pushbullet	
Pushover	
Reddit	
Rocket Chat	
Viber	
Webpush	

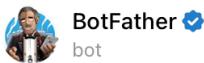


# Telegram

## Step 1: Get API Token

First, you need to create an account on [Telegram](#). After creating an account, you should contact the [BotFather](#), which is a bot used to create other bots.

To create your bot, use the command `/newbot`:



### Games

[/mygames](#) - edit your games  
[/newgame](#) - create a new game  
[/listgames](#) - get a list of your games  
[/editgame](#) - edit a game  
[/deletegame](#) - delete an existing game



usher  
[/newbot](#)



BotFather

Alright, a new bot. How are we going to call it? Please choose a name for your bot.



usher  
casdoor



BotFather

Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris\_bot.



usher  
CasdoorBot



BotFather

Done! Congratulations on your new bot. You will find it at [t.me/CasdoorBot](https://t.me/CasdoorBot). You can now add a description, about section and profile picture for your bot, see [/help](#) for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:

**6531753859:AAEJV6-fopJLLyNa3mSn\_H-dFIPO8VXwEto**

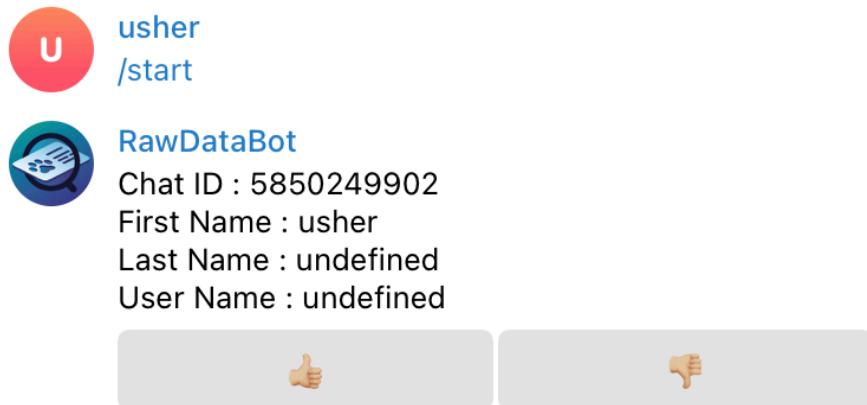
Keep your token **secure** and **store it safely**, it can be used by anyone to control your bot.

For a description of the Bot API, see this page: <https://core.telegram.org/bots/api>

Your bot should have two attributes: a `name` and a `username`. After creating the bot, you will receive an `API Token`.

## Step 2: Get Chat ID

To find your chat ID, use [RawDataBot](#).



## Step 3: Configure Casdoor Telegram Provider

There are three required fields: `App Key`, `Content`, and `Chat ID`. The relationship between the fields and Telegram is as follows:

Name	Name in Telegram
Secret key	API Token
Chat ID	Chat ID

Name	Name in Telegram
Content	

Name ? : telegram

Display name ? : telegram

Organization ? : admin (Shared)

Category ? : Notification

Type ? :  Telegram

Secret key ? : \*\*\*

Content ? : test

Chat ID ? : 5850249902 Send Testing Notification

Provider URL ? : <https://github.com/organizations/xxx/settings/applications/1234567>

# Custom HTTP

① 备注

Casdoor supports Custom HTTP Notification Provider. You can use it to send messages to specific HTTP addresses.

## Configure Casdoor Custom HTTP Provider

There are three required fields: Method, Parameter name, Content, and Chat ID.

Name	Description
Method	Select GET or POST method.
Parameter name	URL query parameter name or body parameter, depending on the method.
Content	The message you want to send.
Chat ID	Your HTTP address

Name <small>②</small> :	custom_http
Display name <small>②</small> :	custom_http
Organization <small>②</small> :	admin (Shared)
Category <small>②</small> :	Notification
Type <small>②</small> :	 Custom HTTP
Method <small>②</small> :	POST
Parameter <small>②</small> :	test
Content <small>②</small> :	test
Endpoint <small>②</small> :	<input type="text" value="http://localhost:12345"/>
	<button style="background-color: #007bff; color: white; border-radius: 5px; padding: 5px;" type="button">Send Testing Notification</button>
Provider URL <small>②</small> :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>

In my example, when I click `Send Notification Message`, I receive this request:

```
Listening on :12345...
Received a request:
Method: POST
URL: /
Body: test
```

# Slack

## Step 1: Config Slack App

First, you need to create an app on [Slack API](#). Give your bot/app the following OAuth permission scopes: `chat:write`, `chat:write.public`

The screenshot shows the 'Scopes' section of the Slack API configuration. It lists two selected OAuth scopes: 'chat:write' and 'chat:write.public'. Both scopes have a trash can icon to their right, indicating they can be removed. A red box highlights the 'chat:write' scope. Below the table is a button labeled 'Add an OAuth Scope'.

OAuth Scope	Description	
<code>chat:write</code>	Send messages as @casdoor	
<code>chat:write.public</code>	Send messages to channels @casdoor isn't a member of	

Add an OAuth Scope

## Step 2: Get Bot User OAuth Access Token and Channel ID

Copy your `Bot User OAuth Access Token` for usage below.

**Features**

- App Home
- Org Level Apps
- Incoming Webhooks
- Interactivity & Shortcuts
- Slash Commands
- Workflow Steps

**OAuth & Permissions**

- Event Subscriptions
- User ID Translation
- App Manifest NEW
- Beta Features

**Submit to App Directory**

- Review & Submit

Give feedback

[Slack ❤️](#)

Help

Contact

Policies

Our Blog

**Opt In**

**⚠️ At least one redirect URL needs to be set below before this app can be opted into token rotation**

---

## OAuth Tokens for Your Workspace

These tokens were automatically generated when you installed the app to your team. You can use these to authenticate your app. [Learn more.](#)

**User OAuth Token**

xoxp-5865439759200-5827199080935-5865457291440-67810c12dfcae [Copy](#)

Access Level: Workspace

**Bot User OAuth Token**

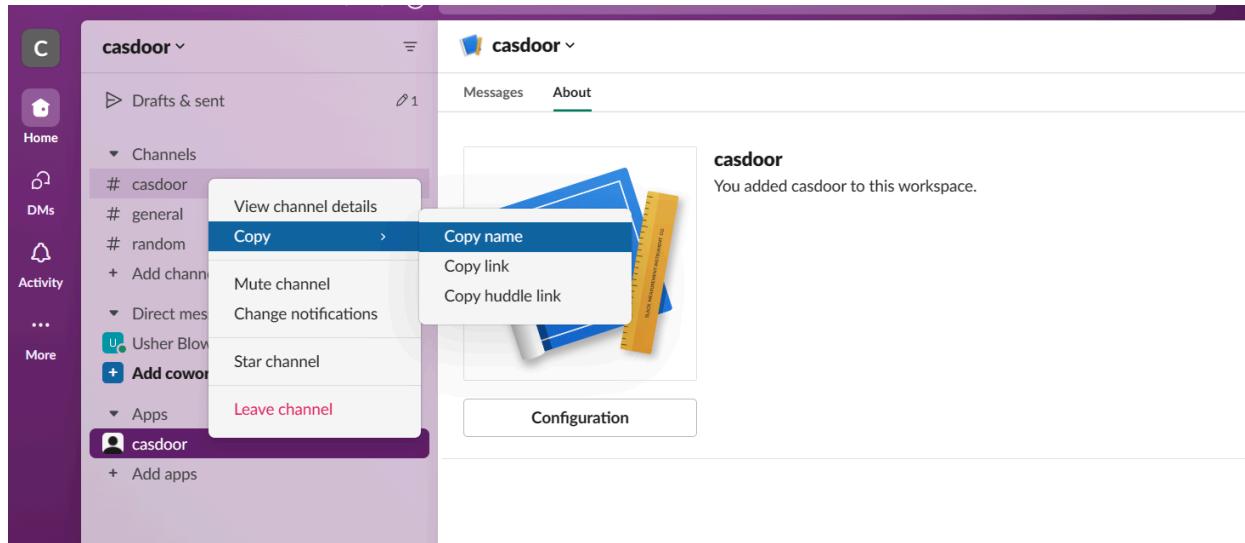
xoxb-5865439759200-5865457299776-2tbOAVTNPpG7vmLoG7OFJS5t [Copy](#)

Access Level: Workspace

[Reinstall to Workspace](#)

## Redirect URLs

Copy the Channel ID of the channel you want to post a message to. You can grab the Channel ID by right clicking a channel and selecting [copy name](#)



# Step 3: Configure Casdoor Slack Provider

There are three required fields: `App Key`, `Content`, and `Chat ID`. The relationship between the fields and Slack is as follows:

Name	Name in Slack
Secret key	Access Token
Chat ID	Channel ID
Content	

---

Name ②:

Display name ②:

Organization ②:

Category ②:

Type ②:

Secret key ②:

Content ②:

Chat ID ②:  Send Testing Notification

Provider URL ②:

---

# Google Chat

## Step 1: Get Application Default Credentials

In order to integrate notify with a Google Chat Application, [Application Credentials](#) must be supplied. For more information on Google Application credential JSON see: [How Application Default Credentials works](#)

The json will look like this:

```
{  
  "type": "service_account",  
  "project_id": "",  
  "private_key_id": "",  
  "private_key": "",  
  "client_email": "",  
  "client_id": "",  
  "auth_uri": "",  
  "token_uri": "",  
  "auth_provider_x509_cert_url": "",  
  "client_x509_cert_url": ""  
}
```

## Step 3: Configure Casdoor Google Chat Provider

Fill in the [Application credential](#) in the metadata.

Name [?](#):

Display name [?](#):

Organization [?](#):

Category [?](#):

Type [?](#):  

Metadata [?](#): 

```
{  
  "type": "service_account",  
  "project_id": "",  
  "private_key_id": ""  
}'
```

Content [?](#):

Test Notification [?](#):

Provider URL [?](#):  

# Twitter

## Step 1: Get the configuration items from twitter

First, sign up for a Twitter developer account, create a Twitter App within the developer portal refer to the documentation: [Authentication](#)

Copy your [API Key](#) and [API Secret](#), [Access Token](#) and [Access Token Secret](#)

The screenshot shows the Twitter Developer Portal interface. On the left is a dark sidebar with navigation links: Dashboard, Projects & Apps (selected), Products (NEW), and Account. The main content area has a light background. At the top, there are two tabs: 'Settings' and 'Keys and tokens' (which is underlined, indicating it's active). Below this, the 'Consumer Keys' section is visible, featuring a red-bordered box around the 'API Key and Secret' row. This row contains the text 'API Key and Secret ⓘ', a 'Reveal API Key hint' link, and a 'Regenerate' button. The 'Authentication Tokens' section follows, showing a card for a 'Bearer Token' generated on September 3, 2023, with 'Revoke' and 'Regenerate' buttons. Another red-bordered box surrounds a card for an 'Access Token and Secret' generated on the same date for the user '@Allcompleteness'. This card also includes the text 'Created with Read Only permissions'.

## Step 2: Get Twitter ID

Twitter ID can't be obtained directly, you can get it from some third-party tools.

- [TweeterID](#)
- [Twiteridfinder](#)

## Step 3: Configure Casdoor Twitter Provider

There are five required fields: Client ID, Client secret, Client ID 2, Client secret 2 and Chat ID. The relationship between the fields and Twitter is as follows:

Name	Name in Twitter
Client ID	API Key
Client secret	API Secret
Client ID 2	Access Token
Client secret 2	Access Token Secret
Chat ID	Twitter ID

Name ? :

Display name ? :

Organization ? :

Category ? :

Type ? :  

Client ID ? :

Client secret ? :

Client ID 2 ? :

Client secret 2 ? :

Content ? :

Chat ID ? :  Send Testing Notification

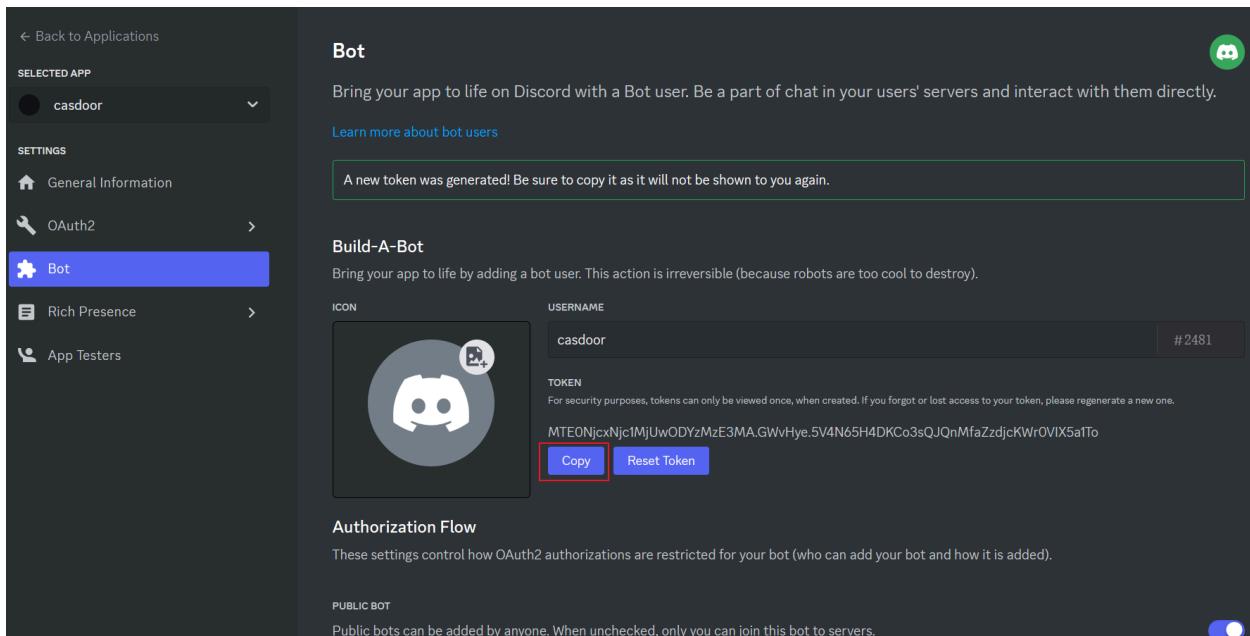
Provider URL ? :

# Discord

## Step 1: Get Token from Discord

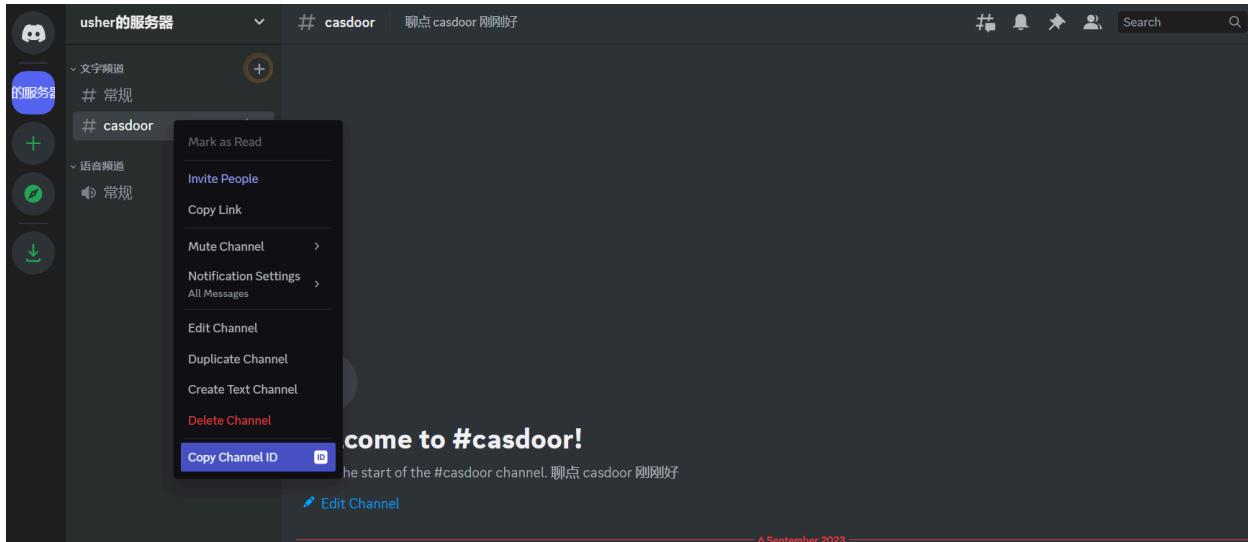
First, sign up for the Discord Developer portal, create a new application, navigate to the “Bot” tab to configure it.

Copy your Bot [token](#)



## Step 2: Get Channel ID

Copy the Channel ID of the channel you want to post a message to. You can grab the Channel ID by right clicking a channel and selecting [Copy Channel ID](#)



## Step 3: Configure Casdoor Discord Provider

There are three required fields: `App Key`, `Content`, and `Chat ID`. The relationship between the fields and Discord is as follows:

Name	Name in Slack
Secret key	Token
Chat ID	Channel ID
Content	

Name ⓘ :	discord
Display name ⓘ :	discord
Organization ⓘ :	admin (Shared)
Category ⓘ :	Notification
Type ⓘ :	 Discord
Secret key ⓘ :	***
Content ⓘ :	test
Chat ID ⓘ :	1146715329133821972
Provider URL ⓘ :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>
	<button type="button">Send Testing Notification</button>

# 存储

## 概述

Setting up a storage provider for uploading files in Casdoor

## Local File System

Using the Local File System as a storage provider for Casdoor

## Amazon S3

Using Amazon S3 as a storage provider for Casdoor

## Azure Blob

使用 Azure Blob 作为Casdoor的存储提供商



## Google Cloud Storage

Using Google Cloud Storage as a storage provider for Casdoor



## MinIO

Using MinIO as a storage provider for Casdoor



## Alibaba Cloud OSS

Using Alibaba Cloud OSS as a storage provider for Casdoor



## Tencent Cloud COS

Using Tencent Cloud COS as a storage provider for Casdoor



## Synology NAS

Using Synology NAS as a storage provider for Casdoor

# 概述

If you need to use file storage services, such as "avatar upload", you will need to set up a storage provider and apply it to your application in Casdoor.

Casdoor supports two types of storage: Local and Cloud. In this chapter, you will learn how to add a storage provider to use these services.

## 项目

- Client ID: A unique identifier provided by the cloud storage provider.
- Client secret: A secure value known only to Casdoor and the cloud storage service.
- Endpoint: The public URL or domain of the cloud storage service.
- Endpoint (Intranet): The internal or private URL or domain of the cloud storage service.
- Path prefix: Path prefix for the file location.

### 信息

The default `Path prefix` is "/". For example, when the `Path prefix` is empty, a default file path would be:

```
https://cdn.casbin.com/casdoor/avatar.png
```

You can fill it with "abcd/xxxx", and then you can store your avatar in:

<https://cdn.casbin.com/abcd/xxxx/casdoor/avatar.png>

- **Bucket:** A container used for storing files and data.
- **Domain:** The custom domain name of the CDN for your cloud storage.
- **Region ID:** An identifier used to specify the data center region where the cloud storage service is located.

## 本地设置

We support uploading files to the local system.

## Cloud

We support AWS S3, Azure Blob Storage, MinIO, Alibaba Cloud OSS, Tencent Cloud COS, and we are constantly adding more Cloud storage services.

# Local File System

## ① 信息

The Local File System provider will store your uploaded files in the Casdoor `files` folder.

For example, when your Casdoor is located in the `/home/user/casdoor` directory, the uploaded files will be stored in the `/home/user/casdoor/files` folder.

## Configure the Casdoor provider

Name <a href="#">?</a> :	localfile
Display name <a href="#">?</a> :	localfile
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	Storage
Type <a href="#">?</a> :	Local File System
Path prefix <a href="#">?</a> :	
Domain <a href="#">?</a> :	http://localhost:8000
Provider URL <a href="#">?</a> :	<a href="#"></a>

The **Path prefix** is the prefix of the location path for your files. You can fill it in as needed. In the following example, you can see the difference with or without the prefix.

## With prefix

Path prefix [?](#) :

images

casdoor > files > images > resource > built-in > admin > withPrefix.png

## Without prefix

Path prefix [?](#) :

casdoor > files > resource > built-in > admin > withoutPrefix.png

# Amazon S3



This is an example of Amazon S3.

## Create security credentials

Follow the document: [Managing access keys](#) to create and save your `access key` and `secret access key` in the Amazon console.

## Configure your bucket

In your bucket permissions options, uncheck the "block" option and save the changes.

## Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

### **Block all public access**

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

#### **Block public access to buckets and objects granted through *new* access control lists (ACLs)**

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

#### **Block public access to buckets and objects granted through *any* access control lists (ACLs)**

S3 will ignore all ACLs that grant public access to buckets and objects.

#### **Block public access to buckets and objects granted through *new* public bucket or access point policies**

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

#### **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

[Cancel](#)

[Save changes](#)

Edit the object ownership and check ACLs enabled.

## Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

**ACLs disabled (recommended)**

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

**ACLs enabled**

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

**⚠️** We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.

### Object Ownership

**Bucket owner preferred**

If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

**Object writer**

The object writer remains the object owner.

**ⓘ** If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more ↗](#)

Cancel

Save changes

## Configure Casdoor

Name	Name in Amazon	Is Required
Client ID	Access key	Required
Client secret	Secret access key	Required
Endpoint	Endpoint	Required
Endpoint (intranet)	VPC endpoint	

Name	Name in Amazon	Is Required
Bucket	Bucket name	Required
Path prefix		
Domain	CloudFront domain	
Region ID	AWS region	Required

Fill in the necessary information, including the `Client ID` and `Client Secret` obtained from the `access key` and `secret access key` in the previous step. You can refer to this documentation for information on the formatting of the `endpoint`: [Website endpoints](#)

Name <a href="#">?</a> :	awss3
Display name <a href="#">?</a> :	awss3
Organization <a href="#">?</a> :	admin (Shared)
Category <a href="#">?</a> :	Storage
Type <a href="#">?</a> :	AWS S3
Client ID <a href="#">?</a> :	AKIAYOMMXHVZC5CHBPNR
Client secret <a href="#">?</a> :	***
Endpoint <a href="#">?</a> :	<a href="http://kininaru.s3-website.ap-northeast-1.amazonaws.com">http://kininaru.s3-website.ap-northeast-1.amazonaws.com</a>
Endpoint (Intranet) <a href="#">?</a> :	
Bucket <a href="#">?</a> :	kininaru
Path prefix <a href="#">?</a> :	
Domain <a href="#">?</a> :	
Region ID <a href="#">?</a> :	ap-northeast-1
Provider URL <a href="#">?</a> :	<a href="#">🔗</a>

## (Optional) Use VPC

You can refer to the official documentation for configuration: [Access AWS services through AWS PrivateLink](#)

## (Optional) Use CloudFront distribution

Follow the document to configure CloudFront: [Configure CloudFront](#)

In the domain field, enter your distribution domain name.

Endpoint [?](#) : <http://kininaru.s3-website.ap-northeast-1.amazonaws.com>

Bucket [?](#) : kininaru

Path prefix [?](#) :

Domain [?](#) : <https://d20zlk9foisfk0.cloudfront.net>

Region ID [?](#) : ap-northeast-1

Provider URL [?](#) : [🔗](#)

# Azure Blob

① 备注

This is an example of Azure Blob.

- 您必须拥有一个 [Azure storage](#) 帐户。

## Step 1: Select Azure Blob

Select Azure Blob as the storage type.

Edit Provider Save Save & Exit

Name ② :	provider_ftfzes
Display name ② :	New Provider - ftfzes
Category ② :	Storage
Type ② :	Azure Blob
Client ID ② :	Local File System AWS S3
Client secret ② :	Aliyun OSS Tencent Cloud COS
Endpoint ② :	Azure Blob

## Step 2: Fill in the necessary information in Casdoor

There are four required fields: `Client ID`, `Client secret`, `Endpoint`, and `Bucket`. The corresponding relationship to the Azure Blob account is as follows:

Field Name	Azure Name	Required
Client ID	AccountName	Required
Client secret	AccountKey	Required
Endpoint	ContainerUrl	Required
Endpoint (intranet)	PrivateEndpoint	
Bucket	ContainerName	Required
Path prefix		
Domain	DomainName	

- AccountName

The `AccountName` is your AccountName.

- AccountKey

The `AccountKey` is your key to access the Azure API.

 备注

You can obtain your account key from the Azure Portal under the "Access Keys" section on the left-hand pane of your storage account.

The screenshot shows the 'Access keys' section of the Azure Storage account settings for 'casbin'. The left sidebar lists 'Containers', 'File shares', 'Queues', and 'Tables' under 'Data storage'; 'Networking', 'Azure CDN', and 'Access keys' under 'Security + networking'; and 'Shared access signature', 'Encryption', and 'Microsoft Defender for Cloud' under 'Data management'. The 'Access keys' section is highlighted with a red box. It displays two key pairs: 'key1' (last rotated 0 days ago) and 'key2' (last rotated 0 days ago). Each key has a 'Show' button to reveal the value. Below each key pair is a 'Connection string' input field with a 'Show' button.

Storage account name: casbin

key1 Rotate key

Last rotated: 2023/7/22 (0 days ago)

Key: [REDACTED] [Show](#)

Connection string: [REDACTED] [Show](#)

key2 Rotate key

Last rotated: 2023/7/22 (0 days ago)

Key: [REDACTED] [Show](#)

Connection string: [REDACTED] [Show](#)

- ContainerUrl

You can obtain the ContainerUrl from your container properties.

 default | Properties

Container

Search Refresh Give feedback

Overview Diagnose and solve problems Access Control (IAM)

Shared access tokens Access policy Properties Metadata

**NAME**  
default

**URL**  
`https://casbin.blob.core.windows.net/default`

**LAST MODIFIED**  
7/22/2023, 5:18:03 PM

**ETAG**  
0x8DB8A948D644055

- (Optional) PrivateEndpoint

Azure Private Endpoint is a feature that allows connecting Azure services to Azure Virtual Network (VNet) private subnets. You can refer to the official Azure documentation for configuration: [private endpoint config](#)

- ContainerName

In this example, a default container called 'default' is created.

Home > casbin

## casbin | Containers

Storage account

Search (Ctrl+ /) Container Change access level Restore containers Refresh Delete

Overview Activity log Tags Diagnose and solve problems Access Control (IAM) Data migration Events Storage browser (preview)

**Data storage**

- Containers** (highlighted with a red circle)
- File shares
- Queues
- Tables

Name

Name	Created
\$logs	2022-04-23
<b>default</b>	2022-04-26

- (Optional) DomainName

The custom domain name in your Azure CDN.

Home > fd-profile

## fd-profile | Front Door and CDN profiles

Front Door and CDN profiles

Search (Ctrl+ /) Purge cache Origin response timeout Delete Refresh

Overview Activity log Access control (IAM) Tags Diagnose and solve problems

**Settings**

- Front Door manager
- Domains
- Origin groups
- Rule set
- Security policies
- Optimizations
- Secrets
- Properties
- Locks

**Analytics**

- Reports
- Monitoring

**Essentials**

Resource group (move)	: default
Status	: Active
Location	: Global
Subscription (move)	: Visual Studio Enterprise
Subscription ID	[REDACTED]
Tags (edit)	: Click here to add tags

**Properties**

Endpoints	Endpoint hostname	Route name
	endpoint-hth4eebgcdzc2ex.z01.azurefd.net	default-route

**Custom domains**

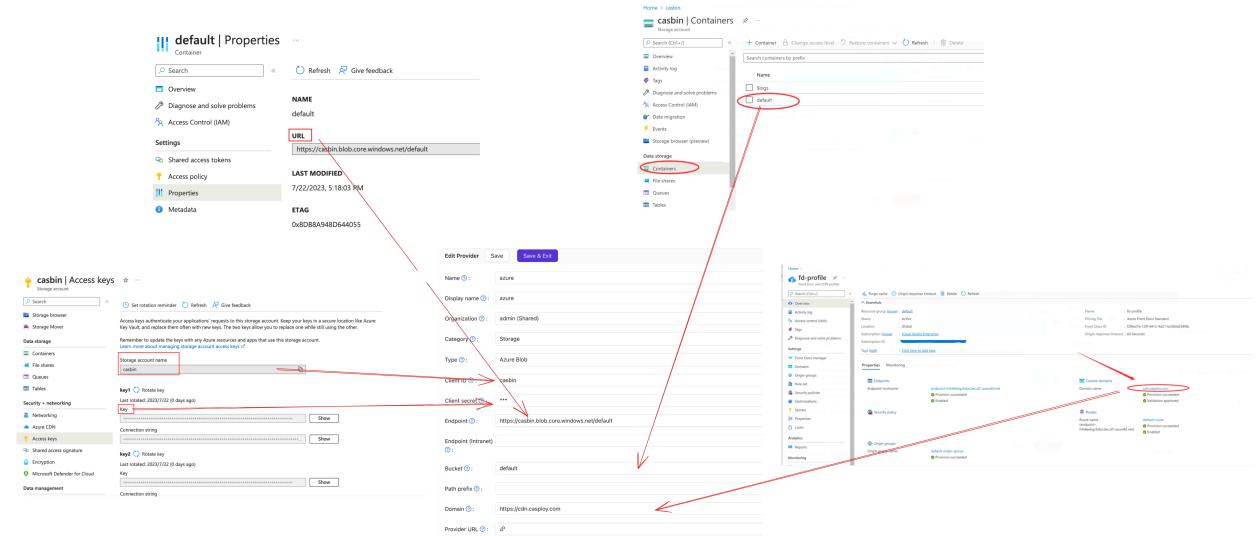
Domain name
cdn.casploy.com

**Routes**

Route name	Endpoint
default-route	endpoint-hth4eebgcdzc2ex.z01.azurefd.net

# Step 3: Save your configuration

The final result is as follows:



Now you can use Azure Blob Storage services in your application.

# Google Cloud Storage

 备注

This is an example of Google Cloud Storage.

## Create security credentials

Follow the document: [Cloud Storage Authentication](#) to create a [service account](#) with the correct [IAM permissions](#) to access the bucket in the GCP console.

## Configure Casdoor

Name	Name in Google	Is Required
Service Account JSON	Service Account Key	Required
Endpoint	Endpoint	
Bucket	Bucket name	Required

Name ⓘ :

Display name ⓘ :

Organization ⓘ :  ▾

Category ⓘ :  ▾

Type ⓘ :  ▾

Service account JSON ⓘ :

Endpoint ⓘ :

Bucket ⓘ :

Path prefix ⓘ :

Provider URL ⓘ :

# MinIO

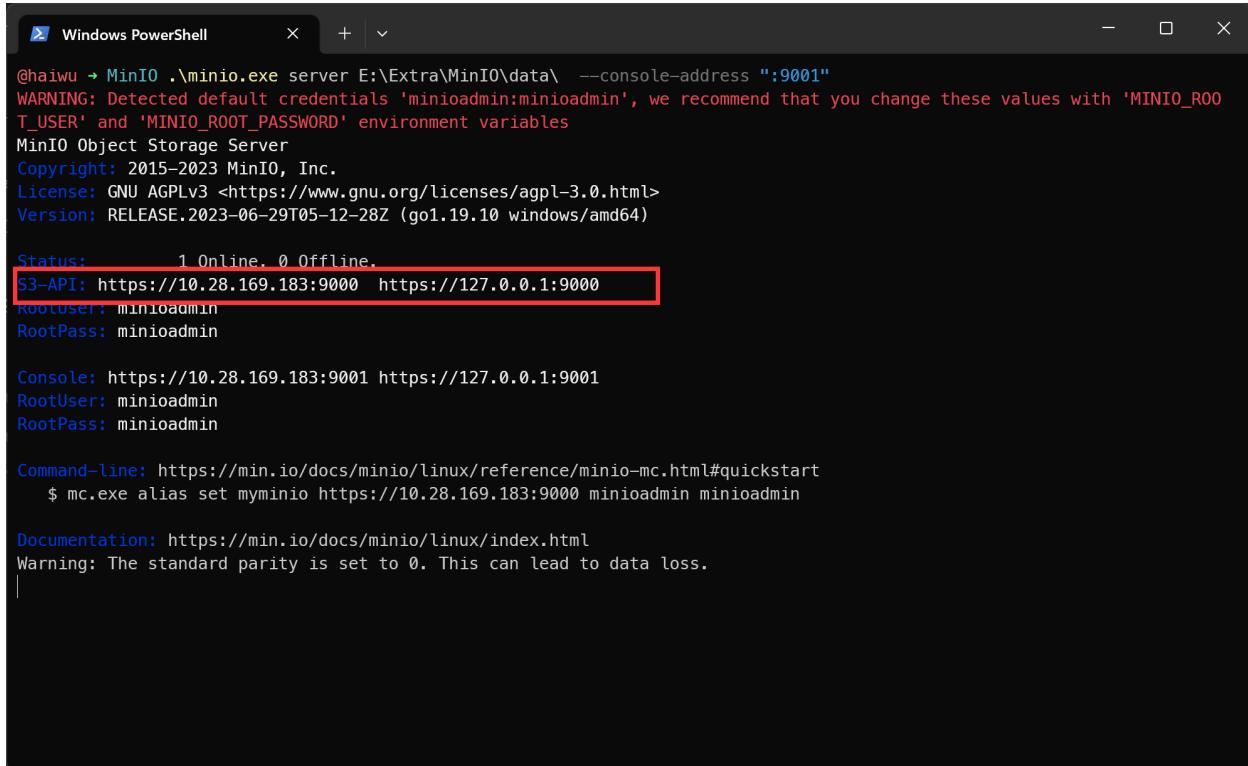
## ⓘ 备注

This is an example of how to configure a MinIO provider.

MinIO is a high-performance object storage service that is API compatible with Amazon S3 cloud storage service.

## Step 1: Deploy the MinIO service

First, deploy the MinIO service with TLS enabled. You can obtain the [API address](#) from the console.



```
Windows PowerShell

@haiwu ~ MinIO .\minio.exe server E:\Extra\MinIO\data\ --console-address ":9001"
WARNING: Detected default credentials 'minioadmin:minioadmin', we recommend that you change these values with 'MINIO_ROOT_USER' and 'MINIO_ROOT_PASSWORD' environment variables
MinIO Object Storage Server
Copyright: 2015-2023 MinIO, Inc.
License: GNU AGPLv3 <https://www.gnu.org/licenses/agpl-3.0.html>
Version: RELEASE.2023-06-29T05-12-28Z (go1.19.10 windows/amd64)

Status: 1 Online, 0 Offline.
S3-API: https://10.28.169.183:9000 https://127.0.0.1:9000
RootUser: minioadmin
RootPass: minioadmin

Console: https://10.28.169.183:9001 https://127.0.0.1:9001
RootUser: minioadmin
RootPass: minioadmin

Command-line: https://min.io/docs/minio/linux/reference/minio-mc.html#quickstart
$ mc.exe alias set myminio https://10.28.169.183:9000 minioadmin minioadmin

Documentation: https://min.io/docs/minio/linux/index.html
Warning: The standard parity is set to 0. This can lead to data loss.
```

Second, create the Access Key and Secret key.

The screenshot shows the MinIO Object Store interface. On the left, a sidebar menu includes 'User' (Object Browser, Access Keys, Documentation), 'Administrator' (Buckets, Policies, Identity, Monitoring, Events, Tiering, Site Replication, Settings), and 'Subscription' (License, Health, Performance). The 'Access Keys' option is highlighted with a red box. The main content area is titled 'Create Access Key'. It has fields for 'Access Key' (containing 'ulqg2f67OrI9mmTnphPA') and 'Secret Key' (redacted). A 'Restrict beyond user policy' toggle switch is set to 'OFF'. Below these are 'Clear' and 'Create' buttons. To the right, a panel titled 'Learn more about Access Keys' contains links for 'Create Access Keys' and 'Assign Custom Credentials', along with detailed descriptions of how access keys inherit policies and support programmatic access.

Third, create the Bucket.

The screenshot shows the MinIO Object Store interface. The sidebar menu is identical to the previous screenshot. The 'Buckets' option in the 'Administrator' section is highlighted with a red box. The main content area is titled 'Create Bucket'. It has a 'Bucket Name' field containing 'casdoor'. Below it is a 'View Bucket Naming Rules' link. A 'Features' section includes 'Versioning' (ON), 'Object Locking' (ON), and 'Quota' (OFF). At the bottom are 'Clear' and 'Create Bucket' buttons. To the right, a panel titled 'Buckets' explains that buckets organize objects like a folder in a filesystem. It details 'Versioning' for keeping object versions, 'Object Locking' for preventing deletion, and 'Quota' for data limits. It also mentions 'Retention' for object deletion rules.

## Step 2: Create a MinIO provider in Casdoor

Now create a MinIO provider in Casdoor. Fill in the necessary information.

Name	Name in MinIO
Category	choose Storage
Type	choose MinIO
Client ID	Access Key obtained from Step 1
Client secret	Secret Key obtained from Step 1
Endpoint	API address obtained from Step 1
Bucket	Bucket obtained from Step 1

## Step 3: Use MinIO storage service in your application

Now you can use the MinIO storage service in your application.

# Alibaba Cloud OSS

① 备注

This is an example of Alibaba Cloud OSS.

The AccessKey is your key to access Alibaba Cloud API with full account permissions.

To create an AccessKey, follow the instructions in the [Alibaba Cloud workbench](#).

Next, create the OSS service:

创建 Bucket

⑦ 创建存储空间 X

注意: Bucket 创建成功后, 您所选择的 存储类型、区域、存储冗余类型 不支持变更。

Bucket 名称	mycasdoor	9/63
地域	华北2 (北京)	▼
相同区域内的产品内网可以互通; 订购后不支持更换区域, 请谨慎选择。		
Endpoint	oss-cn-beijing.aliyuncs.com	

Fill in the necessary information in Casdoor and save:

Name <a href="#">?</a> :	provider_storage_aliyun_oss
Display name <a href="#">?</a> :	Storage Aliyun OSS
Category <a href="#">?</a> :	Storage
Type <a href="#">?</a> :	Aliyun OSS
Client ID <a href="#">?</a>	LTAIxFoNpNAnPoiT
Client secret <a href="#">?</a>	***
Endpoint <a href="#">?</a> :	oss-cn-beijing.aliyuncs.com
Endpoint (Intranet) <a href="#">?</a> :	oss-cn-beijing-internal.aliyuncs.com
Bucket <a href="#">?</a> :	casbin
Domain <a href="#">?</a> :	<a href="https://cdn.casbin.com/casdoor/">https://cdn.casbin.com/casdoor/</a>
Provider URL <a href="#">?</a> :	<a href="https://oss.console.aliyun.com/bucket/oss-cn-beijing/casbin/object">https://oss.console.aliyun.com/bucket/oss-cn-beijing/casbin/object</a>

You can now use Alibaba Cloud cloud storage services in your application.

# Tencent Cloud COS

ⓘ 备注

This is an example of Tencent Cloud COS.

## Fill in the necessary information in Casdoor

There are five required fields: `client ID`, `Client secret`, `Endpoint`, `Bucket`, and `Region ID`. The corresponding relationship to the Tencent Cloud COS account is as follows:

Name	Name in Tencent	Required
Client ID	SecretId	Yes
Client secret	SecretKey	Yes
Endpoint	Endpoint	Yes
Bucket	BucketName	Yes
Path prefix		
Domain	CDNDomain	
Region ID	Region	Yes

## Tencent Cloud COS information

- SecretId and SecretKey

The screenshot shows the Tencent Cloud API Key Management interface. On the left sidebar, under '访问管理' (Access Management), 'API密钥管理' (API Key Management) is selected. The main content area is titled 'API密钥管理' (API Key Management). It contains two sections: '安全提示' (Security Tips) and '使用提示' (Usage Tips). Below these is a table titled '新建密钥' (Create New Key) showing the details of a newly created key:

APPID	密钥	创建时间	最近访问时间	状态
1319606438	<div style="border: 1px solid red; padding: 2px;">SecretId: AKIDdAlMuNrJn8GHI6mLi6NSWbheNr7MViec</div> <div style="border: 1px solid red; padding: 2px;">SecretKey: ***** 显示</div>	2023-07-22 19:01:...	2023-07-22 22:09	已启用

- Endpoint, BucketName, and Region

The screenshot shows the Tencent Cloud COS bucket management interface. On the left sidebar, under '存储' (Storage), '桶管理' (Bucket Management) is selected. The main content area shows basic information about a bucket named 'casdoor-1319606438':

基本信息	域名信息
存储桶名称: casdoor-1319606438 (存储桶不支持改名)	访问域名: https://casdoor-1319606438.cos.ap-guangzhou.myqcloud.com 使用访问域名进行内网访问
所属地域: 广州 (中国) (ap-guangzhou)	自定义CDN加速域名: 0条
创建时间: 2023-07-22 18:57:50	自定义源站域名: 0条
访问权限: 私有读写	全球加速域名: 未开启
	静态网站域名: 未开启

- (Optional) CDNDomain

You can refer to the official documentation for configuration: [Config CDN](#)

## Configure Casdoor provider

The screenshot shows two overlapping interfaces. The top interface is the 'API密钥管理' (API Key Management) in the Tencent Cloud console, displaying an access key with a secret key. The bottom interface is the 'Casdoor provider' configuration page in the Casdoor UI, showing fields like Name, Display name, Organization, Category, Type, Client ID, Client secret, Endpoint, Bucket, Path prefix, Domain, Region ID, and Provider URL.

Red arrows point from specific fields in the Casdoor provider configuration to their corresponding values in the API Key Management interface:

- Client ID: Points to the 'Secret AKIDAMuJn8GH8mLjNSWbheN7Mveic' field in the API Key Management interface.
- Client secret: Points to the 'SecretKey' field in the API Key Management interface.
- Endpoint: Points to the 'Endpoint casdoor-1319606438.cos.ap-guangzhou.myqcloud.com' field in the Casdoor provider configuration.
- Bucket: Points to the 'Bucket casdoor-1319606438' field in the Casdoor provider configuration.
- Region ID: Points to the 'Region ID ap-guangzhou' field in the Casdoor provider configuration.

# Synology NAS

ⓘ 备注

This is an example of Synology NAS.

## Fill in the necessary information in Casdoor

There are five required fields: `client ID`, `Client secret` and `Endpoint`. The corresponding relationship to the Synology NAS account is as follows:

Name	Name in Tencent	Required
Client ID	SecretId	Yes
Client secret	SecretKey	Yes
Endpoint	Endpoint	Yes
Bucket		
Path prefix		
Domain		
Region ID		

## Configure Casdoor provider

The screenshot illustrates the configuration of a Casdoor provider (provider\_synology) and its connection to a Synology NAS. The provider configuration includes:

- Name: provider\_synology
- Display name: New Provider - synology
- Organization: built-in
- Category: Storage
- Type: Synology
- Client ID: password\_ls\_xiaokonglong
- Client secret: \*\*\*
- Endpoint: http://169.254.0.2:5000

The Synology Assistant window shows the following details:

- Endpoint: http://169.254.0.2:5000 (highlighted in red)
- SynologyNAS [169.254.0.2] Manual 已就绪 00.11.32.C:A6.03 6.1.7-15284 DS3617xs A8DDN02468

Red arrows highlight the Client ID, Client secret, and Endpoint fields in the provider configuration, pointing to their respective values in the Synology Assistant window.

You can refer to the official documentation for configuration: [link](#)



&gt; 提供商

&gt; SAML

# SAML

## 概述

使用来自支持SAML 2.0 的外部身份提供商的身份

## Custom

Configure your SAML Custom Provider

## Keycloak

使用 Keycloak 验证用户

## Alibaba Cloud IDaaS

Using Alibaba Cloud IDaaS to authenticate users

# 概述

Casdoor can be configured to support user login to the UI using identities from external identity providers that support SAML 2.0. In this configuration, Casdoor never stores any credentials for the users.

Now, Casdoor supports multiple SAML application providers. Icons of the providers will be displayed on the login page after being added to Casdoor. Here are the providers that Casdoor supports:

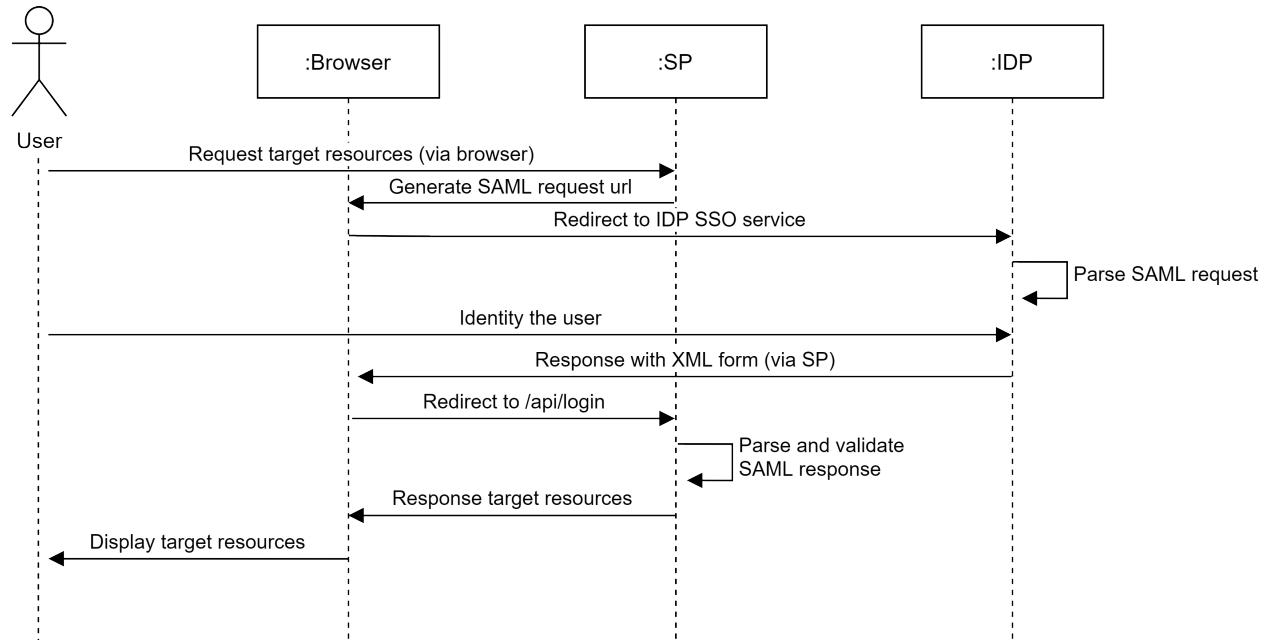
Alibaba Cloud IDaaS	Keycloak	Custom
		
		

# 条款

- 身份提供商 (IDP) —— 储存身份数据库并向Casdoor提供身份和认证服务的服务。
- Service Provider (SP) - The service that provides resources to the end user, in this case, the Casdoor deployment.
- 申述消费者服务——身份提供者提出的SAML断言的消费者。

# SAML 集成工作方式

When using SAML SSO, users log into Casdoor via the identity provider without ever passing credentials to Casdoor. 进展情况见下图表。



# Custom

Casdoor supports configuring SAML Custom Provider, and you can use Casdoor as a Service Provider (SP) to connect any Identity Provider (IDP) that support SAML 2.0 protocol.

## Step1. Get the metadata of IDP

First, you need to obtain the metadata of IDP, which is a XML document used to describe the configuration information of the services provided by IDP. It needs to include information such as `EntityID`, `SSO Endpoint`, etc.

Some IDPs, such as Keycloak, require SP information to provide metadata. You can refer to the document [Keycloak](#).

You can use [oktadev](#) to test the SAML Custom Provider, here is the [metadata](#).

## Step2. Configure SAML Custom Provider

After obtain the metadata of IDP, create a SAML Custom Provider and fill the neccessary information.

Field	Description
Category	Choose <a href="#">SAML</a>

Field	Description
Type	Choose <code>Custom</code>
Favicon.URL	The URL of the IDP logo
Metadata	The metadata of IDP

Then click `Parse` button, and fields `Endpoint`, `IdP`, `Issuer URL`, `SP ACS URL` and `SP Entity ID` will be automatically parsed.

Name [?](#):

Display name [?](#):

Organization [?](#):

Category [?](#):

Type [?](#):  Custom

User mapping [?](#): ?:

Username [?](#):

Display name [?](#):

Email [?](#):

Avatar [?](#):

Favicon [?](#):

Preview: 

Client ID [?](#):

Client secret [?](#):

Sign request [?](#):

Metadata [?](#):   
[Parse](#)

Endpoint [?](#):

IdP [?](#):

Issuer URL [?](#):

SP ACS URL [?](#):

SP Entity ID [?](#):

Provider URL [?](#):

Finally, add the SAML Custom Provider to **Providers** of the application.

Providers <a href="#">?</a>	Add <a href="#">?</a>	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action
provider_storage_minio_s3		Storage		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">^</a> <a href="#">v</a> <a href="#">o</a>
provider_oauth_jarck		OAuth		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">^</a> <a href="#">v</a> <a href="#">o</a>
provider_email_rq		Email		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">^</a> <a href="#">v</a> <a href="#">o</a>
provider_web3_metamask		Web3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">^</a> <a href="#">v</a> <a href="#">o</a>
provider_google_oauth		OAuth		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">One Tap</a> <a href="#">^</a> <a href="#">v</a> <a href="#">o</a>
provider_web3_onboard		Web3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">^</a> <a href="#">v</a> <a href="#">o</a>
saml_custom_provider_oktadev		SAML		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">^</a> <a href="#">v</a> <a href="#">o</a>
saml_custom_provider_keycloak		SAML		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">^</a> <a href="#">v</a> <a href="#">o</a>

# Keycloak

The JBoss [Keycloak](#) system is a widely used and open-source identity management system that supports integration with applications via SAML and OpenID Connect. It can also operate as an identity broker between other providers such as LDAP or other SAML providers and applications that support SAML or OpenID Connect.

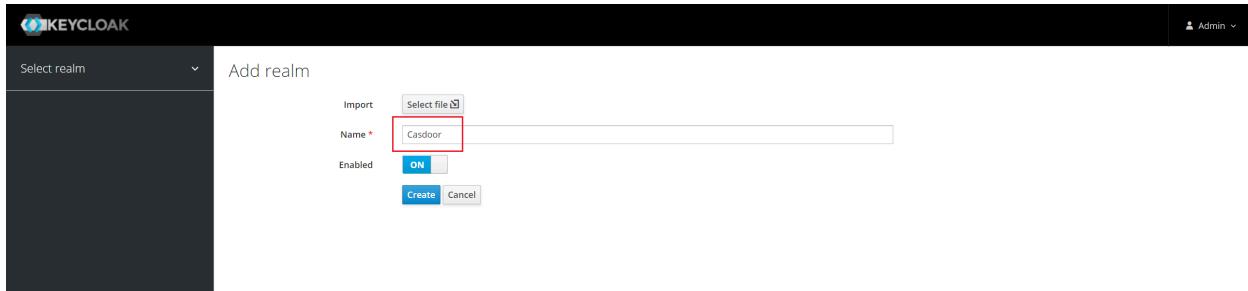
Here is an example of how to configure a new client entry in Keycloak and configure Casdoor to use it to allow UI login by Keycloak users who are granted access via Keycloak configuration.

## Configure Keycloak

For this example, let's make the following configuration choices and assumptions:

- Assume that you are running Casdoor in dev mode locally. The Casdoor UI is available at `http://localhost:7001` and the server is available at `http://localhost:8000`. Replace with the appropriate URL as needed.
- Assume that you are running Keycloak locally. The Keycloak UI is available at `http://localhost:8080/auth`.
- 在此基础上，用于此部署的SPACS URL将是：`http://localhost:8000/api/acs`。
- Our SP Entity ID will use the same URL: `http://localhost:8000/api/acs`.

You can use the default realm or create a new realm.



# 在 Keycloak 中添加客户端条目

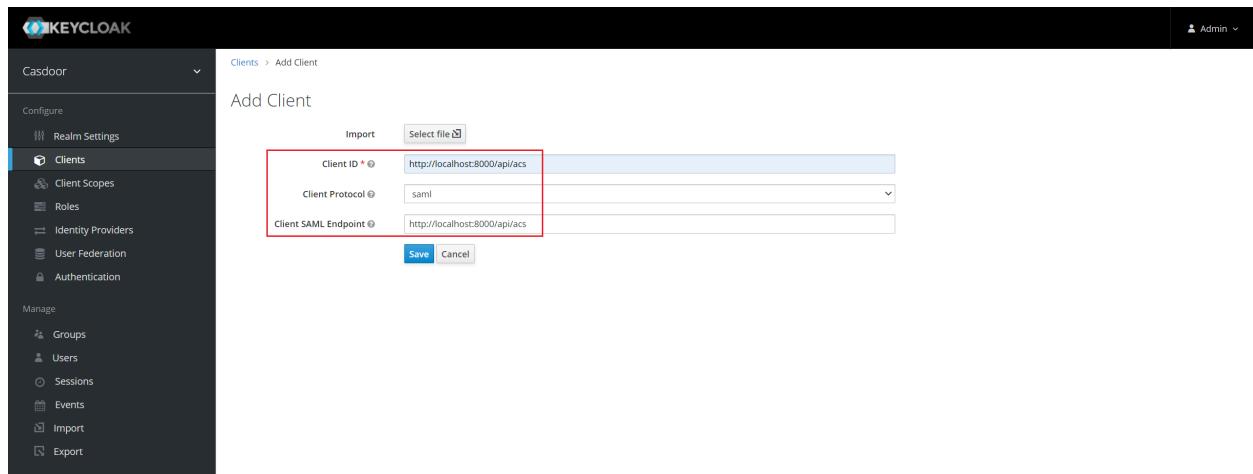
## ① 信息

For more details about Keycloak Clients, refer to the [Keycloak documentation](#).

在菜单中点击 **客户端** 然后点击 **创建** 去到 **添加客户端** 页面。 Fill in the fields as follows:

- **客户端 ID:** `http://localhost:8000/api/acs` - 这将是以后在 Casdoor 配置中使用的 SP 实体ID。
- **Client Protocol:** `saml`.
- **Client SAML Endpoint:** `http://localhost:8000/api/acs` - This URL is where

you want the Keycloak server to send SAML requests and responses. Generally, applications have one URL for processing SAML requests. Multiple URLs can be set in the Settings tab of the client.



单击 **Save** (保存)。此动作创建客户端并将您带到 **设置** 选项卡。

The following are part of the settings:

1. 名称 - Casdoor. This is only used to display a friendly name to Keycloak users in the Keycloak UI. You can use any name you prefer.
2. Enabled - Select  on.
3. Include Authn Statement - Select  on.
4. Sign Documents - Select  on.
5. Sign Assertions - Select  off.
6. Encrypt Assertions - Select  off.
7. Client Signature Required - Select  off.
8. Force Name ID Format - Select  on.
9. Name ID Format - Select  username.
10. 有效重定向 URI - 添加  http://localhost:8000/api/acs.
11. Master SAML 处理 URL -  http://localhost:8000/api/acs.

## 12. 精良的谷物SAML端点配置

- i. 声明消费者服务公开绑定URL - `http://localhost:8000/api/acs`。
- ii. 声明消费者服务重定向绑定URL - `http://localhost:8000/api/acs`。

保存该配置。

KEYCLOAK Admin

Clients > http://localhost:8000/api/acs

Http://localhost:8000/api/acs

Configure

Realm Settings

Clients

Client Scopes

Roles

Identity Providers

User Federation

Authentication

Manage

Groups

Users

Sessions

Events

Import

Export

Settings Roles Client Scopes Mappers Scope Sessions Offline Access Clustering Installation

Client ID: http://localhost:8000/api/acs

Name: Casdoor

Description:

Enabled: ON

Always Display in Console: OFF

Consent Required: OFF

Login Theme:

Client Protocol: saml

Include AuthnStatement: ON

Include OneTimeUse Condition: OFF

Force Artifact Binding: OFF

Sign Documents: ON

Optimize REDIRECT signing key lookup: OFF

Sign Assertions: OFF

Signature Algorithm: RSA\_SHA256

SAML Signature Key Name: KEY\_ID

Canonicalization Method: EXCLUSIVE

Encrypt Assertions: OFF

Client Signature Required: OFF

Force POST Binding: OFF

Front Channel Logout: ON

Force Name ID Format: ON

Name ID Format: username

Root URL:

Valid Redirect URIs: http://localhost:8000/api/acs

Base URL:

Master SAML Processing URL: http://localhost:8000/api/acs

IDP Initiated SSO URL Name:

IDP Initiated SSO Relay State:

Fine Grain SAML Endpoint Configuration

Assertion Consumer Service POST Binding URL: http://localhost:8000/api/acs

Assertion Consumer Service Redirect Binding URL: http://localhost:8000/api/acs

Logout Service POST Binding URL:

Logout Service Redirect Binding URL:

Logout Service ARTIFACT Binding URL:

Artifact Binding URL:

Artifact Resolution Service:

Advanced Settings

Authentication Flow Overrides

Save Cancel



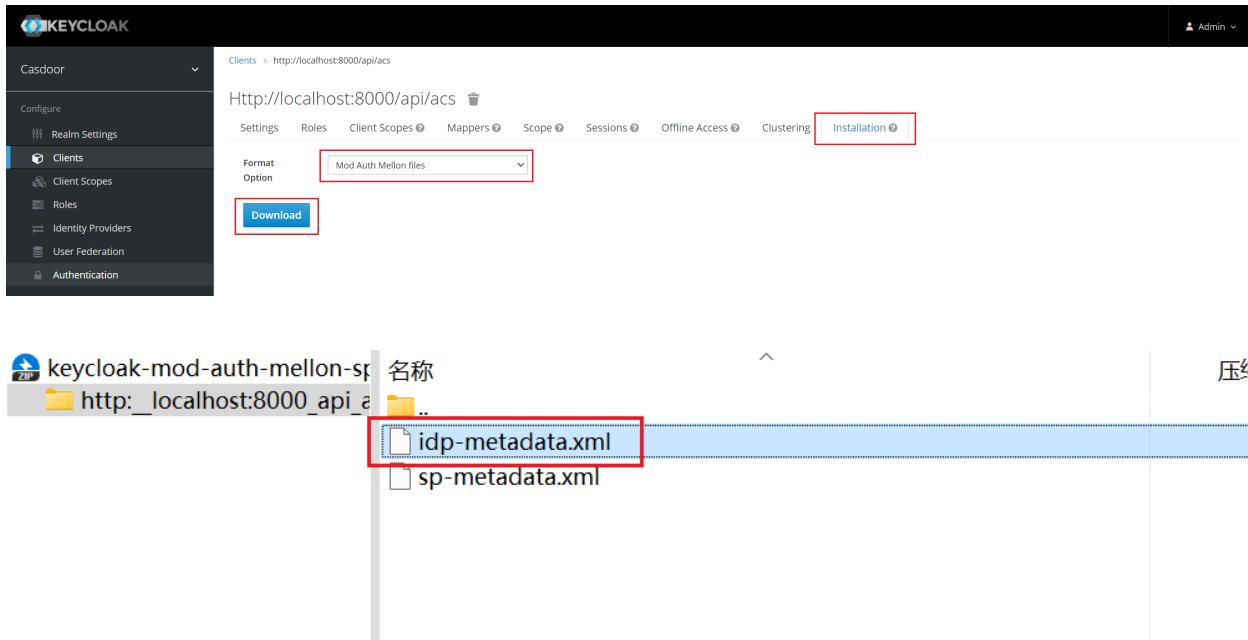
提示

If you want to sign the authn request, you need to enable the **Client Signature Required** option and upload the certificate generated by yourself. The private key and certificate used in Casdoor, `token_jwt_key.key` and `token_jwt_key.pem`, are located in the `object` directory. In Keycloak, you need to click the **Keys** tab, click the **Import** button, select **Archive Format** as **Certificate PEM**, and upload the certificate.

点击 安装 标签页。

For Keycloak <= 5.0.0, select Format Option - SAML Metadata IDPSSODescriptor and copy the metadata.

对于Keycloak 6.0.0+, 选择格式选项 - **Mod Mellon 文件** 并点击 **下载**。Unzip the downloaded.zip, locate `idp-metadata.xml`, and copy the metadata.



The image consists of two screenshots. The top screenshot shows the Keycloak Admin UI for a client named 'Casdoor'. The 'Installation' tab is selected. A dropdown menu is open, showing 'Mod Auth Mellon files' as the selected option. A red box highlights this dropdown and the 'Download' button below it. The bottom screenshot shows the contents of a ZIP file named 'keycloak-mod-auth-mellon-sp'. Inside, there is a folder structure for 'http://localhost:8000/api/acs'. The file 'idp-metadata.xml' is highlighted with a red box, indicating it is the metadata file to be copied.

# 在Casdoor配置

在 Casdoor 中创建一个新的提供商。

选择分类为 SAML, 输入 Keycloak. Copy the content of metadata and paste it into the Metadata field. The values of Endpoint, IdP, and Issuer URL will be generated automatically after clicking the Parse button. Finally, click the Save button.



如果您在 Keycloak 中启用 **客户端签名需要** 选项并上传证书, 请在 Casdoor 中启用 **签名请求** 选项。

Name: keycloak-casdoor  
Display name: keycloak-casdoor  
Category: SAML  
Type: Keycloak  
Client ID:  
Client secret:  
Sign request:   
Metadata:  
<md:EntityDescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" xmlns:ds="http://www.w3.org/2000/09/xmldsig#"  
entityID="http://localhost:8080/auth/realm/casdoor"><md:IDPSSODescriptor WantAuthnRequestsSigned="true" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor use="signing">  
<ds:KeyInfo><ds:KeyName>zqm-3k76jG-na5Zc3ulPDl7bp-4wYtMbWMfPzwqUHAY</ds:KeyName><ds:X509Data>  
<ds:X509Certificate>MIICnTCAYUCBgf9pAmxSDANBgkqhkiG9w0BAQsfADASMRAwDgYDVQQDAdjYXNkb29yMB4XDThMTIxMDExMDg1OFoXDTMxMTIxMDExMTA2OFowEjEQMA4GA1UEAwwHY2FzZG9vcjCCASlwDQYKoZlhvcNA  
Parse  
Endpoint: http://localhost:8080/auth/realm/casdoor/protocol/saml  
IdP: MIICnTCAYUCBgf9pAmxSDANBgkqhkiG9w0BAQsfADASMRAwDgYDVQQDAdjYXNkb29yMB4XDThMTIxMDExMDg1OFoXDTMxMTIxMDExMTA2OFowEjEQMA4GA1UEAwwHY2FzZG9vcjCCASlwDQYKoZlhvcNAQEBBQADggEPADCCAQ  
Issuer URL: http://localhost:8080/auth/realm/casdoor  
SP ACS URL: http://localhost:8000/api/acs  
SP Entity ID: http://localhost:8000/api/acs  
Provider URL: https://github.com/organizations/xxx/settings/applications/1234567

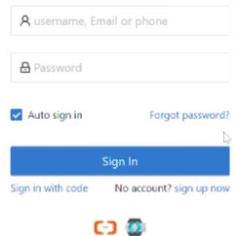
编辑您想要在 Cassdoor 中配置的应用程序。 Select the provider you just added and click the Save button.

Providers	Add	Name	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
casdoor-idaas		SAML							
keycloak-casdoor		SAML							

# 验证效果

Go to the application you just configured and you will find a Keycloak icon on the login page.

Click the icon and you will be redirected to the Keycloak login page. After successful authentication, you will be logged into Casdoor.

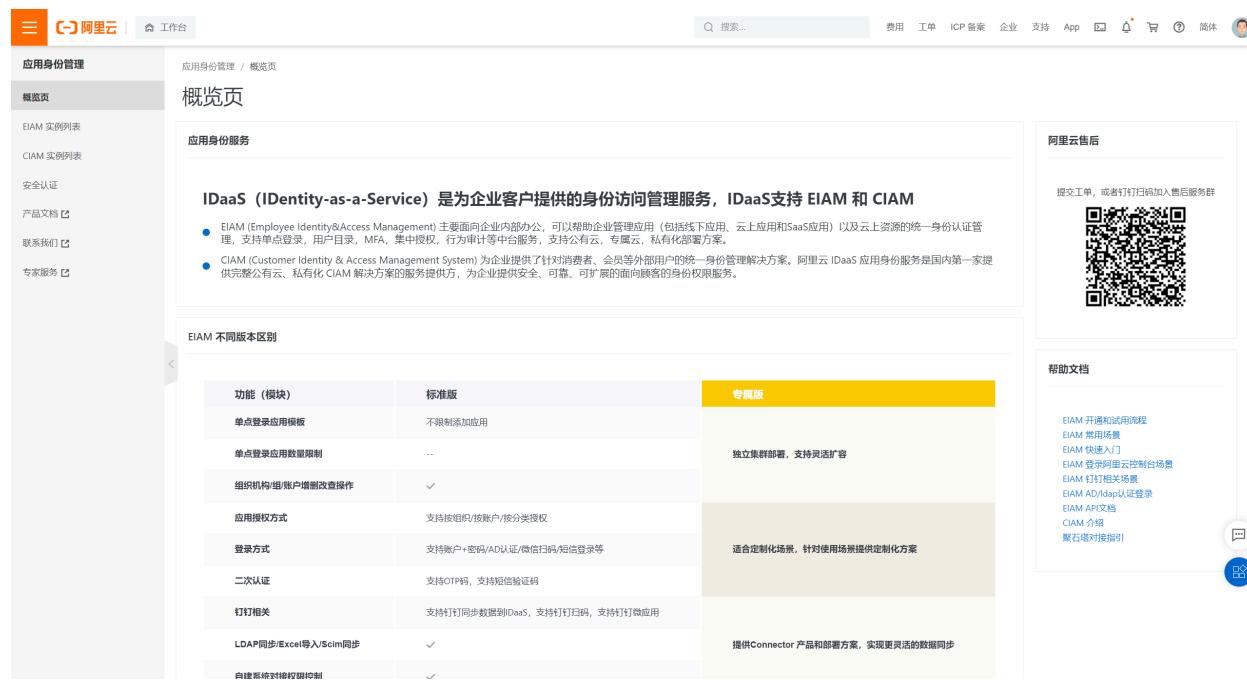


A screenshot of the Casdoor login page. It features a large Casbin logo at the top. Below it are two input fields: one for "username, Email or phone" and one for "Password". Underneath these fields are two buttons: "Auto sign in" (with a checked checkbox) and "Forgot password?". A central "Sign In" button is prominently displayed. At the bottom of the form, there are links for "Sign in with code" and "No account? sign up now". To the right of the "Sign In" button, there are icons for SAML and OpenID Connect.

# Alibaba Cloud IDaaS

## Create SAML application in Alibaba Cloud IDaaS

Login to the [Alibaba Cloud management console](#), search and go to the Application Identity Service (IDentity-as-a-Service, IDaaS).



The screenshot shows the Alibaba Cloud Management Console interface. The top navigation bar includes links for Home, Workbench, Search, and various services like Costs, Bills, ICP备案, Business, Support, App, and Help. The main content area is titled "概览页" (Overview Page) under "应用身份管理" (Application Identity Management). It features a section about IDaaS (Identity-as-a-Service) and compares EIAM (Employee Identity & Access Management) and CIAM (Customer Identity & Access Management System). A table compares EIAM Standard Edition and Premium Edition across various features like single sign-on, user quota, and integration with DingTalk. On the right side, there's a "阿里云售后" (AliCloud After-sales) section with a QR code for service requests and a "帮助文档" (Help Documentation) sidebar with links to EIAM documentation.

点击 **EIAM 实例列表** 并打开免费版本。

打开后将自动创建并运行一个实例。点击实例名称或 管理 按钮来输入 IDaaS 管理控制台。

输入了 IDaaS 管理控制台后，点击 添加应用程序, 搜索 SAML, 然后点击 添加应用程序

应用图标	应用名称	应用ID	标签	描述	应用类型	操作
	云安全访问服务SASE	plugin_saml_casas	SASE, SD-WAN安全, ZTNA, SAML	云安全访问服务SASE (Cloud Access Service Edge) 是阿里云首个一站式SaaS化办公安全管控平台，企业无需再投资复杂的自建的安全硬件设备，即可快速构建云上零信任内网访问、上网行为管理与审计、数据防泄漏、终端安全防护、SD-WAN安全、办公访问加速等在内的办公安全体系。兼容标准的SAML协议，可无缝兼容阿里云DaaS的身份体系，用户可根据在DaaS内配置的员工身份与组织架构，在云安全访问服务SASE产品中配置安全策略。	Web应用	<a href="#">添加应用</a>
	阿里云RAM-用户SSO	plugin_alyun	SSO, SAML, 阿里云	基于 SAML 协议，实现由 DaaS 单点登录到阿里云控制台；使用该模板，需要在 RAM 中为每个用户单独创建 RAM 用户，DaaS 账户和 RAM 账户通过映射实现单点登录到 RAM。	Web应用	<a href="#">添加应用</a>
	阿里云RAM-角色SSO	plugin_alyun_role	SSO, SAML, Aliyun	基于 SAML 协议，实现由 DaaS 单点登录到阿里云控制台；使用该模板，需要在 RAM 中创建 RAM 角色，不需要为每个用户单独创建 RAM 用户，DaaS 账户和 RAM 角色通过映射实现单点登录到 RAM。	Web应用	<a href="#">添加应用</a>
	阿里邮箱	plugin_aimail	SSO, 用户同步, SAML, 阿里云, 邮箱	基于 SAML 协议，实现由 DaaS 到阿里邮箱的单点登录和用户同步。	Web应用	<a href="#">添加应用</a>
	WordPressSaml	plugin_wordpress_saml	SSO, SAML, CMS	WordPress 是全世界最广泛的通用 CMS (Content Management System, 内容管理系统)，它通过非常强大的插件系统和界面自然的操作界面，允许千万技术或非技术人员生产、管理各种类型的网站。从商业网站、政府网站到个人博客、主题论坛，WordPress 所支持的形式非常多样。IDaaS 支持通过 SAML 协议单点登录到 WordPress 网站。	Web应用	<a href="#">添加应用</a>
	SAML	plugin_saml	SSO, SAML	SAML (Security Assertion Markup Language, 安全断言标记语言，版本 2.0) 基于 XML 会议，使用包含断言 (Assertion) 的安全令牌，在授予权 (Identity) 和资源方 (应用) 之间传递身份信息。实现基于网络安全的单点登录。SAML 协议是成熟的认证协议，在国内外的公有云和私有云中有非常广泛的运用。	Web应用	<a href="#">添加应用</a>
	GitLab	plugin_saml_gitlab	SSO, SAML	GitLab 是一个用于仓库管理系统的开源项目。使用 Git 作为代码管理工具，并在此基础上搭建起来的 web 服务。	Web应用	<a href="#">添加应用</a>

点击 **添加签名密钥**。

### 添加应用 (SAML)

X

导入SigningKey	添加SigningKey	别名	序列号	有效期	秘钥算法	算法长度	操作
暂无数据							

填写所有必需的信息并提交。

### 添加SigningKey

X

* 名称	CASDOOR-TEST
部门名称	请输入部门名称
公司名称	请输入公司名称
* 国家	CN
* 省份	Beijing
城市	请输入城市
* 证书长度	1024
* 有效期	3 年

**提交** **取消**

选择添加的签名密钥。

### 添加应用 (SAML)

X

导入SigningKey	添加SigningKey				
别名	序列号	有效期	秘钥算法	算法长度	操作
CN=CASDOOR-TEST, ST=Beijing, C=CN	3322747020095790430	1095	RSA	1024	<button>选择</button> <button>导出</button>

填写下面所需的所有信息并保存。

- IDP 身份ID：保持与签发者地址在 Casdoor 中相同。
- SP 实体 ID & SP ACS URL (SSO 定位)：现在填写您想要的任何东西。完成 Casdoor 配置后，您需要修改。
- 描述属性：直接填充为用户名。
- 账户关联模式：账户协会

## 添加应用 (SAML)

X

图片大小不超过1MB

应用ID

idaas-cn-shanghai-pvl0hq0ojppugin\_saml

\* 应用名称

CASDOOR-SAML

\* IDP IdentityId

CASDOOR

IDP IdentityId is required

\* SP Entity ID

http://localhost

SP Entity ID is required

\* SP ACS URL(SSO Location)

http://localhost

\* NameIdFormat

urn:oasis:names:tc:SAML:2.0:nameid-format:transient

v

\* Binding

POST

v

SP 登出地址

请输入 SP 登出地址

Assertion Attribute

username

应用子账户

v

-

+

断言属性。设值后，会将值放入SAML断言中。名称为自定义名称，值为账户的属性值。

Sign Assertion



IDaaS发起登录地址

IDaaS发起登录地址

以 http://、https:// 开头，填写后使用 IDaaS 发起登录将会跳转到该地址，而不会使用 SAML 的idp发起登录流程

\* 账户关联方式

账户关联 (系统按主子账户对应关系进行手动关联，用户添加后需要管理员审批)

账户映射 (系统自动将主账户名称或指定的字段映射为应用的子账户)

提交

取消

## 帐户授权 & 关联

在SAML应用成功添加后，授权提示会高亮。现在不要授权它，添加一个帐户，然后授权它。

转到组织和组，然后点击新帐户。

The screenshot shows the Alibaba Cloud Organization Structure Management interface. On the left sidebar, under the '账户' (Account) category, the '机构及组' (Organizational Units and Groups) option is selected and highlighted with a red box. In the main content area, there is a large callout box titled '机构及组' (Organizational Units and Groups) with the following text:  
管理员在当前页面对组织架构、部门及其包含的组、账户进行管理，也可以使用AD、LDAP或Excel文件的方式配置导入或同步。  
在左侧的组织架构树中，可以右键点击某个部门对其进行操作，也可以左键选择某个部门，并在右侧为其进行创建账户、创建组、创建部门等操作。

The main panel is titled '组织架构' (Organizational Structure) and contains a search bar with '新账账户' (Create New Account) highlighted by a red box. Below the search bar is a table with one row of data:

编号	账户名称	显示名称	类型	目录	操作
1	idaas_manager	默认管理员	自建账户	/	<a href="#">修改</a> <a href="#">账户同步</a> <a href="#">同步记录</a>

At the bottom right of the table, there is a pagination bar showing '共 1 条' (1 item), page number '1', and a '10 条/页' (10 items per page) dropdown.

填写所有必需的信息并提交。

## 新建账户

X

### 账户属性

### 扩展属性

### 父级组

父级

阿里云IDaaS

\* 账户名称

casdoor

账户名称不能以特殊字符开始，可包含大写字母、小写字母、数字、中划线(-)、下划线(\_)、点(.)，长度至少 4 位

\* 显示名称

casdoor

\* 密码

\*\*\*\*\*

密码中必须包含大小写字母+数字+特殊字符的组合;长度至少 10 位，密码不能包含"<"和">"。

邮箱

请输入有效的邮箱地址

手机号或邮箱至少填写一个。

手机号

+86 ▾ 151 123456789

手机号或邮箱至少填写一个。

外部ID

外部ID

IDaaS 平台中的唯一身份标识, 若不填将由系统自动生成。

过期时间

过期时间

不填将使用系统默认过期时间 2116-12-31

备注

备注

用户备注信息

提交

取消

转到 **应用程序授权**, 选择您想要授权的帐户, 然后点击 **保存**。

The screenshot shows the 'Application Authorization' section of the Alibaba Cloud console. On the left sidebar, '应用授权' (Application Authorization) is selected. In the main area, a table lists accounts under the application 'CASDOOR-SAML'. One account, 'casdoor', is highlighted with a red box. A blue '保存' (Save) button is at the bottom of the table. The search bar at the top right contains the text '自身赋予的权限资源'.

前往 应用程序列表, 点击 查看应用子账户, 然后点击 添加帐户关联。

The screenshot shows the 'Application List' section of the Alibaba Cloud console. On the left sidebar, '应用列表' (Application List) is selected. In the main area, a table lists applications. For the application 'CASDOOR-SAML', the '操作' (Operation) column shows a '授权' (Authorization) button and a '详情' (Details) button. The '查看详情' (View Details) link in the '账户信息 - 子账户' (Account Information - Sub-account) section is highlighted with a red box. The search bar at the top right contains the text 'IDaaS发起地址'.

The screenshot shows the Alibaba Cloud IDaaS application management interface. On the left, there is a sidebar with various navigation options like '概览', '快速入门', '应用' (selected), '账户', '认证', '授权', etc. The main content area has a title '应用列表 / 子账户' and a sub-section '← 子账户'. A modal window titled '子账户' is open, explaining what a子账户 is and how it relates to the main account. At the top right of the main page, there are buttons for '添加账户关联' (highlighted with a red box), '批量导入', and '批量导出'. Below the modal, there is a table titled 'CASDOOR-SAML' with columns: 账户名称, 显示名称, 子账户, 子账户密码, 是否关联, 审批状态, 关联时间, 操作. The table shows '暂无数据'. At the bottom right of the page, there are pagination controls.

填写需要关联的主账户和子账户，然后点击 **保存**。

主账户存在于IDaaS中，子账户是Cassdoor用户的ID。

The screenshot shows the 'Add Account Association' dialog box. It has two input fields: one for the primary account labeled '主账户' containing 'casdoor', and another for the child account labeled '子账户' containing '52908237-fa4c-4681-b636-a6afce22fb2e'. At the bottom, there are two buttons: a blue '保存' button and a white '返回' button.

## 导出 IDaaS 元数据

转到 **应用程序列表**, 点击 **查看应用程序详细信息** 然后点击 **导出 IDaaS SAML Metadada**

The screenshot shows the Alibaba Cloud IaaS application management interface. On the left, there is a sidebar with various navigation options under categories like Application, Account, Authentication, Authorization, Audit, and Others. The main area is titled "应用列表" (Application List) and shows a table with columns for Application Icon, Application Name, and Application ID. One row is selected, showing the details for "CASDOOR-SAML". The right side of the screen is titled "应用详情 (CASDOOR-SAML)" and displays the configuration details for this application. The "图标" (Icon) section shows a blue square icon with a white letter "S" and the text "SAML". The "应用ID" (Application ID) is "idaas-cn-shanghai-[REDACTED]\_jin\_saml". The "应用名称" (Application Name) is "CASDOOR-SAML". The "应用Uuid" (Application Uuid) is "d9bd59093c5c031b7abbb0efa849f7bRxS506SQ3y7". Below these, there are two tabs: "应用信息" (Application Information) and "认证信息" (Authentication Information). The "应用信息" tab contains fields like "应用的详细信息" (Detailed information about the application), "查看详情" (View details), "修改应用" (Edit application), and "删除应用" (Delete application). The "认证信息" tab contains fields like "应用的单点登录地址" (Single sign-on address for the application), "IDaaS发起地址" (Initiator address for IDaaS), and a link to "CASDOOR [Redacted] 导出 IDaaS SAML 元配置文件" (Export IDaaS SAML metadata configuration file). Other sections include "授权信息" (Authorization Information) and "审计信息" (Audit Information), which contain links to "查看日志" (View logs) and "查看同步记录" (View sync records). The "SP ACS URL" (Service Provider Assertion Consumer Service URL) is set to "http://localhost". The "IDP IdentityId" (Identity Provider Identity ID) is "CASDOOR [Redacted]". The "SP Entity ID" (Service Provider Entity ID) is "http://localhost". The "Binding" (Binding) is "POST". The "Sign Assertion" (Sign Assertion) is "是" (Yes). The "Assertion Attribute" (Assertion Attribute) is "username:APPLICATIONUSERNAME". The "IDaaS发起登录地址" (IDaaS initiator login address) is "[REDACTED]". The "SP发起地址" (SP initiator address) is "https://dmoykbkx.login.aliyunidaas.com/enduser/api/application/plugin\_saml/idaas-cn-shanghai-[REDACTED]/login\_saml/sp\_sso?SAMLRequest=xxx&RelayState=yyy".

# 在 Casdoor 配置

在 Casdoor 中创建一个新的提供商。

Select category as SAML, type as Alibaba Cloud IaaS. 复制元数据内容并粘贴到 元数据 输入。 端点, 的值, IdP 和 发行商URL 将在点击 分析 按钮后自动生成。

Name ⓘ :	casdoor-idaas	
Display name ⓘ :	casdoor-idaas	
Category ⓘ :	SAML	
Type ⓘ :	Aliyun IDaaS	
Client ID ⓘ :		
Client secret ⓘ :		
Metadata ⓘ :	<pre>&lt;md:SingleSignOnService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://aliyunidaas.com/charon/api/applications/plugin_idaas/charon"&gt;     &lt;md:SPSSOServiceBinding Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="https://aliyunidaas.com/charon/api/applications/plugin_idaas/charon"/&gt; &lt;/md:SingleSignOnService&gt; &lt;/md:IDPSSODescriptor&gt; &lt;/md:EntityDescriptor&gt;</pre>	
<input style="margin-right: 10px;" type="button" value="Parse"/> <span style="color: green;">Success</span>		
Endpoint ⓘ :	<a href="https://dvmoykbkx.login.aliyunidaas.com/enduser/api/application/plugin_saml/idaas-cn-shanghai">https://dvmoykbkx.login.aliyunidaas.com/enduser/api/application/plugin_saml/idaas-cn-shanghai</a>	
IdP ⓘ :	MIIByzCAVcBgAwIBAgIILhzEz2NMHV4wDQYJKoZIhvNAQEFBQAwNjELMAkGA1UEBhMCQ04xE DAOBgNVBAgT B0JlaWppbmcxFTATBgNVBAMTDENBU0RPTIItVEVTVDaeFw0yMTEyMDkwNzEyMTFaFw0yNDEyMDgwNzEyMTFaMDYxCzAJE	
Issuer URL ⓘ :	CASDOOR	
SP ACS URL ⓘ :	<a href="http://localhost:8000/api/acs">http://localhost:8000/api/acs</a>	<input type="button" value="Copy"/>
SP Entity ID ⓘ :	<a href="http://localhost:8000/api/acs">http://localhost:8000/api/acs</a>	<input type="button" value="Copy"/>
Provider URL ⓘ :	<a href="https://github.com/organizations/xxx/settings/applications/1234567">https://github.com/organizations/xxx/settings/applications/1234567</a>	

复制 SP ACS URL 和 SP 实体 ID 并点击 保存 按钮

编辑您想要在 Cassdoor 中配置的应用程序。选择刚刚添加的提供者，然后点击按钮 **保存**。

Providers	②
Providers	Add
Name	Category
casdoor-idaas	SAML
Preview	②
<a href="#">Test signup page.</a>	<a href="#">Test signin page.</a>

# Modify SAML application in Alibaba Cloud IDaaS

禁用应用程序，然后点击 **修改应用程序**。

The screenshot shows the Alibaba Cloud Application Management interface. On the left, there's a sidebar with categories like Application, Account, Authentication, Authorization, Audit, and Settings. The main area is titled '应用列表' (Application List) and shows a table with columns: 应用图标 (Application Icon), 应用名称 (Application Name), 应用ID (Application ID), 设备类型 (Device Type), 应用状态 (Application Status), 二次认证状态 (Two-factor authentication status), and 操作 (Operations). A red box highlights the '操作' column for the first application, 'CASDOOR-SAML'. Below the table, there are tabs for '应用信息' (Application Information), '认证信息' (Authentication Information), '账户信息 - 同步' (Account Information - Sync), and '账户信息 - 子账户' (Account Information - Sub-account). The '应用信息' tab is active, showing fields like '应用的详细信息' (Detailed information about the application), '应用的单点登录地址' (Single sign-on address), and 'IDaaS发起地址' (IDaaS initiation address). The '认证信息' tab shows 'SCIM协议设置以及把组织机构、组同步推送至应用' (SCIM protocol settings and pushing organization structures and groups to the application). The '账户信息' tabs show '同步机构' (Sync organization) and 'SCIM配置' (SCIM configuration). The '授权信息' (Authorization Information) and '审计信息' (Audit Information) tabs are also visible. At the bottom, there are buttons for '查看详情' (View details), '修改应用' (Edit application), and '删除应用' (Delete application). The '操作' (Operations) section includes '授权' (Authorization) and '详情' (Details) buttons. The bottom right corner has pagination controls: '共 1 条' (1 page), '1' (page number), and '10 条/页' (10 items per page).

填写 SP 实体 ID and SP ACS URL (SSO location) 将内容复制到Casdoor。 提交并启用应用程序。

## 修改应用 (CASDOOR-SAML)

X

图标



上传文件

图片大小不超过1MB

应用ID

idaas-cn-shanghai-...\\login\_saml

\* 应用名称

CASDOOR-SAML

\* IDP IdentityId

CASDOOR

IDP IdentityId is required

\* SP Entity ID

http://localhost:8000/api/acs

SP Entity ID is required

\* SP ACS URL(SSO Location)

http://localhost:8000/api/acs

\* NameIdFormat

urn:oasis:names:tc:SAML:2.0:nameid-format:transient

\* Binding

POST

SP 登出地址

请输入SP 登出地址

Assertion Attribute

username

应用子账户



断言属性。设值后，会将值放入SAML断言中。名称为自定义名称，值为账户的属性值。

Sign Assertion



IDaaS发起登录地址

IDaaS发起登录地址

以 http://、https://开头，填写后使用 IDaaS 发起登录将会跳转到该地址，而不会使用 SAML 的idp发起登录流程

## 验证效果

转到您刚刚配置的应用程序，您可以在登录页面找到一个图标。

Click the icon and jump to the Alibaba Cloud IDaaS login page, and then successfully login to the Casdoor after authentication.



---

username, Email or phone

Password

Auto sign in      [Forgot password?](#)

[Sign In](#)

[Sign in with code](#)      No account? [sign up now](#)

A row of social media icons for GitHub, LinkedIn, and others.

---

# 支付

## Overview

Add Payment providers to your application

## PayPal

Add PayPal as a payment provider to your application

## Stripe

Add Stripe payment provider to your application

## 支付宝

Add Alipay payment provider to your application

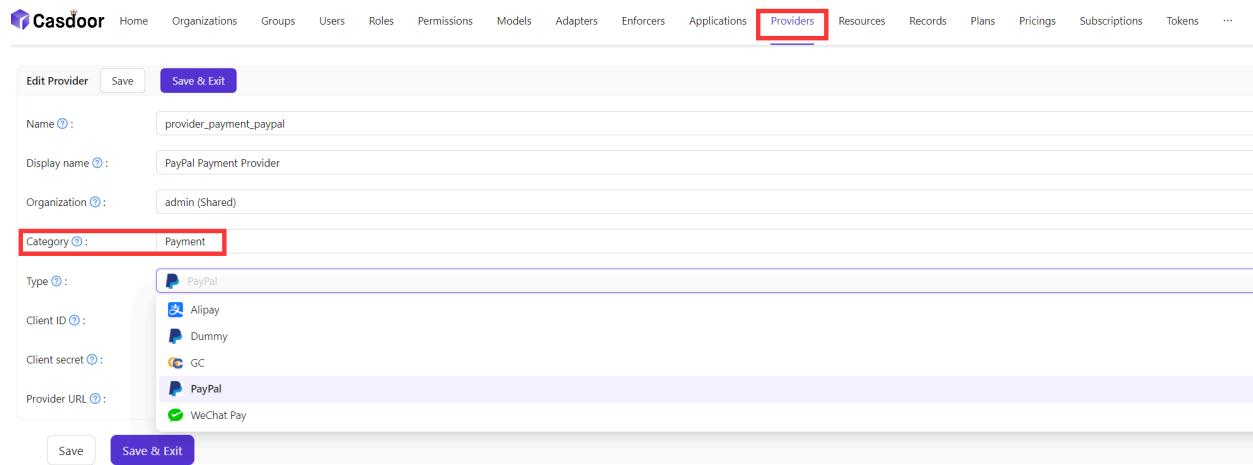


## WeChat Pay

Add WeChat Pay payment provider to your application

# Overview

If you want to use payment services in Casdoor, you need to create a Payment provider and add it to your products.



The screenshot shows the Casdoor interface for managing payment providers. The top navigation bar includes links for Home, Organizations, Groups, Users, Roles, Permissions, Models, Adapters, Enforcers, Applications, **Providers** (which is highlighted with a red box), Resources, Records, Plans, Pricings, Subscriptions, Tokens, and more. Below the navigation is a form titled "Edit Provider". The "Name" field contains "provider\_payment\_paypal", and the "Display name" field contains "PayPal Payment Provider". The "Organization" field is set to "admin (Shared)". The "Category" field is set to "Payment" (also highlighted with a red box). Under the "Type" section, "PayPal" is selected from a dropdown menu. Other options like "Alipay", "Dummy", "GC", and "WeChat Pay" are also listed. At the bottom of the form are two buttons: "Save" and "Save & Exit".

To learn how to configure a product, refer to [Product](#). After configuring a product, you can add Payment providers for the product so that users can purchase the product through the Payment providers.

# PayPal

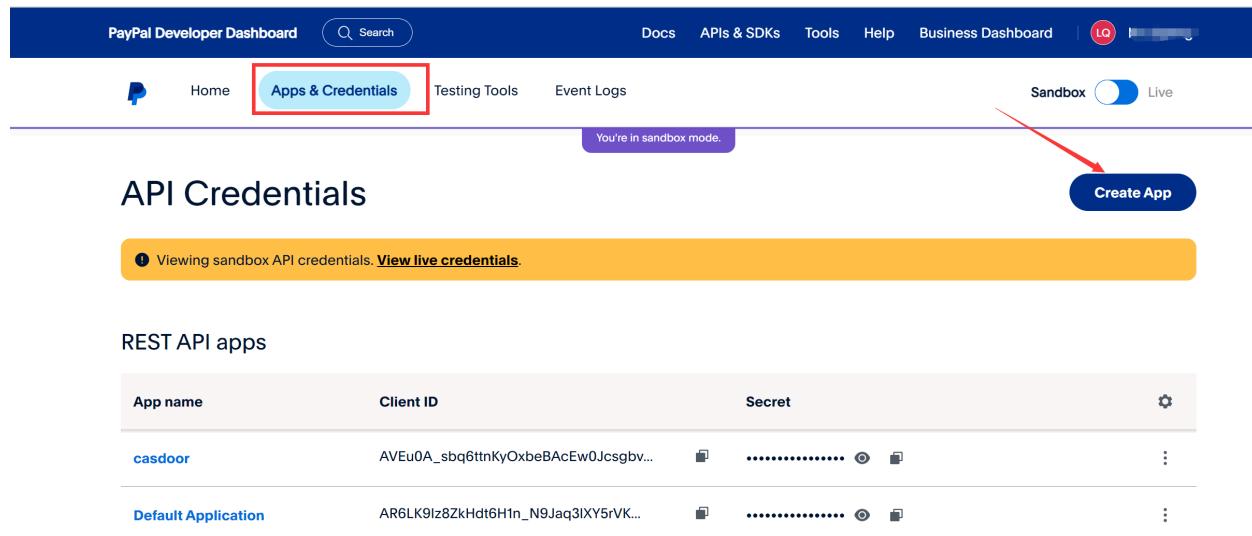
## ⓘ 备注

This is an example of how to configure the PayPal payment provider.

## Step 1: Create a PayPal application

First, you need to create an application in PayPal. To access the PayPal Developer site, you should have a PayPal business account. If you don't have an account, [create one first](#).

After you create a PayPal business account, log in to the [Developer Dashboard](#) using your account and then click on [Create App](#) under [Apps & Credentials](#).



The screenshot shows the PayPal Developer Dashboard interface. At the top, there's a navigation bar with links for Docs, APIs & SDKs, Tools, Help, and Business Dashboard. A user profile icon is also present. Below the navigation, there are tabs for Home, Apps & Credentials (which is highlighted with a red box), Testing Tools, and Event Logs. A status message "You're in sandbox mode." is displayed. On the right side, there's a toggle switch for "Sandbox" which is set to "Live". A prominent blue button labeled "Create App" is located on the right. The main content area is titled "API Credentials" and contains a yellow banner stating "Viewing sandbox API credentials. [View live credentials](#)". Below this, there's a section for "REST API apps" with a table showing two entries:

App name	Client ID	Secret	⋮
casdoor	AVEu0A_sbq6tnKyOxbeBAcEw0Jcsgbv...	.....	⋮
Default Application	AR6LK9lz8ZkHdt6H1n_N9Jaq3IXYrVK...	.....	⋮

You can find the [Client ID](#) and [Secret key](#) in the basic information of your

application.

← Back

## casdoor

Viewing sandbox API credentials. [View live credentials.](#)

### API credentials

App name	casdoor
Client ID	AVEu0A_sbq6trnKyOxbeBAcEw0Jcsgbv2JZvQAtK JFnaULI-EK-U2XIXcEpEouO9olknbU7c3m_lRT5
Secret key 1	***** :

+ Add Second Key

### Sandbox account info

[View details](#)

Sandbox URL	<a href="https://sandbox.paypal.com">https://sandbox.paypal.com</a>
Sandbox Region	C2
Email	sb-qqaiv26894991@business.example.com
Password	*****

### Features

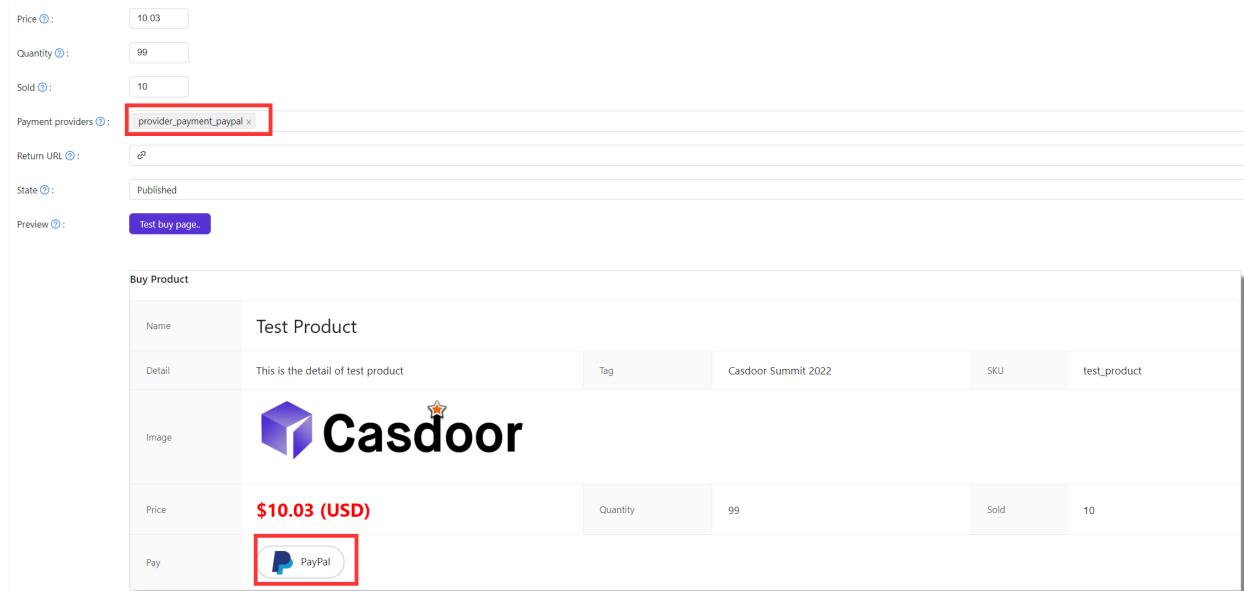
## Step 2: Create a PayPal payment provider

Next, create a PayPal payment provider in Casdoor. Fill in the necessary information:

Name	Name in PayPal
Category	Choose <a href="#">Payment</a>
Type	Choose <a href="#">PayPal</a>
Client ID	Use the <a href="#">Client ID</a> obtained from Step 1
Client secret	Use the <a href="#">Secret key</a> obtained from Step 1

## Step 3: Add the PayPal payment provider for your product

Finally, add the PayPal payment provider for your product so that users can purchase the product using PayPal.



The screenshot shows two parts of the Casdoor interface:

- Product Configuration Form:**
  - Price: 10.03
  - Quantity: 99
  - Sold: 10
  - Payment providers: provider\_payment\_paypal (highlighted with a red box)
  - Return URL: (empty)
  - State: Published
  - Preview: Test buy page...
- Buy Product Page:**

Buy Product					
Name	Test Product				
Detail	This is the detail of test product		Tag	Casdoor Summit 2022	SKU
Image	 Casdoor				
Price	\$10.03 (USD)		Quantity	99	Sold
Pay					

① 备注

The above operations are all performed in PayPal's `Sandbox` mode. If you want to use it in a live production environment, you need to create an application in PayPal's `Live` mode and set `runmode=prod` in Casdoor's configuration file `conf/app.conf`.

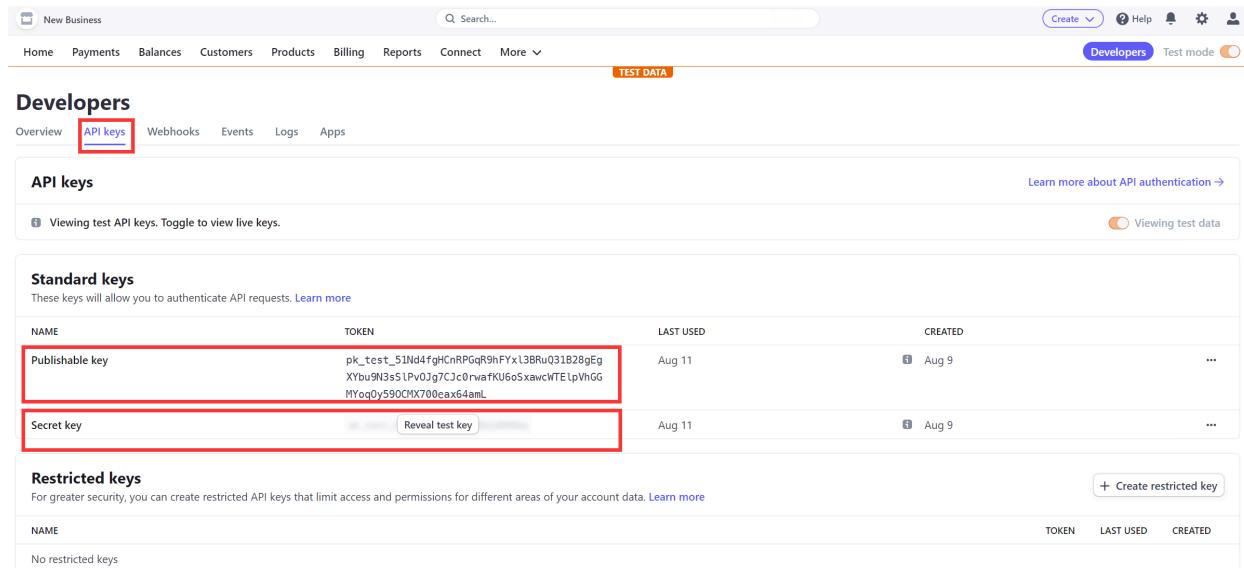
# Stripe

## ⓘ 备注

This is an example of how to configure a Stripe payment provider.

## Step 1. Get Publishable Key and Secret Key

First, you need to have an account at [Stripe](#). After creating a Stripe account, log in to the [Developer Dashboard](#) using your account credentials. You can find the [Publishable key](#) and [Secret key](#) under the [API keys](#) tab.



The screenshot shows the Stripe Developer Dashboard. At the top, there's a navigation bar with links for Home, Payments, Balances, Customers, Products, Billing, Reports, Connect, More, and a search bar. On the right, there are buttons for Create, Help, Notifications, Settings, and Profile. Below the navigation, a banner says "TEST DATA". The main area is titled "Developers" and has tabs for Overview, API keys (which is selected and highlighted with a red box), Webhooks, Events, Logs, and Apps. Under the API keys tab, there's a section for "Standard keys" with a note: "These keys will allow you to authenticate API requests. [Learn more](#)". It lists two keys: "Publishable key" and "Secret key", both of which are also highlighted with red boxes. The "Publishable key" row shows the token: pk\_test\_51Nd4fgHCnRPGgR9hFYxL3BRu031B28gEgXYbu9N3s51Pv0Jg7Cj0rvwfKU6o5xawchTElpVhGGMYogQy59OCMX700ea64amL. The "Secret key" row shows the token: sk\_test\_51Nd4fgHCnRPGgR9hFYxL3BRu031B28gEgXYbu9N3s51Pv0Jg7Cj0rvwfKU6o5xawchTElpVhGGMYogQy59OCMX700ea64amL. There are "Reveal test key" and "Reveal live key" buttons next to the tokens. Below this, there's a section for "Restricted keys" with a note: "For greater security, you can create restricted API keys that limit access and permissions for different areas of your account data. [Learn more](#)". It shows a table with one row: "No restricted keys". A button "+ Create restricted key" is located at the bottom right of this section.

## Step 2. Create a Stripe Payment provider

Next, create a Stripe Payment provider in Casdoor by filling in the necessary information.

Name	Name in Stripe
Category	choose <code>Payment</code>
Type	choose <code>Stripe</code>
Client ID	<code>Publishable key</code> obtained from Step 1
Client secret	<code>Secret key</code> obtained from Step 1

Edit Provider Save Save & Exit

Name ⓘ :	provider_payment_stripe
Display name ⓘ :	Stripe Payment Provider
Organization ⓘ :	admin (Shared)
Category ⓘ :	Payment
Type ⓘ :	<span>S</span> Stripe
Client ID ⓘ :	pk_test_51Nd4fgHCnRPGqR9hFYxl3BRuQ31B28gEgXYbu9N3sSIPvOJg7CJc0rwafKU6oSxawcWTElpVhGGMYoqOy59OCMX700eax64amL
Client secret ⓘ :	***
Provider URL ⓘ :	<code>Ø</code>

Save Save & Exit

# Step 3. Add the Stripe Payment provider for your product

Finally, add the Stripe Payment provider for your product so that users can purchase the product using Stripe.

Currency (1):

Price (1):

Quantity (1):

Sold (1):

Payment providers (1):  provider\_payment\_stripe x

Return URL (1):

State (1): Published

Preview (1): [Test buy page.](#)

**Buy Product**

Name	Test Product	Detail	This is the detail of test product	Tag	Casdoor Summit 2022	SKU	test_product
Image							
Price	\$10.04 (USD)						
Pay	 PayPal	 Stripe					



# 支付宝

## Step 1. Preparation

First, you need to have a merchant account at Alipay Open Platform.

Before accessing the Alipay, there are some preparations that need to be done.

You can refer to the documentation [preparation before access](#) for more information.

### 1.1 Get APPID

Login the Alipay Open Platform Console and [create an application](#).

How to get the `APPID` : [Alipay APPID Query Guide](#)

### 1.2 Configure Cert

Generate an RSA2 certificate based on the [document](#) and then you can obtain the `appPrivateKey.txt` and `appPublicKey.txt`.

Upload the certificate to the applicaiton and then you can download three files: `alipayRootCert.crt`, `appCertPublicKey.crt`, `alipayCertPublicKey.crt`.

Create a Cert called `App Cert` at Casoor:

Name	Name in Alipay
Type	choose Payment
Certificate	content of appCertPublicKey.crt
Private key	content of appPrivateKey.txt

Create a Cert called Root Cert at Casoor:

Name	Name in Alipay
Type	choose Payment
Certificate	content of alipayCertPublicKey.crt
Private key	content of alipayRootCert.crt

## Step 2. Create an Alipay Payment provider

Next, create an Alipay Payment provider in Casdoor by filling in the necessary information.

Name	Name in Alipay
Category	choose <b>Payment</b>
Type	choose <b>Alipay</b>
Client ID	<b>APPID</b> obtained from Step 1.1
Cert	<b>App Cert</b> configured at Step 1.2
Root Cert	<b>Root Cert</b> configured at Step 1.2

Edit Provider
Save
Save & Exit

---

Name <small>②</small> :	provider_payment_alipay
Display name <small>②</small> :	Alipay Payment Provider
Organization <small>②</small> :	admin (Shared)
Category <small>②</small> :	Payment
Type <small>②</small> :	 Alipay
Client ID <small>②</small> :	2021003117621368
Client secret <small>②</small> :	
Cert <small>②</small> :	cert_alipay_app
Root Cert <small>②</small> :	cert_alipay_root
Provider URL <small>②</small> :	

---

Save
Save & Exit

## Step 3. Add the Alipay Pay Payment provider for your product

Finally, add the Alipay Payment provider for your product so that users can purchase the product using Alipay.

Quantity ②:
99

Sold ②:
10

Payment providers ②:

 provider\_payment\_paypal
  provider\_payment\_stripe
  provider\_payment\_wechat
  provider\_payment\_alipay

Return URL ②:


State ②:
Published

Preview ②:
[Test Buy page](#)

---

**Buy Product**

Name	Test Product	Detail	Tag	Casdoor Summit 2022	SKU	test_product
	<b>Casdoor</b>	This is the detail of test product				
Image						
Price	<b>¥ 0.01 (CNY)</b>	Quantity	99	Sold	10	
Pay	<input type="button" value="PayPal"/> <input type="button" value="Stripe"/> <input checked="" type="button" value="WeChat Pay"/> <input checked="" type="button" value="Alipay"/>					

Save
Save & Exit



# WeChat Pay

## Step 1. Preparation

First, you need to have a merchant account at [WeChat Merchant Platform](#).

Before accessing the WeChat Pay, there are some preparations that need to be done.

You can refer to the documentation [preparation before access](#) for more information.

### 1.1 Get API Key v3

Log in to WeChat Merchant Platform, select `Account Settings > API Security > Set APIv3 Secret`, and click `Set APIv3 secret` to get the `API Key v3`.

The screenshot shows the left sidebar with navigation options: Security Center, API Security (highlighted in green), Account Management, Staff Management, Settlement Info, and Agreement. The main content area has two sections:

- API certificate**: Shows a success message: "You have successfully applied for the certificate at 2020-03-13 17:17". A green checkmark icon and "View" and "Change" buttons are present.
- Set APIv3 Secret**: Contains a note: "This key is used to encrypt messages in APIv3's 'download platform certificate' and 'payment callback notification'". It includes a "Set APIv3 Secret" button.

How to get API Key v3 : [APIv3 Key Settings](#)

## 1.2 Get Merchant Certificate

You can log in to WeChat Merchant Platform, and select `Account Settings > API Security > API Certificate` to download the certificate.

The screenshot shows the WeChat Pay API Security settings page. The left sidebar has a green header 'API Security' and includes links for Operating Certificate, API Security, Staff Management, Settlement Info, Agreement, and Merchant Information. The main content area has a title 'API certificate' and a sub-section 'API certificate'. It contains a note about API certificates identifying merchant IDs and preventing ID theft, followed by a box for 'API certificate (CA issued)' which says WeChat provides an API certificate from June 2018, with an 'Apply' button. Below this is another section titled 'API security' with a note about API key signature verification.

After download the certificate, get the [Certificate Serial Number](#) according to [How to view the Certificate Serial Number](#) and [Private Key](#) according to [How to get Private Key of Certificate](#).

Then, create a [Cert](#) at Casdoor and fill the necessary information.

The screenshot shows the 'Edit Cert' page in Casdoor. The left panel contains fields for certificate details: Organization (admin (Shared)), Name (cert\_wechatpay), Display name (Cert Wechatpay), Scope (JWT), Type (Payment, highlighted with a red box), Crypto algorithm (RS256), Bit size (4096), and Expiry in years (20). Below these are buttons for 'Copy certificate' and 'Download certificate', with the certificate URL highlighted in a red box. The right panel shows a 'Private key' section with a 'Copy private key' and 'Download private key' button, and a large text area containing a long base64-encoded private key.

## 1.3 Get Merchant ID and App ID

How to get Merchant ID : [WeChat Pay Merchant ID Query Guide](#)

How to get App ID : [WeChat Pay APPID Query Guide](#)

## Step 2. Create a WeChat Pay Payment provider

Next, create a WeChat Pay Payment provider in Casdoor by filling in the necessary information.

Name	Name in WeChat Pay
Category	choose <a href="#">Payment</a>

Name	Name in WeChat Pay
Type	choose WeChat Pay
Client ID	Merchant ID obtained from Step 1.3
Client secret	API Key v3 obtained from Step 1.1
App ID	App ID obtained from Step 1.3
Cert	Cert configured at Step 1.2

Edit Provider
Save
Save & Exit

---

Name ?:

provider\_payment\_wechat

---

Display name ?:

Wechat Payment Provider

---

Organization ?:

admin (Shared)

---

Category ?:

Payment

---

Type ?:

WeChat Pay

---

Client ID ?:

1619999244

---

Client secret ?:

\*\*\*

---

App ID ?:

wxe933a9cd81c396d1

---

Cert ?:

cert\_wechatpay

---

Provider URL ?:

---

Save
Save & Exit

# Step 3. Add the WeChat Pay Payment provider for your product

Finally, add the WeChat Pay Payment provider for your product so that users can purchase the product using WeChat Pay.

The screenshot shows the Casdoor product management interface. At the top, there are fields for Currency (CNY), Price (0.01), Quantity (99), and Sold (10). Under 'Payment providers', three options are listed: 'provider\_payment\_paypal' (disabled), 'provider\_payment\_stripe' (disabled), and 'provider\_payment\_wechat' (enabled, highlighted with a red box). Below these are fields for 'Return URL' (http://localhost:8080/casdoor/test) and 'State' (Published). A 'Preview' button is at the bottom left. To the right, a modal window titled 'Buy Product' displays a product named 'Test Product' with a detailed description: 'This is the detail of test product'. It shows an image of the Casdoor logo, a price of '¥0.01 (CNY)', and quantities of 99 and 10. At the bottom, there are payment method icons for PayPal, Stripe, and WeChat Pay, with 'WeChat Pay' also highlighted with a red box.

## Support for JSAPI payment

Currently, Casdoor supports [JSAPI payment](#) and [Native payment](#) in WeChat Pay.

To support JSAPI payment, you should configure a [WeChat OAuth Provider](#) which supports [WeChat Media Platform](#). The `Client ID` of WeChat OAuth Provider and the `App ID` of WeChat Pay Payment Provider need to be same.

Edit Provider		Save	Save & Exit
Name ②:	provider_casdoor_wechat		
Display name ②:	Casdoor WeChat		
Organization ②:	admin (Shared)		
Category ②:	OAuth		
Type ②:	 WeChat		
Client ID ②:	wx049c70e6c2027b0b		
Client secret ②:	***		
Client ID 2 ②:	wxе933а9cd81c396d1		
Client secret 2 ②:	***		
Enable QR code ②:	<input checked="" type="checkbox"/>		
Provider URL ②:	<a href="https://open.weixin.qq.com/">https://open.weixin.qq.com/</a>		
<input type="button" value="Save"/>		<input type="button" value="Save &amp; Exit"/>	

Edit Provider		Save	Save & Exit
Name ②:	provider_payment_wechatpay		
Display name ②:	Payment - WeChatPay		
Organization ②:	admin (Shared)		
Category ②:	Payment		
Type ②:	 WeChat Pay		
Client ID ②:	1619999244		
Client secret ②:	***		
App ID ②:	wxе933а9cd81c396d1		
Cert ②:	cert_wechatpay		
Provider URL ②:	<a href="https://pay.weixin.qq.com/index.php/core/cert/api_cert/#/">https://pay.weixin.qq.com/index.php/core/cert/api_cert/#/</a>		
<input type="button" value="Save"/>		<input type="button" value="Save &amp; Exit"/>	

After log in via WeChat(in the mobile scenario: e.g. the WeChat built-in browser inside the WeChat mobile app), users can purchase product using WeChat Pay based on JSAPI payment.

# 验证码

## 概述

添加验证码到您的应用程序

## 默认

Using Casdoor's default captcha in your application

## Cloudflare Turnstile

Add Cloudflare Turnstile to your application

## reCAPTCHA

添加reCAPTCHA 到您的应用程序

 hCaptcha

将 hCaptcha 添加到您的应用程序

 阿里云 Captcha

将阿里云 Captcha 添加到您的应用程序

 Geetest

Add Geetest Captcha to your application

# 概述

Casdoor can be configured to support different captchas to verify if the operation is performed by a human. By adding a captcha provider and applying it in the application, when users login, register, or forget their password and need to send a code, a captcha check dialog will appear to verify if the operation is performed by a human.

Currently, Casdoor supports multiple captcha providers. The following are the providers supported by Casdoor:

默认	Cloudflare Turnstile	reCAPTCHA	hCaptcha	Alibaba Cloud Captcha	Geetest
					
					

We will show you how to apply a captcha and add it to Casdoor.

## 添加验证码提供商

1. Navigate to your Casdoor index page.
2. Click on [Providers](#) in the top bar.
3. Click on [Add](#), then you will see a new provider in the top list.
4. Click on the new provider to modify it.

5. Select **Captcha** in the **Category**.
6. Choose the captcha provider you need in the **Type**.
7. Fill in the most important information. Different captcha providers may require different information to be filled in.

## Applying in the application

1. Click on **Application** in the top bar and choose one application to edit.
2. Click on the provider add button and select the provider you just added.
3. 完成!

# 默认

The default captcha implementation generates and verifies an image. In the default captcha image, a sequence of digits 0-9 is used with a defined length of 5.

## Configuring in Casdoor

To configure the default captcha in Casdoor, follow these steps:

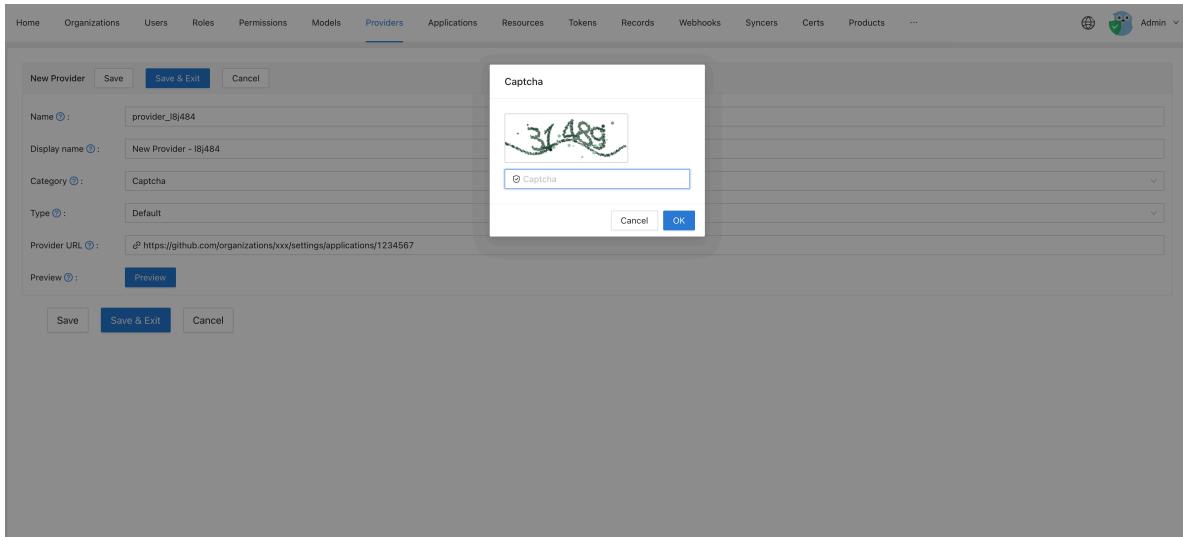
1. 在 Casdoor 中创建一个新的提供商。
2. Select the category as Captcha, and the type as Default.

The screenshot shows the Casdoor web interface with the 'Providers' tab selected. A new provider is being created with the following details:

- Name: provider\_18484
- Display name: New Provider - 18484
- Category: Captcha
- Type: Default
- Provider URL: https://github.com/organizations/xxx/settings/applications/1234567

At the bottom of the form, there are three buttons: 'Save', 'Save & Exit' (highlighted in blue), and 'Cancel'.

3. Click on the Preview button to preview the style of this captcha.



# Applying in your application

To apply the default captcha in your application, do the following:

1. 编辑您想要在 Cassdoor 中配置的应用程序。
2. Select the provider that you just added. There are three types of rules available:
  - **Always**: Always requires human-machine verification during login.
  - **None**: Never requires human-machine verification. The account will be blocked if it attempts to login with the wrong password for the 5th time within 15 minutes. The block will be lifted after 15 minutes.
  - **Dynamic**: After 5 failed login attempts, human-machine verification will be required but the account will not be blocked.

Providers		Add	Name	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Rule	Action		
provider_4olfdm				Captcha		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Always			
provider_casdoor_github				OAuth		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				
provider_casdoor_google				OAuth		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				

We also provide a demo video to demonstrate the differences in rules, which we hope will be helpful to you.

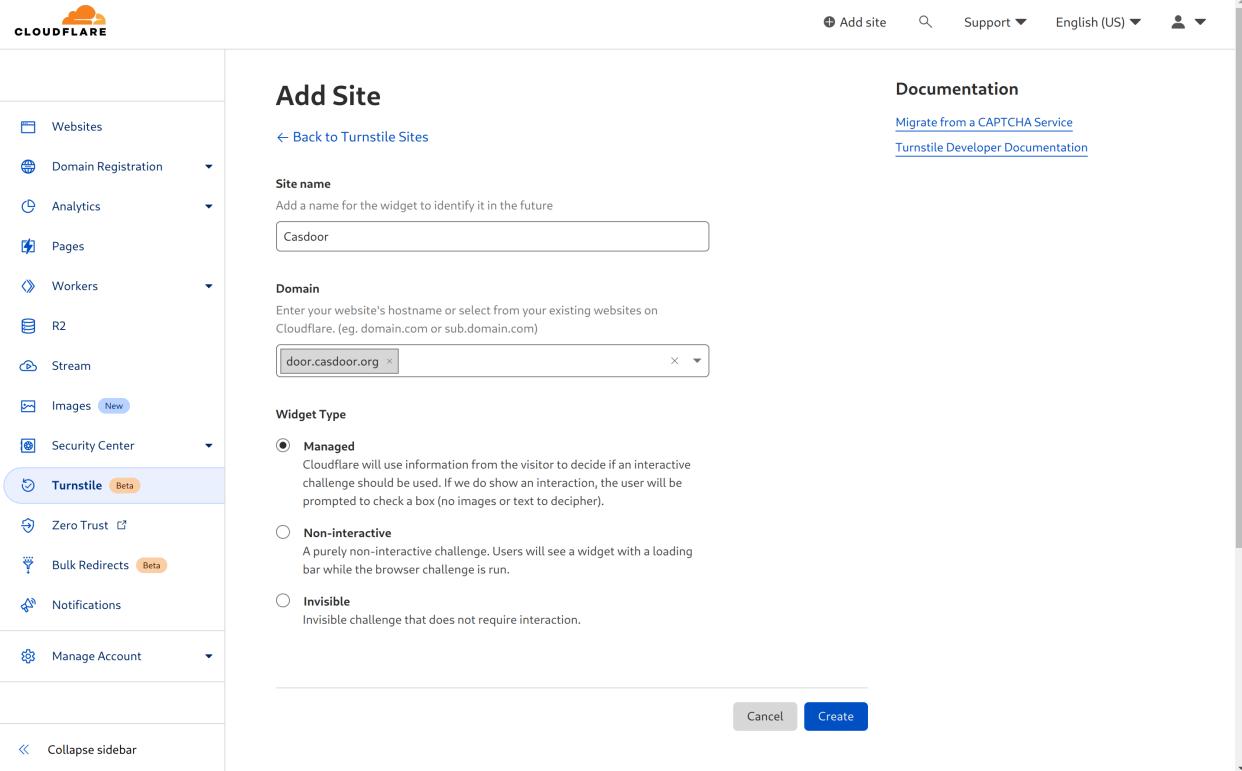
# Cloudflare Turnstile

Cloudflare Turnstile is a CAPTCHA service provided by Cloudflare, which is a user-friendly, privacy-preserving alternative to CAPTCHA. You can find more details in the [Turnstile Docs](#).

## Create an API key pair

To start using Cloudflare Turnstile, you need to [create a Cloudflare account](#), navigate to the [Turnstile](#) tab on the navigation bar, and obtain the Site Key and Secret Key.

First, add a name for the widget to identify it in the future and enter your website's hostname. Then choose the widget type. It is recommended to choose [Managed](#). Finally, click [Create](#).



The screenshot shows the Cloudflare dashboard interface. On the left, a sidebar lists various services: Websites, Domain Registration, Analytics, Pages, Workers, R2, Stream, Images (New), Security Center, Turnstile (Beta), Zero Trust, Bulk Redirects (Beta), Notifications, and Manage Account. The 'Turnstile' option is highlighted with a blue border. Below the sidebar, there's a 'Collapse sidebar' link.

The main content area is titled 'Add Site' and includes a 'Back to Turnstile Sites' link. It has sections for 'Site name' (containing 'Casdoor') and 'Domain' (containing 'door.casdoor.org'). Under 'Widget Type', the 'Managed' option is selected, with a description explaining Cloudflare's decision-making process for interactive challenges. Other options like 'Non-interactive' and 'Invisible' are also listed. At the bottom right are 'Cancel' and 'Create' buttons.

You will then be able to obtain a site key and a secret key.

The screenshot shows the Cloudflare dashboard with the sidebar open. The 'Turnstile' option under the 'Security Center' section is selected, indicated by a blue background and a 'Beta' badge. The main content area is titled 'Add Site' and shows fields for 'Site Key' (containing '0xAAAAA/...') and 'Secret Key' (containing '0xAAAAAAAB...'). Below these are 'Client side integration code' and 'Server side integration code' sections, each containing a snippet of JavaScript. A 'Done' button is at the bottom right. The top navigation bar includes links for 'Add site', 'Support', 'English (US)', and user profile.

# Configure in Casdoor

Create a new provider in Casdoor.

Select the category as **Captcha** and the type as **Cloudflare Turnstile**. Fill in the site key and the secret key that you obtained in the previous step.

 Casdoor Home Organizations Users Roles Permissions Models Adapters Applications Providers Resources ... Admin

Edit Provider

Name ②: Cloudflare Turnstile

Display name ②: Cloudflare Turnstile

Organization ②: admin (share)

Category ②: Captcha

Type ②: Cloudflare Turnstile

Site key ②: 0x4AAAAAAABXhq3vOlgpUTmk

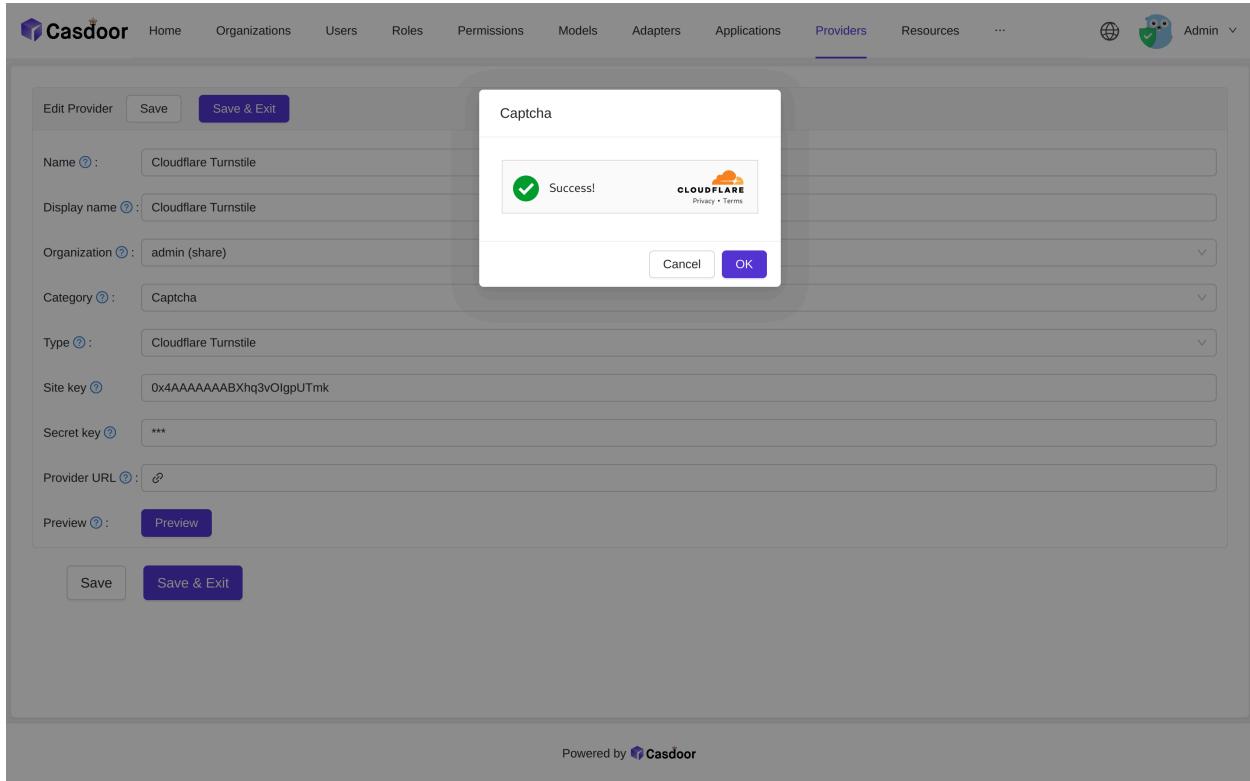
Secret key ②: \*\*\*

Provider URL ②:

Preview ②:

Powered by  Casdoor

You can click the Preview button to see a preview of the style of this CAPTCHA.



# Application Integration

Edit the application you want to configure in Casdoor. Select the provider that you just added and click the **Save** button.

Providers								
Name	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action
Cloudflare Turnstile	Captcha						None	

# reCAPTCHA

reCAPTCHA is provided by Google, and we use reCAPTCHA v2 Checkbox. You can find more details about it at this [link](#).

## 创建一个 API 密钥对

要开始使用 reCAPTCHA，您需要 [注册您的站点的 API 密钥对](#)。配对的密钥由一个站点密钥组成。The site key is used to invoke the reCAPTCHA service on your site or mobile application. 密钥授权您的应用程序后端和 reCAPTCHA 服务器之间的通信来[验证用户的回应](#)。

First, choose the [type of reCAPTCHA](#) and then fill in the authorized domains or [package names](#). After you have accepted the terms of service, click Register to obtain a new API key pair.

Google reCAPTCHA

← Register a new site

Get unlimited assessments using [reCAPTCHA Enterprise](#)

Label ⓘ  
reCaptcha

reCAPTCHA type ⓘ  
 reCAPTCHA v3 Verify requests with a score  
 reCAPTCHA v2 Verify requests with a challenge  
 "I'm not a robot" Checkbox Validate requests with the "I'm not a robot" checkbox  
 Invisible reCAPTCHA badge Validate requests in the background  
 reCAPTCHA Android Validate requests in your android app

Domains ⓘ  
+ casdoor.org

Owners  
resultlee@gmail.com (You)  
Enter email addresses

Accept the reCAPTCHA Terms of Service

By accessing or using the reCAPTCHA APIs, you agree to the Google APIs [Terms of Use](#), Google [Terms of Use](#), and to the Additional Terms below. Please read and understand all applicable terms and policies before accessing the APIs.

You will then receive a site key and a secret key.

The screenshot shows the Google reCAPTCHA registration interface. At the top, it says 'Adding reCAPTCHA to your site'. Below that, it states "'reCaptcha' has been registered.''. It provides instructions to 'Use this site key in the HTML code your site serves to users.' with a 'See client side integration' link. A 'COPY SITE KEY' button is next to a text input field containing a long string of characters. Below it, another set of instructions says 'Use this secret key for communication between your site and reCAPTCHA.' with a 'See server side integration' link. A 'COPY SECRET KEY' button is next to a text input field containing a long string of characters. At the bottom, there are 'GO TO SETTINGS' and 'GO TO ANALYTICS' buttons.

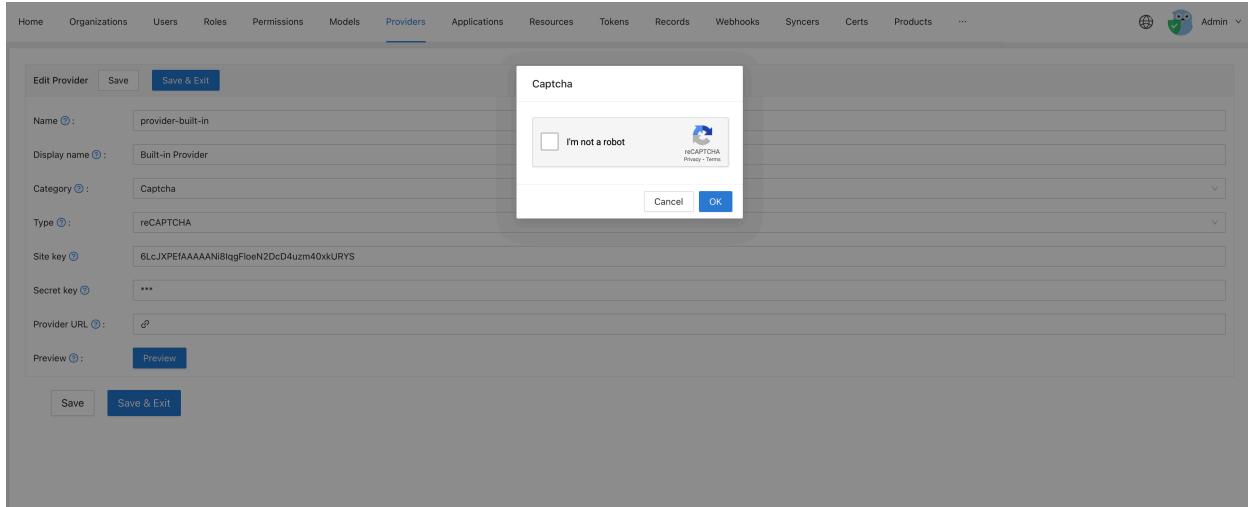
## 在 Casdoor 配置

在 Casdoor 中创建一个新的提供商。

Select the category as Captcha and the type as reCAPTCHA. You need to provide the site key and secret key created in the previous step.

The screenshot shows the Casdoor provider configuration interface. The top navigation bar includes Home, Organizations, Users, Roles, Permissions, Models, Providers (which is selected), Applications, Resources, Tokens, Records, Webhooks, Syncers, Certs, Products, and more. On the right, there is a user icon labeled 'Admin'. The main form is titled 'New Provider' and contains fields for Name (reCaptcha), Display name (reCaptcha), Category (Captcha), Type (reCAPTCHA), Site key (a redacted URL), Secret key (a redacted URL), Provider URL (https://github.com/organizations/xx/settings/applications/1234567), and Preview (a button). There are 'Save', 'Save & Exit', and 'Cancel' buttons at the bottom.

You can click the Preview button to see the style of this captcha.



# Apply in the application

编辑您想要在 Cassdoor 中配置的应用程序。 Select the provider you just added and click the Save button.

Providers	Add	Name	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
		reCaptcha	Captcha						

# hCaptcha

hCaptcha is a captcha service provider, similar to reCAPTCHA. You can find more details about hCaptcha [here](#).

## 创建一个 API 密钥对

To start using hCaptcha, you need to sign up for an API key pair for your site. You can obtain your site key on your [profile page](#).

Once you have signed up, you will receive a site key and a secret key.

## 在 Casdoor 配置

To configure hCaptcha in Casdoor, create a new provider.

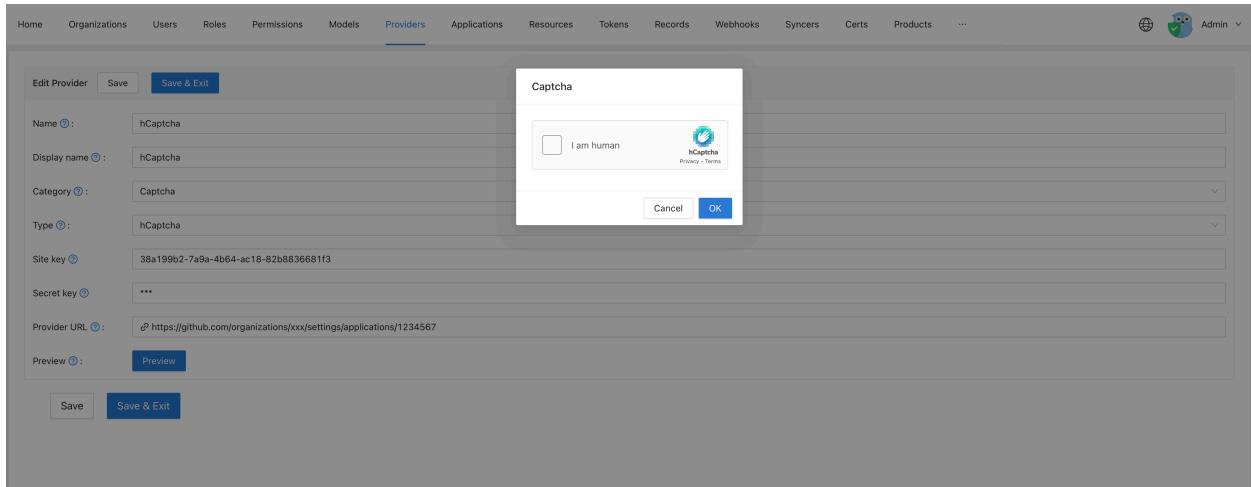
Select the category as Captcha and the type as hCaptcha. Fill in the site key and secret key obtained in the previous step.

The screenshot shows the Casdoor web interface with the 'Providers' tab selected. A new provider is being created with the following details:

- Name: hCaptcha
- Display name: hCaptcha
- Category: Captcha
- Type: hCaptcha
- Site key: 38a199b2-7a9a-4b64-ac18-82b8836681f3
- Secret key: 38a199b2-7a9a-4b64-ac18-82b8836681f3
- Provider URL: https://github.com/organizations/xx/settings/applications/1234567

At the bottom, there are 'Save', 'Save & Exit', and 'Cancel' buttons.

You can click the Preview button to see how the captcha style will look.



## Apply in your application

Go to the application you want to configure in Casdoor. Select the provider you just added and click the Save button.

Providers	Add	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
hCaptcha	✓	Captcha						

# 阿里云 Captcha

阿里云 Captcha 是由 阿里云 提供的验证码服务。 它提供两种验证方式："滑动验证" 和 "智能验证"。 您可以从此 [链接](#) 中查询更多详细信息。

## ① 信息

Currently, only [Alibaba Cloud Captcha 1.0](#) is supported. [Alibaba Cloud Captcha 2.0](#) is currently in the public testing phase, so there are no plans for adaptation in the near term.

## Add Captcha Configuration in Alibaba Cloud

To add the Captcha configuration, log in to the [Alibaba Cloud management console](#), search for and go to the Captcha Service. Then, click on Confirm Open to enable the Captcha Service.



Once you have entered the Captcha management console, click on Add configuration.

The screenshot shows the Alibaba Cloud Workbench interface. On the left, there's a sidebar with '验证码' (Verification Code) selected. The main area has a title '验证码' and a sub-section '配置管理'. A note at the top says '公告: 2021年3月18日起, 人核验证产品统一更名为验证码。'. Below is a table with two rows of configuration data:

配置名称	appkey	scene	验证方式	业务类型	使用场景	最后更新	操作
测试验证码	F [REDACTED] B	nc_other	滑动验证	PC	其它	2022-06-20 23:47:37	<a href="#">自定义样式</a> <a href="#">系统代码集成</a>
测试智能验证码	FF [REDACTED] B	lc_other	智能验证	PC	其它	2022-06-21 00:44:08	<a href="#">自定义样式</a> <a href="#">系统代码集成</a>

At the bottom right, there are navigation buttons: '共有2条' (2 items), a page number '1', and arrows for '上一页' (prev) and '下一页' (next). To the right of the table are three small circular icons: a magnifying glass, a printer, and a clipboard.

Fill in all the required information and submit the form.

The screenshot shows the 'Add Configuration' form. It consists of three steps:

- 1 配置服务内容**:
  - 配置名称:
  - 高峰期QPS:
  - 业务类型:  PC网页  H5 (移动端WAP + APP)
  - 验证方式:  滑动验证  智能验证  无痕验证
  - 使用场景:  登录  注册  活动  论坛  短信  其它
- 2 系统代码集成&测试**
- 3 完成**

Below the form, there's a preview section labeled '产品形态预览:' with a button 'Demo页'.

Now, you can view the **Scene** and **App key** in your console.

配置名称	appkey	scene	验证方式	业务类型	使用场景	最后更新	操作
测试验证码	XXXXXXXXXXXXXXXXXXXX8B	nc_other	滑动验证	PC	其它	2022-06-20 23:47:37	自定义样式   系统代码集成
测试智能验证码	XXXXXXXXXXXXXXXXXXXX8B	lc_other	智能验证	PC	其它	2022-06-21 00:44:08	自定义样式   系统代码集成

Also, the `Access key` and `Secret access key` can be found in your profile.

## 在 Casdoor 配置

在 Casdoor 中创建一个新的提供商。

Select the category as `Captcha`, and the type as `hCaptcha`. Then, choose the sub-type: "Sliding Validation" or "Intelligent Validation". Make sure to fill in the `Access key`, `Secret access key`, `Scene`, and `App key` that you created in the previous step.

The screenshot shows the 'New Provider' configuration page. The provider type is set to 'Aliyun Captcha'. The 'Sub type' is 'Sliding Validation'. The 'Access key' and 'Secret access key' fields contain placeholder text. The 'Scene' field is set to 'nc\_other'. The 'Provider URL' field contains a link to a GitHub organization settings page. A 'Preview' button is visible at the bottom left of the form.

New Provider Save Save & Exit Cancel

Name : Aliyun\_Captcha

Display name : Aliyun\_Captcha

Category : Captcha

Type : Aliyun Captcha

Sub type : Sliding Validation

Access key : LTAI4G7CngW2Pp5p4E4yS7gF

Secret access key : iPXXXXXXXXXXXXXN

Scene : nc\_other

App key : FXXXXXXXXXXXXX8

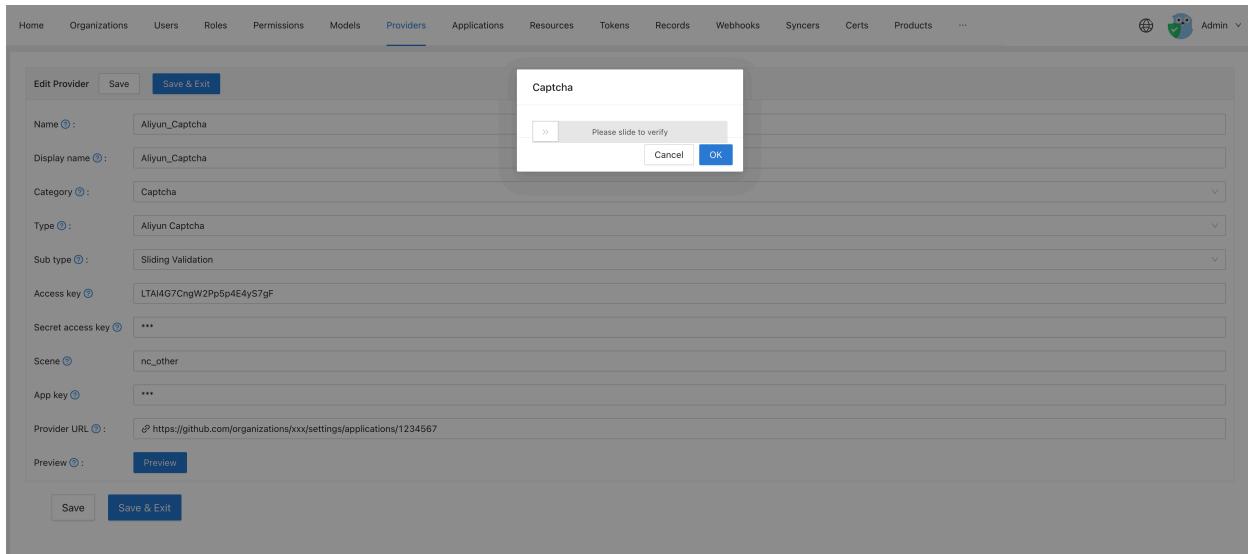
Provider URL : https://github.com/organizations/xxx/settings/applications/1234567

Preview : Preview

Save Save & Exit Cancel

You can click on the **Preview** button to see the style of this captcha.

The following image shows the preview of "Sliding Validation":



And this image shows the preview of "Intelligent Validation":

The screenshot shows the Casdoor provider configuration interface. A modal window titled "Captcha" is open, prompting the user to "Click the button to start". The main configuration form contains the following fields:

- Name: Aliyun\_Captcha
- Display name: Aliyun\_Captcha
- Category: Captcha
- Type: Aliyun Captcha
- Sub type: Intelligent Validation
- Access key: LTAI4G7CngW2Pp5p4E4yS7gF
- Secret access key: ...
- Scene: ic\_other
- App key: ...
- Provider URL: https://github.com/organizations/xxx/settings/applications/1234567
- Preview: Preview

At the bottom of the configuration form are "Save" and "Save & Exit" buttons.

# Application Integration

Edit the application in which you want to configure Casdoor. Select the newly added provider and click on the Save button.

The screenshot shows the Casdoor provider list table. It displays the following information for the provider "Aliyun\_Captcha":

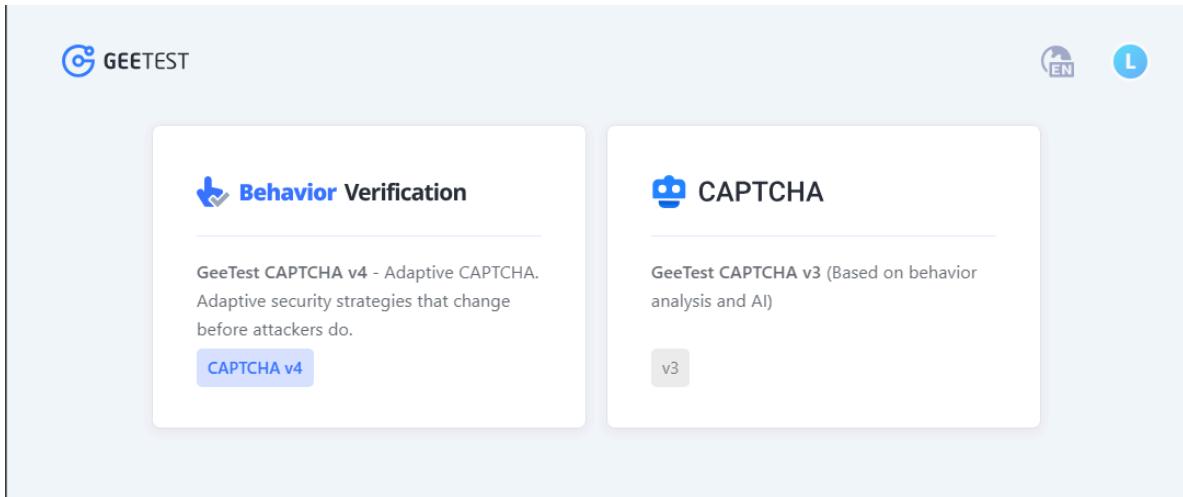
Name	Category	Type	canSignUp	canSignIn	canUnlink	prompted	Action
Aliyun_Captcha	Captcha	Intelligent Validation					

# Geetest

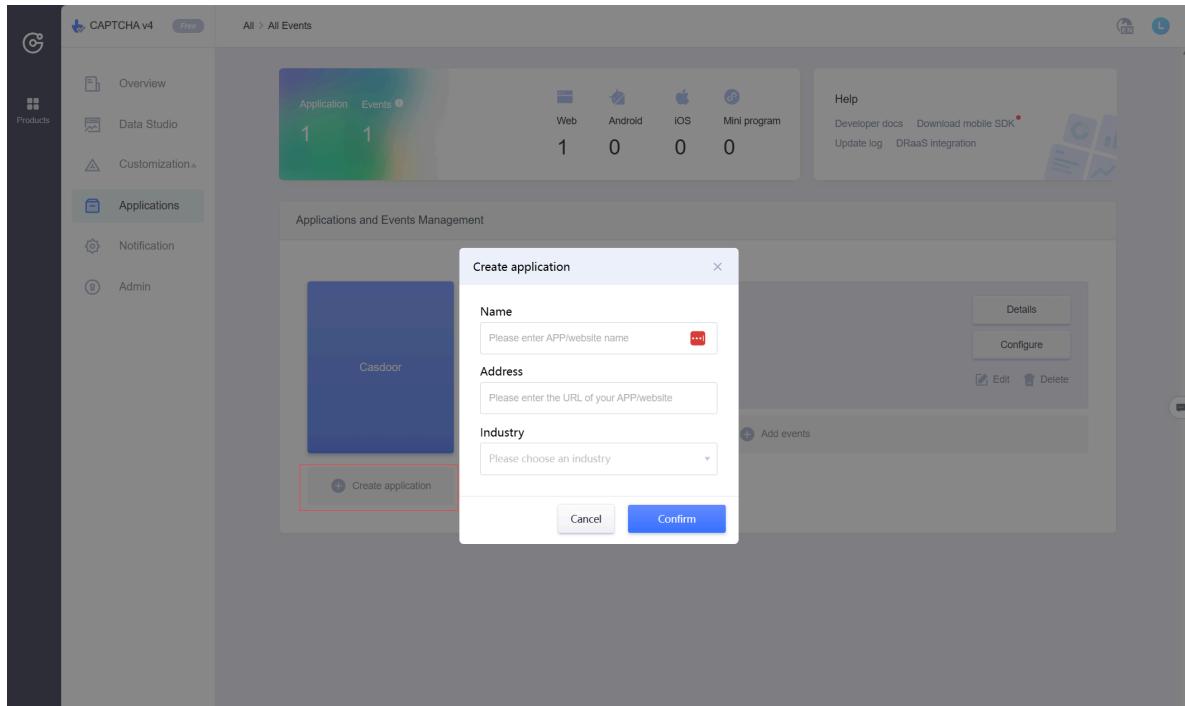
## Configure Geetest

To configure Geetest and obtain the public and secret key, follow these steps:

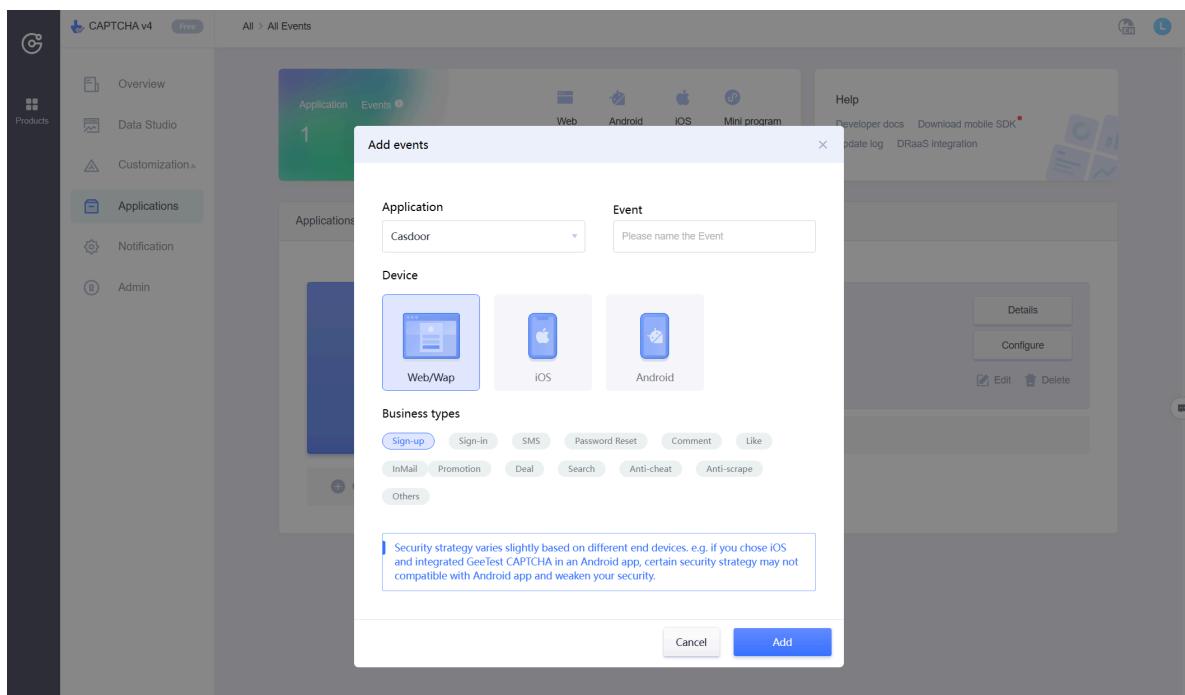
1. Go to the Geetest CAPTCHA V4 section on the [Geetest product page](#).



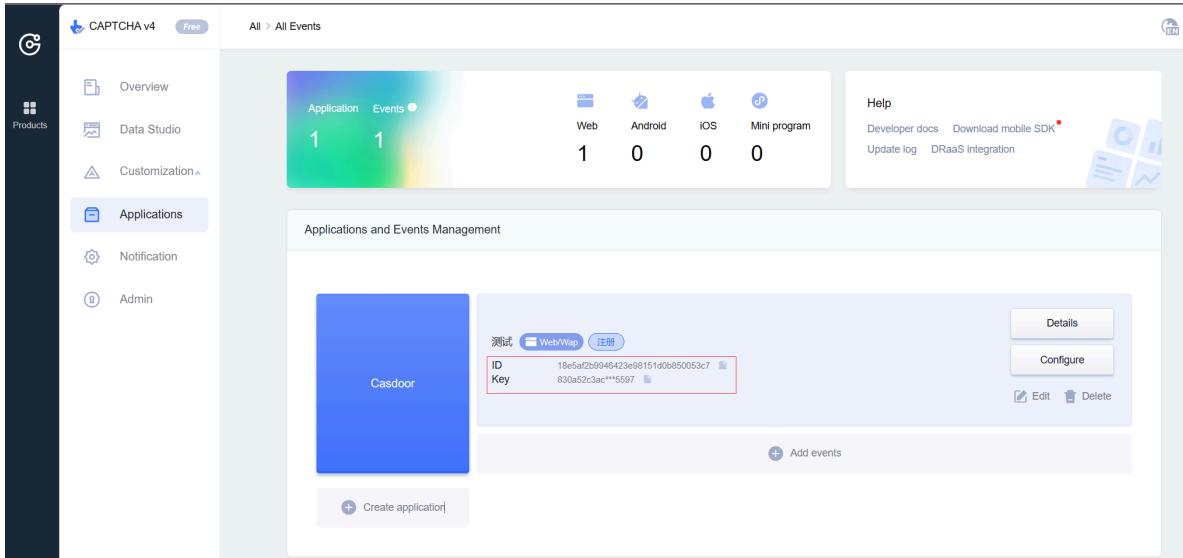
2. Create an application by entering the name and address for your application.



### 3. Add events and choose "web" for the device.



### 4. Retrieve the `ID` and `Key`.



# Configure Casdoor

Follow these steps to configure Casdoor:

1. Create a new provider in Casdoor.

Set the category as Captcha and the type as Geetest. Fill in the `Site key` and `Secret key` with the ID and Key obtained from Geetest.

2. Click the Preview button to preview the style of this captcha.

The screenshot shows the Casdoor web interface for managing providers. The top navigation bar includes links for Home, Organizations, Users, Roles, Permissions, Models, Providers (which is the active tab), Applications, Resources, Tokens, Records, and more. A user icon labeled 'Admin' is visible on the right.

The main content area is titled 'Edit Provider' and contains the following fields:

- Name: provider\_geetest
- Display name: provider\_geetest
- Category: Captcha
- Type: GEETEST
- Site key: 489b713a684726d851073a7e8cf8442a
- Secret key: \*\*\*
- Provider URL: <https://github.com/organizations/xxx/settings/applications/1234567>

Buttons at the bottom include 'Edit Provider', 'Save', and 'Save & Exit'. A note at the bottom right says 'Made with ❤ by Casdoor'.

# Apply in your application

To apply the Geetest configuration in your application:

Edit the application you want to configure in Casdoor. Select the provider you just added and click the Save button.

The screenshot shows a table of providers. The first column is 'Providers' with a link to 'Add'. The second column is 'Name' containing 'geetest'. The third column is 'Category' with a dropdown set to 'Captcha'. The fourth column is 'Type' with a blue 'G' icon. The remaining columns ('Can signup', 'Can signin', 'Can unlink', 'Prompted', 'Rule', and 'Action') are empty or show icons for edit and delete.

Providers	Add	Name	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action
		geetest	Captcha	G						edit delete



&gt; 提供商

&gt; Web3

# Web3



## MetaMask

Adding the MetaMask Web3 provider to your application



## Web3-Onboard

Add the Web3-Onboard Web3 provider to your application

# MetaMask

ⓘ 备注

This is an example of how to configure MetaMask as a Web3 provider.

MetaMask is a browser extension and app that functions as both a cryptocurrency wallet and a gateway to blockchain apps. Casdoor allows you to use MetaMask as an identity provider and enables Web3 login with MetaMask.

## Step 1: Create a MetaMask Web3 provider

To start, you need to create a MetaMask Web3 provider in Casdoor.

Name	Description
Category	Choose <a href="#">Web3</a>
Type	Choose <a href="#">MetaMask</a>

Edit Provider
Save
Save & Exit

---

Name <span>?</span> :	metamask_provider
Display name <span>?</span> :	MetaMask Provider
Organization <span>?</span> :	admin (Shared)
Category <span>?</span> :	Web3
Type <span>?</span> :	MetaMask
Provider URL <span>?</span> :	<a href="#">🔗</a>

---

Save
Save & Exit

## Step 2: Add the provider to your application

Next, add the MetaMask Web3 provider to your application.

The screenshot shows a provider configuration interface with the following details:

- Providers:** A list of providers including "provider\_storage\_minio\_s3", "provider\_oauth\_lark", "provider\_email\_qq", and "metamask\_provider". The "metamask\_provider" is highlighted with a red box.
- Preview:** Buttons for "Copy SAML metadata URL" and "Copy sign-in page URL".
- Table:** A table showing provider settings:
 

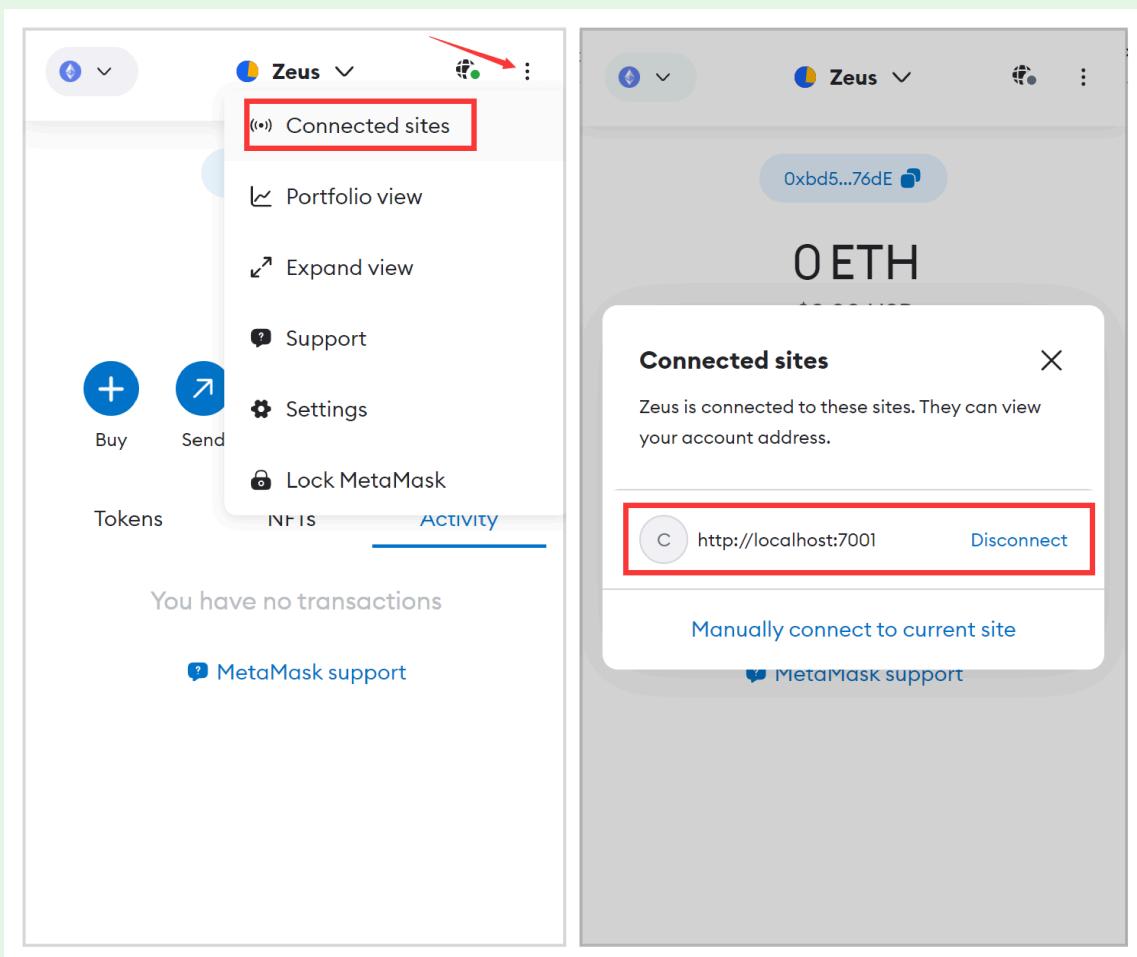
Name	Category	Type	Can sign-up	Can sign-in	Can un-link	Prompted	Rule	Action
metamask_provider	Web3	MetaMask	On	On	On	Off		Up/Down

## Step 3: Login with MetaMask

You can now log in with MetaMask. Here is a demo video.

 提示

1. When logging in with MetaMask, please authorize only one Ethereum address. Casdoor will only bind one Ethereum address per user.
2. If you want to switch to another Ethereum address for login, please disconnect the connection between the current Ethereum address and Casdoor first.

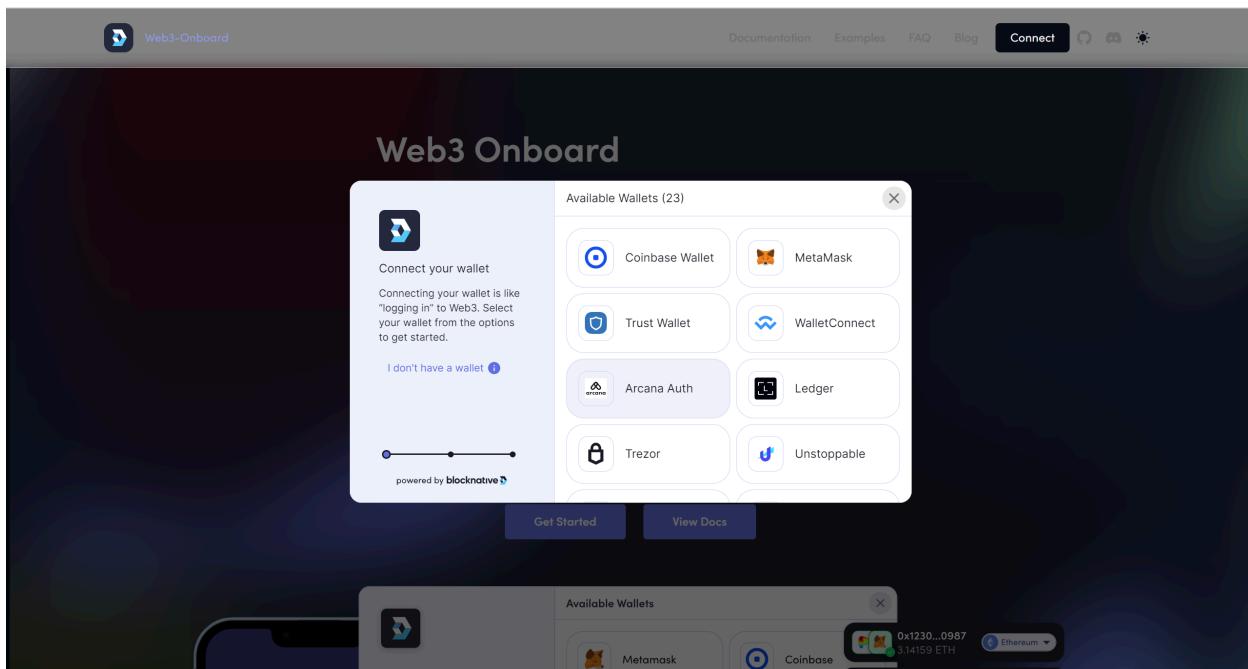


# Web3-Onboard

## ⓘ 备注

This is an example of how to configure Web3-Onboard as a Web3 provider.

[Web3-Onboard](#) can help users use different wallets for Web3 login. Casdoor allows using Web3-Onboard as an identity provider and enables Web3 login with Web3-Onboard.



## Step 1: Create a Web3-Onboard Web3 provider

First, you need to create a Web3-Onboard Web3 provider in Casdoor.

Name	Description
Category	Choose <code>Web3</code>
Type	Choose <code>Web3-Onboard</code>
Wallets	Choose the wallets that are allowed to log in

Edit Provider
Save
Save & Exit

---

Name ? :

Display name ? :

Organization ? :

Category ? :

Type ? :  `Web3-Onboard`

Wallets ? :  Injected  Coinbase  Trust  Gnosis  Sequence  Taho  Frontier  Infinity Wallet

Provider URL ? :

---

Save
Save & Exit

Currently, Casdoor only supports the wallets shown in the image above. The `Injected` wallets represent browser-injected wallets such as `MetaMask` or `Coinbase`.

## Step 2: Add the provider to your application

Second, add the Web3-Onboard Web3 provider to your application.

Providers [?](#) [Add](#)

Name	Category	Type	Can signup	Can signin	Can unlink	Prompted	Rule	Action
provider_storage_minio_s3	Storage		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">▲</a> <a href="#">▼</a> <a href="#">trash</a>
provider_oauth_lark	OAuth		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">▲</a> <a href="#">▼</a> <a href="#">trash</a>
provider_email_qq	Email		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">▲</a> <a href="#">▼</a> <a href="#">trash</a>
provider_web3_metamask	Web3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">▲</a> <a href="#">▼</a> <a href="#">trash</a>
provider_google_oauth	OAuth		<input checked="" type="checkbox"/> One Tap	<a href="#">▲</a> <a href="#">▼</a> <a href="#">trash</a>				
provider_web3_onboard	Web3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<a href="#">▲</a> <a href="#">▼</a> <a href="#">trash</a>

## Step 3: Login with Web3-Onboard

Now you can log in through Web3-Onboard. Here is a demo video.



&gt;

资源

# 资源

## 概述

上传资源到Casdoor

# 概述

You can upload resources in Casdoor. Before uploading resources, you need to configure a storage provider. Please refer to the [Storage Provider](#) section for more information.

Once you have configured at least one storage provider and added it to your application, you can proceed.

Providers [②](#):

Name	Category	Type	操作
Provider_azure	Storage		<a href="#">编辑</a> <a href="#">删除</a>
Github_1	OAuth		<a href="#">编辑</a> <a href="#">删除</a>
provider_Alipay	Payment		<a href="#">编辑</a> <a href="#">删除</a>

Great! Now let's take a look at an example of how to [upload](#) and [delete](#) resources.

## Uploading Resources

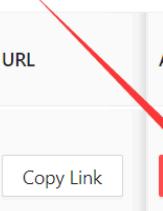
Users can upload various types of resources, such as files and images, to the [cloud storage](#) that you have configured.



Resources		<a href="#">Upload a file...</a>			
Provider	File Name	Content	Created time	Tag	
provider_storage_aliyun_oss	source/casbin/leo220yuyaodog/2022_ICM_Problem_D.pdf		2022-05-18 17:25:21	custom	
provider_storage_aliyun_oss	source/built-in/admin/美的2021&22Q1交流.pdf		2022-05-18 12:28:01	custom	
provider_storage_aliyun_oss	source/casbin/admin/solo.svg		2022-05-17 16:25:39	custom	

## Deleting Resources

If you no longer need a particular resource, you can choose to delete it by clicking the "Delete" button.



Created time	Tag	Type	Format	File size	Preview	URL	Action
2022-05-19 23:16:55	custom	image	.jpg	70.3 KB		<a href="#">Copy Link</a>	<a href="#">Delete</a>



&gt;

产品

# 产品



产品

添加您想要销售的产品



支付

查看支付中产品的交易信息

# 产品

您可以添加您想要销售的产品 (或服务)。 The following will guide you through the process of adding a product.

## Configuring Product Attributes

First, you need to understand the basic properties of the product:

- Tag
- Detail
- Currency
- Price
- Quantity
- Sold

Tag <a href="#">?</a> :	Casdoor Summit 2022
Detail <a href="#">?</a> :	This is a description
Currency <a href="#">?</a> :	USD
Price <a href="#">?</a> :	19
Quantity <a href="#">?</a> :	100
Sold <a href="#">?</a> :	10

## 支付服务提供商

In addition to setting these properties, you also need to add payment providers to the product. Multiple payment providers can be added to a product.

To learn how to configure a payment provider, refer to [Payment Provider](#)

Payment providers <a href="#">?</a> :	provider_Alipay <a href="#">×</a>
Return URL <a href="#">?</a> :	<a href="http://localhost:8000/products/callback">http://localhost:8000/products/callback</a>

Finally, fill in the **Return URL**. This is the URL to which the payment provider page will redirect after the payment is completed.

# 预览产品

You're done! Review the details and save:

Preview ⓘ:

[Test buy page.](#)

Buy Product					
Name	Product				
Detail	This is a subscription.	Tag	Casdoor Summit 2022	SKU	product
Image	 Casdoor				
Price	\$300 (USD)	Quantity	99	Sold	10
Pay	Alipay				

# 支付

After the payment is successfully processed, you will be able to view the transaction information of the products in the Payment section. This information will include details such as the organization, user, purchase time, and product name.

# 发票

To issue an invoice, navigate to the edit screen:

Type	Product	Price	Curren	Action
	A notebook computer	300	USD	<button>Result</button> <button>Edit</button> <button>Delete</button>

On the edit screen, you will need to fill in the relevant invoice information. There are two invoice types available: [individual](#) and [organization](#).

To complete the process, simply click on the "issue invoice" button.

Please let us know if you have any further questions or concerns.



&gt;

定价

# 定价



## Overview

Casdoor Pricing Overview



## Plan

Casdoor Plan Overview



## Pricing Overview

An Overview of Casdoor Pricing



## Subscription

Casdoor Subscription Overview

# Overview

Casdoor can be used as a subscription management system through its [Plan](#), [Pricing](#), and [Subscription](#) features.

You can choose which plans to include in your price list, as shown in the pictures below:

## Casdoor Pricing

Casdoor hosting services provided by Casbin Inc.

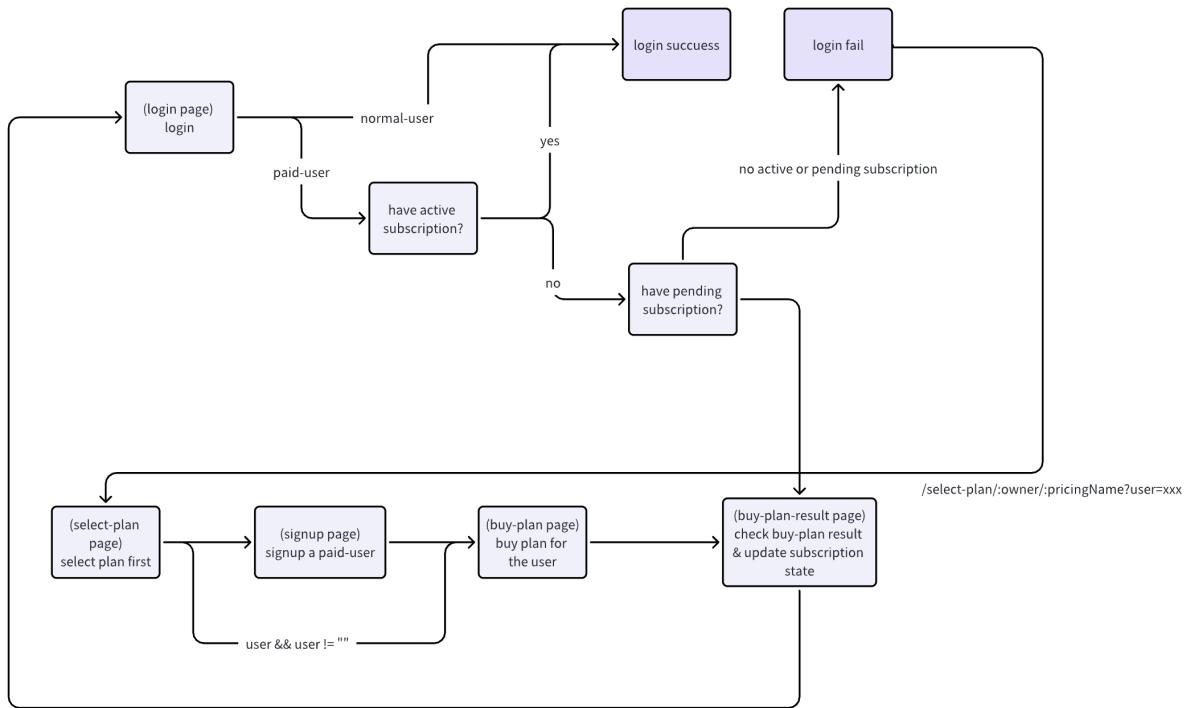
Basic Plan	Premium Plan	Enterprise Plan
<b>\$ 10.01</b> per month	<b>\$ 20.02</b> per month	<b>\$ 30.03</b> per month
For small teams, with limited technical support	For fast growing start-ups, with full technical support	For large & medium-sized enterprise, with full technical support
<a href="#">Getting started</a>	<a href="#">Getting started</a>	<a href="#">Getting started</a>

*Free 7-days trial available!*

Each [Pricing](#) belongs to a specific [Application](#). Users can select a plan and sign up as a [paid-user](#) through the corresponding [pricing page URL](#) of the [Pricing](#).

## General flow

The general flow looks like this:



1. Users enter the select-plan page of the Pricing by accessing the pricing page URL shared by the admin.

**Pricing Configuration:**

Setting	Value
Organization	built-in
Name	pricing_casdoor
Display name	Casdoor Pricing
Description	Casdoor hosting services provided by Casbin Inc.
Application	app-built-in
Plans	plan_basic X plan_premium X plan_enterprise X
Trial duration	7
Is enabled	Enabled (checkbox)
Preview	<a href="#">Copy pricing page URL</a> (button)

**Casdoor Pricing Preview:**

**Casdoor Pricing**  
Casdoor hosting services provided by Casbin Inc.

Plan	Price	Description
Basic Plan	\$ 10.01 per month	For small teams, with limited technical support
Premium Plan	\$ 20.02 per month	For fast growing start-ups, with full technical support
Enterprise Plan	\$ 30.03 per month	For large & medium-sized enterprise, with full technical support

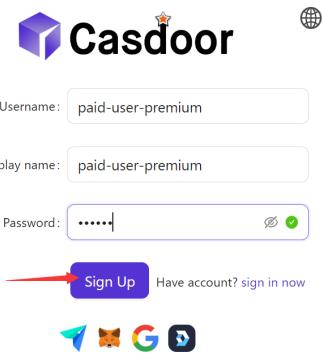
2. Users select a Plan to subscribe and complete the signup process, becoming a paid-user.

## Casdoor Pricing

Casdoor hosting services provided by Casbin Inc.

Basic Plan	Premium Plan	Enterprise Plan
<b>\$ 10.01</b> per month For small teams, with limited technical support	<b>\$ 20.02</b> per month For fast growing start-ups, with full technical support	<b>\$ 30.03</b> per month For large & medium-sized enterprise, with full technical support
<a href="#" style="background-color: #007bff; color: white; padding: 5px 10px; text-decoration: none; border-radius: 5px;">Getting started</a>	<a href="#" style="background-color: #007bff; color: white; padding: 5px 10px; text-decoration: none; border-radius: 5px;">Getting started</a>	<a href="#" style="background-color: #007bff; color: white; padding: 5px 10px; text-decoration: none; border-radius: 5px;">Getting started</a>

Free 7-days trial available!



3. After signing up, users will be redirected to the buy-plan page for the selected Plan to proceed with the payment.

### Buy Product

Name	Auto Created Product for Plan built-in/plan_premium(Premium Plan)	Tag	auto_created_product_for_plan	SKU	product_6g2mcm
Detail	This Product was auto created for Plan built-in/plan_premium(Premium Plan)				
Image					
Price	\$20.02 (USD)	Quantity	999	Sold	0
Pay	 Stripe  PayPal				

- Once the payment is successfully completed, the user's **Subscription** for the **Plan** is activated. Now, users can log in to Casdoor as a **paid-user**.



You have successfully completed the payment: Auto Created Product for Plan built-in/plan\_premium(Premium Plan)

Please click the below button to return to the original website

[Return to Website](#)

Here is a demo video:

# Plan

The Plan describes a list of features for an application, each with its own name and price.

The features of a Plan depend on the Casdoor Role, which comes with a set of Permissions.

This allows for the independent description of a Plan's features, regardless of naming and pricing.

For example, a Plan may have different prices depending on the country or date.

The following picture illustrates the relationship between a Plan and a Role.

## Plan

- Display Name
  - Price per month
- ...

## Role

permission 1  
permission 2  
...  
permission N

# Plan Properties

Every `Plan` has the following properties:

- `Organization`
- `Name`
- `CreatedTime`
- `DisplayName`
- `Role`
- `PricePerMonth`
- `Currency`
- `PaymentProviders`: Users can purchase the `Plan` through the Payment providers. For information on how to configure a Payment provider, see [Payment provider](#).
- `IsEnabled`

The screenshot shows a user interface for editing a plan. At the top, there are three buttons: 'Edit Plan', 'Save', and 'Save & Exit'. The 'Save & Exit' button is highlighted with a purple background.

The form fields and their values are:

- Organization**: built-in
- Name**: plan\_enterprise
- Display name**: Enterprise Plan
- Role**: (empty)
- Description**: For large & medium-sized enterprise, with full technical support
- Price per month**: 30.03
- Price per year**: 100
- Currency**: USD
- Payment providers**: provider\_payment\_stripe × provider\_payment\_paypal ×
- Is enabled**:

At the bottom, there are two buttons: 'Save' and 'Save & Exit'.

When a **Plan** is created through Casdoor, a related **Product** is automatically created.

The information configured for the **Plan** will be automatically synchronized to the **Product**.

When users buy a **Plan**, they are essentially purchasing the related **Product** of the selected **Plan**.

Product Configuration:

Name (必填):	product_4f9fak
Display name (必填):	Auto Created Product for Plan built-in/plan_enterprise(Enterprise Plan)
Organization (必填):	built-in
Image (必填):	URL: https://cdn.casbin.org/img/casdoor-logo_1185x236.png
Preview:	
 Casdoor	
Tag (必填):	auto_created_product_for_plan
Detail (必填):	This Product was auto created for Plan built-in/plan_enterprise(Enterprise Plan)
Description (必填):	
Currency (必填):	USD
Price (必填):	30.03
Quantity (必填):	999
Sold (必填):	0
Payment providers (必填):	provider_payment_stripe   provider_payment_paypal
Return URL (必填):	http://
State (必填):	Published
Preview (必填):	<a href="#">Test buy page.</a>

Buy Product Page Preview:

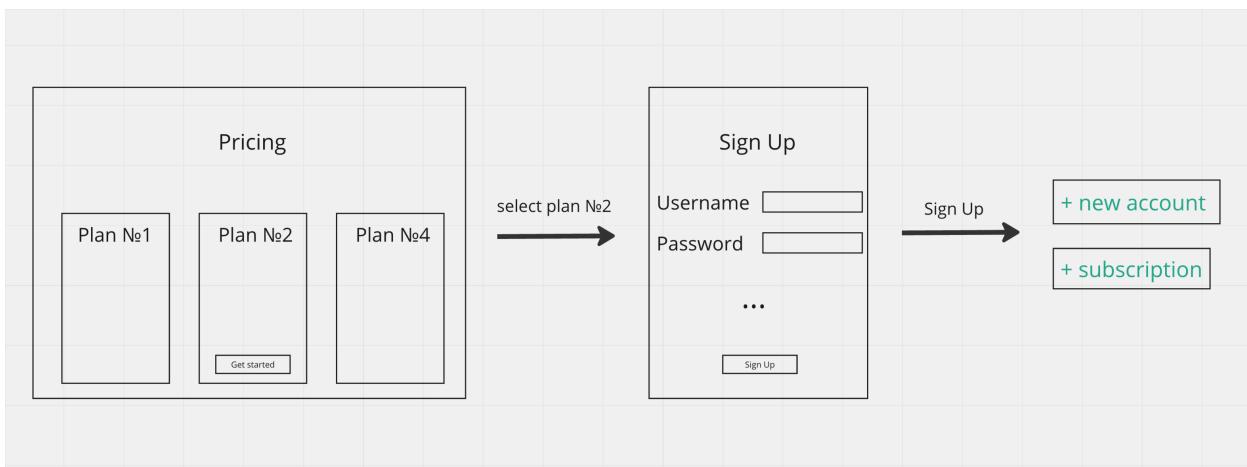
**Buy Product**

Name	Auto Created Product for Plan built-in/plan_enterprise(Enterprise Plan)
Detail	This Product was auto created for Plan built-in/plan_enterprise(Enterprise Plan)
Image	 Casdoor
Price	<b>\$30.03 (USD)</b>
Pay	 Stripe    PayPal
Quantity	999
Sold	0

# Pricing Overview

The **Pricing** feature contains one or more **Plan** options, allowing users to sign up for **Applications** at different price-points.

The general flow of pricing options is depicted in the image below:



## Pricing Properties

Every **Pricing** subscription has the following properties:

- **Organization**
- **Name**
- **CreatedTime**
- **DisplayName**
- **Description**
- **Plans**: An array of Plans.

- `IsEnabled`
- `Application`

To see an example of the pricing interface, refer to the image below:

The screenshot shows the 'Edit Pricing' page. It includes fields for Organization (built-in), Name (pricing\_casdoor), Display name (Casdoor Pricing), Description (Casdoor hosting services provided by Casbin Inc.), Application (app-built-in), Plans (plan\_basic, plan\_premium, plan\_enterprise), Trial duration (7 days), Is enabled (checked), and a Preview button.

## Casdoor Pricing

Casdoor hosting services provided by Casbin Inc.

Basic Plan	Premium Plan	Enterprise Plan
<b>\$ 10.01</b> per month	<b>\$ 20.02</b> per month	<b>\$ 30.03</b> per month
For small teams, with limited technical support	For fast growing start-ups, with full technical support	For large & medium-sized enterprise, with full technical support
<a href="#">Getting started</a>	<a href="#">Getting started</a>	<a href="#">Getting started</a>

# Subscription

The **Subscription** feature helps in managing a user's selected **Plan**, making it easy to control the access to **Application** features.

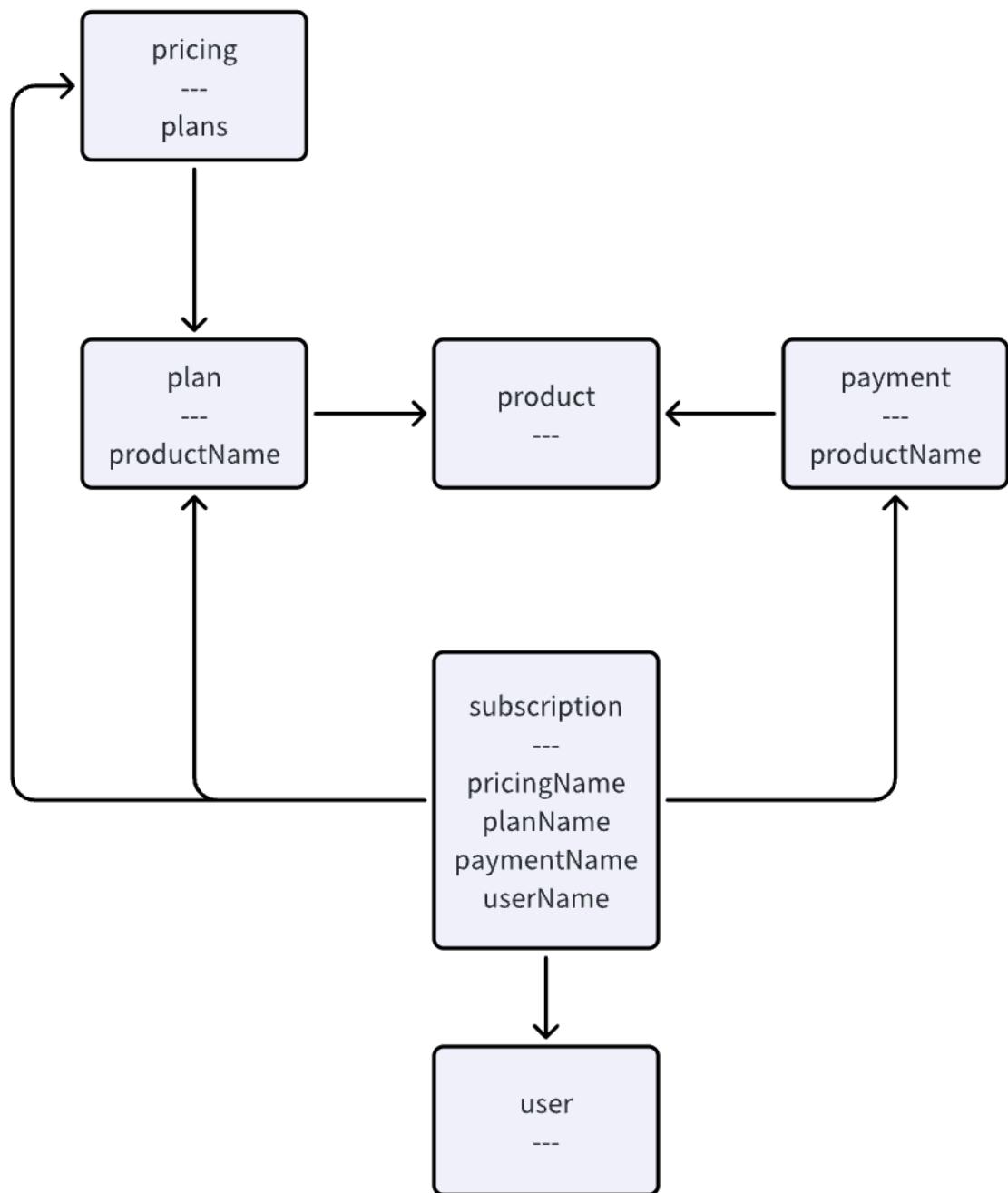
## 提示

Since each **Plan** is based on a **Role**, you can assign the Plan's Role to a user and use the enforce API for permission checking.

A **Subscription** can be created in three ways:

- Manually by an admin
- Via the Pricing flow (after signing up as a **paid-user** and purchasing the selected **Plan**)
- Via API

The relationship between **Pricing**, **Plan**, **Subscription**, **Product**, and **Payment** is as follows:



# Subscription properties

Every Subscription has these properties:

- Owner
- Name
- CreatedTime
- DisplayName
- Description
- Duration: The duration of the Subscription.
- StartTime: The starting time for the Subscription to take effect.
- EndTime: The end time for the Subscription to take effect.
- Pricing: The related Pricing.
- Plan: The related Plan.
- Payment: The related Payment.
- User: The user who holds this Subscription.
- State: Currently, the Subscription has the following states: Pending, Error, Suspended, Active, Upcoming, Expired.

Edit Subscription

Organization <small>②</small> :	built-in
Name <small>②</small> :	sub.e719e2
Display name <small>②</small> :	New Subscription - e719e2
Duration <small>②</small> :	30
Start time <small>②</small> :	2023-08-25 <input type="button" value=""/>
End time <small>②</small> :	2023-09-24 <input type="button" value=""/>
User <small>②</small> :	paid-user-x
Pricing <small>②</small> :	pricing_casdoor
Plan <small>②</small> :	plan_premium
Payment <small>②</small> :	payment_20230825_160124_2d18867
Description <small>②</small> :	
State <small>②</small> :	<input checked="" type="radio"/> Active <input type="radio"/> Pending <input type="radio"/> Upcoming <input type="radio"/> Expired <input type="radio"/> Error <input type="radio"/> Suspended



&gt;

用户

# 用户

## 概述

Managing Users in Casdoor

## MFA / 2FA

Secure your account with MFA / 2FA

## User Roles

Roles assigned to users

## 权限

User Permissions

# 概述

## User Properties

作为一个认证平台，Casdoor 能够管理用户。 Every user has the following properties:

- Owner: The organization that owns the user
- Name: The unique username
- CreatedTime 创建时间
- UpdatedTime
- Id: Unique identifier for each user
- Type
- Password
- PasswordSalt
- PasswordOptions: Password complexity options
- DisplayName: Displayed in the user interface
- FirstName
- LastName
- Avatar: A link to the user's avatar
- PermanentAvatar
- Email
- Phone
- Location
- Address

- `Affiliation`
- `Title`
- `IdCardType`
- `IdCard`
- `Homepage`
- `Bio`
- `Tag`
- `Region`
- `Language`
- `Gender`
- `Birthday`
- `Education`
- `Score`
- `Karma`
- `Ranking`
- `IsDefaultAvatar`
- `IsOnline`
- `IsAdmin`: Indicates whether the user is an admin of their organization
- `IsGlobalAdmin`: Indicates whether the user has permission to manage the Casdoor
- `IsForbidden`
- `IsDeleted`
- `SignupApplication`
- `Hash`
- `PreHash`
- `CreatedIp`
- `LastSigninTime`

- `LastSigninIp`
- `Roles`: An array of the user's roles
- `Permissions`: An array of the user's permissions

Unique IDs for social platform logins:

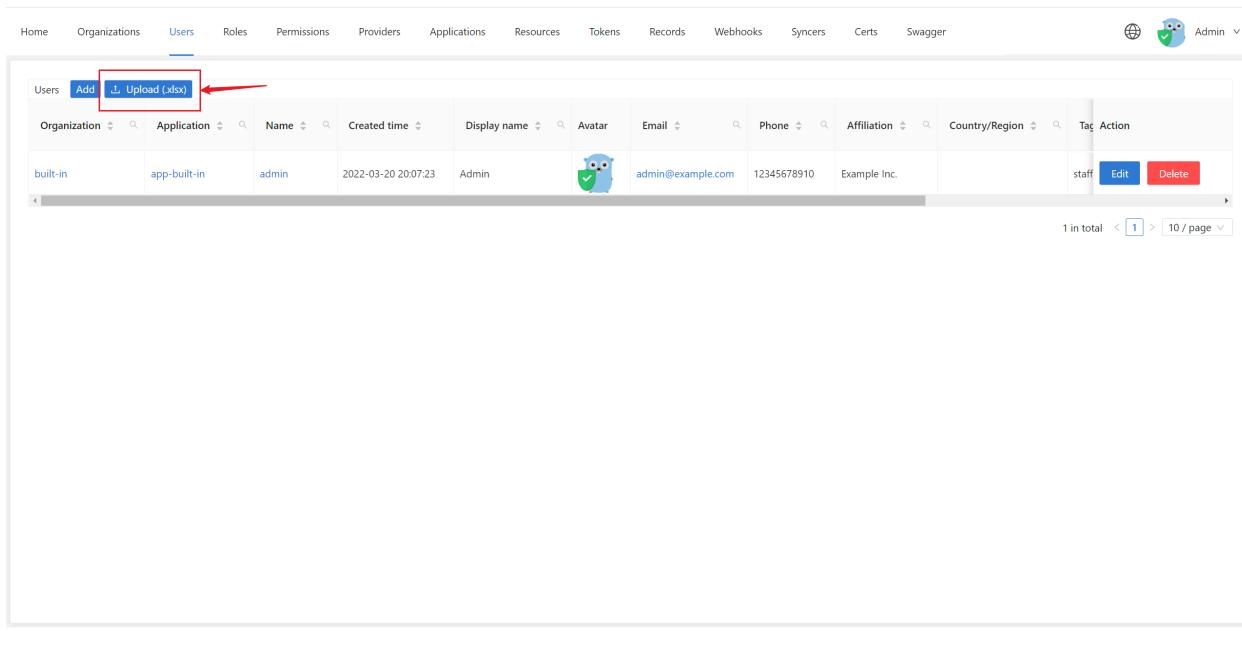
- `Github`
- `Google`
- `QQ`
- `微信`
- `Facebook`
- `钉钉`
- `微博`
- `Gitee`
- `LinkedIn`
- `Wecom`
- `Lark`
- `Gitlab`
- `Adfs`
- `Baidu`
- `Casdoor`
- `Infoflow`
- `Apple`
- `Azure AD`
- `Azure AD B2C`
- `Slack`
- `Steam`

- `Ldap`
- `Properties`: A string->string map that stores any additional properties.

## Importing Users from XLSX File

You can add new users or update existing Casdoor users by uploading an XLSX file containing user information.

In the Admin Console, go to Users and click the **Upload (.xlsx)** button.



The screenshot shows the Casdoor Admin Console interface. The top navigation bar includes links for Home, Organizations, Users (which is the active tab), Roles, Permissions, Providers, Applications, Resources, Tokens, Records, Webhooks, Syncers, Certs, and Swagger. On the far right, there is a globe icon, a user profile icon, and the word "Admin". Below the navigation bar is a search bar with fields for Organization, Application, Name, Created time, Display name, Avatar, Email, Phone, Affiliation, Country/Region, and Tag. The main area displays a table of users. One user is listed: "admin" (Organization: built-in, Application: app-built-in, Name: admin, Created time: 2022-03-20 20:07:23, Display name: Admin, Avatar: a blue owl icon, Email: admin@example.com, Phone: 12345678910, Affiliation: Example Inc., Country/Region: , Tag: staff). Action buttons for Edit and Delete are shown next to the user row. At the bottom of the table, it says "1 in total" and has navigation controls for pages 1 through 10. A small note at the bottom center says "Made with ❤️ by Casdoor".

Select your XLSX file and click Open. The users will be imported.

我们提供了 模板XLSX 文件 `user_test.xlsx` 在 `xlsx` 文件夹中。The template includes 5 test users and headers for some required user properties.

Users												Action
	Organization	Application	Name	Created time	Display name	Avatar	Email	Phone	Affiliation	Country/Region	Tag	Action
built-in	app-built-in	tesla	2022-03-20 20:49:03	Nikola Tesla		9v73hn@example.com	40738134827	Example Inc.	United States of America	scientist	<button>Edit</button>	<button>Delete</button>
built-in	app-built-in	gauss	2022-03-20 20:48:33	Carl Friedrich Gauss		vqdsan@example.com	98621482844	Example Inc.	Germany	mathematician	<button>Edit</button>	<button>Delete</button>
built-in	app-built-in	galileo	2022-03-20 20:47:58	Galileo Galilei		8p4f38@example.com	22596937332	Example Inc.	Italy	scientist	<button>Edit</button>	<button>Delete</button>
built-in	app-built-in	euler	2022-03-20 20:47:08	Leonhard Euler		3dzw4j@example.com	74409642681	Example Inc.	Switzerland	mathematician	<button>Edit</button>	<button>Delete</button>
built-in	app-built-in	einstein	2022-03-20 20:46:29	Albert Einstein		z6mive@example.com	60062541396	Example Inc.	Germany	scientist	<button>Edit</button>	<button>Delete</button>
built-in	app-built-in	admin	2022-03-20 20:07:23	Admin		admin@example.com	12345678910	Example Inc.		staff	<button>Edit</button>	<button>Delete</button>

Made with ❤ by [Casdoor](#)

# Bypass password encryption

When migrating users from an external database to Casdoor, there might be situations where you want to bypass or control the default encryption method provided by `organization.default.Password type` method.

This can be achieved by using the `passwordType` field during user import.

## ⓘ 备注

User with Bcrypt password

Below is an example of a POST body request for the API route `/api/add-user`.

```
{
}
```

Here, the user's password is already encrypted using the bcrypt algorithm, so we specify the `passwordType` as "bcrypt" to inform Casdoor not to encrypt it again.

# MFA / 2FA

## About multi-factor authentication

MFA (Multi-Factor Authentication) is a security measure that can enhance the security of users and systems. It requires users to provide two or more factors of authentication to verify their identity when logging in or performing sensitive operations.

For Casdoor, the second form of authentication is a code that is sent as a text message or email. Once you enable MFA, Casdoor generates an authentication code every time someone attempts to sign in to your account. The only way someone can sign in to your account is if they know both your password and have access to the authentication code.

## Configuring MFA

1. On the user profile page, you can see the configuration of multi-factor authentication. If you cannot see it, make sure the organization has added the multi-factor authentication item in the account items table.

Managed accounts (1) Managed accounts [Add](#)

Application	Username	Password	Action
			No data

Multi-factor authentication (1) Multi-factor methods

Type : sms	<a href="#">Setup</a>
Type : email	<a href="#">Setup</a>

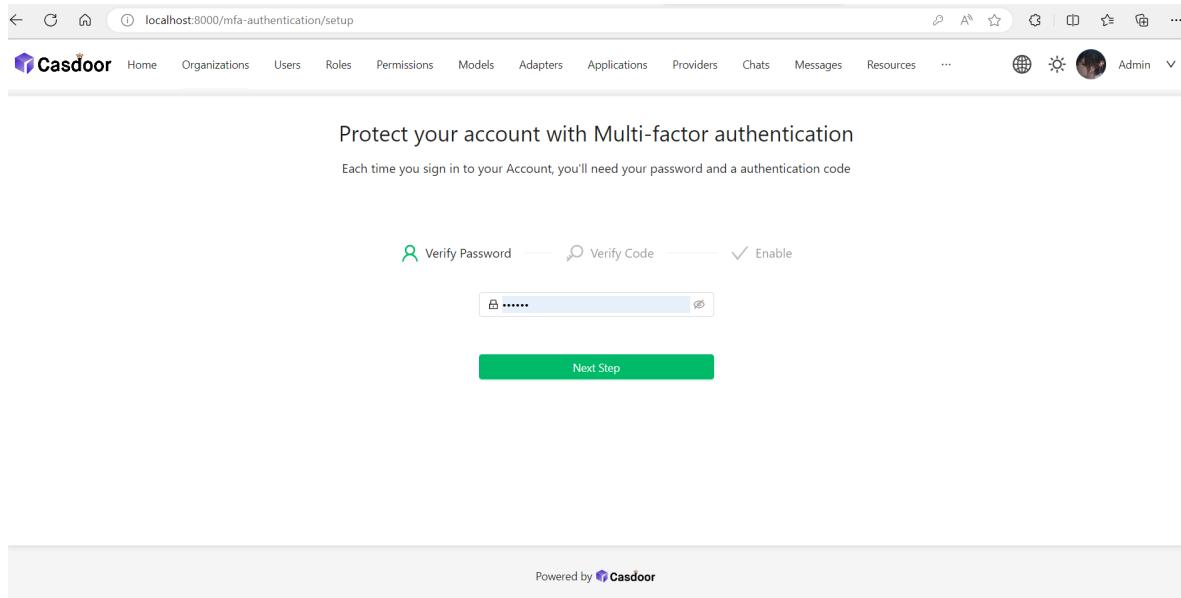
ID card (1) [Edit](#)

2. Click the "setup" button.

Multi-factor authentication (1) Multi-factor methods

Type : sms	<a href="#">Setup</a>
Type : email	<a href="#">Setup</a>

3. Type your password and click "Next Step".



## Configuring multi-factor authentication using a TOTP mobile app

A time-based one-time password (TOTP) application automatically generates an authentication code that changes after a certain period of time. We recommend using:

- [Google Authenticator](#)
- [Microsoft Authenticator](#).

### 💡 提示

To configure authentication via TOTP on multiple devices, during setup, scan the QR code using each device at the same time. If 2FA is already enabled, and you want to add another device, you must reconfigure your TOTP app from the user profile page.

## Protect your account with Multi-factor authentication

Each time you sign in to your Account, you'll need your password and a authentication code

 Verify Password —  Verify Code —  Enable



Scan the QR code with your authenticator app

Or copy the secret to your authenticator app

P757K7XT5MIO5RPZQYSC



 Passcode

Next Step

[Use email](#)    [Use SMS](#)

1. In the "Verify Code" step, do one of the following:
  - Scan the QR code with your mobile device's app. After scanning, the app displays a six-digit code that you can enter on Casdoor.
  - If you cannot scan the QR code, you can manually copy and enter the secret in your TOTP app instead.
2. The TOTP mobile application saves your account on Casdoor and generates a new authentication code every few seconds. On Casdoor, type the code into the "Passcode" field and click "Next Step".
3. Above the "Enable" button, copy your recovery codes and save them to your device. Save them to a secure location because your recovery codes can help

you regain access to your account if you lose access.

## Protect your account with Multi-factor authentication

Each time you sign in to your Account, you'll need your password and a authentication code

 Verify Password —  Verify Code —  Enable

Please save this recovery code. Once your device cannot provide an authentication code, you can reset mfa authentication by this recovery code

ad30de29-3ce0-4e39-a97f-ceff1d503d3c

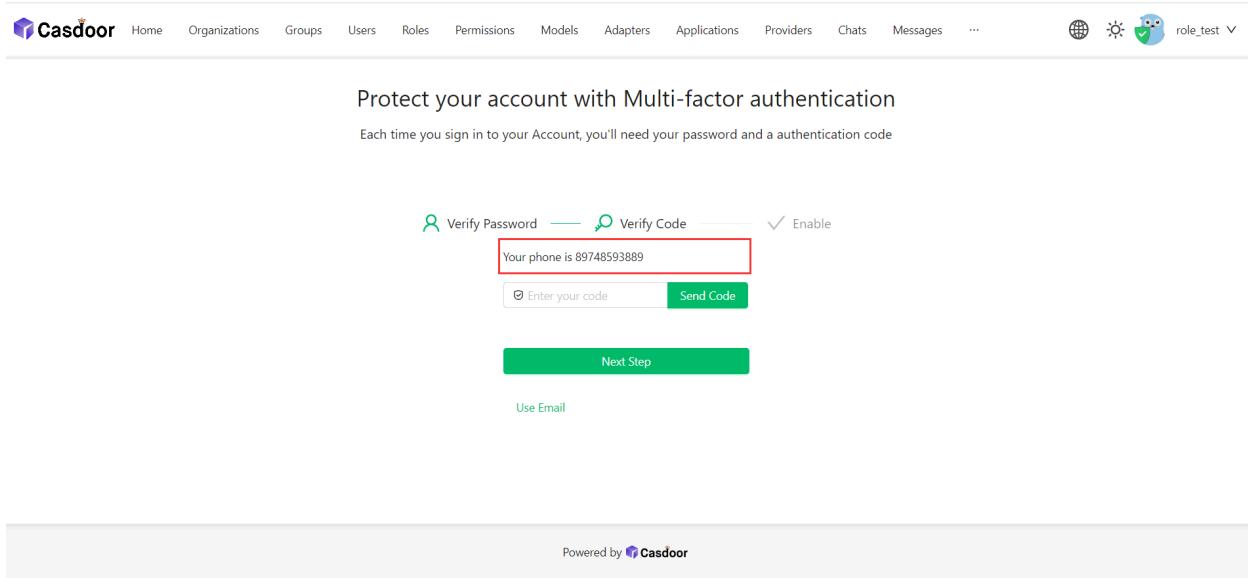
Enable

### 注意事项

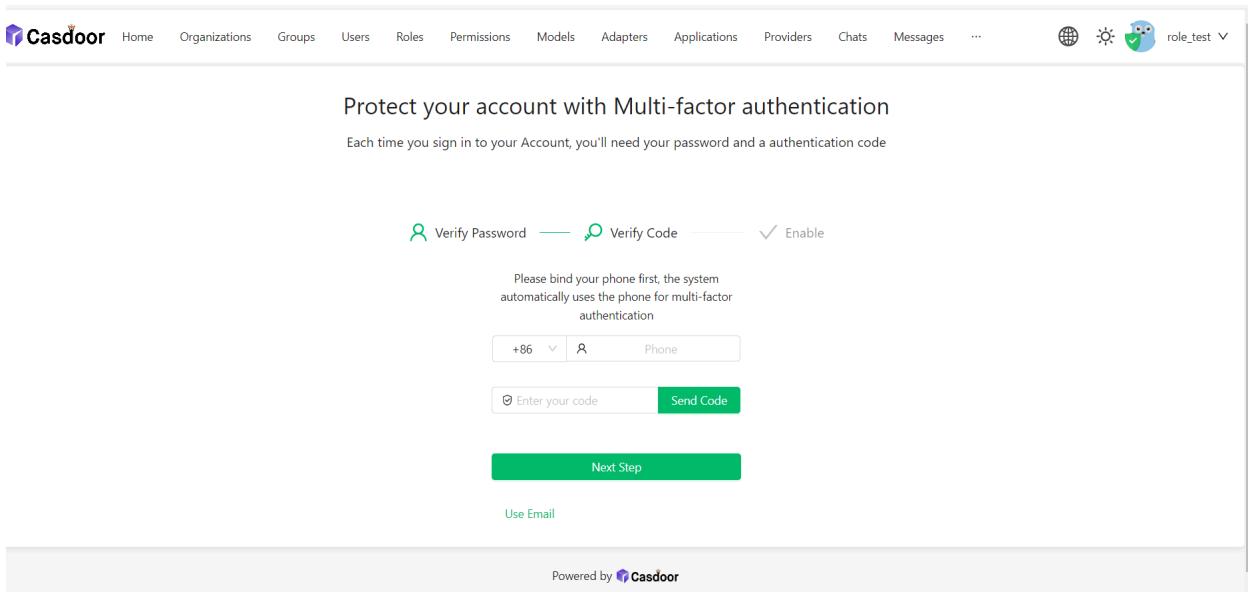
Each recovery code can only be used once. If you use a recovery code to sign in, it will become invalid.

## Configuring multi-factor authentication using text messages

If you have added your mobile phone number, Casdoor will use it to send you a text message.



If you have not added your mobile phone number, you need to add it first.



1. Select your country code and enter your mobile phone number.
2. Check if your information is correct and click "Send Code".
3. You will receive a text message with a security code. Then enter the code into the "Enter your code" field and click "Next Step".

4. Above the "Enable" button, copy your recovery codes and save them to your device. Save them to a secure location because your recovery codes can help you regain access to your account if you lose access.

## Configuring multi-factor authentication using email

Configuring email as your multi-factor authentication method is similar to using text messages.

1. Use your current email or enter your email address and click "Send Code".
2. Then enter the code into the "Enter your code" field and click "Next Step".
3. Above the "Enable" button, copy your recovery codes and save them to your device. Save them to a secure location because your recovery codes can help you regain access to your account if you lose access.

## Changing your preferred MFA method

You can add multiple MFA methods. Only the preferred method will be used when you sign in.

If you want to set a preferred MFA method, click the "Set preferred" button.

The screenshot shows a user interface for managing multi-factor authentication methods. At the top, there is a section titled "Multi-factor authentication" with a "Multi-factor methods" sub-section. Below this, two methods are listed:

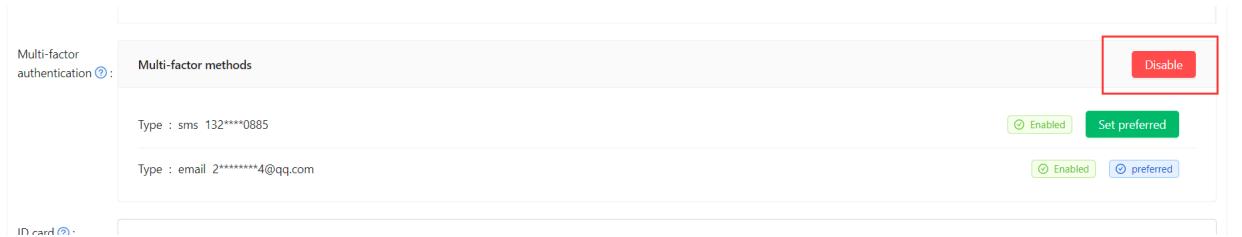
- Type : sms 132\*\*\*\*0885
- Type : email 2\*\*\*\*\*4@qq.com

For each method, there are two buttons: "Enabled" (green) and "preferred" (blue). The "preferred" button for the Email method is highlighted with a red box. At the bottom of the interface, there is a section labeled "ID card" with a dropdown menu.

A "Preferred" label will be displayed on your preferred method.

# Disabling multi-factor authentication

If you want to disable multi-factor authentication, click the "Disable" button. All your multi-factor authentication settings will be deleted.



# User Roles

Each user can have multiple roles. You can view the roles assigned to a user on their profile.

Bio [?](#) :

Tag [?](#) :

Signup application [?](#) :

Roles [?](#) : role\_test role\_test2

Permissions [?](#) : permission\_test

3rd-party logins [?](#) :

Is admin [?](#) :

Is global admin [?](#) :

Is forbidden [?](#) :

Is deleted [?](#) :

## Role Properties

Every role has the following properties:

- 所有者
- 名称
- 创建时间
- 显示名称
- 已启用
- Users: An array of sub users belonging to this role

- Roles: An array of sub roles belonging to this role

# 权限

每个用户可能有多个权限。 You can view the user's permissions on their profile.

The screenshot shows a user profile edit form. At the top, there are fields for Bio (empty), Tag (staff), Signup application (app-built-in), and Roles (role\_test, role\_test2). Below these, a section for Permissions is highlighted with a red box around the label and the value permission\_test. Further down, there are fields for 3rd-party logins, Is admin (unchecked), Is global admin (unchecked), Is forbidden (unchecked), and Is deleted (unchecked).

Bio ⓘ :

Tag ⓘ : staff

Signup application ⓘ : app-built-in

Roles ⓘ : role\_test role\_test2

Permissions ⓘ : permission\_test

3rd-party logins :

Is admin ⓘ :

Is global admin ⓘ :

Is forbidden ⓘ :

Is deleted ⓘ :

## Permission Properties

Permissions have the following properties:

- 所有者
- 名称
- 创建时间
- 显示名称
- 已启用
- 模型
- Users: An array of this role's sub-users

- Roles: An array of this role's sub-roles
- 资源类型
- Resources: An array of the resources
- Actions: An array of actions
- 效果



&gt;

Invitations

# Invitations



## Overview

Managing invitations in casdoor

# Overview

Currently casdoor already supports a more flexible invitation code method for user registration. Once the administrator opens the registration page with the invitation code as a mandatory option, users can only register if they have a valid invitation code.

Signup items <a href="#">?</a> :							
Name	Visible	Required	Prompted	Label	Placeholder	Regex	Rule
ID	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Random
Username	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
Display name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				None
Password	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
Confirm	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
Email	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Normal
Phone	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				None
Agreement	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				Only signup
Invitation code	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				

There are two main ways to use invitation codes, the default added is a random string code, composed of random numbers and letters. In order to be more flexible, the invitation code also supports regular matching to match multiple different invitation codes.

Invitations <a href="#">Add</a>										
Name	Organization	Updated time	Display name	Code	Quota	Used count	Application	Action		
invitation_794tdt	built-in	2024-02-04 20:52:15	New Invitation - 794tdt	319jed4tjk	1	0	All	<a href="#">Edit</a>	<a href="#">Delete</a>	
invitation_147y39	built-in	2024-02-04 20:47:47	New Invitation - 147y39	[a-zA-Z]2333	1	0	All	<a href="#">Edit</a>	<a href="#">Delete</a>	

## Invitation Properties

Casdoor manages invitations through the following properties

- **Organization**: The organization that owns the invitation
- **Name**: The unique invitation name
- **Display name**: Displayed Invitation Name
- **Code**: Invitation code, you can fill in the specific invitation code string, you can also fill in the regular expression
- **Default code**: Used to populate the default invitation code in the invitation link. For randomly generated invitation codes, the default code is the same as the invitation code. For code in regular expression form, you need to fill in the default code by yourself that matches the regular expression rule in the code
- **Quota**: Maximum number of times an invitation code can be used
- **Used count**: Number of times the invitation code has been used
- **Application**: Allow applications that use this invitation code. Selecting **ALL** makes it available to all apps under the organization
- **Username**: Specific username required when registering with this invitation
- **Email**: Specific email required when registering with this invitation
- **Phone**: Specific phone required when registering with this invitation
- **State**: Status of invitation

## Default Invitation

The invitation code in the default invitation is a randomly generated string of numbers and letters, and with **Quota** set to 1, it can only be used once. Application are set to **ALL** by default, which means that all apps under the organization corresponding to this invitation can use this invitation code.

New Invitation	Save	Save & Exit	Copy signup page URL	Cancel
Organization <small>?</small> :	built-in			
Name <small>?</small> :	invitation_lxxgdh			
Display name <small>?</small> :	New Invitation - lxxgdh			
Code <small>?</small> :	yo8hrfa7sf			
Default code <small>?</small> :	yo8hrfa7sf			
Quota <small>?</small> :	1			
Used count <small>?</small> :	0			
Application <small>?</small> :	All			
Username <small>?</small> :				
Email <small>?</small> :				
Phone <small>?</small> :				
State <small>?</small> :	Active			

If the invitation code is set for a specific user and you want the user to register with the given `username`, `email`, `phone` and `invitation code`, you can restrict the user's registration by filling in the corresponding fields. If the fields are empty or if they are not configured on the registration page, casdoor does not force validation of these fields

Username <small>?</small> :	Bob
Email <small>?</small> :	bob@qq.com
Phone <small>?</small> :	15295959595

When it is necessary to reuse an invitation code, you can set `Quota` to a larger value, for example, if you want this invitation code to be used 10 times, then you can set `Quota` to 10. When you wish to stop registering with this invitation code, you can also do this by modifying the status of the invitation to `Suspended`.

Quota ⓘ :	<input type="text" value="10"/>
Used count ⓘ :	<input type="text" value="0"/>
Application ⓘ :	<input type="text" value="application_phs9si"/>
Username ⓘ :	<input type="text"/>
Email ⓘ :	<input type="text"/>
Phone ⓘ :	<input type="text"/>
State ⓘ :	<input type="text" value="Suspended"/>

### ⚠ 注意事项

When `username`, `email`, or `phone` is configured in the invitation, the `quota` should not be greater than one. This is because the user's `username`, `email`, and `phone` should be unique, and multiple users should not be able to register using the same `username`, `email`, or `phone`.

## Regular Match Invitation

Sometimes there is a need for a large number of invitation codes for user registration, and generating invitation codes one by one can be very inefficient. Casdoor supports validating invitation codes through regular expression matching. For example, by setting the `Code` as `"[a-z]2333"`, any invitation code that matches this regular expression will be successfully matched as a valid invitation code.

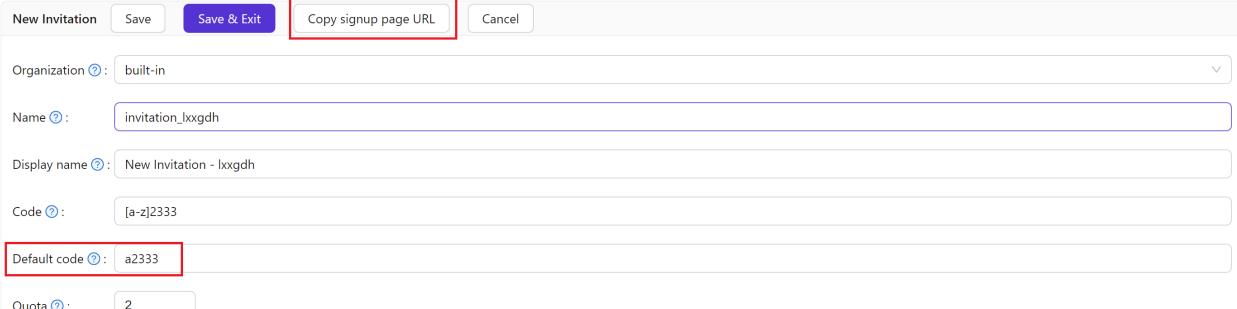
Code ⓘ :	<input type="text" value="[a-z]2333"/>
Default code ⓘ :	<input type="text" value="a2333"/>
Quota ⓘ :	<input type="text" value="2"/>
Used count ⓘ :	<input type="text" value="0"/>

### ⓘ 备注

When using regular expressions to validate invitation codes, each invitation code that matches the regular expression can only be used once, and the Quota can still limit the number of usages. For example, when the Code is "[a-z]2333" and the Quota is 2, only a maximum of two invitation codes that match the regular expression can be successfully used.

## Invitation Link

Casdoor supports copying the invitation link corresponding to an invitation. The invitation code in the invitation link corresponds to the Default code field. Therefore, for invitations that use regular expressions, the Default code must be manually filled in to generate the correct invitation link. Additionally, when registering using an invitation link, the registration page will automatically populate certain field information set by the invitation corresponding to the invitation code.



The screenshot shows the 'New Invitation' form in Casdoor. The fields are as follows:

- New Invitation (button)
- Save (button)
- Save & Exit (button)
- Copy signup page URL (button, highlighted with a red box)
- Cancel (button)
- Organization: built-in
- Name: invitation\_lxxgdb
- Display name: New Invitation - lxxgdb
- Code: [a-z]2333
- Default code: a2333 (highlighted with a red box)
- Quota: 2



\* Username:

\* Password:

\* Email:

\* Phone:

\* Invitation code:

[Sign Up](#) Have account? [sign in now](#)

Powered by Casdoor

# Demo



&gt;

同步器

# 同步器

## 概述

Synchronizing users in Casdoor

## 数据库

Using Database Syncer to synchronize databases

## Keycloak

使用 Keycloak Syncer 同步 Keycloak

## WeCom

Using WeCom Syncer to synchronize databases

# 概述

As an authentication platform, Casdoor can easily manage users stored in databases.

## 同步器

Casdoor stores users in the user table. So, when you plan to use Casdoor as an authentication platform, there is no need to worry about migrating your application's user data into Casdoor. Casdoor provides a **syncer** to quickly help you synchronize user data to Casdoor.

You need to specify the database and user table that you want to synchronize with Casdoor, and the syncer will take care of syncing the data at regular intervals. For more details, refer to the [database syncer](#).

## 同步哈希值

Casdoor uses a hash function to determine how to update a user. This hash value is calculated for each user in the table, using information such as the password or mobile phone number.

If the calculated hash value of a user with a specific `Id` changes compared to the original value, Casdoor confirms that the user table has been updated. Subsequently, the database updates the old information, thereby achieving **bilateral synchronization** between the Casdoor user table and the original user table.

# 数据库

## 数据库同步器

The users table we created as a demo is imported from the [template XLSX file](#).

owner	name	created_time	updated_time	id	type	password	password_salt	display_name	first_name	last_name	avatar	permanent_avatar_email
built-in	einstein	2022-03-20T20:46:29+08:00		1c57cc37-37f5-4def-9e9f-082189ef63d2	normal-user	123		Albert Einstein			https://casbin.org	z6mive@
built-in	euler	2022-03-20T20:47:08+08:00		bb7831b4-0d24-4e96-b043-f8fd8d15eb	normal-user	123		Leonhard Euler			https://casbin.org	3dzw4j@
built-in	galileo	2022-03-20T20:47:58+08:00		7920eb6c-f9f5-40ef-8e18-3ac99f49bd15	normal-user	123		Galileo Galilei			https://casbin.org	8p4f38@
built-in	gauss	2022-03-20T20:48:33+08:00		f0c28816-2c0d-479b-b545-cb4cf96db36	normal-user	123		Carl Friedrich Gauß			https://casbin.org	vqdsan@
built-in	tesla	2022-03-20T20:49:03+08:00		687c3068-fd21-4d32-b2ba-e13e8b369a	normal-user	123		Nikola Tesla			https://casbin.org	9v73hn@

To create a new syncer, go to the **Syncers** tab and fill in all the required information as shown below. Then, save the changes.

Organization ② :	built-in																																		
Name ② :	syncer_qmpox9																																		
Type ② :	Database																																		
Host ② :	localhost																																		
Port ② :	3306																																		
User ② :	root																																		
Password ② :	password																																		
Database type ② :	MySQL																																		
Database ② :	auth																																		
Table ② :	user																																		
Table columns ② :	<table border="1"> <thead> <tr> <th>Column name</th> <th>Add</th> <th>Column type</th> <th>Casdoor column</th> <th>Is key</th> <th>Is hashed</th> <th>Action</th> </tr> </thead> <tbody> <tr> <td>name</td> <td></td> <td>string</td> <td>Name</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td><td></td><td></td></tr> <tr> <td>id</td> <td></td> <td>string</td> <td>Id</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td><td></td><td></td></tr> <tr> <td>first_name</td> <td></td> <td>string</td> <td>FirstName</td> <td><input checked="" type="checkbox"/></td> <td><input checked="" type="checkbox"/></td> <td></td><td></td><td></td></tr> </tbody> </table>	Column name	Add	Column type	Casdoor column	Is key	Is hashed	Action	name		string	Name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				id		string	Id	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>				first_name		string	FirstName	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
Column name	Add	Column type	Casdoor column	Is key	Is hashed	Action																													
name		string	Name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>																														
id		string	Id	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>																														
first_name		string	FirstName	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>																														



In general, you need to fill in at least the **ID** and **Name** in the Casdoor

columns. Other important fields include `createdTime`, `Password`, and `DisplayName`.

The following fields are required:

- `Organization`: The organization that the user will be imported to
- `Name`: 同步器名称
- `Type`: Select "database"
- `Host`: 原始数据库主机地址
- `Port`: 原始数据库端口
- `User`: 原始数据库用户名
- `Password`: 原始数据库密码
- `Database type`: All Xorm-supported databases such as MySQL, PostgreSQL, SQL Server, Oracle, and SQLite
- `Database`: 原始数据库名称
- `Table`: The original user table name
- `表格列`
- `Column name`: The original user column name
- `Column type`: The original user column type
- `Casdoor Column`: The Casdoor user column name

Optional fields:

- `Is hashed`: 是否计算哈希值. When this option is enabled, the syncer will only synchronize the user if the field of the user in the origin table is updated. If this option is disabled, the syncer will still synchronize the user even if only the field is updated. In short, the user will not be synchronized until the fields involved in the hash calculation (enabled "Is hashed") are updated.
- `Is key`: Whether it is the primary key of the user in the origin table and the

user in the Casdoor table. When synchronizing the database, it is determined based on the field whose "Is key" option is selected. At least one of the "Is key" buttons for fields should be selected. If none are selected, the first "Is key" option is selected by default.

- **Avatar base URL**: When syncing users, if the **Avatar base URL** is not empty and the origin `user.avatar` does not have the prefix "http", the new `user.avatar` will be replaced by **Avatar base URL** + `user.avatar`.
- **Affiliation table**: It is used to sync the affiliation of the user from this table in the database. Since the affiliation may be a code of type int in the "Affiliation table", it needs to be mapped to a string. Refer to [getAffiliationMap\(\)](#). Casdoor has some redundant fields to borrow, so [here](#) we use **score** to map the int code to a string name.

Once you have configured the syncer, enable the **Is enable** option and save. The syncer will start working.

Syncers											Add	
Name	Organization	Created time	Type	Host	Port	User	Password	Database type	Database	Action		
syncer_qmpox9	built-in	2023-08-09 18:57:36	Database	localhost	3306	root	password	mysql	auth	<button>Sync</button>	<button>Edit</button>	<button>Delete</button>

You can also use the "Sync" button for database synchronization.

## Update

When the **Table columns** are set as shown in the following figure, the update operation is performed.

Table columns <small>(?)</small>											
Table columns			Column type		Casdoor column		Is key	Is hashed	Action		
Column name	Column type										
name	string		Name				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			
id	string		Id				<input type="checkbox"/>	<input checked="" type="checkbox"/>			
first_name	string		FirstName				<input type="checkbox"/>	<input checked="" type="checkbox"/>			
password	string		Password				<input type="checkbox"/>	<input checked="" type="checkbox"/>			

If the data in the two tables is different for the key, you can synchronize the data

between the two tables based on the primary key.

- Update user in the original table
- Update user in the Casdoor table

## Add

When the `Table columns` are set as shown in the following figure, the add operation is performed.

Table columns (2):						
Column name	Column type	Casdoor column	Is key	Is hashed	Action	
name	string	Name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
id	string	Id	<input type="checkbox"/>	<input checked="" type="checkbox"/>		
first_name	string	FirstName	<input type="checkbox"/>	<input checked="" type="checkbox"/>		
password	string	Password	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

If the number of data between the two tables is different, add the data to the table with the lower number of data based on the primary key.

- Add user in the original table
- Add user in the Casdoor table

# Keycloak

## Keycloak 同步器

The Keycloak syncer is essentially the same as the [database syncer](#), with the added functionality of automatic configuration for Keycloak [Tables](#) and [Table columns](#).

Furthermore, the Keycloak syncer will fetch data from the [credential](#) table, [keycloak\\_group](#) table, and [user\\_group\\_membership](#) table, as user information in Keycloak is stored across multiple tables.

Table columns	Add	Column name	Column type	Casdoor column	Is hashed	Action
ID	string	Id	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
USERNAME	string	Name	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
EMAIL	string	DisplayName	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
EMAIL_VERIFIED	boolean	Email	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
FIRST_NAME	string	EmailVerified	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
LAST_NAME	string	FirstName	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
CREATED_TIMESTAMP	string	LastName	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
ENABLED	boolean	CreatedTime	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		
		IsForbidden	<input checked="" type="checkbox"/>	<span>^</span> <span>v</span> <span>o</span>		

# WeCom

## WeCom Syncer

By using WeCom syncer, you can sync WeCom user and department data to Casdoor's user table and group table.

The following fields are required:

- Organization: The organization that the user will be imported to
- Name: The syncer's name
- Type: Select "WeCom"
- User: Your WeCom Company ID
- Password: Your WeCom App secret
- ClientSecret: Your WeCom Sync of Contacts secret

Follow the steps below to configure.

### Step 1: Get WeCom Syncer configuration items

- In your WeCom management platform, navigate to My Company, get Company ID in Company Information.

WeCom

Service Provider Console | API Documentation | CSR | Quit

Homepage Contacts Collaboration App Management Customers and Partners Advanced Features Security and Management My Company

**Company Information**

**Company Information**

Permissions	Company Logo		Go to verify
Chat Management	Company short name	usher <span style="border: 1px solid red; padding: 2px;">未认证</span> <a href="#">Modify</a>	Company not verified. After verification, the number of users can be increased.
Contacts Management	Company address	<a href="#">Add</a>	
Workspace Management	Phone No.	<a href="#">Add</a>	
WeChat Workplace	Company Domain Na...	<a href="#">Add</a>	
External Communication Management	Company member	1 member(s) <a href="#">Statistics</a>	
Security and Confidentiality	Company Department	1 dept(s)	
Setting	Added/Max.	1/1000 <a href="#">Verify now to increase the limit</a>	
	Invoice Title	<a href="#">Add</a> Set VAT invoice titles for company members <span style="color: blue;">?</span>	
	Industry Type	Internet and Related Services <a href="#">Modify</a>	
	Company Size	1-50 <a href="#">Modify</a>	
	Creation time	2023-6-18	
	Company ID	ww752595f99d89b1ca	

Already a WeCom service provider. Go to Service Provider Platform

- In your Self-build App, get **App secret**.

WeCom

Service Provider Console | API Documentation | CSR | Quit

Homepage Contacts Collaboration **App Management** Customers and Partners Advanced Features Security and Management My Company

[« Back](#) Casdoor

 Casdoor <a href="#">View</a>	No app info	Enabled <input checked="" type="checkbox"/>
AgentId	1000003	<a href="#">Edit</a>
Secret	<a href="#">View</a>	
Allowed users	 usher	

- In Sync of Contacts Management Tool, get **Sync of Contacts secret**.

Sync of Contacts

**Sync of Contacts**

Companies can sync contacts through APIs or authorized third-party service providers [API Documentation](#)

Sync method      API Sync

Permission      [Read and edit Contacts](#) [Edit](#) [View Permission Details](#)

Secret      [View](#) [Send again](#)

Company's Trusted IP      3 IP address(es) are configured. [Settings](#)  
Only configured IP addresses can access company data via API.

Set event receiving server      The added member or department will be pushed to the following URL in the form of event to ensure the Contacts is synced. [Learn More](#)

[Disable API sync](#)

## Step2: Config Casdoor WeCom Syncer

Go to Syncers tab, select **WeCom** type and fill in the required information as shown below. Then, save the changes.

[Edit Syncer](#) [Save](#) [Save & Exit](#)

Organization [?](#) :

Name [?](#) :

Type [?](#) :

Database type [?](#) :

Host [?](#) :

Port [?](#) :

User [?](#) :  Company ID

Password [?](#) :  App secret

Client secret [?](#) :  Sync of contacts secret

Database [?](#) :

Table [?](#) :



&gt;

令牌

# 令牌



## 概述

Casdoor令牌简介

# 概述

Casdoor is built on OAuth and utilizes tokens as users' OAuth tokens.

The following are the available token fields in Casdoor:

- Owner
- Name
- CreatedTime
- Application
- Organization
- User
- Code
- AccessToken
- ExpireIn (Tokens will expire in hours)
- Scope (Scope of authorization)
- TokenType (e.g., Bearer type)

After logging into the application, there are three options to generate a JWT Token:

- JWT
- JWT - Empty
- JWT - Custom

The options are as follows: JWT will generate a token containing all User fields, JWT-Empty will generate a token with all non-empty values for the user, and JWT-Custom will generate a token containing custom User Token fields (you can

choose attributes in the Token fields).

Token format [?](#) : JWT-Custom

Token fields [?](#) : [Owner](#) [x](#) [CreatedTime](#) [x](#) [DisplayName](#) [x](#) [UpdatedTime](#) [x](#) |

Token expire [?](#) :

Refresh token expire [?](#) :

Failed signin limit [?](#) :

Failed signin frozen time [?](#) :

Owner	Name	✓
CreatedTime	UpdatedTime	✓
Id	Type	✓
Password	PasswordSalt	



&gt;

Webhooks

# Webhooks

## 概述

Adding Webhooks in Casdoor

# 概述

## 概述

Event systems enable you to create integrations that subscribe to specific events in Casdoor. When one of these events is triggered, a JSON payload will be sent to the configured URL via a POST request. The application will parse the JSON payload and execute the specified function. Events include signup, login, logout, and user updates, all of which are stored in the action field of the record. 事件系统可以用来更新用户的外部问题。



&gt;

LDAP

# LDAP

## 概述

Casdoor cooperates with an LDAP server

## Configuring and Syncing LDAP Users

Configuring LDAP in Casdoor for user synchronization

## LDAP 服务器

How to connect LDAP client in Casdoor

# 概述

Support for an LDAP server has been introduced into Casdoor. Casdoor is able to synchronize users from LDAP servers to Casdoor in order to use them as user accounts for logging in, and authenticate them using the LDAP servers. Casdoor also supports setting up cron jobs to synchronize users automatically on a regular basis.

## Details about Casdoor-LDAP synchronization mechanism

The way Casdoor cooperates with an LDAP server is described as follows:

1. Synchronization: Casdoor can connect to an LDAP server, fetch users' information, and read all users' information (including `uidNumber`, `uid`, `cn`, `gidNumber`, `mail`, `email`, `emailAddress`, `telephoneNumber`, `mobile`, `mobileTelephoneNumber`, `registeredAddress`, `postalAddress`). It then creates Casdoor accounts for each user in the LDAP, and stores these accounts in the database.
2. Authentication: As we have seen, Casdoor does not fetch LDAP users' passwords. When an account that is synchronized from the LDAP server tries to log in through Casdoor, instead of checking the password stored in Casdoor's database, Casdoor connects to the LDAP server and verifies whether the user's password is correct.
3. User identification: Casdoor uses `uid` to exclusively identify a user. Therefore, please ensure that every LDAP user has a unique `uid`.

Once a user is synchronized into Casdoor, their information is independent from the LDAP user. This means that if you modify the user's information in Casdoor, the user's information in the LDAP will not be modified, and vice versa (except for the LDAP user's password, as we rely on it to authenticate the user).

# Configuring and Syncing LDAP Users

LDAP configurations are specific to each organization, as LDAP users will be synchronized into them.

To modify the configuration, you need to use a global admin user. Enter the following information for LDAP user synchronization on the "organization" page.

LDAP :		添加			
LDAP服务器	服务器	基本DN	自动同步	最近同步	操作
Example LDAP Server	example.com:389	ou=People,dc=example,dc=com	Disable		<button>同步</button> <button>编辑</button> <button>删除</button>

# Configuring Connection to LDAP Server

Edit LDAP    Save    **Save & Exit**    Sync LDAP

Organization ② :	built-in
ID ② :	691edec0-f1ab-4e23-8f9f-a824a383032f
Server name ② :	Example LDAP Server
Server host ② :	example.com
Server port ② :	389
Enable SSL ② :	<input checked="" type="checkbox"/>
Base DN ② :	ou=built-in,dc=example,dc=com
Search Filter ② :	(objectClass=posixAccount)
Filter fields ② :	<input type="checkbox"/> uid <input type="checkbox"/> Email
Admin ② :	cn=admin,dc=example,dc=com
Admin Password ② :	.....
Auto Sync ② :	0 mins

## Server Name

A friendly name that managers can use to identify different servers.

Example: Example LDAP Server

## Server Host

The host or IP address of the LDAP server.

Example: `example.com`

## Server Port

The port number of the LDAP server. Only numeric values are allowed.

Example: `389`

## Base DN

Casdoor uses Sub search mode by default when searching in LDAP. The Base DN is the basic distinguished name used for the search. Casdoor will return all users under the specified Base DN.

The admin account configured in Casdoor should have at least read-only permissions at the Base DN.

Example: `ou=Example, dc=example, dc=com`

## Search Filter

Casdoor uses a search filter to query LDAP users.

Example: `(objectClass=posixAccount)`

## Filter Fields

Filter fields are the identifiers of the user in the LDAP server. When logging in to Casdoor as an LDAP user, the entered login username is regarded as the `uid` of the LDAP user. You can also configure other fields, such as `mail` or `mobile`.

The screenshot shows the Casdoor web application interface for managing LDAP configurations. The main title is "Edit LDAP". The form fields include:

- Organization: built-in
- ID: 691edec0-f1ab-4e23-8f9f-a824a383032f
- Server name: Example LDAP Server
- Server host: 1
- Server port: 389
- Enable SSL: (checkbox)
- Base DN: ou=built-in,dc=example,dc=com
- Search Filter: (objectClass=inetOrgPerson)
- Filter fields: (empty)
- Admin: cn=admin,dc=example,dc=com
- Admin Password: (redacted)

Buttons at the top right include "Save", "Save & Exit", and "Sync LDAP".

## Admin

An account that can log in to the specified LDAP server.

The login method (DN or ID) depends on the LDAP server settings you want to connect to.

Example: `cn=manager,dc=example,dc=com`

## Admin Password

The password for the LDAP server Admin account.

## Auto Sync

Set to `0` to disable auto sync. Any other value means Sync every few minutes.

# Synchronizing Users

The sync table displays all the users obtained from the LDAP server within the specific `ou`. If the users have already been synced, the checkbox will be disabled. You can select the users by checking the box, and then sync the selected users from the LDAP server.

Example LDAP Server		Sync	Edit LDAP			
<input type="checkbox"/>	CN	UidNumber / Uid	Group Id	Email	Phone	Address
<input type="checkbox"/>	zhan san	1000 / zsan	500			
<input type="checkbox"/>	li si	1001 / lsi	500			
<input type="checkbox"/>	a dmin	1002 / admin	500			
<input type="checkbox"/>	tom brown	1007 / jery	500			
<input type="checkbox"/>	wrie jerry	1003 / wjerry	500			
<input type="checkbox"/>	admin2	1004 / admin2	500			
<input type="checkbox"/>	yyyy	1005 / yyyy	500			

< 1 > 10 / page ▾

## ⚠ 注意事项

If the `uid` of a user in the LDAP server is the same as the `name` of an existing user in the Casdoor organization, Casdoor will create a new user with a `name` that includes the `uid` and a random string. However, this user may not be able to log in because the name of the newly synced user does not exist in the LDAP server. Therefore, it is recommended to avoid this situation.

# LDAP 服务器

Many systems, like [Nexus](#), support LDAP authentication. Casdoor also implements a simple LDAP server, which supports bind and search operations.

This document describes how to connect to the LDAP server in Casdoor and implement simple login authentication.

## LDAP 服务器端口号

The LDAP server listens on port [389](#) by default. You can change the default port by modifying the [ldapServerPort](#) value in [conf/app.conf](#).

## How it Works

Similar to the LDAP client in Casdoor, the users in the LDAP server are all subclasses of [posixAccount](#).

When the server receives a set of data transmitted by the LDAP, it will parse the [cn](#) and [ou](#), where [cn](#) represents the username and [ou](#) represents the organization name. The [dc](#) does not matter.

If it is a bind operation, the server will use Casdoor to verify the username and password and grant the user permission in Casdoor.

If it is a search operation, the server will check whether the search operation is legal, according to the permissions granted to the client by the bind operation, and return a response.

## ① 信息

We only support Simple Authentication.

## How to Bind

In Casdoor LDAP server, we only recognize `DN` similar to this format:

`cn=admin,ou=built-in,dc=example,dc=com`.

Please set the `DN` of the admin user to the above format. Then, you can use this `DN` to bind to the LDAP server with the user's password to log in to Casdoor for verification. If the server verification is successful, the user will be granted authority in Casdoor.

## How to Search

Once the bind operation completes successfully, you can perform the search operation. There are some differences between search and bind operations.

- To search for a certain user, such as `Alice` under the `built-in` organization, you should use a `DN` like this: `ou=built-in,dc=example,dc=com`, and add `cn=Alice` in the Filter field.
- To search for all users under a certain organization, such as all users in `built-in`, you should use a `DN` like this: `ou=built-in,dc=example,dc=com`, and add `cn=*` in the Filter field.
- To search for all users in all organizations (assuming the user has sufficient permissions), you should use a `DN` like this: `ou=*,dc=example,dc=com`, and add `cn=*` in the Filter field.



&gt;

RADIUS

# RADIUS



## Overview

Use Casdoor as RADIUS server

# Overview

You can use Casdoor as a RADIUS server. RADIUS is a client/server protocol, the client can be a NAS or any computer running RADIUS client software.

## Congiure

Before deploying Casdoor, you need to modify the RADIUS-related configurations in the `conf/app.conf` file, including the server port and secret:

```
radiusServerPort = 1812  
radiusSecret = "secret"
```

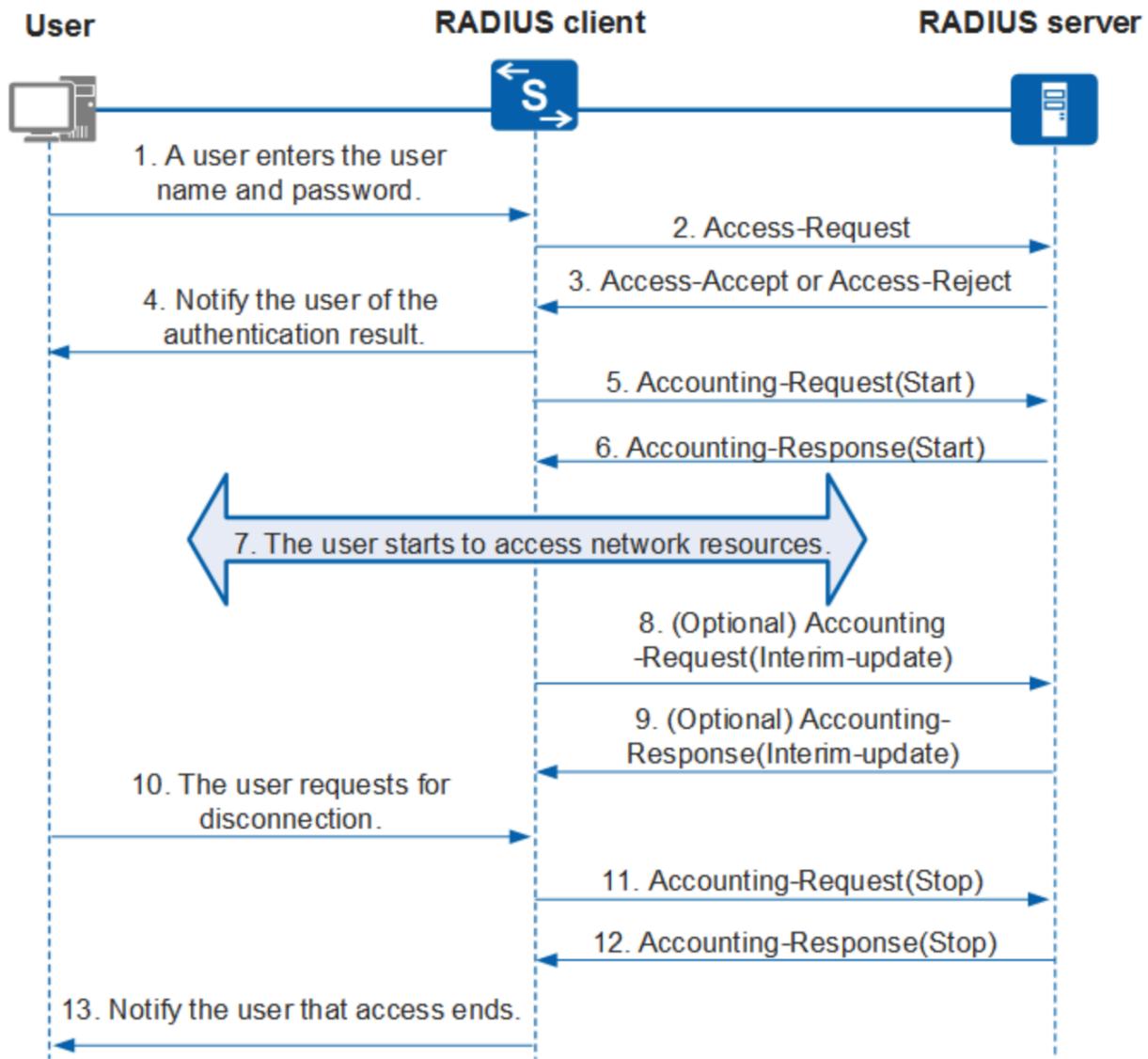
Now you can use Casdoor as RADIUS server.

## Use Casdoor as RADIUS server

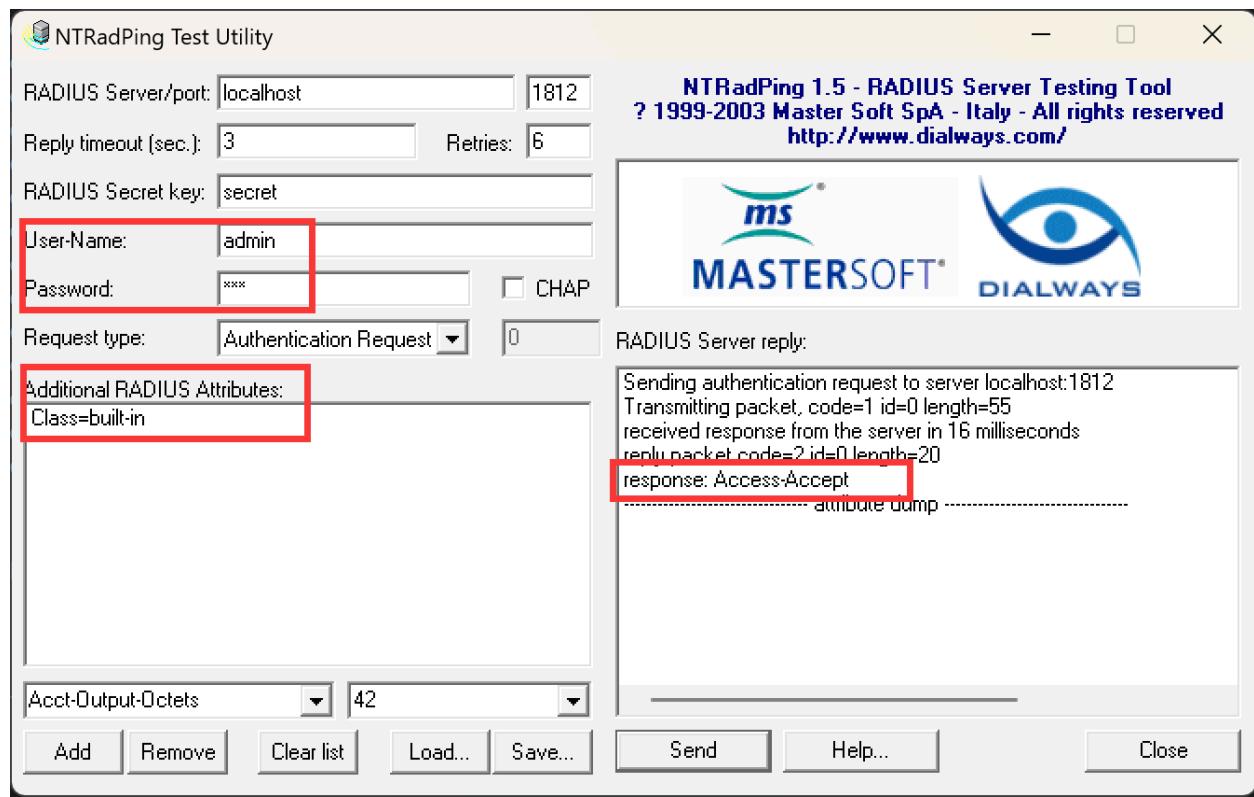
Casdoor currently can support follow standard RADIUS request:

- `Access-Request` : The authentication request message is sent by the RADIUS client to the Casdoor. Casdoor determines whether to allow access based on the user information carried in the message and reply with `Access-Reject` or `Access-Accept`.
- `Accounting-Request` : When a user starts or stops accessing network resources, the RADIUS client will send accounting request (Start/Interim-update/Stop) message to Casdoor. Casdoor will record relevant accounting

request message and reply with `Accounting-Response`.



Since Casdoor use Organization to manage User, where each User belongs to a specific Organization, the `Class` attribute in the request needs to be set as the User's Organization.





&gt;

SCIM

# SCIM



## Overview

Use Casdoor as SCIM service provider

# Overview

The SCIM protocol is a HTTP-based protocol for provisioning and managing identity data specified through SCIM schemas. You can use Casdoor as a SCIM service provider.

## Use Casdoor as SCIM service provider

Currently Casdoor only support `User Resource Schema`, you can manage users through SCIM User operations. You can interact with the Casdoor through the following endpoints:

Endpoint	Method	Description
/scim/ ServiceProviderConfig	GET	Provide details about the features of the SCIM standard that are supported, for example, the resources that are supported.
/scim/Schemas	GET	Provide details about the service provider schemas.
/scim/ResourceTypes	GET	Specify metadata about each resource.
/scim/Users/:id	GET	Retrieve a user with resource identifier <code>id</code> .

Endpoint	Method	Description
/scim/Users	GET	Query users with query parameters (currently only support <code>startIndex</code> and <code>count</code> ).
/scim/Users	POST	Create a user.
/scim/Users/:id	PUT	Update a user with resource identifier <code>id</code> .
/scim/Users/:id	PATCH	Modify a user with resource identifier <code>id</code> by PATCH operation.
/scim/Users/:id	DEL	Delete a user with resource identifier <code>id</code> .

For more details, please refer to [rfc7644](#).

## User Resource

Casdoor implements the mapping between `User Resource Schema` (SCIM) and `User` (Casdoor). The mapping relationship between attributes is as follows:

User Resource Schema (SCIM)	User (Casdoor)
<code>id</code>	<code>Id</code>
<code>meta.created</code>	<code>CreatedTime</code>

User Resource Schema (SCIM)	User (Casdoor)
meta.lastModified	UpdatedTime
meta.version	UpdatedTime
externalId	ExternalId
userName	Name
password	Password
displayName	DisplayName
profileUrl	Homepage
userType	Type
name.givenName	FirstName
name.familyName	LastName
emails[0].value	Email
phoneNumbers[0].value	Phone
photos[0].value	Avatar
addresses[0].locality	Location
addresses[0].region	Region

User Resource Schema (SCIM)	User (Casdoor)
addresses[0].country	CountryCode

Since Casdoor use Organization to manage User, where each User belongs to a specific Organization, the `organization` attribute should be passed in `Enterprise User Schema Extension` (identified by the schema URI `urn:ietf:params:scim:schemas:extension:enterprise:2.0:User`). Here is a User Resource Schema SCIM representation in JSON format:

```
{
  "active": true,
  "addresses": [
    {
      "country": "CN",
      "locality": "Shanghai",
      "region": "CN"
    }
  ],
  "displayName": "Bob~",
  "emails": [
    {
      "value": "test1@casdoor.com"
    }
  ],
  "externalId": "1234123543234234",
  "id": "ceacbc6-40d0-48f1-af23-0990232d570a",
  "meta": {
    "resourceType": "User",
    "created": "2023-10-08T23:51:55+08:00",
    "lastModified": "2023-10-12T20:38:49+08:00",
    "location": "Users/ceacbc6-40d0-48f1-af23-0990232d570a",
    "version": "2023-10-12T20:38:49+08:00"
  },
  "name": {
    "familyName": "Doe",
    "givenName": "Bob"
  }
}
```





&gt;

集成

# 集成



3 个项目



1 个项目



9 个项目



17 个项目



## JavaScript

2 个项目



## Lua

1 个项目



## PHP

4 个项目



## Ruby

1 个项目



## Haskell

1 个项目



Python

1 个项目



> 集成 >

C++

# C++

## Nginx

Using Casdoor with Nginx

## NginxCommunityVersion

Using Casdoor with Nginx (Not Nginx-Plus) and Oauth2-Proxy

## Envoy

Using Casdoor in Envoy

# Nginx

Enable OpenID Connect-based single sign-on for applications proxied by NGINX Plus using Casdoor as the identity provider (IdP).

This guide explains how to enable single sign-on (SSO) for applications that are being proxied by NGINX Plus. The solution uses OpenID Connect as the authentication mechanism, with [Casdoor](#) as the identity provider (IdP), and NGINX Plus as the relying party.

See Also: You can find more information about the NGINX Plus OpenID Connect integration in the project's GitHub repository.

## Prerequisites

The instructions assume that you have the following:

- A running Casdoor server. Refer to the Casdoor documentation for [Server Installation](#) and [Try with Docker](#).
- An NGINX Plus subscription and NGINX Plus R15 or later. For installation instructions, see the [NGINX Plus Admin Guide](#).
- The [NGINX JavaScript module](#), which is required for handling the interaction between NGINX Plus and the IdP. After installing NGINX Plus, install the module using the appropriate command for your operating system.

For Debian and Ubuntu:

```
sudo apt install nginx-plus-module-njs
```

For CentOS, RHEL, and Oracle Linux:

```
sudo yum install nginx-plus-module-njs
```

- The following directive should be included in the top-level (“main”) configuration context in `/etc/nginx/nginx.conf` in order to load the NGINX JavaScript module:

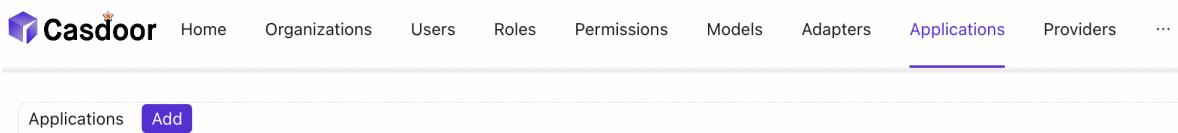
```
load_module modules/ngx_http_js_module.so;
```

## Configuring Casdoor

**Note:** The following procedure reflects the Casdoor GUI at the time of publication, but the GUI is subject to change. Use this guide as a reference and adapt to the current Casdoor GUI as necessary.

To create a Casdoor client for NGINX Plus in the Casdoor GUI, follow these steps:

1. Log in to your Casdoor account at <http://your-casdoor-url.com/login/>.
2. In the top navigation column, click Application. On the Application page that opens, click the Add button in the upper left corner.



3. On the Edit Application page that opens, change the value in the Name and Display name fields to the name of the application for which you’re enabling

SSO. Here, we're using NGINX Plus.

Name  :

NGINX Plus

Display name

NGINX Plus

 :

In the Redirect URLs field, type the URL of the NGINX Plus instance including the port number, and ending in `/_codexch` (in this guide it is [https://your-site-url.com:443/\\_codexch](https://your-site-url.com:443/_codexch)).

Redirect URLs

 :

Redirect URLs

Add

Redirect URL

 [https://my-nginx.example.com:443/\\_codexch](https://my-nginx.example.com:443/_codexch)

Notes:

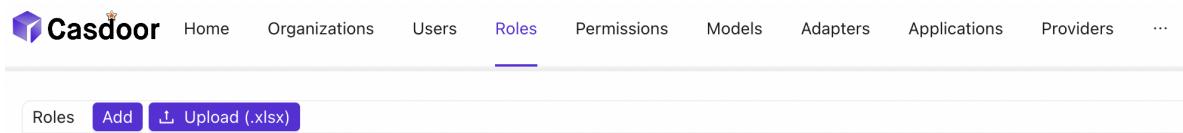
- For production, we strongly recommend that you use SSL/TLS (port 443).
- The port number is mandatory even when you're using the default port for HTTP (80) or HTTPS (443).

4. Record the values in the Client ID and Client Secret fields. You will copy them into the NGINX Plus configuration file in [Step 4 of Configuring NGINX Plus](#).

Client ID [?](#) : 200c96d5ce5f11111111111111111111

Client secret [?](#) : 58f13a80b877e7e7e7e7e7e7e7e7e7e7e

5. Click Roles in the top navigation column, then click the Add button in the upper left corner of the page that opens.



6. On the Add page that opens, type a value in the Name and Display Name fields (here it is nginx-casdoor-role) and click the Save button.

Name [?](#) : nginx-casdoor-role

Display name [?](#) : nginx-casdoor-role

7. In the top navigation column, click Users. On the Users page that opens, either click Edit to edit one of the existing users or click the Add button in the upper left corner to create a new user.
8. On the Add page that opens, modify the Name and Display Name as you like (here it is user1).

Name  : user1

Display name user1

 :

Select NGINX Plus in the Signup application.

Signup application  : NGINX Plus

In the Managed accounts field, select NGINX Plus in Application and fill in the username and password.

Managed accounts  :	Managed accounts	Add
Application	Username	Password
NGINX Plus	<input type="text"/>	<input type="password"/>

9. Go back to the Roles page and click Edit on the nginx-casdoor-role row. In the opened page, in the Sub users field, select the username you just created (here it is built-in/user1).

Sub users  : built-in/user1 

# Configuring NGINX Plus

To configure NGINX Plus as the OpenID Connect relying party, follow these steps:

1. Start by creating a clone of the [nginx-openid-connect](#) GitHub repository:

```
git clone https://github.com/nginxinc/nginx-openid-connect
```

2. Copy the following files from the clone to the `/etc/nginx/conf.d` directory:

- `frontend.conf`
- `openid_connect.js`
- `openid_connect.server_conf`
- `openid_connect_configuration.conf`

3. Retrieve the URLs for the authorization endpoint, token endpoint, and JSON Web Key (JWK) file from the Casdoor configuration. Open a terminal and execute the following `curl` command, piping the output to the indicated `python` command to generate a readable configuration format. For brevity, we have truncated the output to display only the relevant fields.

```
curl http://<casdoor-server-address>/.well-known/openid-configuration | python -m json.tool  
...
```

4. Open `/etc/nginx/conf.d/openid_connect_configuration.conf` using your preferred text editor. Modify the "default" parameter value for each of the following `map` directives with the specified values:
  - `map $host $oidc_authz_endpoint` – Use the value of the `authorization_endpoint` from [Step 3](#) (in this guide, `https://<casdoor-server-address>/login/oauth/authorize`)
  - `map $host $oidc_token_endpoint` – Use the value of the `token_endpoint` from [Step 3](#) (in this guide, `http://<casdoor-server-address>/api/login/oauth/access_token`)
  - `map $host $oidc_client` – Use the value in the Client ID field from [Step 4 of Configuring Casdoor](#)
  - `map $host $oidc_client_secret` – Use the value in the Client Secret field from [Step 2 of Configuring Casdoor](#)
  - `map $host $oidc_hmac_key` – Use a unique, long, and secure phrase
5. Configure the JWK file based on the version of NGINX Plus being used:
  - In NGINX Plus R17 and later, NGINX Plus can directly read the JWK file from the URL specified as `jwks_uri` in [Step 3](#). Make the following changes to `/etc/nginx/conf.d/frontend.conf`:
    - a. Comment out (or remove) the `auth_jwt_key_file` directive.
    - b. Uncomment the `auth_jwt_key_request` directive. (The parameter `_jwks_uri` refers to the value of the `$oidc_jwt_keyfile` variable, which will be set in the next step.)
    - c. Update the "default" parameter of the `map $host $oidc_jwt_keyfile` directive to the value obtained from the `jwks_uri` field in [Step 3](#) (in this guide, `http://<casdoor-server-address>/.well-known/jwks`).
  - In NGINX Plus R16 and earlier, or if preferred, the JWK file must be located

on the local disk. Follow these steps:

- a. Copy the JSON contents from the JWK file specified in the `jwks_uri` field in [Step 3](#) (in this guide, `http://<casdoor-server-address>/.well-known/jwks`) to a local file (e.g., `/etc/nginx/my_casdoor_jwk.json`).
  - b. In `/etc/nginx/conf.d/openid_connect_configuration.conf`, change the "default" parameter of the `map $host $oidc_jwt_keyfile` directive to the path of the local file.
6. Ensure that the user specified in the `user` directive within the NGINX Plus configuration file (usually `/etc/nginx/nginx.conf`) has read permissions for the JWK file.

## Testing

Open a browser and enter the address of your NGINX Plus instance. Then, attempt to log in using the credentials of a user who has been assigned the NGINX Plus role.



# Casdoor



username, Email or phone



Password



Auto sign in

[Forgot password?](#)

Sign In

No account? [sign up now](#)

## Troubleshooting

Please refer to the [Troubleshooting](#) section in the nginx-openid-connect repository on GitHub.

# NginxCommunityVersion

## Prerequisites

This guide assumes that you have the following conditions:

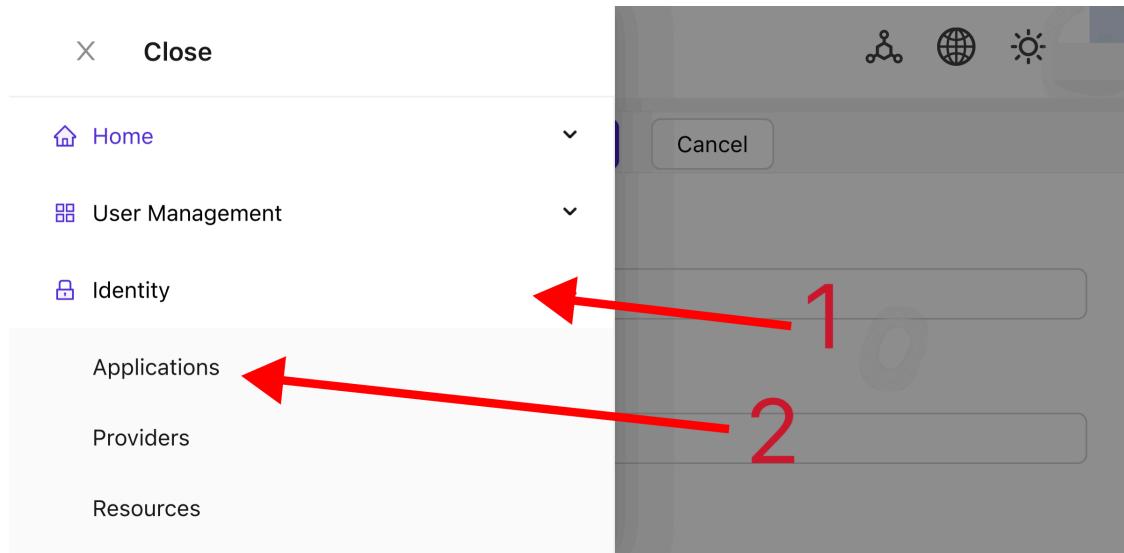
- Running Casdoor service. If you haven't installed Casdoor service yet, please refer to [Server Installation](#) or [Try with Docker](#).
- Nginx open-source edition with `ngx_http_auth_request_module` module enabled at compile time. If you don't know how to enable the `ngx_http_auth_request_module` module, please refer to the [Nginx Module Document](#).
- The website on which you want to enable authentication is successfully deployed on Nginx, with a configured domain name (instead of using an IP address), and can be accessed normally.
- OAuth2-Proxy tool (currently, the following two popular projects with high stars are available on GitHub, and you need to choose one of them):
  1. oauth2-proxy/oauth2-proxy (used in this article) [GitHub](#) OR [Official-Website](#)
  2. vouch/vouch-proxy [GitHub](#)

## I. Configure CasDoor

**Note:** The operations in this article are based on the Casdoor GUI at the time of publication, but the Casdoor GUI may change depending on the version. Please follow the references provided in this article to configure your deployed Casdoor version.

**Note:** The keys, passwords, usernames, and other confidential information mentioned in this article are all examples. For security reasons, you must replace them with your own relevant content when deploying.

1. Log in to your Casdoor admin account.
2. In the top bar, select "Identity Authentication" > "Applications", and then click "Add" on the "Applications" page.



3. Complete the application configuration based on your project information. In this article, we use "Nginx-Community" as the example application name.

☰ Menu

New Application

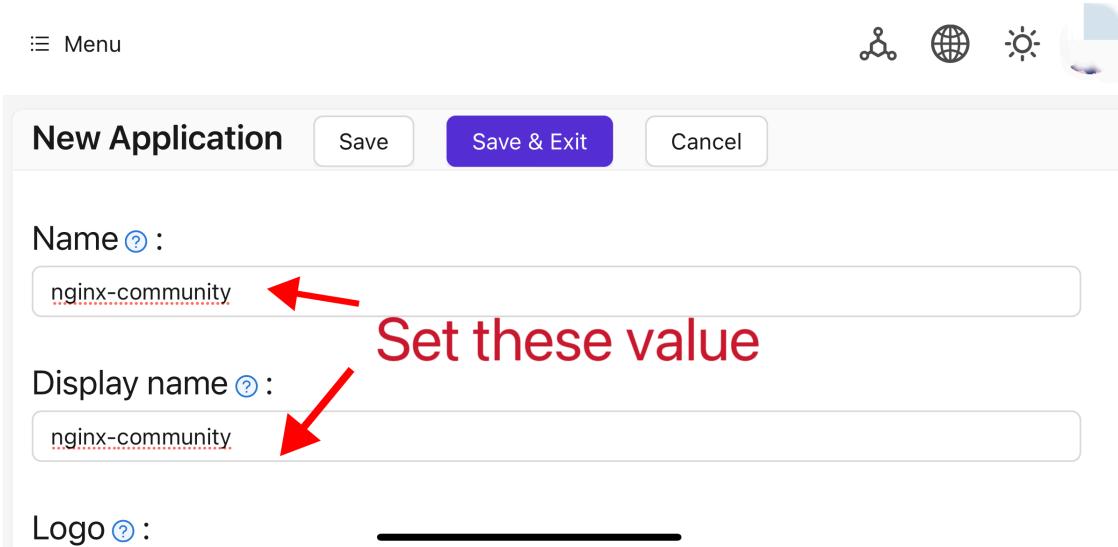
Save Save & Exit Cancel

Name ② : nginx-community

Display name ② : nginx-community

Logo ② :

**Set these value**



4. Take note of the values of the "Client ID" and "Client Secret" fields. They will be used when configuring OAuth2-Proxy later. Then configure the "Redirect URL" as `https://project.yourdomain.com/oauth2/callback/`.

Client ID ② : 811a0b0

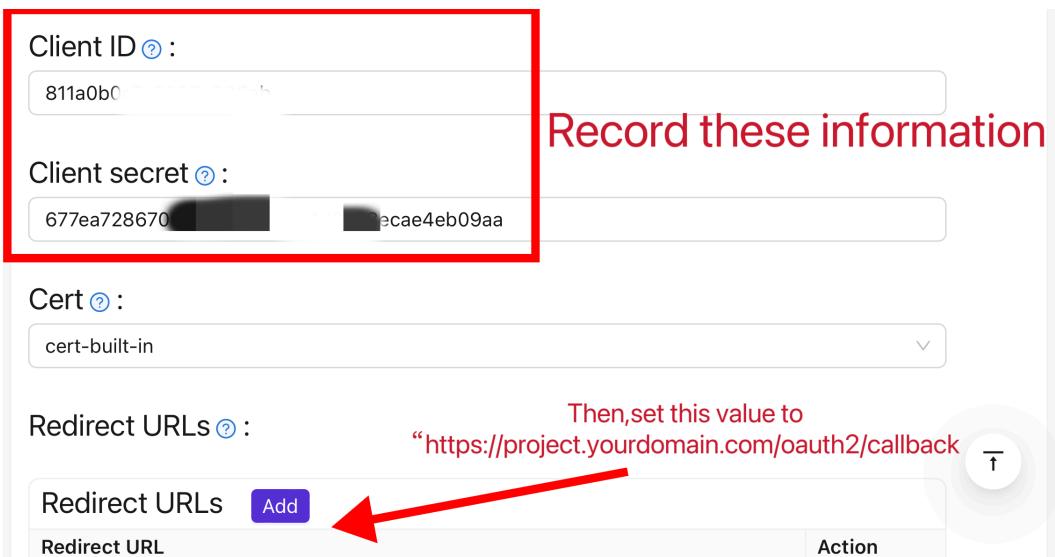
Client secret ② : 677ea728670 [REDACTED] ecae4eb09aa

Cert ② : cert-built-in

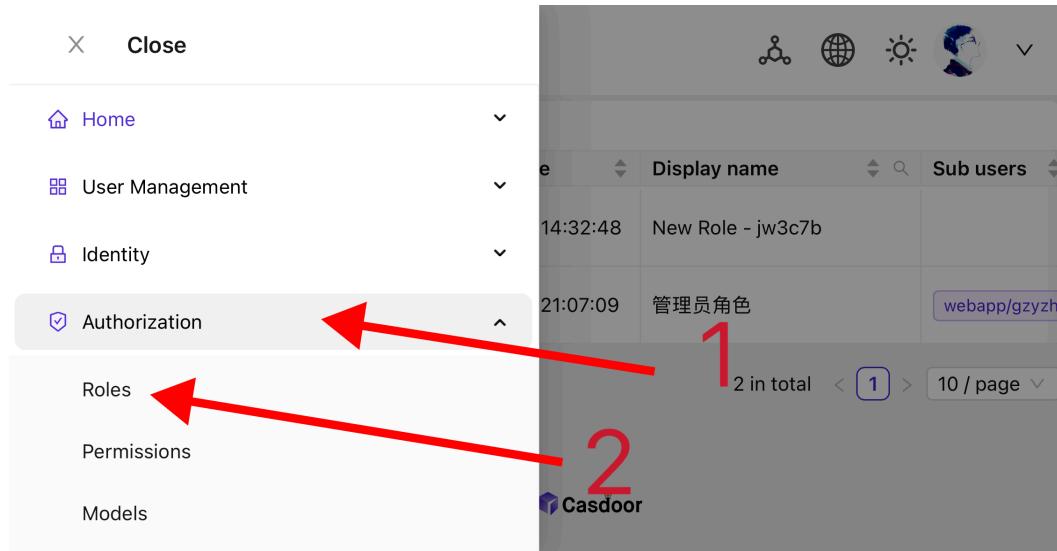
Redirect URLs ② : Then, set this value to  
“`https://project.yourdomain.com/oauth2/callback`”

Redirect URLs Add Redirect URL Action

**Record these information**



5. In the top bar, select "Casbin Permission Management" > "Roles", and then click "Add" on the "Roles" page.



6. Complete the role configuration based on your project information. In this article, we use "nginx\_role" as the example role name.

**New Role** Save Save & Exit Cancel

Organization ? : webapp

Name ? : **nginx\_community**

Display name ? : **nginx\_community**

Set these value  
Then, press “save&exit”

7. (Optional) In the top bar, select "User Management" > "Users", and then add new users based on your needs. If the users you need already exist, you can skip this step. In this article, we create an example user named "user".
8. Go back to the "Roles" page mentioned in step 5, edit the `nginx_role` role, and add the users you need to the "Included Users" option. In this article, we

add the previously created `builtin/user` here.

## II. Configure Oauth2-Proxy

Note: This article uses the Oauth2-Proxy project as an example. If you want to use Vouch instead of Oauth2-Proxy, please refer to their official documentation on [GitHub](#).

Note: This article assumes that your site is configured with a trusted SSL certificate and only allows HTTPS access, or that you have set up automatic redirection from HTTP visitors to HTTPS. This helps maximize the protection of cookies and prevents malicious reading of login tokens. If your site needs to be accessed via the insecure HTTP protocol, please modify the relevant commands accordingly. For more help with deploying via HTTP, please refer to the official documentation of Oauth2-Proxy on [GitHub](#).

Tips: [OAuth2-Proxy](#) provides various deployment methods (such as source code compilation, Docker installation, etc.). For ease of explanation, this article uses the "pre-built binary" for deployment.

1. Go to the [GitHub Releases](#) page and download the binary package corresponding to your operating system and CPU architecture. As of January 1, 2024, the latest release version of OAuth-Proxy is `v7.5.1`. If you want to download the binary package for this version, you can execute the following command for Linux with AMD64:

```
 wget -O oauth2-proxy-linux.tar.gz https://github.com/
 oauth2-proxy/oauth2-proxy/releases/download/v7.5.1/
 oauth2-proxy-v7.5.1.linux-amd64.tar.gz
```

It is strongly recommended that you check the `SHA256SUM` value provided by

the official website on the [GitHub Releases](#) page after downloading the compressed package and compare it with the `SHA256SUM` value of the package you downloaded, character by character.

2. Extract the downloaded package:

```
tar -zxvf oauth2-proxy-*.tar.gz
```

3. Enter the extracted directory:

```
cd oauth2-proxy-v7.5.1.linux-amd64
```

4. Move the obtained binary file to `/usr/local/bin` and configure it with executable permissions. You may need to elevate permissions using `sudo` depending on your situation.

```
cp ./oauth2-proxy /usr/local/bin  
cd /usr/local/bin  
chmod +x ./oauth2-proxy
```

5. Test the binary installation. If the installation is successful, after executing the following command, you should see output similar to `oauth2-proxy v7.5.1 (built with go1.21.1)`.

```
cd ~  
oauth2-proxy --version
```

6. Run `oauth2-proxy` with command-line parameters. Parameters marked with [required] must be configured according to your specific situation, while

parameters marked with [optional] can optimize performance but can also be omitted. To ensure that oauth2-proxy can run in the background, you can use process monitoring tools like `Screen` or `Supervisor` or terminal tools.

```
oauth2-proxy \
--provider=oidc \ #[required] Do not change
--client-id=abc123456def \ #[required] "Client ID" obtained in
step I.4 above
--client-secret=abc123456def \ #[required] "Client Secret"
obtained in step I.4 above
--oidc-issuer-url=https://auth.yourdomain.com \ #[required]
Your Casdoor URL (domain name or public IP)
--redirect-url=https://project.yourdomain.com/oauth2/callback
\ #[required] https://domain-of-the-project-to-protect/oauth2/
callback
--scope=email+profile+groups+openid \ #[required] Obtained
from Casdoor: user email, user profile, groups, and login
authentication
--cookie-domain=project.yourdomain.com \ #[required] Domain
name of the project you want to protect
--whitelist-domain=project.yourdomain.com \ #[required] Domain
name of the project you want to protect
--cookie-secret=abc123456def \ #[required] Please generate a
random string of numbers and letters and fill it in here
--email-domain=* \ #[required] List of acceptable user email
domains (* means accept all domains). If the user's email
suffix is not in this list, a 403 error will be returned even
if the login is successful.
--insecure-oidc-allow-unverified-email=true \ #[required]
Whether to accept users with unverified email addresses
--http-address=http://127.0.0.1:65534 \ #[required] Address
that oauth2-proxy listens on. The port number here can be set
arbitrarily. Please record the value you set, as it will be
needed for configuring Nginx later.
--cookie-expire=24h0m0s \ #[optional] Cookie expiration time.
After this period, users will need to log in again.
```

# III. Configure Nginx

**Note:** Please confirm again that your Nginx has enabled the `ngx_http_auth_request_module` module when compiling and installing from source code (the compilation command includes `--with_http_auth_request_module`). If you don't know how to enable the `ngx_http_auth_request_module` module, please refer to the [Nginx Module Document](#).

**Tips:** Nginx installed using the Baota panel tool does not enable this module by default.

1. Open the configuration file of the website you have already deployed and want to protect, and make the following modifications:

**Note:** You need to adjust this configuration file according to your specific situation. Due to Nginx versions and other factors, this configuration file may not work smoothly on all Nginx instances. Please adjust the relevant content based on your own Nginx information.

```
server {
    listen 443 ssl http2;

    include /path/to/ssl.conf;

    # Add the following content
    location ^~ /oauth2/ {
        proxy_pass      http://127.0.0.1:65534; # Change this
        to the "--http-address" configured in step II.6

        proxy_set_header Host                      $host;
        proxy_set_header X-Real-IP                 $remote_addr;
```

2. Save the file and reload your Nginx.

## Testing

- Next, you can test your implementation.
- In normal circumstances, your users will go through the following process when logging in to your service:
  - Open the URL `project.yourdomain.com` in a browser → Only see a page requiring login, including a button named "Sign in with OpenID Connect" → Click the button and be redirected to your Casdoor address, where they will be asked to log in → Users enter their username and password, and Casdoor verifies their credentials → Automatically redirect back to your URL `project.yourdomain.com` → Successfully access your service → Users will be asked to log in again when the `--cookie-expire` time you set expires.

## Troubleshooting

- If your project is not running as expected, please check your Nginx configuration and Oauth2-Proxy configuration parameters for correctness.
- You can also refer to the official documentation of Oauth2-Proxy on [GitHub](#).
- If you find any errors in this document, please feel free to request edits on GitHub.



# Envoy

## Prerequisites

A running Casdoor server. Please refer to the Casdoor documentation for [Server Installation](#) and [Try with Docker](#).

## Configuring Casdoor

1. Add the Envoy application. In the Redirect URLs field, enter the URL of the Envoy instance including the port number, and ending with /oauth2/callback (e.g., http://%REQ(:authority)%/oauth2/callback). Make a note of the values in the Client ID and Client Secret.
2. Add the `envoy-casdoor-role` role.
3. Add the `user1` user. Select Envoy in the Signup application. In the Managed accounts field, select Envoy in the Application dropdown and fill in the username and password. Go back to the Roles page and click "Edit" on the `envoy-casdoor-role` row. In the opened page, in the Sub users field, select the username you just created (in this case, it is built-in/user1).

## Configure Envoy

1. Modify the `token_endpoint`, `authorization_endpoint`, and `client_id` in the `envoy.yaml` file.
2. Modify the `inline_string` in the `token-secret.yaml` file to the Client Secret of Envoy from Casdoor.

3. Modify the `inline_bytes` in the `hmac-secret.yaml` file with a unique, long, and secure phrase.
4. Add the `envoy.yaml`, `token-secret.yaml`, and `hmac-secret.yaml` files to your Envoy path.

## How to Run

1. Start Envoy using the `envoy.yaml` file.
2. Go to the website where Envoy is listening. You should immediately be redirected to Casdoor for user authentication.

# C#

## Unity

Use the Casdoor-dotnet-sdk for Unity development.

# Unity

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed.

You can refer to the Casdoor official documentation for the [Server Installation](#). Please deploy your Casdoor instance in production mode.

After a successful deployment, ensure that:

- Open your favorite browser and visit <http://localhost:8000>, you will see the login page of Casdoor.
- Input `admin` and `123` to test the login functionality.

Alternatively, you can use the [official Casdoor demo station](#) for a quick start.

## Step 2: Import Casdoor.Client

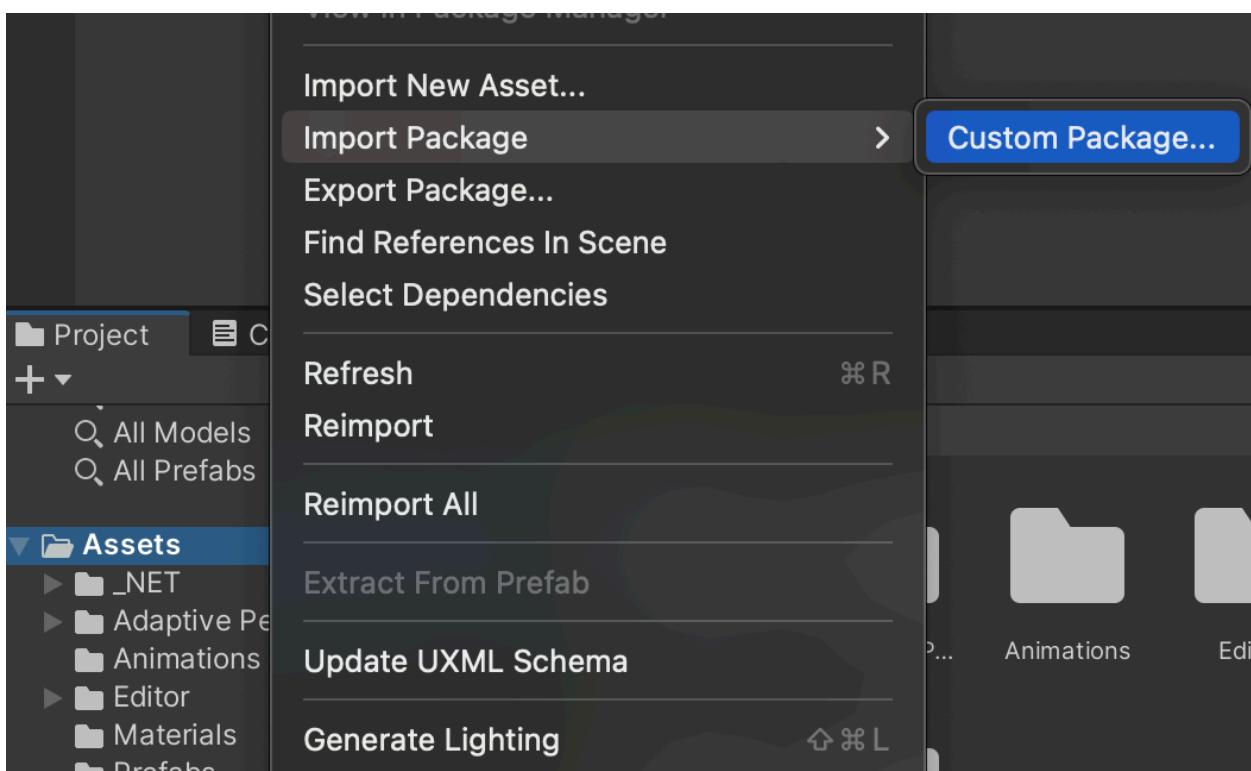
Import [Casdoor.Client](#) for `.NET` in the [Casdoor-dotnet-sdk](#).

One optional method is as follows:

- `git@github.com:casdoor/casdoor-dotnet-sdk.git`
- Run `ConsoleApp` in the `Sample` folder.
- Get the `/casdoor-dotnet-sdk/src/Casdoor.Client/bin/Debug/net462` folder.

Now, you can import the `net462` folder into your Unity project through the

method shown in the figure below. Of course, you can also choose folders of other versions.

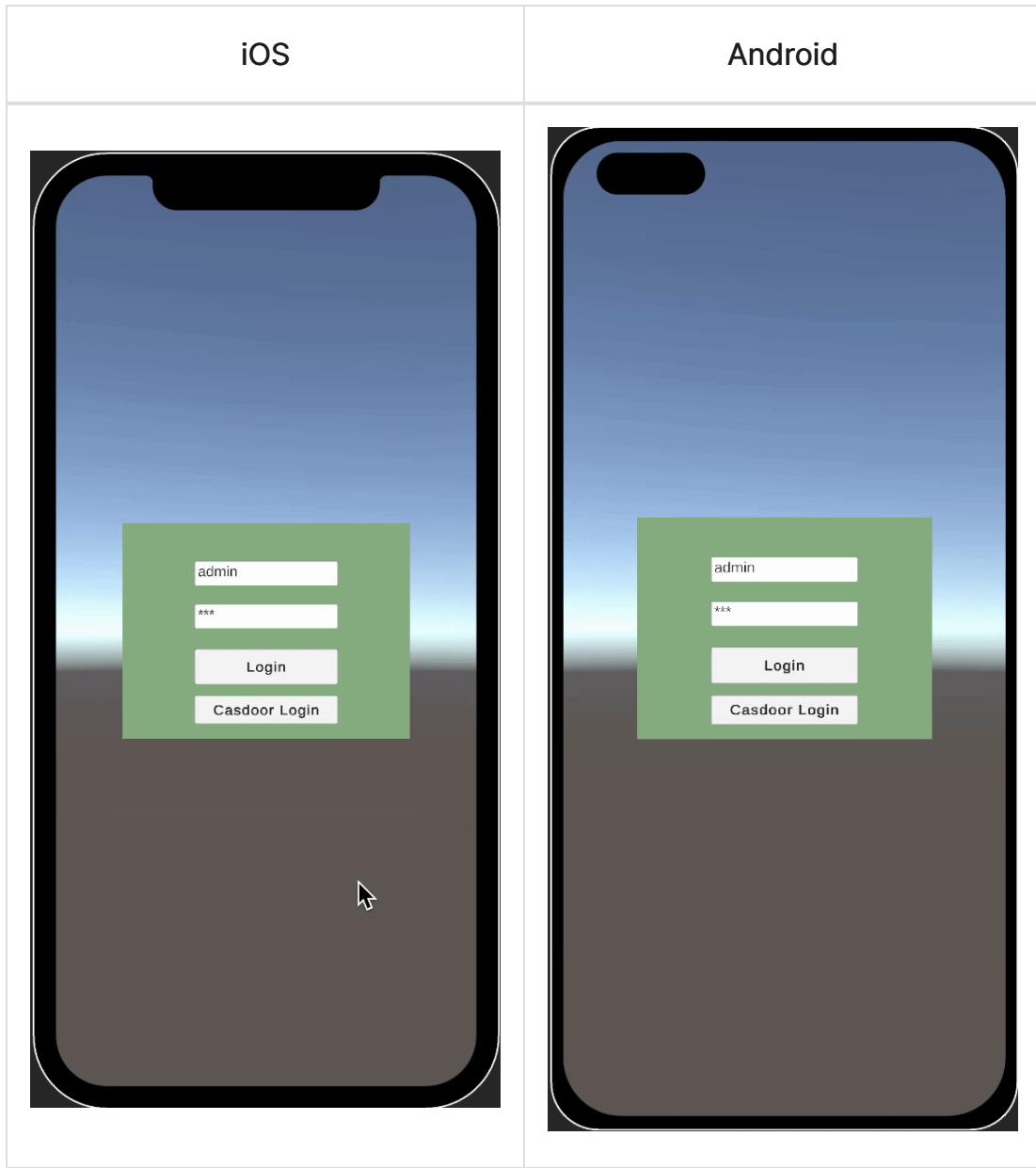


## Step 3: Usage

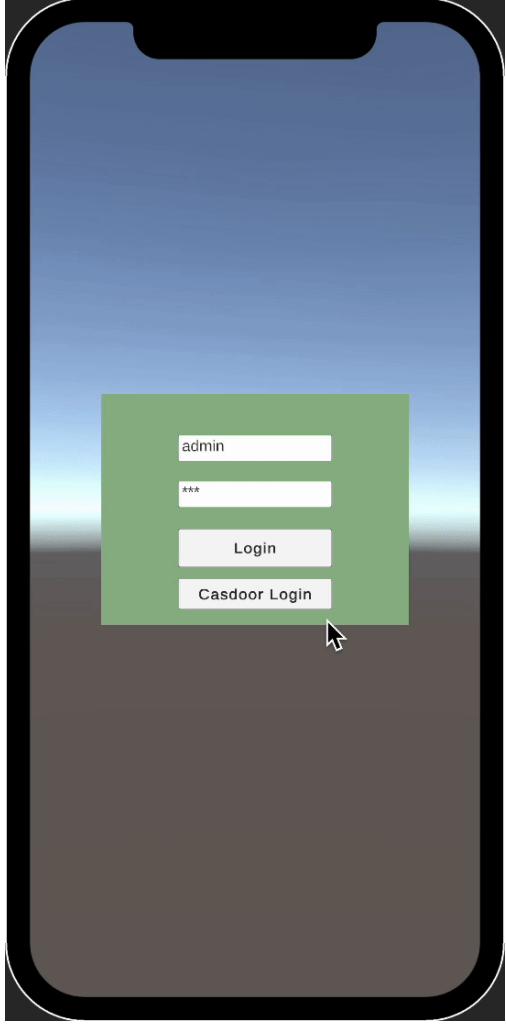
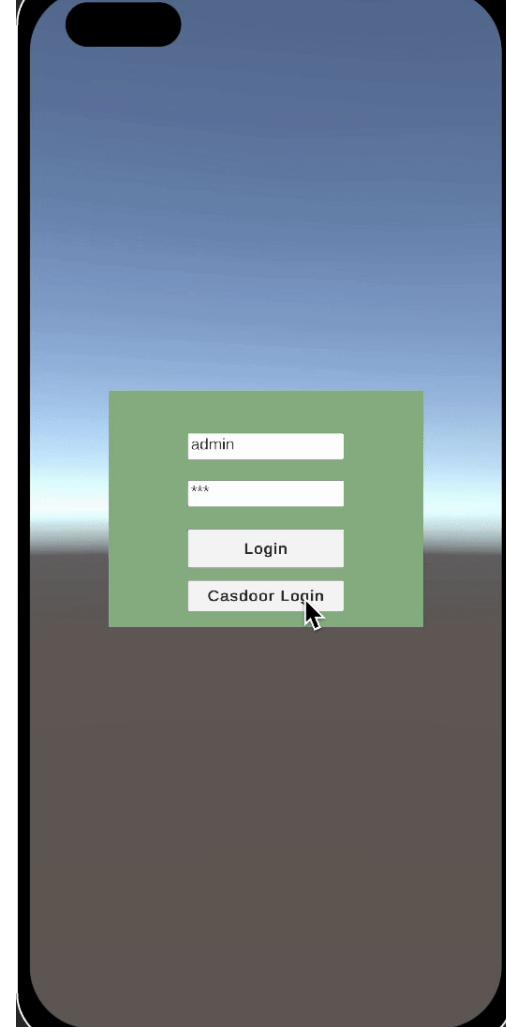
Learn how to use the `Casdoor.Client` SDK for Unity 3D mobile development by looking at [casdoor-unity-example](#).

After running the `casdoor-unity-example`, you will see the following interfaces:

- Login with username and password:



- Login with the Casdoor web page:

iOS	Android
 A screenshot of an iPhone X displaying a login interface. The interface consists of a green rectangular card with rounded corners. It contains four white rectangular input fields: the first labeled "admin", the second with three asterisks (***) for a password, the third labeled "Login", and the fourth labeled "Casdoor Login". A cursor arrow is positioned directly above the "Casdoor Login" button, indicating it is about to be tapped.	 A screenshot of an Android smartphone displaying a similar login interface. The green card has the same four input fields. The "Casdoor Login" button is highlighted with a white mouse cursor arrow pointing directly at its center.

# Go

## Kubernetes

Using Casdoor for Authentication in Kubernetes

## OpenShift

Using Casdoor for authentication in OpenShift

## BookStack

在 BookStack 中使用 Casdoor 进行身份验证

## Bytebase

Using OAuth2 to connect various applications, like Bytebase

 **ELK**

Casdoor/elk-auth-casdoor概览

 **Gitea**

在 Gitea 中使用 Casdoor 进行身份验证

 **Grafana**

在 Grafana 中使用 Casdoor 进行身份验证

 **MinIO**

Configuring Casdoor as an identity provider to support MinIO

 **Portainer**

Using Casdoor for authentication in Portainer

# Kubernetes

According to the [Kubernetes documentation](#), the API Server of Kubernetes can be authenticated using OpenID Connect (OIDC). This article will guide you on how to configure authentication in Kubernetes using Casdoor.

## Environment Requirements

Before starting, please make sure that you have the following environment:

- A Kubernetes cluster.
- A Casdoor application like this [demo website](#).
- `kubectl` command tool (optional).

ⓘ 备注

Kubernetes `oidc-issuer-url` only accepts URLs which use the `https://` prefix. So your Casdoor application should be deployed on an HTTPS website.

## Step 1: Creating a Casdoor App and User Account for Authentication

Go to your Casdoor application and add a new application called **Kubernetes**. Please remember the `Name`, `Organization`, `client ID`, `client Secret`, and add some grant types to this app.

Name [?](#):

Display name [?](#):

Logo [?](#):   
URL [?](#):

Preview: 

Home [?](#):

Description [?](#):

Organization [?](#):

Client ID [?](#):

Client secret [?](#):

Cert [?](#):

Grant types [?](#):

Next, add a new user to the application that you just created. Please note that the **Organization** and **Signup application** used here should correspond to the app registered earlier.

Organization <a href="#">?</a> :	casbin
ID <a href="#">?</a> :	202e02e9-9128-496a-a209-fdb336448f56
Name <a href="#">?</a> :	user_pnvm5i
Display name <a href="#">?</a> :	New User - pnvm5i
Avatar <a href="#">?</a> :	<p>Preview:</p>  <p>Upload a photo...</p>
User type <a href="#">?</a> :	normal-user
Password <a href="#">?</a> :	<a href="#">Modify password...</a>
Email <a href="#">?</a> :	pnvm5i@example.com
Phone <a href="#">?</a> :	+1 <span style="margin-left: 10px;">▼</span> 78005961394
Country/Region <a href="#">?</a> :	Please select country/region
Location <a href="#">?</a> :	
Affiliation <a href="#">?</a> :	Example Inc.
Title <a href="#">?</a> :	
Homepage <a href="#">?</a> :	
Bio <a href="#">?</a> :	
Tag <a href="#">?</a> :	staff
Signup application <a href="#">?</a> :	Kubernetes

# Step 2: Configure Kubernetes API Server with OIDC Authentication

To enable the OIDC plugin, you need to configure the following flags on the API server:

- `--oidc-issuer-url`: URL of the provider that allows the API server to discover public signing keys.
- `--oidc-client-id`: A client id that all tokens must be issued for.

This article uses minikube for demonstration. You can configure the OIDC plugin for the minikube's API server using the following command at startup:

```
minikube start --extra-config=apiserver.oidc-issuer-
url=https://demo.casdoor.com --extra-config=apiserver.oidc-client-
id=294b09fbc17f95daf2fe
```

# Step 3: Test OIDC Authentication

## Obtain Authentication Information

Due to the lack of a frontend in kubectl, authentication can be performed by sending a POST request to the Casdoor server. Here is the code in Python which sends a POST request to the Casdoor server and retrieves the `id_token` and `refresh_token`:

```
import requests
```

After executing this code, you should receive a response similar to the following:

```
{  
  "access_token": "xxx",  
  "id_token": "yyy",  
  "refresh_token": "zzz",  
  "token_type": "Bearer",  
  "expires_in": 72000,  
  "scope": ""  
}
```

Now, you can use the `id_token` that you just obtained to authenticate with the Kubernetes API server.

## HTTP Request-Based Authentication

Add the token to the request header.

```
curl https://www.xxx.com -k -H "Authorization: Bearer $(id_token)"
```

- `https://www.xxx.com` is the Kubernetes API server deployment address.

## Kubectl Client-Based Authentication

### Configuration File Method

Write the following configuration to the `~/.kube/config` file. You should replace each configuration item in the configuration file above with the values you obtained earlier.

```
users:
```

Now, you can directly access your API server using kubectl. Try running a test command.

```
kubectl cluster-info
```

### Command Line Argument Method

Alternatively, you can authenticate by directly adding the `id_token` to the command line parameters of kubectl.

```
kubectl --token=$(id_token) cluster-info
```

# OpenShift

OpenShift supports OIDC, so we can integrate Casdoor with OpenShift. The following steps demonstrate how to integrate Casdoor with OpenShift Local using the [online demo of Casdoor](#).

## Step 1: Create an Casdoor application

Add a new application in Casdoor, noting the following points:

- Remember the `Client ID` and `Client secret` for the next step.
- The format of the Redirect URL is `https://oauth-openshift.apps.<cluster_name>.<cluster_domain>/*`. Fill it in depending on your situation.

Name [?](#) :

Display name [?](#) :

Logo [?](#) :   
Preview: 

Home [?](#) :

Description [?](#) :

Organization [?](#) :

Client ID [?](#) :

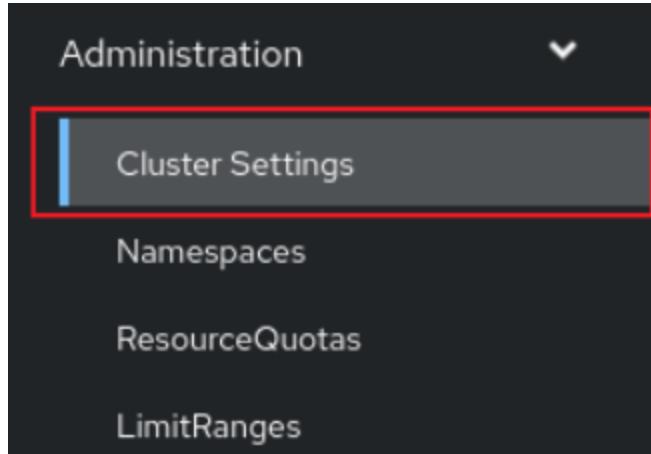
Client secret [?](#) :

Cert [?](#) :

Redirect URLs [?](#) :   
Redirect URLs [Add](#)

## Step 2: OpenShift OAuth Configuration

Now log into the OpenShift Console as Kubeadmin. Once you are logged in, browse to the side menu and locate the Cluster settings.



Under Global Configuration, you will see OAuth.

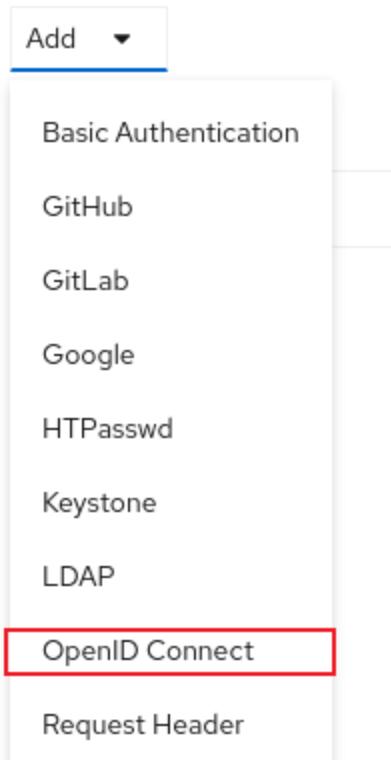
OAuth

OAuth holds cluster-wide information about OAuth. The canonical name is 'cluster'. It is used to configure the integrated OAuth server. This configuration is only honored when the top level Authentication config has type set to IntegratedOAuth. Compatibility level I: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

You will see the Identity Provider section. In the ADD section, select OpenID Connect from the options.

## Identity providers

Identity providers determine ho



Configure OIDC, noting the following points:

- Fill in the `Client ID` and `Client Secret` remembered from the previous step.
- The Issuer URL must use https, in the form `https://<casdoor-host>`, again depending on your situation.

## Add Identity Provider: OpenID Connect

Integrate with an OpenID Connect identity provider using an Authorization Code Flow.

Name \*

casdoor

Unique name of the new identity provider. This cannot be changed later.

Client ID \*

2452f2b5abb6ff131199

Client secret \*

\*\*\*\*\*

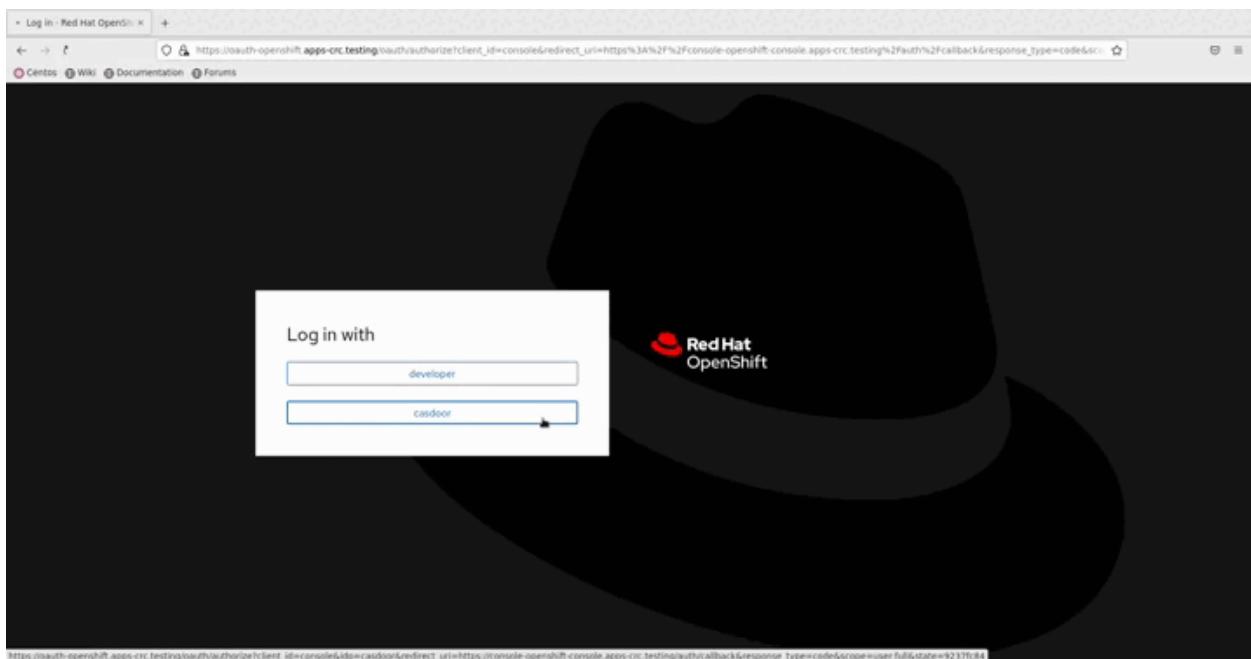
Issuer URL \*

<https://demo.casdoor.com/>

The URL that the OpenID provider asserts as its issuer identifier. It must use the https scheme with no URL query parameters or fragment.

## Step 3: Test OIDC Authentication

Access the OpenShift console in the browser. You will see Casdoor (the name you configured in the previous step). Click on the Casdoor login option. You will be redirected to the Casdoor login page.



# BookStack

## 在 BookStack 中使用 Casdoor 进行身份验证

BookStack is an open-source book and document sharing site, as well as an open-source application developed using the Go language to help you better manage document reading.

BookStack-casdoor has been integrated with Casdoor, and you can now quickly get started with a simple configuration.

### Step 1: Create a Casdoor application

Go to your Casdoor and add a new application called BookStack. Here is an example of creating the BookStack application in Casdoor.

Edit Application Save Save & Exit

Name ? : bookstack

Display name ? : bookstack

Logo ? : URL ? : [https://cdn.casdoor.com/logo/casdoor-logo\\_1185x256.png](https://cdn.casdoor.com/logo/casdoor-logo_1185x256.png)

Preview: 

Home ? :

Description ? :

Organization ? :

Client ID ? :

Client secret ? :

请记住 名称, 组织, 客户端 ID, 和 客户密钥。 You will need them in the next step.

## Step 2: Configure Casdoor Login

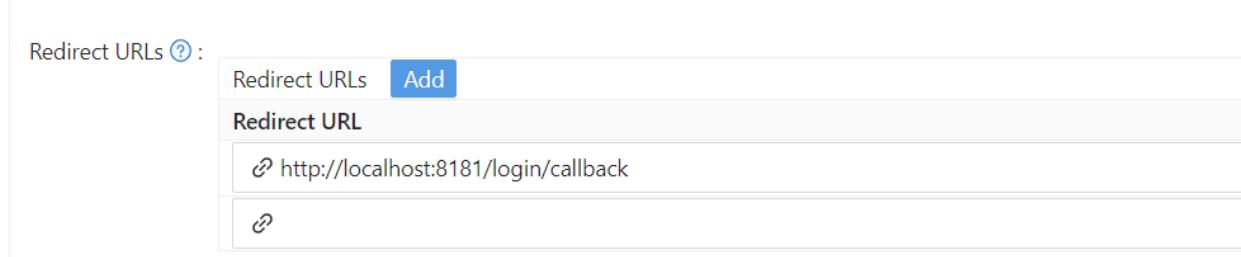
Next, navigate to BookStack and find the file `oauth.conf.example`.

将 `oauth.conf.example` 重命名为 `oauth.conf` 和 **修改配置** By default, the content is as follows:

```
[oauth]
casdoorOrganization = "<Organization>"
casdoorApplication = "bookstack"
casdoorEndpoint = http://localhost:8000
clientId = <client ID>
clientSecret = <client Secret>
redirectUrl = http://localhost:8181/login/callback
```

## Step 3: Fill in the `redirectUrl` in Casdoor

In the final step, go back to the page where you added the BookStack application and fill in the `Redirect URLs`. Make sure the `Redirect URL` is the same as the `redirectUrl` in the `oauth.conf` file.



Now that you have completed the Casdoor configuration!

You can now go back to your BookStack and experience using Casdoor for login authentication once you have successfully deployed BookStack.

# Bytebase

Casdoor can use OAuth2 to connect various applications. In this example, we will use [Bytebase](#) to demonstrate how to use OAuth2 to connect to your applications.

The following are the configuration names:

`CASDOOR_HOSTNAME`: The domain name or IP address where the Casdoor server is deployed.

`Bytebase_HOSTNAME`: The domain name or IP address where Bytebase is deployed.

## Step 1: Deploy Casdoor and Bytebase

Firstly, deploy [Casdoor](#) and [Bytebase](#).

After successful deployment, make sure that:

1. Casdoor can be logged in and used normally.
2. You can set `CASDOOR_HOSTNAME` to `http://localhost:8000` when deploying Casdoor in `prod` mode. See [production mode](#).

## Step 2: Configure Casdoor application

1. Create a new or use an existing Casdoor application.
2. Find the redirect URL: `<CASDOOR_HOSTNAME>/oauth/callback`.
3. Add the redirect URL to the Casdoor application:

The screenshot shows the Casdoor application settings page. It includes fields for Client ID (e828fd6922f4292b979e), Client secret (bab9f6c2fad67471e1bd81e074ea192e4f46dd), Cert (cert-built-in), and Redirect URLs (Redirect URLs: <CASDOOR\_HOSTNAME>/oauth/callback). A red box highlights the Client ID and Client secret fields.

On the application settings page, you will find two values: `Client ID` and `Client secret`. We will use these values in the next step.

Open your favorite browser and visit: `http://<CASDOOR_HOSTNAME>.well-known/openid-configuration`. You will see the OIDC configuration of Casdoor.

## Step 3: Configure Bytebase

- Find SSO and select OAuth 2.0:

The screenshot shows the Bytebase SSO configuration page. On the left sidebar, 'SSO' is highlighted with a red arrow. The main form has 'Type' set to 'OAuth 2.0' (radio button selected) and 'Custom' (radio button selected under 'Use template'). Arrows point to both the selected radio buttons. Below, 'Basic information' shows 'Name' as 'Custom'.

- Configure this app:

The screenshot shows the Casdoor application interface. On the left, there's a sidebar with various navigation items such as Account, Profile, Workspace, General, Members, Projects, Subscription, Debug Log, Security & Policy, SQL Review, Risk Center, Custom Approval, Data Anonymization, Data Access Control, Audit Log, Integration, GRCs, SSO, IM, and Archive. The main content area is titled 'SSO > casdoor' and shows 'Basic Information' for the 'casdoor' provider. It includes fields for 'Name' (casdoor), 'Identity Provider ID' (idp-casdoor-mels), 'Domain' (http://101.43.192.216:8000), and 'Identity provider information' which points to 'The information is provided by your identity provider'. Below this, there are sections for 'Client ID' (e828fd992342926979e), 'Client secret' (sensitive - write only), 'Auth URL' (http://101.43.192.216:8000/login/oauth/authorize), 'Scopes' (openid profile email), 'Token URL' (http://101.43.192.216:8000/api/login/oauth/access\_token), and 'User information URL' (http://101.43.192.216:8000/api/get-account). At the bottom, there are buttons for 'Test Connection', 'Archive this SSO', 'Discard changes', and 'Update'.

### 3. Find the Client ID and Client Secret on the Casdoor application page.

- Token server URL: http://**CASDOOR\_HOSTNAME**/api/login/oauth/access\_token
- Authorization server URL: http://**CASDOOR\_HOSTNAME**/login/oauth/authorize
- User Info server URL: http://**CASDOOR\_HOSTNAME**/api/get-account
- Scopes: address phone openid profile offline\_access email

Log out of Bytebase and test SSO.

The screenshot shows the Bytebase login page. The background features a colorful cartoon illustration of a rocket launching from a planet, with two characters (one holding a flag) watching. The login form has fields for 'Email' (jim@example.com) and 'Password'. There are links for 'Forgot your password?' and 'New to Bytebase? Sign up'. Below the form is a 'Sign in with casdoor' button. The Bytebase logo is at the top right. At the bottom, there are links for 'English' and '简体中文' (Simplified Chinese), and a copyright notice: '© 2023 Bytebase. All rights reserved.'

# ELK

## Casdoor/elk-auth-casdoor概览

One of the biggest drawbacks of ELK (Elasticsearch, Logstash, and Kibana) is that originally these products had no authentication mechanism. As a result, anyone with the URL of Kibana or Elasticsearch could access the Kibana dashboard. Later on, ELK integrated an embedded authentication system called "Xpack." However, its advanced functions (such as OAuth, OIDC, LDAP, SAML) are not free. Only plain authentication, using a set of accounts and passwords, is available free of charge, which is quite inconvenient. This approach does not allow us to provide a unique account for everyone in a corporation.

To address this issue, we have developed an elk authentication solution based on Casdoor. This solution is free, open-source, under ongoing maintenance, and supports a wide range of advanced features. Casdoor is a centralized authentication/Single-Sign-On platform based on OAuth 2.0/OIDC. Casdoor/elk-auth-casdoor serves as a reverse proxy designed to intercept all HTTP data flow towards the ELK/Kibana stack. It guides users who haven't logged in to log in. This reverse proxy operates transparently as long as the user has logged in.

If a user hasn't been correctly authenticated, the request will be temporarily cached, and the user will be redirected to the Casdoor login page. Once the user logs in through Casdoor, the cached request will be restored and sent to Kibana. Therefore, if a POST request (or any other request type besides GET) is intercepted, the user won't need to refill the form and resend the request. 逆向代理将为你记住它。

The casdoor/elk-auth-casdoor repository is located at <https://github.com/>

[casdoor/elk-auth-casdoor](#).

## 如何使用？

0. Ensure that you have the Go programming language environment installed.
1. Go to [casdoor/elk-auth-casdoor](#) and fetch the code.
2. Register your proxy as an app with Casdoor.
3. Modify the configuration.

The configuration file is located at "conf/app.conf". Here is an example, which you should customize based on your specific needs.

```
appname = .  
# port on which the reverse proxy shall be run  
httpport = 8080  
runmode = dev  
# EDIT IT IF NECESSARY. The URL of this reverse proxy.  
pluginEndpoint = "http://localhost:8080"  
# EDIT IT IF NECESSARY. The URL of the Kibana.  
targetEndpoint = "http://localhost:5601"  
# EDIT IT. The URL of Casdoor.  
casdoorEndpoint = "http://localhost:8000"  
# EDIT IT. The clientID of your reverse proxy in Casdoor.  
clientID = ceb6eb261ab20174548d  
# EDIT IT. The clientSecret of your reverse proxy in Casdoor.  
clientSecret = af928f0ef1abc1b1195ca58e0e609e9001e134f4  
# EDIT IT. The application name of your reverse proxy in  
Casdoor.  
appName = ELKProxy  
# EDIT IT. The organization to which your reverse proxy  
belongs in Casdoor.  
organization = built-in
```

4. Visit <http://localhost:8080> (in the above example) and log in following the redirection guidance. You should then see Kibana protected and authenticated by Casdoor.
5. If everything works well, don't forget to block external access to the original Kibana port by configuring your firewall (or another method). This ensures that outsiders can only access Kibana via this reverse proxy.

# Gitea

## 在 Gitea 中使用 Casdoor 进行身份验证

Gitea 是一个社区管理的轻量代码托管解决方案，写入Go。采用MIT开源协议

Gitea支持第三方身份验证，包括Oauth，这样就可以使用Casdoor进行身份验证。以下是操作教程。

### 准备：

要配置 Gitea 使用 Casdoor 作为身份识别提供者，您需要安装 Gitea 以及访问管理员帐户。

关于如何下载、安装和运行 Gitea 的更多信息，请参阅 <https://docs.gitea.io/en-us/install-from-binary/>

您需要在安装过程中创建管理员帐户。如果您已经注册，管理员将是第一个注册用户。请使用此帐户继续以下操作。

### 1. 创建一个Casdoor应用程序

像这样：

Edit Application

Name ⓘ: application\_9p7eai

Display name ⓘ: New Application - 9p7eai

Logo ⓘ: URL ⓘ: https://cdn.casbin.com/logo/logo\_1024x256.png

Preview: 

Home ⓘ: [View](#)

Description ⓘ:

Organization ⓘ: built-in

Client ID ⓘ: 7ceb9b7fda4a9061ec1c

Client secret ⓘ: 3416238e1edf915eac08b8fe345b2b95cdba7e04

Cert ⓘ: cert-built-in

Redirect URLs

Action	Redirect URLs	Add
<a href="#">Edit</a> <a href="#">Delete</a>	http://localhost:3000/user/oauth2/Casdoor/callback	<a href="#">Add</a>

请记住客户端ID和密码，以便下一步操作。

请不要在此步骤中填写回调url。Url取决于下一步Gitea的配置。稍后我们将返回来设置一个正确的回调url。

## 2. 配置 Gitea 使用 Casdoor

以管理员身份登录。通过右上角的下拉菜单转到“站点管理”页面。然后切换到“认证源”页面

你应该看到类似下面的内容：

The screenshot shows the GitHub Authentication Sources management interface. At the top, there are navigation links: Issues, Pull Requests, Milestones, Explore, a notifications icon, a plus sign for creating new items, and a gear icon for settings. Below the header, a secondary navigation bar includes: Dashboard, User Accounts, Organizations, Repositories, Webhooks, Authentication Sources (which is underlined, indicating it's the active tab), User Emails, Configuration, System Notices, and Monitoring. A main content area titled "Authentication Source Management (Total: 0)" contains a table with the following columns: ID, Name, Type, Enabled, Updated, Created, and Edit. There is also a blue "Add Authentication Source" button at the top right of the table area.

按“添加认证源”按钮并填写类似的表单。

The screenshot shows the "Add Authentication Source" form on GitHub. The "Authentication Type" dropdown is set to "OAuth2". The "Authentication Name" input field contains "Casdoor". The "OAuth2 Provider" dropdown is set to "OpenID Connect". The "Client ID (Key)" input field contains "7ceb9b7fda4a9061ec1c". The "Client Secret" input field contains "3416238e1edf915eac08b8fe345b2b95cdba7e04". The "Icon URL" input field is empty. The "OpenID Connect Auto Discovery URL" input field contains "http://localhost:8000/.well-known/openid-configuration". A checkbox labeled "Skip local 2FA" is checked, with a note below stating "Leaving unset means local users with 2FA set will still have to pass 2FA to log on". The "Additional Scopes" input field is empty.

请选择认证类型为“oauth2”。

请输入此认证源的名称并 **记住此名称**。此名称将在下一步骤中用于回调url。

请选择 **OpenID Connect** Oauth2 提供商。

填写上一步中记住的**客户端ID**和**客户端密码**。

在 **openid** 连接中填写自动发现URL，它应该是 `<your endpoint of casdoor>/.well-known /openid-configuration`。

按您的意愿填写其他可选配置项。然后提交它。

提交表单

### 3. 配置后台回调url

返回第2步中的应用程序编辑页面并添加以下回调url：

`<endpoint of gitea>/user/oauth2/<authentication source name>/callback`

`<authentication source name>` 是上一步Gitea认证源的名称。

### 4. 在 Gitea 上试试

退出当前管理员帐户。

您应该在登录页面中看到这一点：

The screenshot shows a top navigation bar with 'Sign In' and 'OpenID' buttons. Below is a 'Sign In' form with fields for 'Username or Email Address' and 'Password'. There is a 'Remember this Device' checkbox, a 'Sign In' button, a 'Forgot password?' link, and a 'Need an account? Register now.' link. At the bottom is an 'OpenID Connect' sign-in button.

Sign In

OpenID

Sign In

Username or Email Address \*

Password \*

Remember this Device

**Sign In**   [Forgot password?](#)

[Need an account? Register now.](#)

Sign In With OpenID Connect

按“使用openid登录”按钮，您将被重定向到casdoor登录页面。

登录后您将看到这个：

The screenshot shows a 'Complete New Account' form within a Gitea interface. It includes fields for 'Username' and 'Email Address', both marked with a red asterisk indicating they are required. A 'Complete Account' button is at the bottom.

Explore Help

Register New Account [Link to Existing Account](#)

Complete New Account

Username \*

Email Address \*

admin@example.com

**Complete Account**

按照指示并用一个新的 Gitea 帐户或现有帐户绑定下级帐户。

然后一切都将正常工作。



# Grafana

## 在 Grafana 中使用 Casdoor 进行身份验证

[Grafana](#) supports authentication via OAuth. Therefore, it is extremely easy for users to use Casdoor to log in to Grafana. Only several steps and simple configurations are needed to achieve that.

Here is a tutorial on how to use Casdoor for authentication in Grafana. Before you proceed, please ensure that you have Grafana installed and running.

### Step 1: Create an app for Grafana in Casdoor

Here is an example of creating an app in Casdoor:

Edit Application Save Save & Exit

Name ⓘ: application\_9p7eai

Display name ⓘ: New Application - 9p7eai

Logo ⓘ: URL ⓘ: [https://cdn.casbin.com/logo/logo\\_1024x256.png](https://cdn.casbin.com/logo/logo_1024x256.png)

Preview:



casbin

Home ⓘ: [/](#)

Description ⓘ:

Organization ⓘ: built-in

Client ID ⓘ: 7ceb9b7fda4a9061ec1c

Client secret ⓘ: 3416238e1edf915eac08b8fe345b2b95cd8a7e04

Cert ⓘ: cert-built-in

Redirect URLs ⓘ:

Action	Redirect URL
<a href="#">Add</a>	<a href="#">http://localhost:3000/login/generic_oauth</a>

Please copy the client secret and client ID for the next step.

Please add the callback URL of Grafana. By default, Grafana's OAuth callback is [/login/generic\\_oauth](/login/generic_oauth). So please concatenate this URL correctly.

## Step 2: Modify the configuration of Grafana

By default, the configuration file for OAuth is located at `conf/defaults.ini` in the workdir of Grafana.

Please find the section `[auth.generic_oauth]` and modify the following fields:

```
[auth.generic_oauth]
name = Casdoor
```

## About HTTPS

If you don't want HTTPS enabled for Casdoor or if you deploy Grafana without HTTPS enabled, please also set `tls_skip_verify_insecure = true`.

## About redirectURI after Sign In With Casdoor

If the redirect URI is not correct after signing in with Casdoor in Grafana, you may want to configure [root\\_url](#).

```
[server]
http_port = 3000
# The public-facing domain name used to access Grafana from a
browser
domain = <your IP here>
# The full public-facing URL
root_url = %(protocol)s://%(domain)s:%(http_port)s/
```

Related links:

1. [Grafana documentation](#)
2. [Grafana defaults.ini](#)

## About Role Mapping

You may want to configure `role_attribute_path` to map your user's role to Grafana via [role\\_attribute\\_path](#).

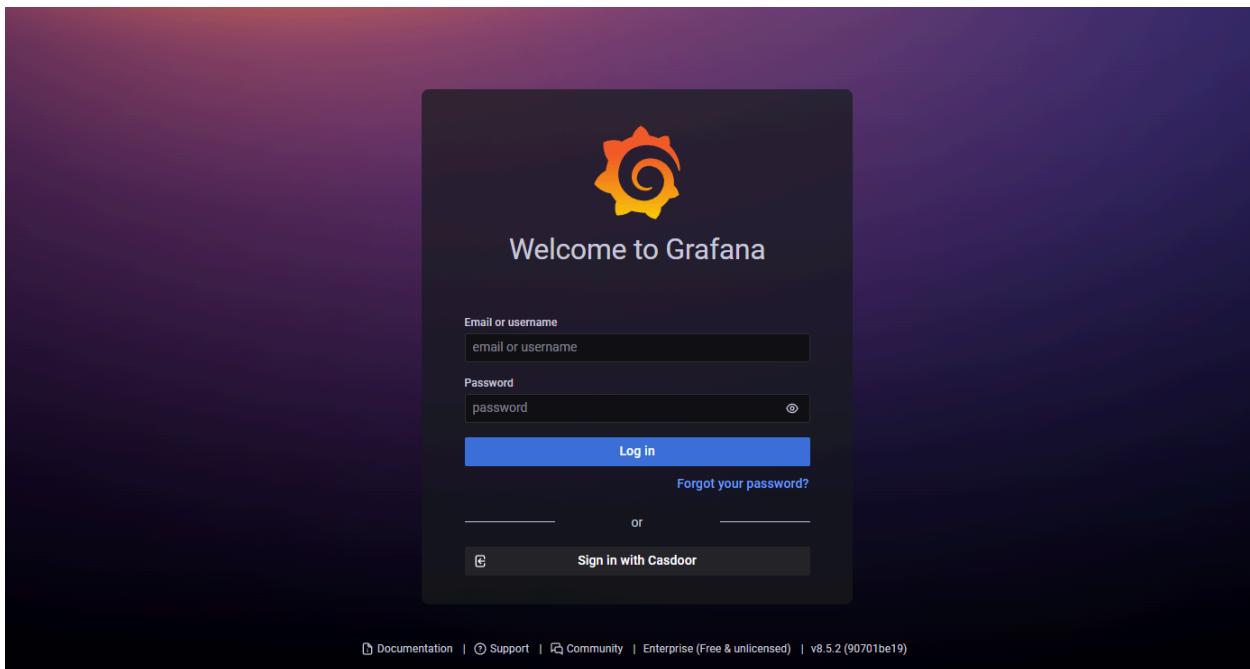
```
[auth.generic_oauth]
role_attribute_path = contains(roles[*].name, 'admin') && 'Admin'
```

The JMESPath expression after `role_attribute_path` is very important here.  
Please refer to the Grafana documentation.

## Step 3: See if it works

Shutdown Grafana and restart it.

Go to the login page. You should see something like this:



# MinIO

MinIO supports external identity management using an OpenID Connect (OIDC)-compatible provider. This document covers the configuration of Casdoor as an identity provider to support MinIO.

## Step 1: Deploy Casdoor & MinIO

First, deploy Casdoor.

You can refer to the Casdoor official documentation for [Server Installation](#).

After a successful deployment, make sure that:

- The Casdoor server is running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001> to see the login page of Casdoor.
- Test the login functionality by entering `admin` and `123`.

Next, you can quickly implement a Casdoor-based login page in your own app by following these steps.

You can refer to [here](#) to deploy your MinIO server and [here](#) for the MinIO client called `mc`.

## Step 2: Configure Casdoor Application

1. Create a new Casdoor application or use an existing one.

2. Add your redirect URL.

Client ID <a href="#">?</a> :	24a25ea0714d92e78595	Client ID
Client secret <a href="#">?</a> :	155 [REDACTED]	Client Secret
Redirect URLs <a href="#">?</a> :	Redirect URLs	Add
	Redirect URL	Add a redirect URL for spring security
	? http://localhost:8082/ui-one/login/oauth2/code/custom	

3. Add the provider you want and provide any necessary settings.

On the application settings page, you will find two values: `Client ID` and `Client secret` (as shown in the picture above). We will use these values in the next step.

Open your favorite browser and visit: [http://CASDOOR\\_HOSTNAME/.well-known/openid-configuration](http://CASDOOR_HOSTNAME/.well-known/openid-configuration) to see the OIDC configuration of Casdoor.

4. This step is necessary for MinIO. As MinIO needs to use a claim attribute in JWT for its policy, you should configure it in Casdoor as well. Currently, Casdoor uses `tag` as a workaround for configuring MinIO's policy.

Tag [?](#) : readwrite

You can find all the supported policies [here](#).

# Step 3: Configure MinIO

You can start a MinIO server using the following commands:

```
导出MINIO_ROOT_USER=minio  
导出 MINIO_ROOT_PASSWORD=minio123  
minio server /mnt/export
```

You can use the `--console-address` parameter to configure the address and port.

Next, add a service alias using the MinIO client `mc`.

```
mc alias set myminio <Your console address> minio minio123
```

Now, configure the OpenID Connect of MinIO. For Casdoor, the command will be:

```
mc admin config set myminio identity_openid  
config_url="http://CASDOOR_HOSTNAME/.well-known/openid-  
configuration" client_id=<client id> client_secret=<client secret>  
claim_name="tag"
```

You can refer to the [official document](#) for more detailed parameters.

Once successfully set, restart the MinIO instance.

```
mc 管理服务重启myminio
```

# Step 4: Try the demo!

Now, open your MinIO console in the browser and click on `Login with SSO`.

You will be redirected to the Casdoor user login page. Upon successful login, you will be redirected to the MinIO page and logged in automatically. You should now see the buckets and objects that you have access to.

## ⚠ 注意事项

If you deploy the frontend and backend of Casdoor on different ports, the login page you are redirected to will be on the backend port and it will display `404 not found`. You can modify the port to the frontend one. Then you can access the Casdoor login page successfully.

# Portainer

## Using Casdoor for authentication in Portainer

Portainer supports authentication via OAuth. Therefore, it is easy for users to use Casdoor to log in to Portainer. Only several steps and simple configurations are needed to achieve that.

Here is a tutorial on how to use Casdoor for authentication in Grafana. Before you proceed, please ensure that you have Portainer installed and running.

The following are the configuration names:

`CASDOOR_HOST`: The domain name or IP address where the Casdoor server is deployed.

`PORTAINER_HOST`: The domain name or IP address where Portainer is deployed.

## Step 1: Create an app for Portainer in Casdoor

Here is an example of creating an app in Casdoor:

Name ⓘ : Portainer\_test

Display name ⓘ : Portainer\_test

Logo ⓘ : URL ⓘ : [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home ⓘ :

Description ⓘ :

Organization ⓘ : built-in

Tags ⓘ :

Client ID ⓘ : 2da468d1968c5f85d6b4

Client secret ⓘ : b4db599c84f978425102f161b833625fa9b6b7c

Cert ⓘ : cert-built-in

Redirect URLs ⓘ : Redirect URLs Add

Redirect URL : [https://<PORTAINER\\_HOST>](https://<PORTAINER_HOST>)

1. Copy the client secret and client ID for the next step.
2. Add a Redirect URL. It's your Portainer host.

## Step 2: Configure Portainer

Expand the Settings from the left navigation bar, click on the Authentication option from this list.

1. Enable Use SSO and Automatic user provisioning:

The screenshot shows the Portainer.io interface with the 'Authentication' tab selected. The main section is titled 'Authentication settings'. Under 'Authentication method', 'Microsoft Active Directory' is highlighted with an orange border. Other options like 'Internal', 'LDAP', and 'OAuth' are also shown. Below this, there are sections for 'Single Sign-On' (with 'Use SSO' turned on) and 'Automatic user provisioning' (with a toggle switch turned on).

## 2. Fill in the necessary information as follows:

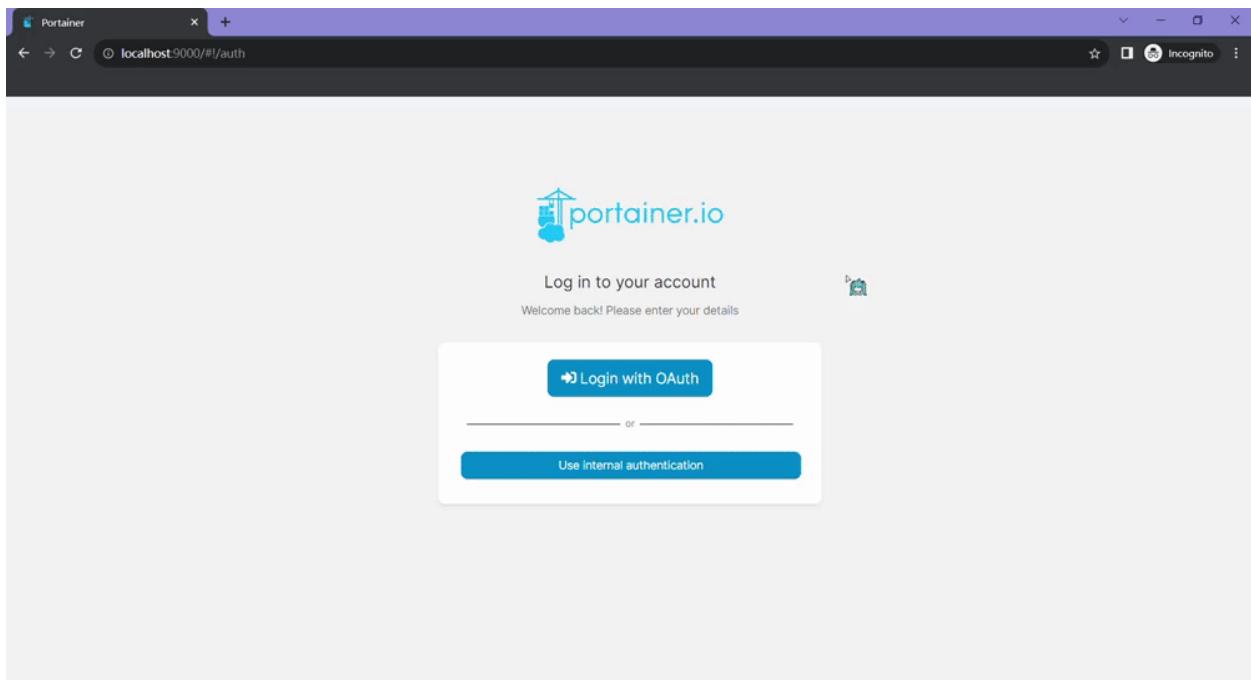
The screenshot shows the Portainer.io interface with the 'Authentication' tab selected. The main section is titled 'OAuth Configuration'. It lists four providers: Microsoft, Google, Github, and Custom. The 'Custom' provider is selected and its configuration fields are filled in:

- Client ID: 89c7dba629b5722d4ea2
- Client secret: (redacted)
- Authorization URL: https://<CASDOOR\_HOST>/login/oauth/authorize
- Access token URL: https://<CASDOOR\_HOST>/api/login/oauth/access\_token
- Resource URL: https://<CASDOOR\_HOST>/api/userinfo
- Redirect URL: https://<PORTAINER\_HOST>
- User identifier: email
- Scopes: openid email profile

A 'Save settings' button is at the bottom.

- Authorization URL: `https://<CASDOOR_HOST>/login/oauth/authorize`
- Access token URL: `https://<CASDOOR_HOST>/api/login/oauth/access_token`
- Resource URL: `https://<CASDOOR_HOST>/api/userinfo`
- Redirect URL: `https://<PORTAINER_HOST>`

Log out of Portainer and test.



# Java

## Spring Boot

Using Casdoor in a Spring Boot project

## Spring Cloud

在Spring Cloud中使用 Casdoor

## Spring Cloud Gateway

在Spring Cloud Gateway中使用 Casdoor

## Spring 安全

2 个项目

## Jenkins Plugin

Using the Casdoor plugin for Jenkins security

## Jenkins OIDC

Using the OIDC protocol as an IDP to connect various applications, like Jenkins

## Jira

2 个项目

## Connecting Applications with OIDC Protocol - Confluence

Learn how to use OIDC protocol as IDP to connect Confluence and other applications.

## RuoYi

在 RuoYi-Cloud 上使用 Casdoor

## Pulsar Manager

Using Casdoor in Pulsar Manager

## 在 ShenYu 中使用 Casdoor

How to use Casdoor with ShenYu

## ShardingSphere

在 ShardingSphere 中使用 Casdoor

## Apache IoTDB

Using Casdoor with Apache IoTDB

## Apache DolphinScheduler

Using Casdoor for DolphinScheduler SSO login



## FireZone

Using the OIDC protocol as the IDP to connect various applications, such as FireZone



## Cloud Foundry

Learn how to integrate Casdoor with Cloud Foundry to secure your applications.



## Thingsboard

Learn how to integrate Casdoor with Thingsboard to secure your applications

# Spring Boot

[casdoor-spring-boot-example](#) is an example of how to use [casdoor-spring-boot-starter](#) in a Spring Boot project. We will guide you through the steps below.

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed.

您可以参考[服务安装](#)的Casdoor官方文档。请在[生产模式](#)中部署您的castor实例。

After a successful deployment, make sure the following:

- Open your favorite browser and visit <http://localhost:8000>. You will see the login page of Casdoor.
- Test the login functionality by entering `admin` as the username and `123` as the password.

Now, you can quickly implement a Casdoor-based login page in your own app using the following steps.

## Step 2: Import casdoor-spring-boot-starter

You can import the casdoor-spring-boot-starter using Maven or Gradle.

Maven      Gradle

---

```
<!-- https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter -->
```

```
<dependency>
    <groupId>org.casbin</groupId>
    <artifactId>casdoor-spring-boot-starter</artifactId>
    <version>1.x.y</version>
</dependency>
```

```
// https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter
```

```
implementation group: 'org.casbin', name: 'casdoor-spring-boot-starter', version: '1.x.y'
```

## Step 3: Initialize Config

Initialization requires 6 string-type parameters in the following order: | Name | Required | 描述 | ----- | ----- | ----- |

----- | | endpoint | 是 | Casdoor  
Server URL, such as http://localhost:8000 | | clientId | 是 | Application client ID  
| | clientSecret | 是 | Application client secret | | certificate | Yes | Application  
certificate | | organizationName | 是 | Application organization | | applicationName  
| 否 | Application name | You can use Java properties or YAML files for  
initialization.

Properties      YML

---

```
casdoor.endpoint = http://localhost:8000
```

```
casdoor:  
  endpoint: http://localhost:8000  
  client-id: <client-id>  
  client-secret: <client-secret>  
  certificate: <certificate>  
  organization-name: built-in  
  application-name: app-built-in
```

### ⚠ 注意事项

Replace the configuration values with your own Casdoor instance, especially the `clientId`, `clientSecret`, and `jwtPublicKey`.

## Step 4: Redirect to the Login Page

When you need to authenticate users who access your app, you can send the target URL and redirect to the login page provided by Casdoor. Make sure you have added the callback URL (e.g. <http://localhost:8080/login>) in the application configuration beforehand.

```
@Resource  
private CasdoorAuthService casdoorAuthService;  
  
@RequestMapping("toLogin")  
public String toLogin() {  
    return "redirect:" +  
    casdoorAuthService.getSigninUrl("http://localhost:8080/login");  
}
```

# Step 5: Get Token and Parse

After the Casdoor verification is passed, it will redirect back to your application with the code and state.

You can get the code and call the `getOAuthToken` method, then parse the JWT token.

`Casdoor User` 包含了由Casdoor提供的有关用户的基本信息。 You can use it to set the session in your application.

```
@RequestMapping("login")
public String login(String code, String state, HttpServletRequest
request) {
    String token = "";
    CasdoorUser user = null;
    try {
        token = casdoorAuthService.getOAuthToken(code, state);
        user = casdoorAuthService.parseJwtToken(token);
    } catch (CasdoorAuthException e) {
        e.printStackTrace();
    }
    HttpSession session = request.getSession();
    session.setAttribute("casdoorUser", user);
    return "redirect:/";
}
```

# Services

Examples of APIs are shown below:

- CasdoorAuthService
  - `String token = casdoorAuthService.getOAuthToken(code, "app-built-in");`
  - `CasdoorUser casdoorUser = casdoorAuthService.parseJwtToken(token);`

- CasdoorUserService
  - `CasdoorUser casdoorUser = casdoorUserService.getUser("admin");`
  - `CasdoorUser casdoorUser = casdoorUserService.getUserByEmail("admin@example.com");`
  - `CasdoorUser[] casdoorUsers = casdoorUserService.getUsers();`
  - `CasdoorUser[] casdoorUsers = casdoorUserService.getSortedUsers("created_time", 5);`
  - `int count = casdoorUserService.getUserCount("0");`
  - `CasdoorResponse response = casdoorUserService.addUser(user);`
  - `CasdoorResponse response = casdoorUserService.updateUser(user);`
  - `CasdoorResponse response = casdoorUserService.deleteUser(user);`

- CasdoorEmailService
  - `CasdoorResponse response = casdoorEmailService.sendEmail(title, content, sender, receiver);`

- CasdoorSmsService
  - `CasdoorResponse response = casdoorSmsService.sendSms(randomCode(), receiver);`

- CasdoorResourceService
  - `CasdoorResponse response = casdoorResourceService.uploadResource(user, tag, parent, fullPath, file);`
  - `CasdoorResponse response =`

```
casdoorResourceService.deleteResource(file.getName());
```

# More Resources

You can explore the following projects/docs to learn more about integrating Java with Casdoor:

- [casdoor-java-sdk](#)
- [casdoor-spring-boot-starter](#)
- [casdoor-spring-boot-example](#)
- [casdoor-spring-security-example](#)
- [casdoor-spring-security-react-example](#)
- [casdoor-spring-boot-shiro-example](#)

# Spring Cloud

In the Spring Cloud microservice system, general authentication occurs at the gateway. Please refer to [casdoor-springcloud-gateway-example](#) for more information.

如果您想要在单个服务中使用Casdoor，您可以参考 [casdoor-spring-boot示例](#)。

Whether it's in the gateway layer or in a single service, both use the [casdoor-spring-boot-starter](#).

## 更多内容

You can explore the following projects/docs to learn more about integrating Java with Casdoor:

- [casdoor-java-sdk](#)
- [casdoor-spring-boot-starter](#)
- [casdoor-spring-boot-example](#)
- [casdoor-spring-security-example](#)
- [casdoor-spring-security-react-example](#)
- [casdoor-spring-boot-shiro-example](#)
- [casdoor-springcloud-gateway-example](#)

# Spring Cloud Gateway

The [casdoor-springcloud-gateway-example](#) is an example of how to use the [casdoor-spring-boot-starter](#) as an OAuth2 plugin in Spring Cloud Gateway. The steps to use it are described below.

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed. You can refer to the official Casdoor documentation for the [Server Installation](#). 请在 生产模式 中部署您的 Casdoor 实例。

After a successful deployment, you need to ensure the following:

- Open your favorite browser and visit <http://localhost:8000>. You will see the login page of Casdoor.
- Input `admin` and `123` to test if the login functionality is working fine.

After that, you can quickly implement a Casdoor-based login page in your own app using the following steps.

## Step 2: Initialize a Spring Cloud Gateway

You can use the code from this example directly or combine it with your own business code.

You need a gateway service and at least one business service. In this example, `casdoor-gateway` is the gateway service and `casdoor-api` is the business service.

## Step 3: Include the dependency

Add the `casdoor-spring-boot-starter` dependency to your Spring Cloud Gateway project.

For Apache Maven:

```
/casdoor-gateway/pom.xml
```

```
<!-- https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter -->
<dependency>
    <groupId>org.casbin</groupId>
    <artifactId>casdoor-spring-boot-starter</artifactId>
    <version>1.x.y</version>
</dependency>
```

For Gradle:

```
// https://mvnrepository.com/artifact/org.casbin/casdoor-spring-boot-starter
implementation group: 'org.casbin', name: 'casdoor-spring-boot-starter', version: '1.x.y'
```

## Step 4: Configure your properties

Initialization requires 6 parameters, all of which are of type string.

名称(按顺序排列)	Required	描述
endpoint	是	Casdoor Server URL, such as <code>http://localhost:8000</code>
clientId	是	Application.client_id
clientSecret	是	Application.client_secret
certificate	是	Application.certificate
organizationName	是	Application.organization
applicationName	否	Application.name

You can use Java properties or YAML files to initialize these parameters.

For properties:

```
casdoor.endpoint=http://localhost:8000
casdoor.clientId=<client-id>
casdoor.clientSecret=<client-secret>
casdoor.certificate=<certificate>
casdoor.organizationName=built-in
casdoor.applicationName=app-built-in
```

For YAML:

```
casdoor:
  endpoint: http://localhost:8000
  client-id: <client-id>
```

In addition, you need to configure Gateway Routing. For YAML:

```
spring:
  application:
    name: casdoor-gateway
  cloud:
    gateway:
      routes:
        - id: api-route
          uri: http://localhost:9091
          predicates:
            - Path=/api/**
```

## Step 5: Add the CasdoorAuthFilter

Add an implementation class of the GlobalFilter interface to the gateway for identity verification, such as the CasdoorAuthFilter used in this example.

If the authentication fails, it returns a 401 status code to the frontend to redirect them to the login interface.

```
@Component
public class CasdoorAuthFilter implements GlobalFilter, Ordered {

    private static final Logger LOGGER =
LoggerFactory.getLogger(CasdoorAuthFilter.class);

    @Override public int getOrder() {
        return 0;
    }

    @Override public Mono<Void> filter(ServerWebExchange exchange,
GatewayFilterChain chain) {
```

# Step 6: Get the Service and use it

Now provide 5 services: `CasdoorAuthService`, `CasdoorUserService`, `CasdoorEmailService`, `CasdoorSmsService`, and `CasdoorResourceService`.

You can create them as follows in the Gateway project.

```
@Resource  
private CasdoorAuthService casdoorAuthService;
```

When you require authentication for accessing your app, you can send the target URL and redirect to the login page provided by Casdoor.

Please make sure that you have added the callback URL (e.g., <http://localhost:9090/callback>) in the application configuration in advance.

```
@RequestMapping("login")  
public Mono<String> login() {  
    return Mono.just("redirect:" +  
        casdoorAuthService.getSignInUrl("http://localhost:9090/callback"));  
}
```

After successful verification by Casdoor, it will be redirected back to your application with a code and state. You can get the code and call the `getOAuthToken` method to parse out the JWT token.

`CasdoorUser` contains the basic information about the user provided by Casdoor. You can use it as a keyword to set the session in your application.

```

@RequestMapping("callback")
public Mono<String> callback(String code, String state,
ServerWebExchange exchange) {
    String token = "";
    CasdoorUser user = null;
    try {
        token = casdoorAuthService.getOAuthToken(code, state);
        user = casdoorAuthService.parseJwtToken(token);
    } catch(CasdoorAuthException e) {
        e.printStackTrace();
    }
    CasdoorUser finalUser = user;
    return exchange.getSession().flatMap(session -> {
        session.getAttributes().put("casdoorUser", finalUser);
        return Mono.just("redirect:/");
    });
}

```

Examples of the APIs are shown below.

- CasdoorAuthService
  - `String token = casdoorAuthService.getOAuthToken(code, "app-built-in");`
  - `CasdoorUser casdoorUser = casdoorAuthService.parseJwtToken(token);`
- CasdoorUserService
  - `CasdoorUser casdoorUser = casdoorUserService.getUser("admin");`
  - `CasdoorUser casdoorUser = casdoorUserService.getUserByEmail("admin@example.com");`
  - `CasdoorUser[] casdoorUsers = casdoorUserService.getUsers();`
  - `CasdoorUser[] casdoorUsers = casdoorUserService.getSortedUsers("created_time", 5);`
  - `int count = casdoorUserService.getUserCount("0");`

- CasdoorResponse response = casdoorUserService.addUser(user);
  - CasdoorResponse response =  
casdoorUserService.updateUser(user);
  - CasdoorResponse response =  
casdoorUserService.deleteUser(user);
- CasdoorEmailService
    - CasdoorResponse response = casdoorEmailService.sendEmail(title, content, sender, receiver);
  - CasdoorSmsService
    - CasdoorResponse response =  
casdoorSmsService.sendSms(randomCode(), receiver);
  - CasdoorResourceService
    - CasdoorResponse response =  
casdoorResourceService.uploadResource(user, tag, parent, fullFilePath, file);
    - CasdoorResponse response =  
casdoorResourceService.deleteResource(file.getName());

## Step 7: Restart the project

After starting the project, open your favorite browser and visit <http://localhost:9090>. Then click any button that requests resources from casdoor-api.



# Casdoor

[Get Resource](#)

[Update Resource](#)

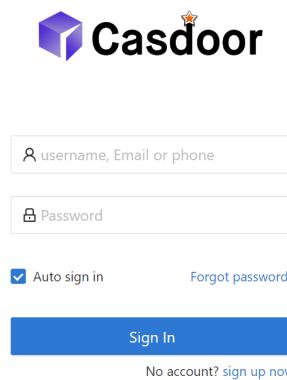
将触发网关认证逻辑。 Since you are not logged in, you will be redirected to the login interface. Click the Login button.



# Click to login

>Login

随后您可以看到Cassdoor统一的登录平台。



After a successful login, you will be redirected to the main interface. Now you can click any button.



# Casdoor

[Get Resource](#)

[Update Resource](#)

"success get resource1"

## 更多内容

您可以探索以下项目/文件来了解更多关于Java与Casdoor一体化的信息。

- [casdoor-java-sdk](#)
- [casdoor-spring-boot-starter](#)
- [casdoor-spring-boot-example](#)
- [casdoor-spring-security-example](#)
- [casdoor-spring-security-react-example](#)
- [casdoor-spring-boot-shiro-example](#)
- [casdoor-springcloud-gateway-example](#)

# Spring 安全

## Spring Security OAuth

Using Spring Security as an example to demonstrate how to use OIDC to connect to your applications

## Spring Security Filter with OIDC integration for Casdoor

This article explains how to use Spring Security Filter to connect your application with Casdoor using OIDC.

# Spring Security OAuth

Casdoor can use the OIDC protocol as an IDP to connect various applications. In this guide, we will use Spring Security as an example to show you how to use OIDC to connect to your applications.

## Step 1: Deploy Casdoor

First, you need to deploy Casdoor.

您可以参考Casdoor 官方文档 [Server Installation](#)。

After successfully deploying Casdoor, make sure:

- The Casdoor server is running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>, where you will see the login page of Casdoor.
- Verify that the login functionality is working fine by entering `admin` and `123`.

Now, you can quickly implement a Casdoor-based login page in your own app by following the steps below.

## Step 2. 配置Casdoor应用程序

1. Create a new Casdoor application or use an existing one.
2. Add your redirect URL (You can find more details on how to obtain the redirect URL in the next section).

The screenshot shows the 'Client ID' field containing '24a25ea0714d92e78595' and the 'Client Secret' field containing '155...'. Below these, the 'Redirect URLs' section shows a table with one row: 'http://localhost:8082/ui-one/login/oauth2/code/custom'.

3. Add the desired provider and fill in any additional settings.

On the application settings page, you will find two values: `Client ID` and `Client secret`, as shown in the image above. We will use these values in the next step.

Open your preferred browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`. Here, you will find the OIDC configuration of Casdoor.

## Step 3. 配置Spring Security

Spring Security natively supports OIDC.

您可以自定义Spring Security OAuth2 客户端的设置：

### ⚠ 注意事项

您应该用您自己的 Casdoor 实例替换配置，特别是 `<client ID>` 等。

`application.yml`

`application.properties`

```
spring:  
  security:  
    oauth2:  
      client:  
        registration:  
          casdoor:
```

```
spring.security.oauth2.client.registration.casdoor.client-id=<Client ID>
spring.security.oauth2.client.registration.casdoor.client-secret=<Client Secret>
spring.security.oauth2.client.registration.casdoor.scope=<Scope>
spring.security.oauth2.client.registration.casdoor.authorization-grant-type=authorization_code
spring.security.oauth2.client.registration.casdoor.redirect-uri=<Redirect URL>

spring.security.oauth2.client.provider.casdoor.authorization-uri=http://CASDOOR_HOSTNAME:7001/login/oauth/authorize
spring.security.oauth2.client.provider.casdoor.token-uri=http://CASDOOR_HOSTNAME:8000/api/login/oauth/access_token
spring.security.oauth2.client.provider.casdoor.user-info-uri=http://CASDOOR_HOSTNAME:8000/api/get-account
spring.security.oauth2.client.provider.casdoor.user-name-attribute=name
```

#### ⚠ 注意事项

For the default situation of Spring Security, the <Redirect URL> should be like

`http://<Your Spring Boot Application Endpoint>/<Servlet Prefix if it is configured>/login/oauth2/code/custom`. 例如，对于下面的演示来说，重定向URL应该是 `http://localhost:8080/login/oauth2/code/code/custom`。 You should also configure this in the `casdoor` application.

You can also customize the settings using `ClientRegistration` in your code. 您可以在这里找到映射

## Step 4: Get Started with a Demo

1. 我们可以创建 Spring Boot 应用程序。
2. We can add a configuration that protects all endpoints except `/` and `/login**` for users to log in.

```
@EnableWebSecurity
```

3. We can add a naive page for the user to log in.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Spring OAuth Client Thymeleaf - 1</title>
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/
bootstrap.min.css" />
</head>
<body>
  <nav
    class="navbar navbar-expand-lg navbar-light bg-light shadow-sm
p-3 mb-5">
    <a class="navbar-brand" th:href="@{/foos/}">Spring OAuth Client
      Thymeleaf - 1</a>
  </nav>
  <div class="container">
    <label>Welcome!</label> <br /> <a th:href="@{/foos/}"
      class="btn btn-primary">Login</a>
  </div>
</body>
</html>
```

When the user clicks the `login` button, they will be redirected to `casdoor`.

4. Next, we can define our protected resources. We can expose an endpoint called `/foos` and a web page for display.

#### Data Model

```
public class FooModel {
  private Long id;
  private String name;

  public FooModel(Long id, String name) {
    super();
    this.id = id;
    this.name = name;
```

## Controller

```
@Controller
public class FooClientController {
    @GetMapping("/foos")
    public String getFoos(Model model) {
        List<FooModel> foos = new ArrayList<>();
        foos.add(new FooModel(1L, "a"));
        foos.add(new FooModel(2L, "b"));
        foos.add(new FooModel(3L, "c"));
        model.addAttribute("foos", foos);
        return "foos";
    }
}
```

## Web page

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Spring OAuth Client Thymeleaf - 1</title>
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/
bootstrap.min.css" />
</head>
<body>
    <nav
        class="navbar navbar-expand-lg navbar-light bg-light shadow-sm
p-3 mb-5">
        <a class="navbar-brand" th:href="@{/foos/}">Spring OAuth Client
            Thymeleaf - 1</a>
        <ul class="navbar-nav ml-auto">
            <li class="navbar-text">Hi, <span
sec:authentication="name">preferred_username</span>&ampnbsp&ampnbsp&ampnbsp</li>
        </ul>
    </nav>
    <div class="container">
        <h1>All Foos:</h1>
        <table class="table table-bordered table-striped">
            <thead>
```

### ⚠ 注意事项

All the web page templates should be placed under `resources/templates`.

## Step 5: Try the demo!

Firstly, you can try opening your favorite browser and directly visiting `/foos`. It will automatically redirect you to Casdoor's login page. You can log in there or from the root page.

If you visit your root page, you will see the Casdoor Application Setting.

Spring OAuth Client Thymeleaf - 1

Welcome !  
[Login](#)

Click the `Login` button and the page will redirect you to Casdoor's login page.



username, Email or phone

Password

Auto sign in      [Forgot password?](#)

[Sign In](#)

[Sign in with code](#)    No account? [sign up now](#)

After logging in, the page will redirect you to </foos>.

Spring OAuth Client Thymeleaf -1

Hi,

Your Username

### All Foos:

ID	Name
1	a
2	b
3	c





# Spring Security Filter with OIDC integration for Casdoor

Casdoor is an open-source IDP that supports OIDC and various other protocols. In this article, we will see how to integrate Casdoor with your application using Spring Security Filter and OIDC.

## Step 1: Deploy Casdoor

First, you need to deploy the Casdoor server. Refer to the [official documentation](#) for server installation instructions. After successful deployment, ensure that:

- The Casdoor server is running at <http://localhost:8000>.
- You can see the Casdoor login page at <http://localhost:7001>.
- You can test the login functionality by logging in with the credentials `admin` and `123`.

After verifying these steps, follow the steps below to integrate Casdoor with your application.

## Step 2: Configure Casdoor Application

- Create a new Casdoor application or use an existing one.
- Add your redirect URL. You can find more information about obtaining the redirect URL in the next section.

Name [?](#) : application\_a6ftas → your application name

Display name [?](#) : New Application - a6ftas

Logo [?](#) : URL [?](#) : [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home [?](#) :

Description [?](#) :

Organization [?](#) : organization\_carg1b → your organization name

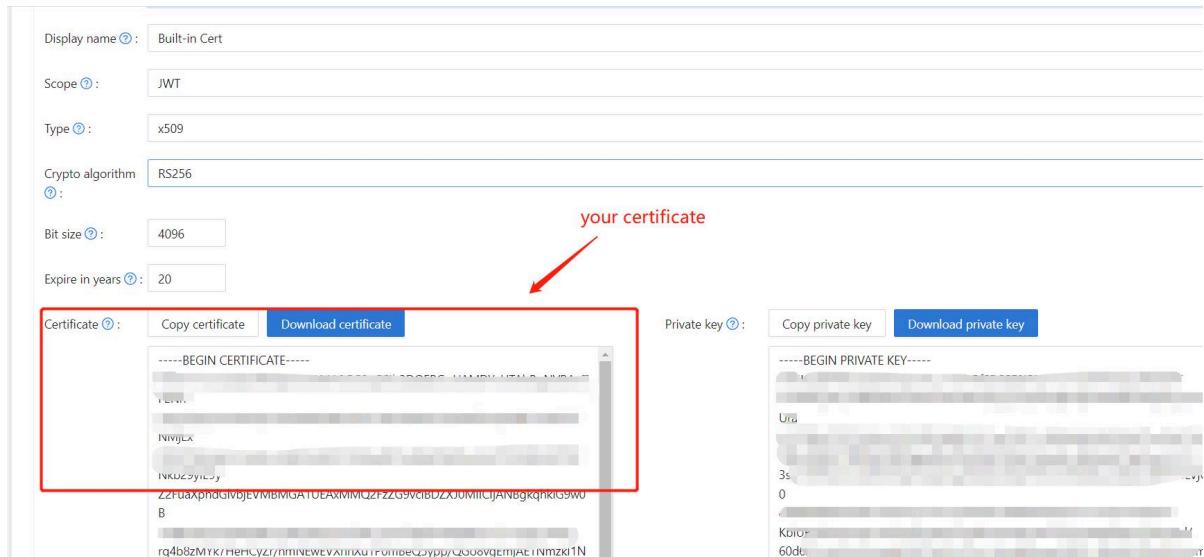
Client ID [?](#) : 3ed7314825ecf955cb19 → your client id

Client secret [?](#) : ee9314ea228 → your client secret

Cert [?](#) : cert-built-in

Redirect URLs [?](#) : Redirect URLs [Add](#)  
Redirect URL [?](#) http://localhost:3000/callback → your redirect url

- Obtain your [Certificate](#) on the certificate editing page.



- Add the provider and other settings as needed.

You can obtain the values for `Application Name`, `Organization Name`, `Redirect URL`, `Client ID`, `Client Secret`, and `Certificate` on the application settings page. We will use them in the next step.

## Step 3: Configure Spring Security

You can customize the settings of the Spring Security filters to process tokens:

### **⚠ 注意事项**

Make sure you replace the configuration values with your own Casdoor instance, especially `<Client ID>` and the others.

```
server:
  port: 8080
casdoor:
  endpoint: http://CASDOOR_HOSTNAME:8000
  client-id: <Client ID>
  client-secret: <Client Secret>
  certificate: <Certificate>
```

### ⚠ 注意事项

对于前端应用程序来说，`<FRONTEND_HOSTNAME>` 的默认值是 `localhost:3000`。In this demo, the redirect URL is `http://localhost:3000/callback`. Make sure to configure this in your `casdoor` application.

## Step 4: Configure Frontend

You need to install `casdoor-js-sdk` and configure the SDK as follows:

1. 安装 `casdoor-js-sdk`。

```
npm i casdoor-js-sdk
# or
yarn add casdoor-js-sdk
```

2. 设置 `SDK`。

```
import Sdk from "casdoor-js-sdk";

// Serverurl is the URL where spring security is deployed
export const ServerUrl = "http://BACKEND_HOSTNAME:8080";

const sdkConfig = {
  serverUrl: "http://CASDOOR_HOSTNAME:8000",
  clientId: "<your client id>",
  appName: "<your application name>",
  organizationName: "<your organization name>",
  redirectPath: "/callback",
};

export const CasdoorSDK = new Sdk(sdkConfig);
```

# Step 5: Set Up a Demo

1. Create a Spring Boot application.
2. Add some configurations to handle JWT.

```
@EnableWebSecurity
public class SecurityConfig {

    private final JwtTokenFilter jwtTokenFilter;

    public SecurityConfig(JwtTokenFilter jwtTokenFilter) {
        this.jwtTokenFilter = jwtTokenFilter;
    }

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        // enable CORS and disable CSRF
        http = http.cors(corsConfig -> corsConfig
            .configurationSource(configurationSource())
            .csrf().disable());

        // set session management to stateless
        http = http
            .sessionManagement()

        .sessionCreationPolicy(SessionCreationPolicy.STATELESS)
            .and();

        // set permissions on endpoints
        http.authorizeHttpRequests(authorize -> authorize
            .mvcMatchers("/api/redirect-url", "/api/signin").permitAll()
            .mvcMatchers("/api/**").authenticated()
        );

        // set unauthorized requests exception handler
    }
}
```

3. Add a simple JWT filter to intercept requests that require token verification.

```
@Component
public class JwtTokenFilter extends OncePerRequestFilter {

    private final CasdoorAuthService casdoorAuthService;

    public JwtTokenFilter(CasdoorAuthService casdoorAuthService) {
        this.casdoorAuthService = casdoorAuthService;
    }

    @Override
    protected void doFilterInternal(HttpServletRequest request,
                                    HttpServletResponse response,
                                    FilterChain chain)
        throws ServletException, IOException {
        // get authorization header and validate
        final String header =
request.getHeader(HttpHeaders.AUTHORIZATION);
        if (!StringUtils.hasText(header) ||
!header.startsWith("Bearer ")) {
            chain.doFilter(request, response);
            return;
        }

        // get jwt token and validate
        final String token = header.split(" ")[1].trim();

        // get user identity and set it on the spring security
        context
        UserDetails userDetails = null;
        try {
            CasdoorUser casdoorUser =
casdoorAuthService.parseJwtToken(token);
            userDetails = new CustomUserDetails(casdoorUser);
        } catch (CasdoorAuthException exception) {
            logger.error("casdoor auth exception", exception);
            chain.doFilter(request, response);
            return;
        }
    }
}
```

When the user accesses the interface requiring authentication, `JwtTokenFilter` will obtain the token from the request header `Authorization` and verify it.

4. Define a `Controller` to handle when the user logs in to Casdoor. After the user logs in, they will be redirected to the server and carry the `code` and `state`. The server then needs to verify the user's identity from Casdoor and obtain the `token` through these two parameters.

```
@RestController
public class UserController {

    private static final Logger logger =
LoggerFactory.getLogger(UserController.class);

    private final CasdoorAuthService casdoorAuthService;

    // ...

    @PostMapping("/api/signin")
    public Result signin(@RequestParam("code") String code,
@RequestParam("state") String state) {
        try {
            String token = casdoorAuthService.getOAuthToken(code,
state);
            return Result.success(token);
        } catch (CasdoorAuthException exception) {
            logger.error("casdoor auth exception", exception);
            return Result.failure(exception.getMessage());
        }
    }

    // ...
}
```

## Step 6: Try the Demo

You can access the frontend application through your browser. If you are not logged in,

you will see a login button. Click on it, and you will be redirected to the Casdoor login page.

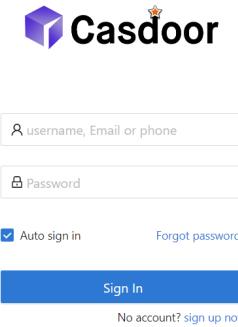
If you visit your root page,

---

[Casdoor Login](#)

Click the [Casdoor Login](#) button, and the page will redirect to Casdoor's login page.

---



---

Made with ❤ by [Casdoor](#)

After logging in, you will be redirected to [/](#).



New User - rtsbx4

[Logout](#)

# Jenkins Plugin

Casdoor provides a plugin that allows users to log in to Jenkins. Here, we will show you how to use the Casdoor plugin for Jenkins security.

The following are some of the configuration settings:

`CASDOOR_HOSTNAME`: The domain name or IP where the Casdoor server is deployed.

`JENKINS_HOSTNAME`: The domain name or IP where Jenkins is deployed.

## Step 1: Deploy Casdoor and Jenkins

Firstly, deploy [Casdoor](#) and [Jenkins](#).

After a successful deployment, ensure the following:

1. Set the Jenkins URL (Manage Jenkins → Configure System → Jenkins Location) to `JENKINS_HOSTNAME`.

Dashboard > configuration

**Jenkins Location**

Jenkins URL  ?

JENKINS\_HOSTNAME  ?

System Admin e-mail address  ?

**Serve resource files from another domain**

Resource Root URL  ?

Without a resource root URL, resources will be served from the Jenkins URL with Content-Security-Policy set.

**Global properties**

Environment variables

**Save** **Apply**

2. Verify that Casdoor can be logged in and used normally.
3. Set the `origin` value of Casdoor (conf/app.conf) to `CASDOOR_HOSTNAME`.

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 | origin = "http://10.144.1.2:8000"| CASDOOR_HOSTNAME
```

## Step 2: Configure the Casdoor Application

1. Create a new Casdoor application or use an existing one.
2. Add a redirect URL: `http://JENKINS_HOSTNAME/securityRealm/finishLogin`

Description [?](#): Casdoor for Jenkins

Organization [?](#): built-in

Client ID [?](#): bbd0bd66696e504dec59 Client ID

Client secret [?](#): d2de01b01REDACTED110b47465c Client secret

Redirect URLs [?](#):

Redirect URLs	Add
Redirect URL	<input type="text" value="http://10.144.125.123:6780/securityRealm/finishLogin"/> <span style="color: red;">Add a redirect url for Jenkins</span>
JENKINS_HOSTNAME	

3. Add the desired provider and provide any additional settings.

On the application settings page, you will find two values: `Client ID` and `Client secret`, as shown in the picture above. We will use these values in the next step.

Open your favorite browser and visit `http://CASDOOR_HOSTNAME/.well-known/openid-configuration` to view the OIDC configuration of Casdoor.

## Step 3: Configure Jenkins

Now, you can install the Casdoor plugin from the marketplace or by uploading its `.jar` file.

After the installation is complete, go to `Manage Jenkins → Configure Global Security`.

**Suggestion:** Back up the Jenkins `config.xml` file and use it for recovery in case of setup errors.

 **Configure Global Security**

**Authentication**

Disable remember me

**Security Realm**

Casdoor Authentication Plugin  
Casdoor Endpoint  
 Casdoor Endpoint is required.

Client ID  
 Client Id is required.

Client Secret  
 Client Secret is required.

JWT Public Key  
 Jwt Public Key is required.

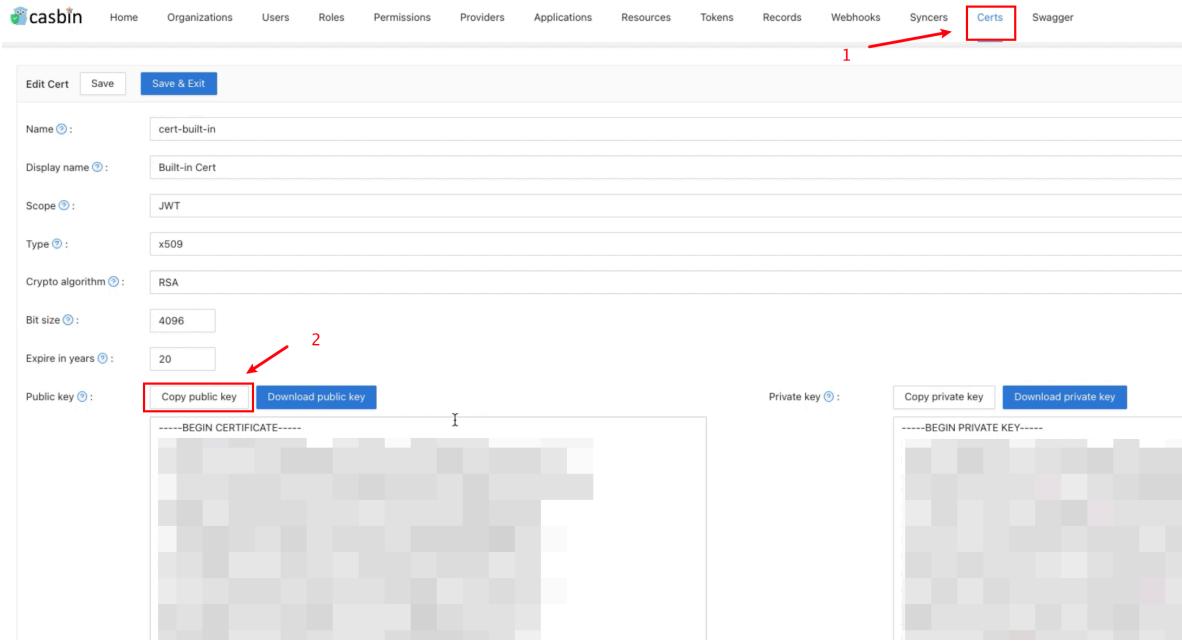
Organization Name

Application Name

Delegate to servlet container  
 Jenkins' own user database

Advanced...  

1. In the Security Realm section, select "Casdoor Authentication Plugin".
2. In the Casdoor Endpoint field, enter the `CASDOOR_HOSTNAME` mentioned earlier.
3. In the Client ID field, enter the `Client ID` mentioned earlier.
4. In the Client secret field, enter the `Client secret` mentioned earlier.
5. In the JWT Public Key field, provide the public key used to validate the JWT token. You can find the public key in Casdoor by clicking on `Cert` at the top. After clicking `edit` on your application, you can copy the public key from the following page.



6. Organization Name and Application Name are optional. You can specify your organization and application to verify users in other organizations and applications. If these fields are left empty, the plugin will use the default organization and application.
7. In the Authorization section, check "Logged-in users can do anything". Disable "Allow anonymous read access".
8. Click **Save**.

Jenkins will now automatically redirect you to Casdoor for authentication.

# Jenkins OIDC

Casdoor can use the OIDC protocol as an IDP to connect various applications. In this example, we will use Jenkins to demonstrate how to use OIDC to connect to your applications.

The following are some of the names used in the configuration:

- `CASDOOR_HOSTNAME`: The domain name or IP where the Casdoor server is deployed.
- `JENKINS_HOSTNAME`: The domain name or IP where Jenkins is deployed.

## Step 1: Deploy Casdoor and Jenkins

Firstly, deploy [Casdoor](#) and [Jenkins](#).

After a successful deployment, ensure the following:

1. Set the Jenkins URL (Manage Jenkins → Configure System → Jenkins Location) to `JENKINS_HOSTNAME`.

The screenshot shows the Jenkins configuration page under 'Dashboard > configuration'. It includes sections for 'Jenkins Location' (with Jenkins URL set to 'http://10.144.125.123:6780') and 'Global properties' (with checkboxes for 'Environment variables' and 'Advanced'). Buttons for 'Save' and 'Apply' are at the bottom.

2. Ensure that Casdoor can be logged in and used normally.
3. Set Casdoor's `origin` value (conf/app.conf) to `CASDOOR_HOSTNAME`.

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 | origin = "http://10.144.1.2:8000"
      CASDOOR_HOSTNAME
```

## Step 2: Configure the Casdoor application

1. Create a new Casdoor application or use an existing one.

2. Add a redirect URL: `http://JENKINS_HOSTNAME/securityRealm/finishLogin`

The screenshot shows the Casdoor application settings page. A specific redirect URL configuration is highlighted with a red box. The URL is `http://10.144.125.123:6780/securityRealm/finishLogin`, which corresponds to the placeholder `JENKINS_HOSTNAME` mentioned in the text above. The 'Add' button for redirect URLs is also highlighted with a red box.

Description : Casdoor for Jenkins

Organization : built-in

Client ID : bbd0bd66696e504dec59 Client ID

Client secret : d2de01b01...110b47465c Client secret

Redirect URLs

Redirect URLs Add

Redirect URL

`http://10.144.125.123:6780/securityRealm/finishLogin` Add a redirect url for Jenkins

`JENKINS_HOSTNAME`

3. Add the provider you want and provide any additional settings.

You will obtain two values from the application settings page: `Client ID` and `Client secret`. We will use these values in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration` to view the OIDC configuration of Casdoor.

## Step 3: Configure Jenkins

First, we need to install [OpenId Connect Authentication](#) as Jenkins does not natively support OIDC.

After the installation is complete, go to [Manage Jenkins → Configure Global Security](#).

The screenshot shows the Jenkins Manage Jenkins dashboard. On the left, there's a sidebar with links like '用户列表' (User List), '构建历史' (Build History), 'Manage Jenkins' (which is highlighted with a red border), and 'My Views'. Below that are sections for '构建队列' (Build Queue) and '构建执行状态' (Build Execution Status). The main area is titled 'System Configuration' and contains several management options: 'Configure System' (global settings), 'Global Tool Configuration' (configure tools and installers), 'Manage Nodes and Clouds' (control various nodes), 'Manage Plugins' (manage Jenkins plugins), 'Configure Global Security' (highlighted with a red border), 'Manage Credentials' (configure credentials), and 'Configure Credential Providers'.

### 💡 提示

Make sure to back up the Jenkins `config.xml` file to recover in case of any setup errors.

1. In Access Control, select `Login with Openid Connect` as the Security Realm.
2. Specify the `Client ID` noted above in the Client ID field.
3. Specify the `Client secret` noted above in the Client secret field.
4. In the Configuration mode, select `Automatic configuration` and enter `http://CASDOOR_HOSTNAME.well-known/openid-configuration` as the Well-known configuration endpoint.

**Security Realm**

- Delegate to servlet container
- Jenkins' own user database
- Login with Openid Connect Select this

Client id ?

`bbd0bd66696e504dec59`

Input your Client ID

Client secret ?

🔒 Concealed

Input your Client secret

Change Password

**Configuration mode**

- Automatic configuration
- Well-known configuration endpoint
- Manual configuration

Well-known configuration endpoint ?

`http://10.144.1.2:8000/.well-known/openid-configuration`

**CASDOOR\_HOSTNAME**

Manual configuration ?

If your Casdoor is deployed locally, you may need to select **Manual configuration** and provide the following information:

- Token server URL: `http://CASDOOR_HOSTNAME/api/login/oauth/access_token`
- Authorization server URL: `http://CASDOOR_HOSTNAME/login/oauth/authorize`
- UserInfo server URL: `http://CASDOOR_HOSTNAME/api/get-account`
- Scopes: `address phone openid profile offline_access email`

Configuration mode

- Automatic configuration
- Manual configuration

Token server url ?

`http://10.144.1.2:8000/api/login/oauth/access_token`

**CASDOOR\_HOSTNAME**

Authorization server url ?

`http://10.144.1.2:8000/login/oauth/authorize`

Userinfo server url ?

`http://10.144.1.2:8000/api/get-account`

Scopes ?

`address phone openid profile offline_access email`

5. Click on **Advanced settings** and fill in the following:

- In the User name field, specify `name`.

- In the Full name field, specify `displayName`.
- In the Email field, specify `email`.

User name field name	<input type="text" value="name"/>
Full name field name	<input type="text" value="displayName"/>
Email field name	<input type="text" value="email"/>
Groups field name ?	<input type="text"/>
Token Field Key To Check ?	<input type="text"/>

6. In the Authorization section, enable “Logged-in users can do anything” and disable “Allow anonymous read access”. You can configure more complex authorization later, but for now, check if OpenID works correctly.

Log out of Jenkins, and it should redirect you to Casdoor for authentication.



Auto sign in

[Forgot password?](#)

[Sign In](#)

[Sign in with code](#)    [No account? sign up now](#)

Q
G

# Jira

## Via Built-in SSO

Using the OIDC protocol as an IDP to connect various applications, such as Jira

## Using the miniOrange Plugin

Connect casdoor and Jira using the OIDC protocol as the IDP

# Via Built-in SSO

This is a free method to connect Casdoor, but your website must use HTTPS.

Casdoor can use the OIDC protocol as an IDP to connect various applications. Here is a [Jira](#) tutorial.

The following are some of the names in the configuration:

- `CASDOOR_HOSTNAME`: Domain name or IP where the Casdoor server is deployed.
- `Jira_HOSTNAME`: Domain name or IP where Jira is deployed.

## Step 1: Deploy Casdoor and Jira

Firstly, deploy [Casdoor](#) and [Jira](#).

After a successful deployment, ensure the following:

1. Casdoor can be logged in and used normally.
2. You can set `CASDOOR_HOSTNAME` to `http://localhost:8000` when deploying Casdoor in `prod` mode. See [production mode](#).

## Step 2: Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Find Authentication methods:

The screenshot shows the Jira Software Administration interface. The left sidebar has sections like Applications, Projects, Issues, Manage apps, User management, and System (which is selected). The main content area is titled "Authentication methods". It includes a warning about making authentication safer, a table of login options (Username and password, casdoor), and a section for "Authentication on API calls" with a toggle switch. A red box labeled "1" highlights the "System" button in the top right corner of the sidebar. A red box labeled "2" highlights the "System" tab in the top navigation bar. A red box labeled "3" highlights the "Authentication methods" link in the sidebar.

3. Add a Configuration and choose OpenID Connection single sign-on in the Authentication method

### Add new configuration

Name \*

Use a unique name for this configuration.

Authentication method

OpenID Connect single sign-on



Users log in using OpenID Connect

4. Find the redirect URL:

Give these URLs to your identity provider

Redirect URL

<https://test.v2tl.com/plugins/servlet/oidc/callback>



Location where the client is sent to after successful account authentication.

## 5. Add a redirect URL:

The screenshot shows the Casdoor application settings page. It includes fields for Client ID (642ec5d6779a2f0e879d), Client secret (26cb47985c47ae3844580536ce2f59872969e109), and Cert (cert-built-in). Below these, there is a section for Redirect URLs with an 'Add' button. A single redirect URL is listed: https://test.v2tl.com/plugins/servlet/oidc/callback. To the right of the list is an 'Action' column with icons for moving up, moving down, and deleting.

Not surprisingly, you can obtain two values on the application settings page:

`Client ID` and `Client secret`, like the picture above. We will use them in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`. You will see the OIDC configuration of Casdoor.

## Step 3: Configure Jira

1. We need to continue configuring our Configuration in Jira

## Edit existing configuration

Name \*

casdoor

Use a unique name for this configuration.

Authentication method

OpenID Connect single sign-on

Users log in using OpenID Connect

### OpenID Connect settings

Issuer URL \*

https://demo.casdoor.com

your casdoor url

The complete URL of the OpenID Provider. Needs to be unique.

Client ID \*

642ec5d6779a2f0e879d

application client ID

The client identifier, as registered with the OpenID Provider.

Client secret \*

.....

application client secret [Change](#)

Client secret is used in conjunction with the Client ID to authenticate the client application against the OpenID Provider.

Username mapping \*

\${preferred\_username}

Used to map IdP claims to the username, e.g. \${sub}

Additional scopes

phone ✕ email ✕ address ✕ profile ✕

✖ ▾

The default scope is 'openid'. Add more scopes if needed to obtain the username claim.

Redirect URL  
 Copy it to casdoor

Location where the client is sent to after successful account authentication.

Initiate login URL  
 Copy

URL used for OpenID Provider-initiated login.

**Additional settings**  
The authorization, token, and user info endpoints will be filled automatically if your Identity provider offers this option. If not, you will be asked to provide this information.

Fill the data automatically from my chosen identity provider.

**JIT provisioning**  
Just-in-time user provisioning allows users to be created and updated automatically when they log in through SSO to Atlassian Data Center applications. [Learn more](#).

Create users on login to the application

**OpenID Connect behaviour**

Remember user logins  
If checked, successful login history will be saved and users will be logged in automatically without the need for reauthentication.

**Login page settings**  
Decide if the IdP should be visible on login page and customize what the user will see on the button.

Show IdP on the login page

Login button text \*

The text is shown to the user on the login page. Remaining characters: 33.

Save configuration Cancel

2. You can configure more complex authorization later. For now, check if OpenID actually works.

⚠ You have temporary access to administrative functions. [Drop access](#) if you no longer require it. For more information, refer to the [documentation](#).



Dashboards ▼ Projects ▼ Issues ▼ Boards ▼ Plans ▼ Create

Search



## Administration

Search Jira admin

Applications Projects Issues Manage apps User management Latest upgrade report System

General configuration

[Find more admin tools](#)

Jira mobile app

SYSTEM SUPPORT

System info

Instrumentation

Monitoring

Database monitoring

Integrity checker

Logging and profiling

Scheduler details

Troubleshooting and support tools

Clean up

Audit log

Clustering

SECURITY

Project roles

Global permissions

### Authentication methods

Add configuration

Manage how users authenticate. Save authentication configurations using SAML, OpenID Connect, or Crowd as the identity provider. [Learn more about using multiple identity providers.](#)

#### ⚠ Make authentication safer

Authenticating with username and password is less secure than through single sign-on. Now that you've configured the latter, consider disabling product login form and basic authentication.

Communicate this change to your users.

[How to disable](#) - Dismiss

#### Login options

Name	Type	Last updated	Show on login page	Actions
Username and password	Product login form	Never	<input checked="" type="checkbox"/>	<span style="font-size: small;">...</span>
casdoor	OpenID Connect	26 April 2023 7:20 PM	<input checked="" type="checkbox"/>	<span style="font-size: small;">...</span>

#### Authentication on API calls



Allow basic authentication on API calls.

You can use personal access tokens as a safer alternative method of authentication. See [Using personal access tokens](#).

# Using the miniOrange Plugin

This tutorial explains how to use [miniOrange](#) to connect casdoor and Jira.

[Casdoor](#) can use the OIDC protocol as the IDP to connect various applications. You can refer to this [Jira](#) tutorial for more information.

The following are some important names in the configuration:

`CASDOOR_HOSTNAME`: The domain name or IP where the Casdoor server is deployed.

`Jira_HOSTNAME`: The domain name or IP where Jira is deployed.

## Step 1: Deploy Casdoor and Jira

Firstly, deploy [Casdoor](#) and [Jira](#).

After successful deployment, make sure:

1. Set Jira URL (Plans → Administration → System → General Configuration) to `Jira_HOSTNAME`.

The screenshot shows the Jira System settings page under the 'General configuration' tab. The 'Base URL' field is highlighted with a red box and contains the value 'http://localhost:8080'. A red arrow points from this field to the placeholder 'Jira\_HOSTNAME' in the 'Email from' field. Other settings visible include 'Title' (JIRA), 'Mode' (Private), 'Maximum Authentication Attempts Allowed' (3), 'CAPTCHA on signup' (OFF), and 'Introduction' (Internationalization).

2. Casdoor 可以正常登录使用。
3. You can set `CASDOOR_HOSTNAME` to `http://localhost:8000` when deploying Casdoor in `prod` mode. 详见 [生产模式](#)。

## Step 2: Configure Casdoor Application and Jira

1. Create a new Casdoor application or use an existing one.
2. Install the [miniOrange](#) app to support OAuth. You can find this app in Plans->Administration->Find new apps->search

The screenshot shows the Atlassian Marketplace for Jira interface. A red arrow points to the 'Manage apps' tab at the top. Another red arrow points to the 'Find new apps' button in the 'ATLASSIAN MARKETPLACE' section. A third red arrow points to the 'Oauth' filter button at the bottom left of the search results. The search results list the 'mO Jira OAuth SSO, Jira OpenID Connect SSO, Jira OIDC SSO' app by miniOrange. The app card includes a logo, developer information ('miniOrange • Supported by vendor • Data Center'), ratings ('★★★★ (56)'), installation count ('607 installations'), and purchase options ('Free trial', 'Buy now'). A red arrow also points to the app name 'mO Jira OAuth SSO, Jira OpenID Connect SSO, Jira OIDC SSO' in the list.

3. Set **Selected Application** to Custom OpenId.

4. Find the redirect URL:

miniOrange OAuth Configuration  
Manage apps Ask Us On Forum Frequently Asked Questions

Back to common setting OAuth/OIDC Configurations

Callback URL: http://localhost:8080/plugins/servlet/oauth/callback

5. Add the redirect URL:

Client ID: 514e09591ee5554b16fe  
Client secret: e7f05b14a68fb23e526f08515aefb73bbab7814a  
Cert: cert-built-in  
Redirect URLs: http://localhost:8080/plugins/servlet/oauth/callback

6. Configure the app as follows:

Selected Application: **Custom OpenId** Import Details

Provider ID: **5c881c25-2e02-42c9-af06-0a71e0beb516**

Custom App Name: casdoor

Client Id: \* 514e09591ee5554b16fe

Client Secret: \* e7f05b14a68fb23e526f08515aefb73bbab7814a

Scope: \* openid email profile address phone offline\_access

Authorize Endpoint: http://localhost:8000/login/oauth/authorize

Access Token Endpoint: http://localhost:8000/api/login/oauth/access\_token

Logout Endpoint: Enter the Logout Endpoint URL

Enter the Logout endpoint of your OAuth/OpenID Provider. Leave blank if Logout endpoint not supported by provider.  
e.g. If Keycloak Logout endpoint is configured with {hostname}/auth/realms/{realm-name}/protocol/openid-connect/logout too.

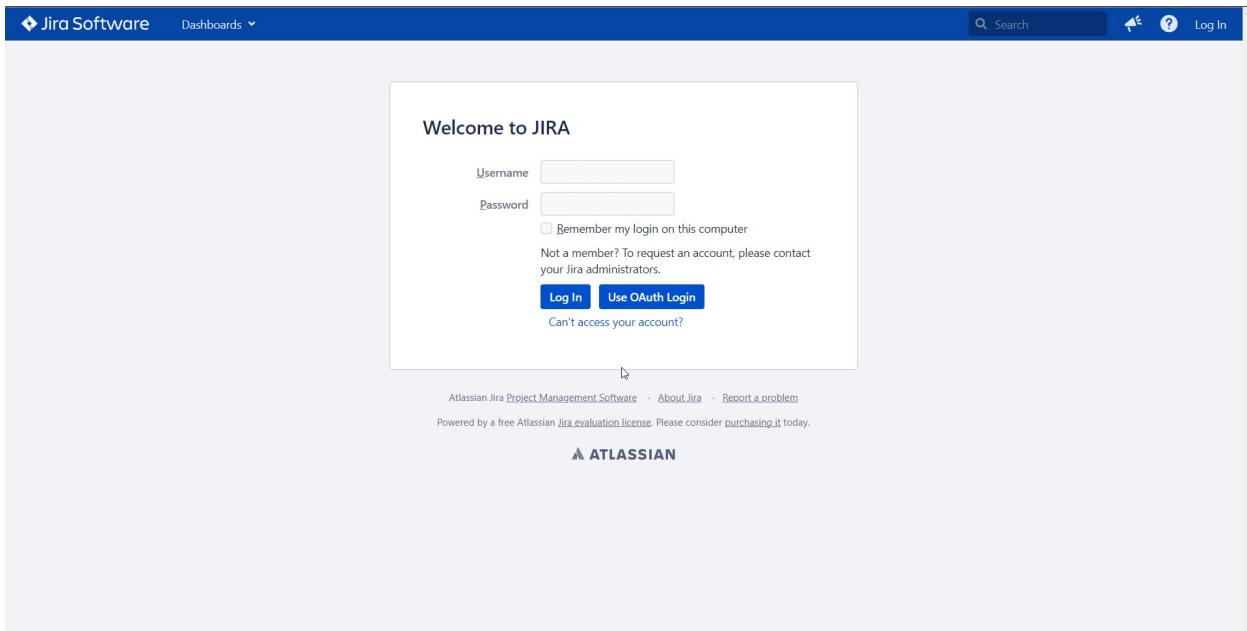
Save Test Configuration

- Token server URL: http://**CASDOOR\_HOSTNAME**/api/login/oauth/**access\_token**

- Authorization server URL: http://**CASDOOR\_HOSTNAME**/login/oauth/authorize
- UserInfo server URL: http://**CASDOOR\_HOSTNAME**/api/get-account
- Scopes: address phone openid profile offline\_access email

Open your favorite browser and visit: http://**CASDOOR\_HOSTNAME**/.well-known/openid-configuration. You will see the OIDC configuration of Casdoor.

Log out of Jira and test SSO.





# Connecting Applications with OIDC Protocol - Confluence

Casdoor can use OIDC protocol as an IDP to connect various applications. In this guide, we will use Confluence as an example to demonstrate how to use OIDC to connect your applications.

To start, make sure you have deployed Casdoor and Confluence successfully. Here are a few configuration names you need to remember:

- `CASDOOR_HOSTNAME`: Domain name or IP where Casdoor server is deployed.
- `Confluence_HOSTNAME`: Domain name or IP where Confluence is deployed.

## Step 1: Deploy Casdoor and Confluence

First, deploy Casdoor and Confluence.

After successful deployment, ensure the following:

1. Set Confluence URL to `Confluence_HOSTNAME`.

The screenshot shows the 'General Configuration' section of the Confluence administration interface. On the left, a sidebar lists various configuration options. The 'General Configuration' option is highlighted with a blue arrow pointing to it. The main content area is titled 'General Configuration' and contains a 'Site Configuration' section. Under 'Site Configuration', there is a 'Server Base URL' field set to 'http://localhost:8090'. A blue arrow points to this field.

2. Casdoor can be logged in and used normally.
3. You can set `CASDOOR_HOSTNAME` to `http://localhost:8000` if you deploy Casdoor in `prod` mode. Refer to the [production mode](#) for more details.

## Step 2: Configure Casdoor application

1. Create a new Casdoor application or use an existing one.
2. Find a redirect URL:

The screenshot shows the 'OAuth/OIDC Configurations' page. On the left, a sidebar lists 'OAuth/OIDC Configurations'. The main content area has a 'Callback URL' field set to 'http://localhost:8090/plugins/servlet/oauth/callback'. A blue arrow points to this field.

3. Add the redirect URL to the application:

The screenshot shows the 'OAuth/OIDC Configurations' page. It includes fields for 'Client ID' (01ae4bd048734ca2dea), 'Client secret' (f26a4115725867b7bb7b668c81e1f8f7fae1544d), 'Cert' (cert-built-in), and 'Redirect URLs'. The 'Redirect URLs' section has an 'Add' button and a field containing 'http://localhost:8090/plugins/servlet/oauth/callback'. Blue arrows point to each of these four fields.

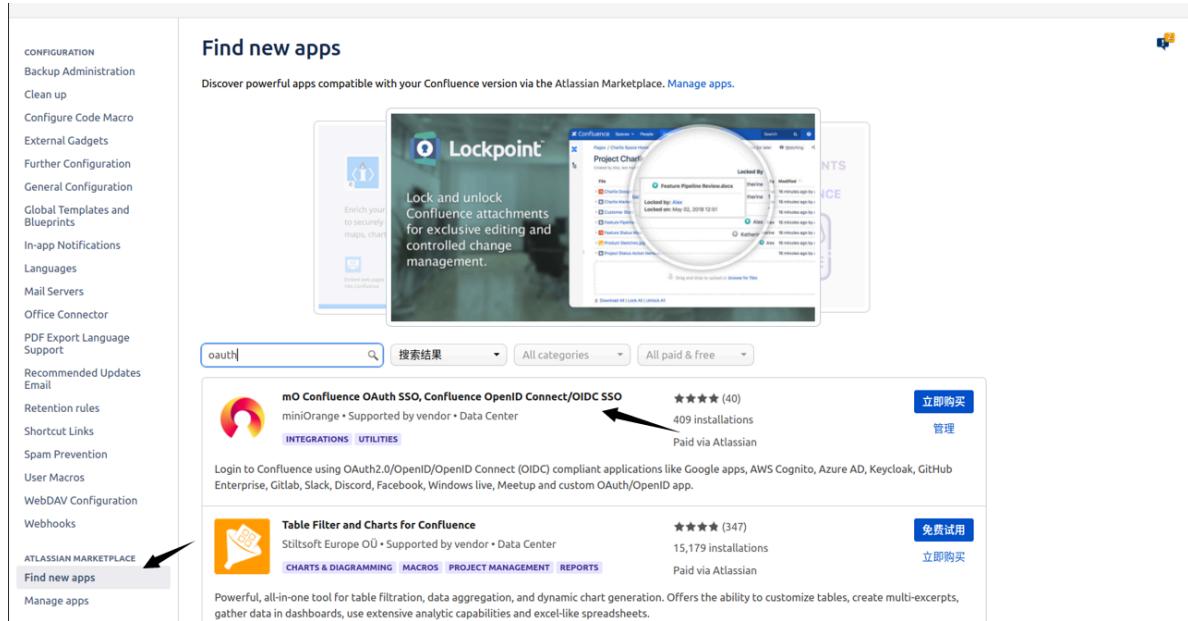
4. Add the desired provider and configure other settings accordingly.

On the application settings page, you will find two values: `Client ID` and `Client Secret`. We will need these in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration` to see the OIDC configuration of Casdoor.

## Step 3: Configure Confluence

1. Install the [miniOrange](#) app to support OAuth. You can find this app in:



The screenshot shows the Atlassian Marketplace interface. On the left, there's a sidebar with various configuration options like 'Backup Administration', 'External Gadgets', and 'ATLASSIAN MARKETPLACE'. The 'ATLASSIAN MARKETPLACE' option is highlighted with a red arrow. In the main area, there's a search bar with 'oauth' typed in. Below it, there are several app listings. One listing for 'mO Confluence OAuth SSO, Confluence OpenID Connect/OIDC SSO' by miniOrange is highlighted with a black arrow. This listing includes a star rating of 4.0, 409 installations, and a 'Buy Now' button. Other visible apps include 'Lockpoint' and 'Table Filter and Charts for Confluence'.

2. Configure the app:

Selected Application:	<b>Custom OpenId</b>	<b>Import Details</b>	<b>Setup Guide</b>
Provider ID: <b>4f6b30c1-eba8-4b89-ac02-4a4b7a137b97</b>			
Custom App Name: <input type="text" value="Casdoor SSO"/>			
Client Id: <input type="text" value="014ae4bd048734ca2dea"/>			
Client Secret: <input type="text" value="f26a4115725867b7bb7b668c81e1f8f7fae1544d"/>			
Scope: <input type="text" value="openid profile email"/>			
Authorize Endpoint: <input type="text" value="https://door.casdoor.com/login/oauth/authorize"/>			
Access Token Endpoint: <input type="text" value="https://door.casdoor.com/api/login/oauth/access_token"/>			
Logout Endpoint: <input type="text" value="Enter the Logout Endpoint URL"/> <p style="font-size: small; margin-top: 5px;">           Enter the Logout endpoint of your OAuth/OpenID Provider. Leave blank if Logout endpoint not supported by provider.            e.g. If Keycloak Logout endpoint is configured with {hostname}/auth/realm/{realm-name}/{protocol}/openid-connect/logout URL then on!         </p>			

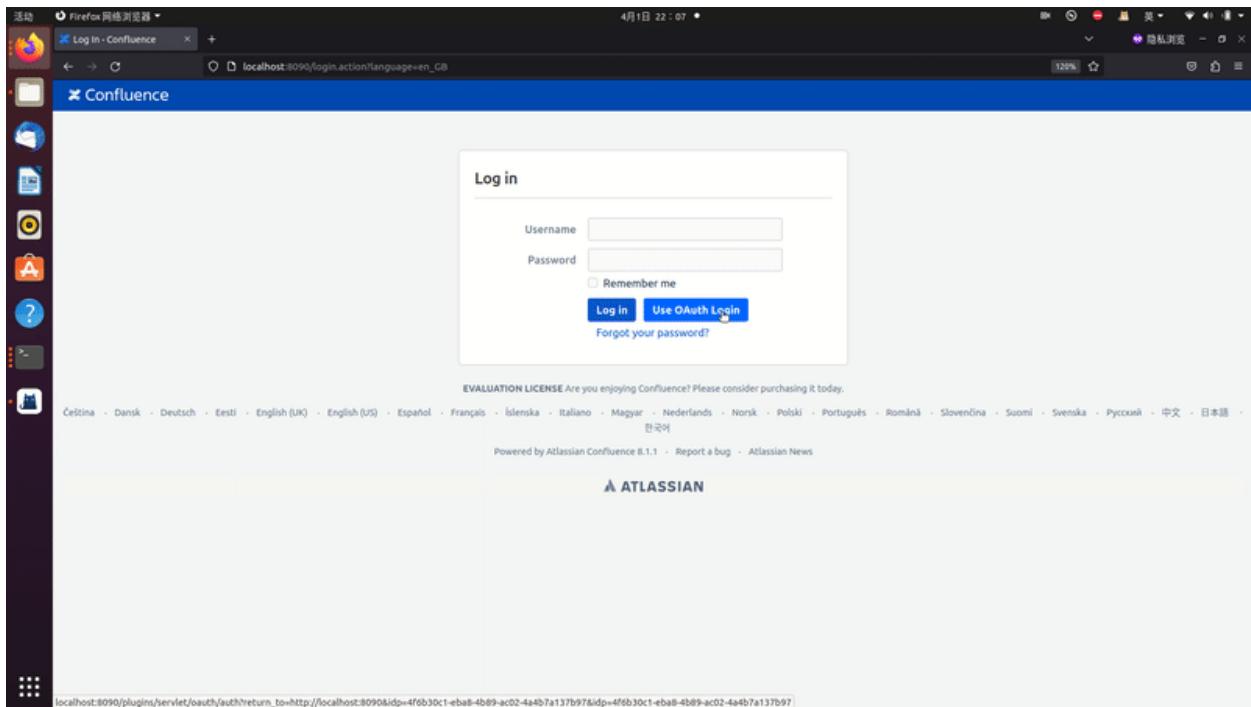
3. Set **Selected Application** to Custom OpenID.
4. Retrieve the Client ID and Client Secret from the Casdoor application page.

Configure the following settings for Confluence:

- **Token server URL**: `http://CASDOOR_HOSTNAME/api/login/oauth/access_token`
- **Authorization server URL**: `http://CASDOOR_HOSTNAME/login/oauth/authorize`
- **User Info server URL**: `http://CASDOOR_HOSTNAME/api/get-account`
- **Scopes**: `address phone openid profile offline_access email`

You can configure more advanced authorization settings later. For now, check if OpenID actually works.

Log out of Confluence and test SSO:



# RuoYi

Casdoor can be easily integrated with RuoYi-cloud.

## Step 1: Deploy Casdoor

First, deploy Casdoor.

您可以参考Casdoor 官方文档 [Server Installation](#)。

After successful deployment, ensure the following:

- The Casdoor server is running at <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001> to access the Casdoor login page.
- Test the login functionality by entering `admin` and `123`.

Next, you can quickly implement a Casdoor-based login page in your own app following these steps.

## Step 2: Configure Casdoor

To configure Casdoor, please follow these steps:

1. Open Casdoor in a browser by clicking [here](#). It is recommended to use a different browser than your development browser.
2. Configure an organization, an application, and the Synchronizer in Casdoor.

You can find detailed instructions on how to do this [here](#).

Here are some additional points to keep in mind:

1. When editing the syncer, make sure to check the table columns:

Table columns 	Add	Column name	Column type	Casdoor column	Is hashed	Action
		user_id	integer	Id	<input checked="" type="checkbox"/>	  
		dept_id	integer	Affiliation	<input checked="" type="checkbox"/>	  
		user_name	string	Name	<input checked="" type="checkbox"/>	  
		nick_name	string	DisplayName	<input checked="" type="checkbox"/>	  
		user_type	string	Type	<input checked="" type="checkbox"/>	  
		email	string	Email	<input checked="" type="checkbox"/>	  
		phonenumer	string	Phone	<input checked="" type="checkbox"/>	  
		sex	string	Gender	<input checked="" type="checkbox"/>	  
		avatar	string	Avatar	<input checked="" type="checkbox"/>	  
		password	string	Password	<input checked="" type="checkbox"/>	  
		del_flag	string	IsDeleted	<input checked="" type="checkbox"/>	  
		login_ip	string	CreatedIp	<input checked="" type="checkbox"/>	  
		create_time	string	CreatedTime	<input checked="" type="checkbox"/>	  
		password	string	Password	<input checked="" type="checkbox"/>	  

2. When editing the organization, make sure to select the correct password type:

Password type  :

3. Lastly, ensure that you have enabled soft deletion.

Please make sure to follow these instructions carefully to properly configure Casdoor.

## 步骤3. 前端改造

### 3.1 Jump to Casdoor's login page

We can use a front-end SDK, taking vue-sdk as an example here. After you initialize vue-sdk, you can obtain the Casdoor login page URL by using the `getSignInUrl()` function.

You can link it in the way you prefer, and feel free to delete any original code from Ruoyi-Cloud that is no longer necessary, such as the original account and password el-input.

## 3.2 Accept the code and state returned by Casdoor

After successfully logging in through Casdoor, Casdoor sends the code and state to the page we set up. We can retrieve the code and state using the create() function.

```
created() {
    let url = window.document.location.href; // get URL
    let u = new URL(url);
    this.loginForm.code = u.searchParams.get('code'); // get code
    and state
    this.loginForm.state = u.searchParams.get('state');
    if (this.loginForm.code != null && this.loginForm.state != null) { // if code and state are not null, execute handleLogin
        this.handleLogin();
    }
}
```

For RuoYi-Cloud, we simply modify its original method of sending the account and password to send the code and state instead. Therefore, the change is only in what is sent to the backend, in relation to the original login.

# Step 4: Refactor your back-end

## 4.1 Accept the code and state returned by the front-end

```
@PostMapping("login")
public R<?> callback(@RequestBody CodeBody code) {
    String token =
        casdoorAuthService.getOAuthToken(code.getCode(), code.getState());
    CasdoorUser casdoorUser =
        casdoorAuthService.parseJwtToken(token);
    if (casdoorUser.getName() != null) {
        String casdoorUserName = casdoorUser.getName();
        if (sysLoginService.getUserByCasdoorName(casdoorUserName)
            == null) {
            sysLoginService.casdoorRegister(casdoorUserName); // Add this user to the database if they don't exist
        }
    }
    LoginUser userInfo =
        sysLoginService.casdoorLogin(casdoorUser.getName()); // Get the user's information from the database
    return R.ok(tokenService.createToken(userInfo));
}
```

In this method, we are using the casdoor-SpringBoot-sdk method and making slight modifications to the RuoYi-Cloud method.

For example, the RuoYi-Cloud original method registers an account with a password. I have changed it to register an account using the `casdoorRegister` method.

I have also added a method `getUserByCasdoorName` to check if the account exists, and changed the method `executeUserInfo` to `executeWithAccount` to reflect this change.

This is an easy modification, as we only need to remove the part that checks the password.

## Step 5: Summary

### 5.1 Front-end

- The existing login and register pages need to be removed.
- Additionally, the front-end needs to accept code and state parameters and send them to the back-end.

### 5.2 Back-end

The RuoYi back-end already has a well-implemented login and registration function. We just need to make some minor modifications, which makes the process highly convenient.

## Step 6: Detailed Steps

1. Deploy and configure Casdoor. Be sure to select the bcrypt password type for the organization, as RuoYi-Cloud also uses bcrypt for passwords.
2. Use Casdoor syncers to copy database users to your Casdoor organization. This will import the original accounts into Casdoor.
3. After deploying Casdoor, make changes to the front-end. Disable the RuoYi

check code.

```
// checkcode switch  
captchaEnabled: false,  
// register switch  
register: true,
```

Note that the RuoYi-Cloud captcha needs to be disabled in Nacos again. Also, the RuoYi-Cloud registration function needs to be enabled by setting `sys.account.registerUser` to `true`.

4. Add a button for users to log in with Casdoor, and modify the data's `loginForm`.

```
<div><a href="http://localhost:7001/login/oauth/authorize?client_id=d509b6b3edc8a3d4cce9&response_type=code&redirect_uri=http%3A%2F%2Flocalhost%3D8080%2Fcasdoor">casdoor</a>
```

```
        loginForm: {  
            code: "",  
            state: ""  
        },
```

Here, I have written the URL, but you can obtain it using the Casdoor-Vue-SDK or Casdoor-SpringBoot-SDK.

5. Since we are no longer using the original login method, delete the cookie and checkcode methods.

The new `created` function should look like this:

```
created() {  
    let url = window.document.location.href; // Get the URL  
    let u = new URL(url);  
    this.loginForm.code = u.searchParams.get('code'); // Get
```

6. In fact, we only need to change the parameter we send to the back-end and delete the unnecessary functions. No other changes are necessary.

```
handleLogin() {
  console.log("进入handleLogin")
  this.$store.dispatch("Login", this.loginForm).then(() => {
    this.$router.push({ path: this.redirect || "/" }).catch(()=>{});
  }).catch(() => {
    this.loading = false;
    if (this.captchaEnabled) {
      this.getCode();
      console.log(this.getCode)
    }
  });
}
```

```
Login({ commit }, userInfo) {
  const code = userInfo.code
  const state = userInfo.state
  return new Promise((resolve, reject) => {
    login(code, state).then(res => {
      console.log("LOGIN")
      let data = res.data
      setToken(data.access_token)
      commit('SET_TOKEN', data.access_token)
      setExpiresIn(data.expires_in)
      commit('SET_EXPIRES_IN', data.expires_in)
      resolve()
    }).catch(error => {
      reject(error)
    })
  })
},
```

```
export function login(code, state) {
  return request({
    url: '/auth/login',
    headers: {
      isToken: false
    },
    method: 'post',
    data: {code, state}
  })
}
```

7. Import the required dependency in the back-end.

pom.xml

```
<dependency>
  <groupId>org.casbin</groupId>
  <artifactId>casdoor-spring-boot-starter</artifactId>
  <version>1.2.0</version>
</dependency>
```

You also need to configure Casdoor in the resource file.

8. Define the callback function as the redirect function. Make changes to some methods in `sysLoginService`. Delete the password check step because it is no longer needed.

```
@PostMapping("login")
public R<?> callback(@RequestBody CodeBody code) {
  // Define a CodeBody entity with code and state
  String token =
```

9. Add new methods to `SysLoginService`.

```
public LoginUser casdoorLogin(String username) {
    R<LoginUser> userResult =
remoteUserService.getUserInfo(username,
SecurityConstants.INNER);
    // Execute the user
    if (R.FAIL == userResult.getCode()) {
        throw new ServiceException(userResult.getMsg());
    }

    if (StringUtils.isNull(userResult) ||
StringUtils.isNull(userResult.getData())) {
        recordLogService.recordLogininfor(username,
Constants.LOGIN_FAIL, "This user does not exist");
        throw new ServiceException("User " + username + " does
not exist");
    }
    LoginUser userInfo = userResult.getData();
    SysUser user = userResult.getData().getSysUser();
    if
(UserStatus.DELETED.getCode().equals(user.getDelFlag())) {
        recordLogService.recordLogininfor(username,
Constants.LOGIN_FAIL, "Sorry, your account has been deleted");
        throw new ServiceException("Sorry, your account " +
username + " has been deleted");
    }
    if (UserStatus.DISABLE.getCode().equals(user.getStatus()))
{
        recordLogService.recordLogininfor(username,
Constants.LOGIN_FAIL, "Your account is disabled. Please
contact the administrator");
        throw new ServiceException("Sorry, your account " +
username + " is disabled");
    }
    recordLogService.recordLogininfor(username,
Constants.LOGIN_SUCCESS, "Login successful");
```

```
public String getUserByCasdoorName(String casdoorUsername) {
    R<LoginUser> userResult =
remoteUserService.getUserInfo(casdoorUsername,
SecurityConstants.INNER);
    if (StringUtils.isNull(userResult) ||
StringUtils.isNull(userResult.getData())) {
        // If the user is not in the Ruoyi-Cloud database but
exists in Casdoor, create the user in the database
        return null;
    }
    String username =
userResult.getData().getSysUser().getUserName();
    return username;
}
```

```
public void casdoorRegister(String username) {
    if (StringUtils.isAnyBlank(username)) {
        throw new ServiceException("User must provide a
username");
    }
    SysUser sysUser = new SysUser();
    sysUser.setUserName(username);
    sysUser.setNickName(username);
    R<?> registerResult =
remoteUserService.registerUserInfo(sysUser,
SecurityConstants.INNER);
    System.out.println(registerResult);
    if (R.FAIL == registerResult.getCode()) {
        throw new ServiceException(registerResult.getMsg());
    }
    recordLogService.recordLoginInfo(username,
Constants.REGISTER, "Registration successful");
}
```

# Pulsar Manager

Casdoor can easily connect to Pulsar Manager.

The code for connecting Casdoor has already been added to Pulsar Manager, so we just need to configure the `application.yml` file in the back-end and enable the front-end switch.

## Step 1: Deploy Casdoor

First, deploy Casdoor.

You can refer to the official Casdoor documentation for the [Server Installation](#).

After a successful deployment, ensure the following:

- The Casdoor server is running successfully at <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>. You should see the login page of Casdoor.
- Test the login functionality by entering `admin` and `123`.

Now, you can quickly implement a Casdoor-based login page in your own app using the following steps.

## Step 2: Configure Casdoor

To configure Casdoor, refer to [Casdoor](#) (it is recommended to use a different browser than your development browser).

You should also configure an organization and an application. You can refer to [Casdoor](#) for detailed instructions.

## Step 2.1: Create an organization

Edit Organization Save Save & Exit

Name ⓘ:	pulsar
Display name ⓘ:	pulsar
Favicon ⓘ:	URL ⓘ: <a href="https://cdn.casbin.org/img/favicon.png">https://cdn.casbin.org/img/favicon.png</a>
Preview:	
Website URL ⓘ:	<a href="http://localhost:9527/#/login?redirect=%2F">http://localhost:9527/#/login?redirect=%2F</a>
Password type ⓘ:	plain
Password salt ⓘ:	
Phone prefix ⓘ:	+ 86

## Step 2.2: Create an application

Name ⓘ: app-pulsar  
Display name ⓘ: app-pulsar  
Logo ⓘ: URL ⓘ: [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home ⓘ:	<a href="#">/</a>				
Description ⓘ:					
Organization ⓘ:	pulsar				
Client ID ⓘ:	6ba06c1e1a30929fdda7				
Client secret ⓘ:	df92bbf913225ebbae9af7ba8d41fe19507eb079				
Cert ⓘ:	cert-built-in				
Redirect URLs ⓘ:	<table border="1"><thead><tr><th>Action</th><th>Add</th></tr></thead><tbody><tr><td>Redirect URL:</td><td><a href="http://localhost:9527/callback">http://localhost:9527/callback</a></td></tr></tbody></table>	Action	Add	Redirect URL:	<a href="http://localhost:9527/callback">http://localhost:9527/callback</a>
Action	Add				
Redirect URL:	<a href="http://localhost:9527/callback">http://localhost:9527/callback</a>				

## Step 3: Enable the Pulsar Manager front-

## end switch

Enable this switch to send code and state to the back-end.

You can find the switch on line 80 of `pulsar-manager/front-end/src/router/index.js`.

```
- // mode: 'history', // require service support
+ mode: 'history', // require service support
```

## Step 4: Configure the back-end code

Configure Casdoor's settings in the `application.properties` file, which can be found on line 154 of `pulsar-manager/src/main/resources/application.properties`.

```
casdoor.endpoint = http://localhost:8000
casdoor.clientId = <client id from previous step>
casdoor.clientSecret = <client secret from previous step>
casdoor.certificate = <client certificate from previous step>
casdoor.organizationName = pulsar
casdoor.applicationName = app-pulsar
```

# 在 ShenYu 中使用 Casdoor

ShenYu has a Casdoor plugin to enable the use of Casdoor.

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed. You can refer to the official Casdoor documentation for [Server Installation](#).

After a successful deployment, please ensure that:

- The Casdoor server is running on <http://localhost:8000>.
- Open your preferred browser and visit <http://localhost:7001> to see the Casdoor login page.
- Login functionality is working fine by inputting `admin` and `123`.

After following the above steps, you can quickly implement a Casdoor-based login page in your app with the following steps.

## Step 2: Configure the Casdoor application

1. Create a new Casdoor application or use an existing one
2. Add your redirect URL

The screenshot shows the Casdoor application configuration interface. Key fields include:

- Name:** app-test (highlighted with a red arrow)
- Display name:** app-test
- Logo:** https://cdn.casbin.org/img/casdoor-logo\_118x256.png
- Preview:** Shows the Casdoor logo and the word "Casdoor".
- Home:** (empty)
- Description:** (empty)
- Organization:** built-in (highlighted with a red arrow)
- Client ID:** 663a84154e73d1fb156a (highlighted with a red arrow)
- Client secret:** 84209d412a338a42b789c05a3446e623cb7262d (highlighted with a red arrow)
- Cert:** cert-built-in
- Redirect URLs:**
  - Add (button)
  - Redirect URL: http://localhost:9195/http/hi (highlighted with a red arrow)
  - Redirect URL: http://localhost:9195/http/hello

### 3. On the certificate editing page, you can view your **Certificate**

**Certificate:**

Copy certificate
Download certificate

```
-----BEGIN CERTIFICATE-----
MIIE+TCCAUgBgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTABBgNVBAoTFENh
c2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENlcnQwHhcNMjEx
MDE1MDgxMTUyWhcNNDEXMDE1MDgxMTUyWjA2M0R0wGwYDVQQKExRDYXNkb29yIe9y
Z2FuaxphdGlvbjEVMBMGA1UEAxMMQ2FzZG9vcibDZXJ0MIIiCjANBgkqhkiG9w0B
AQEFAOCAG8AMiCCgKCAGEAisInpb5E1/ymf01RfSDSSE8I7y+lw+RJi74e5ej
rq4b8zMYk7HeHCyZr/hmNewEVXnhXu1P0mBeQ5ypp/QGo8vgEmjaETNmzkl1NjOQ
CjCYwUrasO/f/Mnl1C0j13vx6mV1kHZjSrKsMhYY1vaxTEP3+VB8Hjg3MHFWrb07
uvFMCJe5W8+0rKErZCKTR8+9VB3janeBz/zQePFVh79bfZate/hLirPK0Go9P1g
OvwloC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDFE7mTstRS8b/wUjNCUBD
PTSLVjc04WIIIf6Nkfx0Z7KvmbPstSj+btvqcsvRAGtvsB9h62Kptjs1Yn7GAuo
I3qt/4zoKbiURYxkQJXlvwCQsEftUuk5ew5zuPSIDRLoLByQTLbx0jLAFNfW3g/
pzSDjgd/60d6HTmvbZni4SmjdyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMc0fx4Gu
hRsnlsmkmUDeylZ9aBL9oj11YEQfM2JZEq+RvtUx+wB4y8K/tD1bcY+lfnG5rBpw
IDpS262boq4SRsvb3Z7bB0w4ZxvOfj/1VLoRftPbLifobhfr/AeZMHpiKOXvfz4
yE+hqzi68wdF0VR9xYc/RbSAf7323OsjYnjjEglnUtRohnRgCpjlk/Mt2Kt84Kb0
wn8CAwEAAaAMQMA4wDAYDVR0TAQH/BAlwADANBgkqhkiG9w0BAQsFAOCAGEAn2lf
DKkLX+F1vKRO/5gJ+Plr8P5NKuQkmwH97b8CS2gS1phDyNgic4/LSdzuf4Awe6ve
C06lVdWSlis8UPUPdjmt2uMPSNjwLxG3QsrimMURNlwFLTfRem/heJe0Zgur9J1M
8haawdSdjH2RgmFoDeE2r8NVRfhbR8KnCO1ddTJKuS1N0/irHz21W4jt4rxzCvl
2nR42Fybap3O/g2jXMhNNRowZmNjgpsF7XVENCSuFO1jTywLaqjuXCg54lL7XVLG
omKNNNcc8h1FCelj/nbbGMhodnFWKDTsJcbNmcoPNHo6ixzqMy/Hqc+mWv7maAG
Jtevs3qgMZ8F9Qzr3HpUc6R3ZYWDY/xxPisuKftOPZgtH979XC4mdf0WPnOBLLqL
2DJ1zaBmjIGolvb7XNVKcUfdXYw85ZTQ5b9cli4e+6bmwyQqlwt+Ati/uFEV
XzCj70B4IALX6xau1kLepV9O1GERizYrz5P9NJNA7Ko05AVMp9w0DQTkt+LbXnZE
HHnWk8xHQKF9sR7YBPGls/Ac6tvivUa150gj/8dLRZ/veyFfGo2yZsl+hKVU5
nCCJHBcAyFnm1hdvdwEdH3jDbjNB6ciotJzrf/3VYaIWSalADosHAgMwfXuWP+h
8XKXmzlxuHbTMQytZPDgsps5aK+S4Q9wb8RRAYo=
-----END CERTIFICATE-----
```

## Step 3: Use the Casdoor plugin in

# ShenYu

## 1. Configure the Casdoor plugin in ShenYu

## Plugin

X

\* Plugin: casdoor

### casdoor Configuration

\* application-name: app-test

\* certificate: -----BEGIN CERTIFICATE-----\nMIIE+TCCAuGgAwI

\* client\_id: 6e3a84154e73d1fb156a

\* client\_secrect: a4209d412a33a842b7a9c05a3446e623cbb7262d

\* casdoor endpoint: http://localhost:8000

\* organization-name: test

\* Role: Authentication

\* Sort: 40

Status:

Cancel

Sure

Note: As ShenYu only has a single line input box, `\n` must be added in every line

of certificate.

## Certificate ? :

[Copy certificate](#)

[Download certificate](#)

-----BEGIN CERTIFICATE-----\nMIIE+TCCAUgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENH\nn\nc2Rvb3lgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENlcnQwHhcNMjEx\nn\nMDE1MDgxMTUyWhcNNDExMDE1MDgxMTUyWja2MR0wGwYDVQQKErDYXNkb29yIe9y\nn\nZ2FuaXphdGlvbjEVMBMGA1UEAxMMQ2FzZG9vcBDZXJ0MIIICjANBgkqhkiG9w0B\nn\nAQEFAAOCAg8AMIICgKCAgEAstnpb5E1/yM0f1RfSDSSE8IR7y+lw+RJi74e5ej\nn\nrq4b8zMjYk7HeHCyZr/hmNEwEVXnhXu1P0mBeQ5yp/QGo8vgEmjAETNmzkl1NjOQ\nn\nCjCYwUrasO/f/Mnl1C0j13vx6mV1kHZjsRsKmhYY1vaxTEP3+VB8Hjg3MHFWrb07\nn\nuvFMCJe5W8+0rKErZCKTR8+9VB3janeBz//zQePFVh79bfZate/hLirPK0Go9P1g\nn\nOvwloC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDFE7mTstRSBb/wUjNCUBD\nn\nPTSLVjc04WIISf6Nkf0Z7KvmbPstSj+btcqvRAgtvdsB9h62Kptjs1Yn7GAuo\nn\nl3qt/4zoKbiURYxkQJXlvwCQsEftUuk5ew5zuPSIDRLoLbyQLtbx0JqlAFNfW3g\nn\npzSDjgd/60d6HTmvbZni4SmjdyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMc0fx4Gu\nn\nhRsnLsmkmUDeylZ9aBL9oj11YEQfm2JZEq+RvtUx+wB4y8K/tD1bcY+lfnG5rBpw\nn\nIDpS262boq4SRsvb3Z7bB0w4ZxvOfJ/1VLoRftjPbLlf0bhfr/AeZMHpIKOXvfz4\nn\nyE+hqzi68wdF0VR9xYc/RbSAf7323OsjYnjjEgInUtRohnRgCpjlk/Mt2Kt84Kb0\nn\nwn8CAwEAAsMQMA4wDAYDVR0TAQH/BAlwADANBgkqhkiG9w0BAQsFAAOCAgEAn2If\nn\nDKkLX+F1vKRO/5gJ+Plr8P5NKuQkmwH97b8CS2gS1phDyNgIc4/LSdzuf4Awe6ve\nn\nC06IVdWSlis8UPUPdmT2uMPNSJwLxG3QsrimMURNwFILTfRem/heJe0ZgurJ1M\nn\n8haawdSdjH2RgmFoDeE2r8NVRfhbR8KnCO1ddTJKuS1N0/irHz21W4jt4rxzCv\nn\n2nR42Fybap3O/g2JXMHNNROwZmNjgpsF7XVENCSuFO1jTywLaajuXCg54IL7XVLG\nn\nomKNNNcc8h1FcEkj/nnbGMhodnFWKDTsJcbNmOPNHo6ixzqMy/Hqc+mWYv7maAG\nn\nJtevs3qgMZ8F9Qzr3HpUc6R3ZYYWDY/xxPisufktpZGtH979XC4mdf0WPnOBQl\nn\n2DJ1zaBmjijGolvb7XNVKcUfDXYw85ZTZQ5b9cl4e+6bmyWqQltlw+Ati/uFEV\nn\nXzCj70B4IALX6xau1kLepV9O1GERizYRz5P9NJNA7Ko05AVMp9w0DQTkt+LbXnZE\nn\nHHnWKy8xHQKZF9sR7YBPGls/Ac6tviv5ua15Ogj/8dLRZ/veyFfGo2yZsl+hKVU5\nn\nnCCJHBcAyFnmlhdvdwEdH33jDBjNB6ciotJzf/3VYalWSalADosHAgMWfXuWP+h\nn\n8XXKXmzlxuHbTMQYtZPDgspS5aK+S4Q9wb8RRAYo=\n

here not need add \n

You can copy it and paste it into the certificate of the ShenYu Casdoor config.

You don't need to save it in the Casdoor certificate editing page, as it is only for copying.

## 2. Configure the ShenYu Casdoor plugin

The screenshot shows the Apache ShenYu Gateway Management System interface. On the left, there's a sidebar with 'Change Mode' (radio button), 'PluginList' (selected), and several sub-options: Mock, Cache, Authentication (Sign and Jwt), and Casdoor (which is highlighted in blue). The main area is titled 'Apache ShenYu Gateway Management System' and has tabs for 'SelectorList' and 'RulesList'. Under 'RulesList', there's a sub-tab 'Synchronous casdoor'. Below these are two search/filter boxes: 'Name' and 'RuleName', each with 'Query' and 'Add' buttons. A table lists rules with columns: Name, Open, Operation, RuleName, Open, UpdateTime, and Operation. One row is shown: '/http/' (Open, Modify Delete), '/http/hi' (Open), 2022-09-28 10:50:02.729, and Modify Delete. At the bottom, there are pagination controls: '1 < 1 > 12 / page'.

You can configure what you need for the Casdoor config.

## 3. Getting the service and using it

### 3.1 Directly visit the Web

The screenshot shows a browser window with the address bar containing 'localhost:9195/http/hi'. The page content displays a JSON response: {"code":401, "message":"Illegal authorization"}

### 3.2 Use Casdoor Login

localhost:7001/login/oauth/authorize?client\_id=6e3a84154e73d1fb156a&response\_type=code&redirect\_uri=http://localhost:9195/http/hi&scope=read&state=app-test A a



Continue with :



Or sign in with another account :

username, Email or phonePassword

Auto sign in [Forgot password?](#)

Sign In

No account? [sign up now](#)



localhost:9195/http/hi?code=822607b015cca2515b2b&state=app-test

hi! null! I'm Shenyu-Gateway System. Welcome!

### 3.3 Carry the token in Headers

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:9195/http/hi`. The Headers section is active, displaying the following configuration:

Key	Description
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.29.2
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
Authorization	eyJhbGciOiJSUzI1NlslmtpZCI6ImNlcnQtYn... <span style="color:red">Your token</span>

The Response tab shows the following JSON output:

```
1  hi! null! I'm Shenyu-Gateway System. Welcome!
```

### 3.4 Save name, ID and organization in Headers

This makes it easier to use them in the future.

# ShardingSphere

[shardingsphere-elasticjob-ui](#) has integrated Casdoor. You can use it after configuring it.

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed.

您可以参考Casdoor 官方文档 [Server Installation](#)。

After a successful deployment, make sure:

- The Casdoor server is successfully running on <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>. You will see the login page of Casdoor.
- Input `admin` and `123` to test if the login functionality is working fine.

Then, you can quickly implement a Casdoor-based login page in your own app with the following steps.

## Step 2: Configure Casdoor application and configure application in ShardingSphere

1. Create or use an existing Casdoor application

Name ⓘ: ShardingSphere 

Display name ⓘ: ShardingSphere

Logo ⓘ:  https://cdn.casbin.org/img/casdoor-logo\_1185x256.png

Preview:

Home ⓘ: 

Description ⓘ:

Organization ⓘ: ShardingSphere 

Client ID ⓘ: 3ed79fa530645fb0d3653 

Client secret ⓘ: 54633c82b7796a4332c6976864c6c16bc3b05556 

Cert ⓘ: cert-built-in 

Redirect URLs ⓘ:  Redirect URLs 

Redirect URL:  http://localhost:8080 

The RedirectURLs depend on the URL you need to redirect to. The selected data will be used in the next step.

2. On the certificate editing page, you can see your **Certificate**

Certificate [?](#) : Private

[Copy certificate](#) [Download certificate](#)

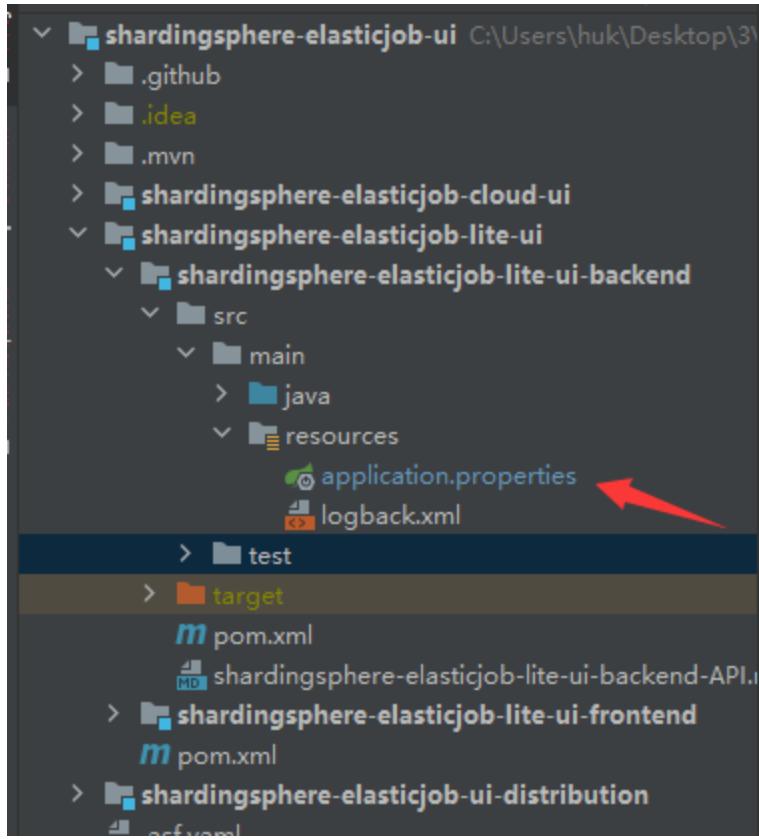
-----BEGIN CERTIFICATE-----

```
MIIIE+TCCAUgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENhc2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENIcnQwHhcNMjExMDE1MDgxMTUyWhcNNDEmDE1MDgxMTUyWjA2MR0wGwYDVQQKExRDYXNkb29yIE9yZ2FuaxphdGlvbjEVMBMGA1UEAxMMQ2FzZG9vcibDZXJ0MIICljanBqkhhkiG9w0BAQEFAOCAg8AMIICCgkCAgEAstnpb5E1/ym0f1RfSDSSE8IR7y+lw+Rjji74e5ejrq4b8zMYk7HeHCyZr/hmNEwEVXnhXu1P0mBeQ5yp/QGo8vgEmjAE TNmzk1NjOQCjCYwUraso/f/Mn1C0j13vx6mV1kHZjSrKsMhYY1vaxTEP3+VB8Hjg3MHFWrb07uvFMCje5W8+0rKErZCKTR8+9VB3janeBz/zQePFVh79bfZate/hLirPK0Go9P1gOvwloC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDFE7mTstRSBb/wUjNCUBDPTSLVjC04WIISf6Nkfx0Z7KvmbPstSj+btcqsvRAGtvdsB9h62Kptjs1Yn7GAuoI3qt+4zoKbiURYxkQJXlvwCQsEftUuk5ew5zuPSIDRLoLByQTLbx0jqLAFNfW3gpzSDjgd/60d6HTmvbZni4SmjdyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMc0fx4GuhRsnLsmkmUDeylZ9aBL9oj11YEQfM2JZEq+RVtUx+wB4y8K/tD1bcY+IfnG5rBpwIDpS262boq4SRsvb3Z7bB0w4ZxvOfj/1VLoRftjPbLif0bhfr/AeZMHpIKOXvfz4yE+hqzi68wdF0VR9xYc/RbSAf7323OsjYnjEgInUtRohnRgCpjlk/Mt2Kt84Kb0wn8CAwEAAaMQMA4wDAYDVR0TAQH/BAIwADANBgkqhkiG9wOBAQsFAAACAgEAn2IfDKkLX+F1vKRO+5gJ+Plr8P5NKuQkmwH97b8CS2gS1phDyNgIc4/Lsdzuf4Awe6veC06IVdWSlis8UPUPdjmt2uMPSNjwLxG3QsrimMURNwFILTfRem/heJe0Zgur9J1M8haawdSdJjH2RgmFoDeE2r8NVRfhbR8KnCO1ddTJKuS1N0/irHz21W4jt4rxzCv12nR42FybaP3O/g2JXMHNNR0wZmNjgpsF7XVENCSuFO1jTywLaqjuXCg54IL7XVLGomKNNNCc8h1FCeKj/nnbGMhodnFWKDTsJcbNmcoPNHo6ixzqMy/Hqc+mWYv7maAGJtevs3qgMZ8F9Qzr3HpUc6R3ZYWDY/xxPisuKftOPZgtH979XC4mdf0WPnOBLql2DJ1za8mjigJolvb7XNVKcUfdXYw85ZTZQ5b9cl4e+6bmyWqQltwt+Ati/uFEVXzCj70B4IALX6xau1kLEpV9O1GERizYRz5P9NJNA7KoO5AVMp9w0DQTkt+LbXnZEHHnWKy8xHQKZF9sR7YBPGLs/Ac6tviv5Ua15OgJ/8dLRZ/veyFfGo2yZsl+hKVU5nCCJHBcAyFn1hdvdwEdH33jDBjNB6ciotJZrf/3VYalWSalADosHAgMWfXuWP+h8XXXmzlkuHbTMQYtZPDgsps5aK+S4Q9wb8RRAYo=
```

-----END CERTIFICATE-----

### 3. Configure the application in ShardingSphere

First, we need to find the application.properties that we need to configure.



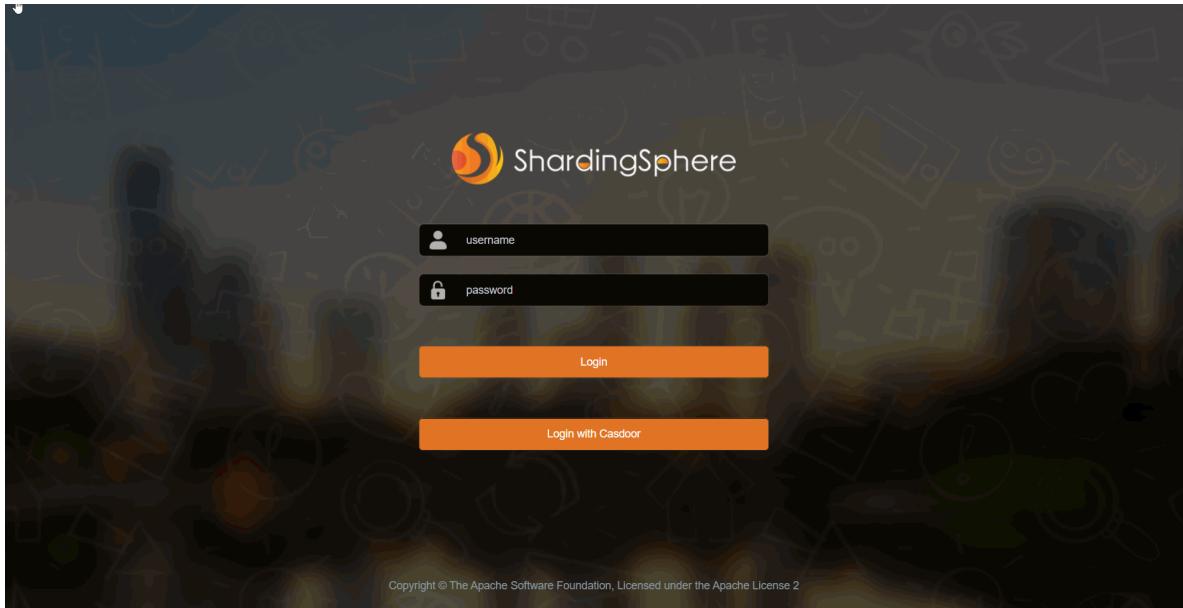
Next, we need to copy the data from the Casdoor application and paste it into the application.

```

casdoor.endpoint=http://localhost:7001
casdoor.client-id=3ed79fa530645fdb3653
casdoor.client-secret=54633c82b7796a4332c6976864c6c16bc3b05556
casdoor.certificate=\n
-----BEGIN CERTIFICATE-----\n\
MIIE+TCCAvGgAwIBAgIDAeJAMA0GCSqGSIb3DQEBCwUAMDYxHTAbBgNVBAoTFENh\n\
c2Rvb3IgT3JnYW5pemF0aW9uMRUwEwYDVQQDEwxDYXNkb29yIENlcnQwHhcNMjEx\n\
MDE1MDgxMTUyWhcNNDExMDE1MDgxMTUyWja2MR0wGwYDVQQKExRDYXNkb29yIE9y\n\
Z2FuaXphdGlvbjEVMBMGa1UEAxMMQ2FzZG9vcIBDZXJ0MIICIjANBqkqhkiG9w0B\n\
AQEFAOCAg8AMIIICgKCAGeAsInpb5E1/ym0f1RfSDSSE8IR7y+lw+RJjI74e5ej\n\
rq4b8zMYk7HeHCyZr/hmNEwEVXnhXu1P0mBeQ5ypp/QGo8vgEmjAE TNmzkI1Nj0Q\n\
CjCYwUras0/f/MnI1C0j13vx6mV1kHZjSrKsMhYY1vaxTEP3+VB8Hjg3MHFWrb07\n\
uvFMCJe5W8+0rKErZCKTR8+9VB3janeBz//zQePFVh79bFZate/hLirPK0Go9P1g\n\
OvwIoC1A3sarHTP4Qm/LQRt0rHqZFybdySpyWAQvhNaDFE7mtStRSBb/wUjNCUBD\n\
PTSLvjC04W1lsf6Nufx0Z7KvmbPstSj+btvccsvRAGtvdsB9h62Kptjs1Yn7GAuo\n\
I3qt/4zoKbiURYxkQJXIvwCQsEftUuk5ew5zuPS1DRLoLByQTLbx0JqLAFNfW3g/\n\
pzSDjgd/60d6HTmvbZni4SmidyFhXCDb1Kn7N+xTojnfaNkwep2REV+RMc0fx4Gu\n\
hRsnLsmkmUDeyIZ9aBL9oj1YEQfM2JZEq+RVtUx+wB4y8K/tD1bcY+IfnG5rBpw\n\
IDpS262boq4SRSpb3Z7bB0w4Zxv0fJ/1VL0RftjPbLIf0bhfr/AeZMHpIK0Xvfz4\n\
yE+hqzi68wdF0VR9xYc/RbSAf73230sjYnjjEgInUtRohnRgCpjIk/Mt2Kt84Kb0\n\
wn8CAwEAaMQMA4wDAYDVR0TAQH/BAIwADANBqkqhkiG9w0BAQsFAAOCAgEAn2lf\n\
DKkLX+F1vKRO/5gJ+Plr8P5NKuQkmwH97b8CS2gS1phDyNgIc4/LSdzuf4Awe6ve\n\
C061VdWSIis8UPUPdjmt2uMPSNjwLxG3QsrimMURNwFLLTfRem/heJe0Zqur9J1M\n\
8aaawdSdJjh2RgmFoDeE2r8NVRfhbR8KnC01ddTJKuS1N0/irHz21W4jt4rxzCvl\n\
2nR42Fybap30/g2JXMhNNR0wZmNjgpsF7XVENCSuF01jTywLaqjuXcg54IL7XVLG\n\
omKNNNcc8h1FCeKj/nnbGMhodnFWKDTsJcbNmcoPNHo6ixzqMy/Hqc+mWYv7maAG\n\
Jtevs3qgMZ8F9Qzr3HpUc6R3ZYWDY/xxPisuKftOPZgtH979XC4mdf0WPn0BLql\n\
2DJ1zaBmjigJolyb7XNVKcUfDXYw85TZQ5b9clI4e+6bmyWqQItlw+Ati/uFEV\n\
XzCj70B4lALX6xau1kEpV901GERizYRz5P9NJNA7Ko05AVMp9w0DQTkt+LbXnZE\n\
HHnWKy8xHQKZF9sR7YBPGls/Ac6tviv5Ua150gJ/8dLRZ/veyFFGo2yZsI+hKVU5\n\
nCCJHBcAyFnmlhdvdwEdH33jDBjNB6ciotJZrf/3VYaIWSalADosHAgMWfXuWP+h\n\
8XKXmzlxuHbTMQYtZPDgspS5aK+S4Q9wb8RRAYo=\n
-----END CERTIFICATE-----\n
casdoor.organization-name=ShardingSphere
casdoor.application-name=ShardingSphere

```

#### 4. Test it



# Apache IoTDB

Casdoor can easily connect to [Apache IoTDB](#).

The code for connecting Casdoor has already been added in [Apache IoTDB Web Workbench](#), so all we need to do is configure the application.yml file in the back-end and activate the front-end switch.

## Step 1: Deploy Casdoor

First, deploy Casdoor.

You can refer to the official Casdoor documentation for the [Server Installation](#).

After deploying successfully, ensure that:

- The Casdoor server is running successfully at <http://localhost:8000>.
- Open your preferred browser and visit <http://localhost:7001>, where you will see the Casdoor login page.
- Test the login functionality by entering `admin` and `123`.

With these steps completed, you can now quickly implement a Casdoor-based login page in your own application.

## Step 2: Configure Casdoor

To configure Casdoor, refer to [casdoor](#) (It is recommended not to use the same browser you are developing in to configure Casdoor's browser).

You should also create an organization and an application. Refer to [casdoor](#) for instructions.

## 2.1 Create an organization

Name ⓘ: IoTDB

Display name ⓘ: IoTDB

Favicon ⓘ: URL ⓘ: <https://cdn.casbin.org/img/favicon.png>

Preview: 

Website URL ⓘ: <https://door.casdoor.com>

Password type ⓘ: plain

Password salt ⓘ:

## 2.2 Create an application

Name ⓘ: app\_IoTDB

Display name ⓘ: app\_IoTDB

Logo ⓘ: URL ⓘ: [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home ⓘ: [/](#)

Description ⓘ:

Organization ⓘ: built-in

Client ID ⓘ: 6f561b83d119be3e1e3c

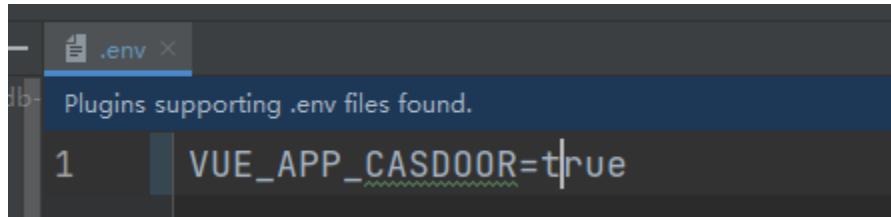
Client secret ⓘ: 242082e823b31a9b0d3a0a4a5a80cd5e415c75f

Cert ⓘ: cert-built-in

# Step 3: Activate Apache IoTDB Web Workbench front-end switch

Open this switch to send the code and state to the back-end.

This switch can be found in iotdb-web-workbench/fronted/.env file.



A screenshot of a code editor showing a file named '.env'. The file contains the line 'VUE\_APP\_CASDOOR=true'. A tooltip above the line says 'Plugins supporting .env files found.'

## Step 4: Configure the back-end code

You need to configure Casdoor's settings in the iotdb-web-workbench/backend/src/main/resources/application.properties file.

```
casdoor.endpoint = http://localhost:8000
casdoor.clientId = <client id from previous step>
casdoor.clientSecret = <client secret from previous step>
casdoor.certificate=<client certificate from previous step>
casdoor.organizationName=IoTDB
casdoor.applicationName=app-IoTDB
```

# Result

The image is a composite of two screenshots. On the left, there is a photograph of a factory conveyor belt system, showing several large, dark cylindrical components and a bright yellow rectangular object being processed. On the right, there is a screenshot of the IoTDB WorkBench login page. The page has a header with the IoTDB logo and "WorkBench". Below the header, it says "Welcome To IoTDB WorkBench". There are two input fields: one for "Account" with the placeholder "Please Input Account" and one for "Password" with placeholder "\*\*\*\*\*". To the right of the password field is a "Forgot Password?" link. At the bottom of the page are two buttons: a teal "Sign In" button and a green "Sign In With Casdoor" button with a small circular icon.

# Apache DolphinScheduler

Casdoor is one of the supported login methods for [Apache DolphinScheduler](#).

## Step 1: Deploy Casdoor

Firstly, Casdoor should be deployed. You can refer to the Casdoor official documentation for [Server Installation](#).

After a successful deployment, please ensure that:

- The Casdoor server is running successfully at <http://localhost:8000>.
- Open your favorite browser and visit <http://localhost:7001>. You will see the login page of Casdoor.
- Test the login functionality by inputting "admin" and "123".

Once the deployment is completed, you can quickly implement a Casdoor-based login page in your own app by following the steps below.

## Step 2: Configure Casdoor Application

1. Create a new Casdoor application or use an existing one.
2. Add your redirect URL (You can find more details about how to obtain the redirect URL in the next section).

Name [?](#) : application\_a6ftas → your application name

Display name [?](#) : New Application - a6ftas

Logo [?](#) : URL [?](#) : [https://cdn.casbin.org/img/casdoor-logo\\_1185x256.png](https://cdn.casbin.org/img/casdoor-logo_1185x256.png)

Preview: 

Home [?](#) :

Description [?](#) :

Organization [?](#) : organization\_carg1b → your organization name

Client ID [?](#) : 3ed7314825ecf955cb19 → your client id

Client secret [?](#) : ee9314ea228... → your client secret

Cert [?](#) : cert-built-in

Redirect URLs [?](#) : Redirect URLs Add  
Redirect URL [?](#) http://localhost:3000/callback → your redirect url

3. Add the desired provider and fill in other necessary settings.

On the application settings page, you will find two important values: `Client ID` and `Client secret`, as shown in the picture above. We will use these values in the next step.

Open your favorite browser and visit `http://CASDOOR_HOSTNAME/.well-known/openid-configuration` to view the OIDC configuration of Casdoor.

# Step 3: Configure DolphinScheduler

| dolphinscheduler-api/src/main/resources/application.yaml

```
security:
  authentication:
    # Authentication types (supported types: PASSWORD, LDAP,
    CASDOOR_SSO)
    type: CASDOOR_SSO
  casdoor:
    # The URL of your Casdoor server
    endpoint:
    client-id:
    client-secret:
    # The certificate may be multi-line; you can use `|-` for ease
    certificate:
    # The organization name you added in Casdoor
    organization-name:
    # The application name you added in Casdoor
    application-name:
    # The DolphinScheduler login URL
    redirect-url: http://localhost:5173/login
```

Now, DolphinScheduler will automatically redirect you to Casdoor for authentication.



# FireZone

[Casdoor](#) can use the OIDC protocol as the IDP to connect various applications. Here, we will use [FireZone](#) as an example to show you how to use OIDC to connect to your applications.

## Step 1: Deploy Casdoor and FireZone

Firstly, Casdoor and FireZone should be deployed.

After a successful deployment, ensure the following:

1. Set the FireZone URL (Sigin → Security → Add OpenID Connect Provider) to FIREZONE\_HOSTNAME.

The screenshot shows the Firezone configuration interface with the following details:

- Left Sidebar:** Configuration, Settings (selected), and Diagnostics.
- Top Bar:** Site Settings
- Content Area:**
  - Site Defaults:**
    - Allowed IPs:** 172.21.0.0/16, 172.16.0.0/16
    - DNS Servers:** 172.16.250.155
    - Endpoint:** FIREZONE\_HOSTNAME (highlighted with a red arrow)
    - Persistent Keepalive:** 0
    - MTU:** 1280

- Casdoor can be logged in and used normally.
- CASDOOR\_HOSTNAME**: <http://localhost:8000>, if you deploy Casdoor using the default `app.conf`.

## Step 2: Configure Casdoor application

- Create a new Casdoor application or use an existing one.
- Add a redirect URL:

For example, if the Configid in the FireZone Provider is TEST, the redirect URL should be `http://[FIREZONE_HOST]/auth/oidc/[PROVIDER_CONFIG_ID]/callback/`.

Home ?:

Description ?:

Organization ?:

Client ID ?:

Client secret ?:

Cert ?:

Redirect URLs ?:  
     
    
  

Open your favorite browser and visit: [http://\[CASDOOR\\_HOSTNAME\]/.well-known/openid-configuration](http://[CASDOOR_HOSTNAME]/.well-known/openid-configuration), and you will see the OIDC configuration of Casdoor.

### 3. Configure FireZone: Security → Add OpenID Connect Provider

OIDC Config

Config ID  
TEST

Label  
TEST

Scope  
openid email profile

Response type  
code

Client ID  
0159c45127541d48e433

Client secret  
add1be9982640e048fcf46770d75674b918484af

Discovery Document URI  
http://localhost:8000/.well-known/openid-configuration

Auto create users

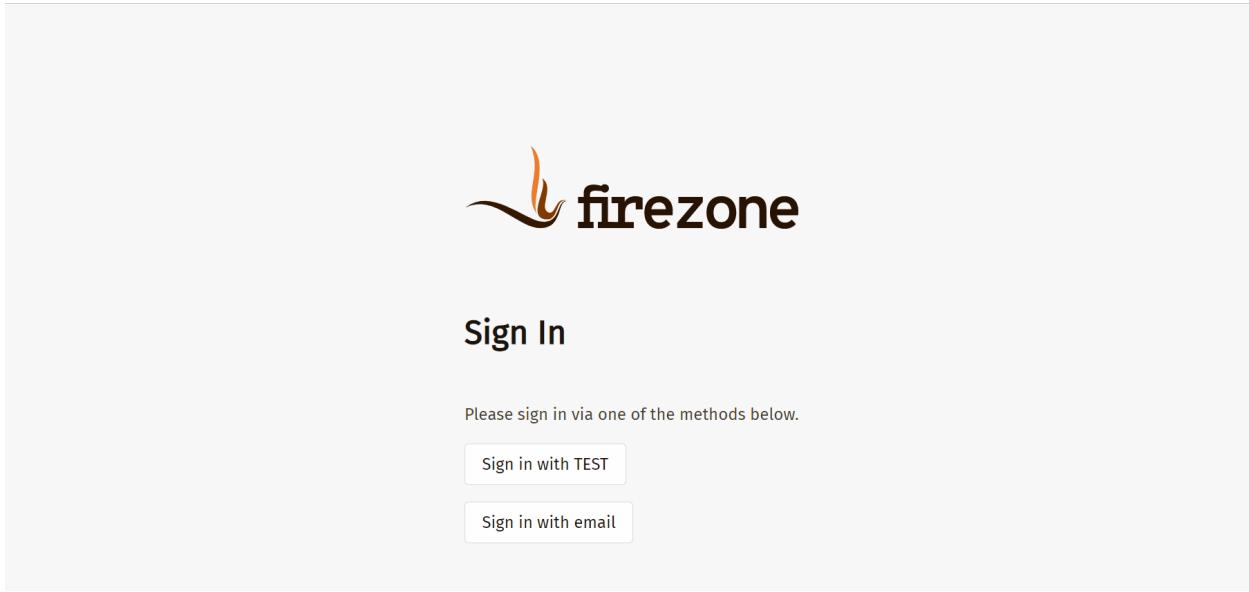
Save

- Discovery Document URI: The FireZone Provider Discovery Document URI should be https://[CASDOOR\_HOST]/.well-known/openid-configuration.
- Scopes: openid email profile
- ConfigID: The ConfigID should be the PROVIDER\_CONFIG\_ID of the

redirect URL and should correspond to the Casdoor redirect URL.

- `Auto-create users`: Successful login will automatically create a user.

## Log out of FireZone and test SSO



# Cloud Foundry

Before the integration, we need to deploy Casdoor locally.

Then, we can quickly implement a Casdoor-based login page in our own app with the following steps.

## Step 1: Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect URL: `http://CASDOOR_HOSTNAME/login`



3. Copy the client ID; we will need it in the following steps.

## Step 2: Add a user in Casdoor

Now that you have the application, but not a user, you need to create a user and assign the role.

Go to the "Users" page and click on "Add user" in the top-right corner. This opens a new page where you can add the new user.

Save the user after adding a username and the organization "Cloud Foundry" (other details are optional).

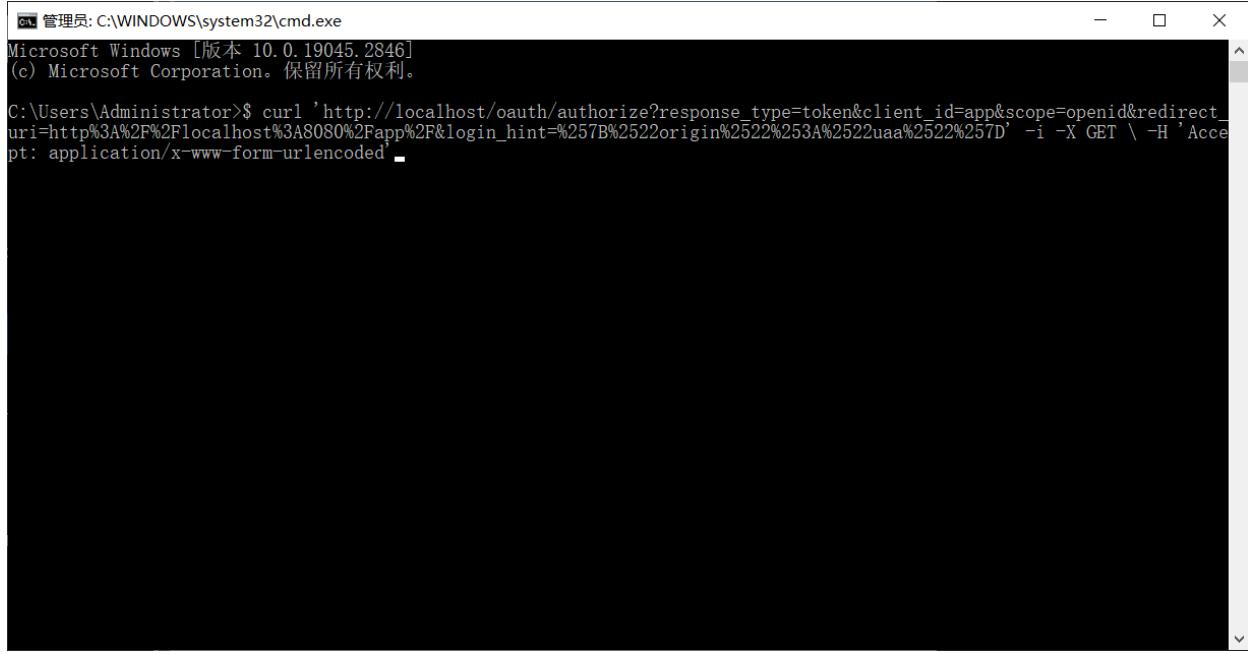
Now, you need to set up a password for your user, which you can do by clicking on "Manage your password".

Choose a password for your user and confirm it.

## Step 3: Build the Cloud Foundry App

Start the Cloud Foundry by following these steps.

- \$ git clone git://github.com/cloudfoundry/uaa.git
- \$ cd uaa
- \$ ./gradlew run

A screenshot of a Windows Command Prompt window titled "管理员: C:\WINDOWS\system32\cmd.exe". The window shows the following command being run:  
C:\Users\Administrator>\$ curl 'http://localhost/oauth/authorize?response\_type=token&client\_id=app&scope=openid&redirect\_uri=http%3A%2F%2Flocalhost%3A8080%2Fapp%2F&login\_hint=%257B%2522origin%2522%253A%2522uaa%2522%257D' -i -X GET \ -H 'Accept: application/x-www-form-urlencoded'  
The command is intended to fetch an OAuth token from a local server, but the output is completely blacked out.

## Step 4: Integrate Casdoor

Now open another command line and input:

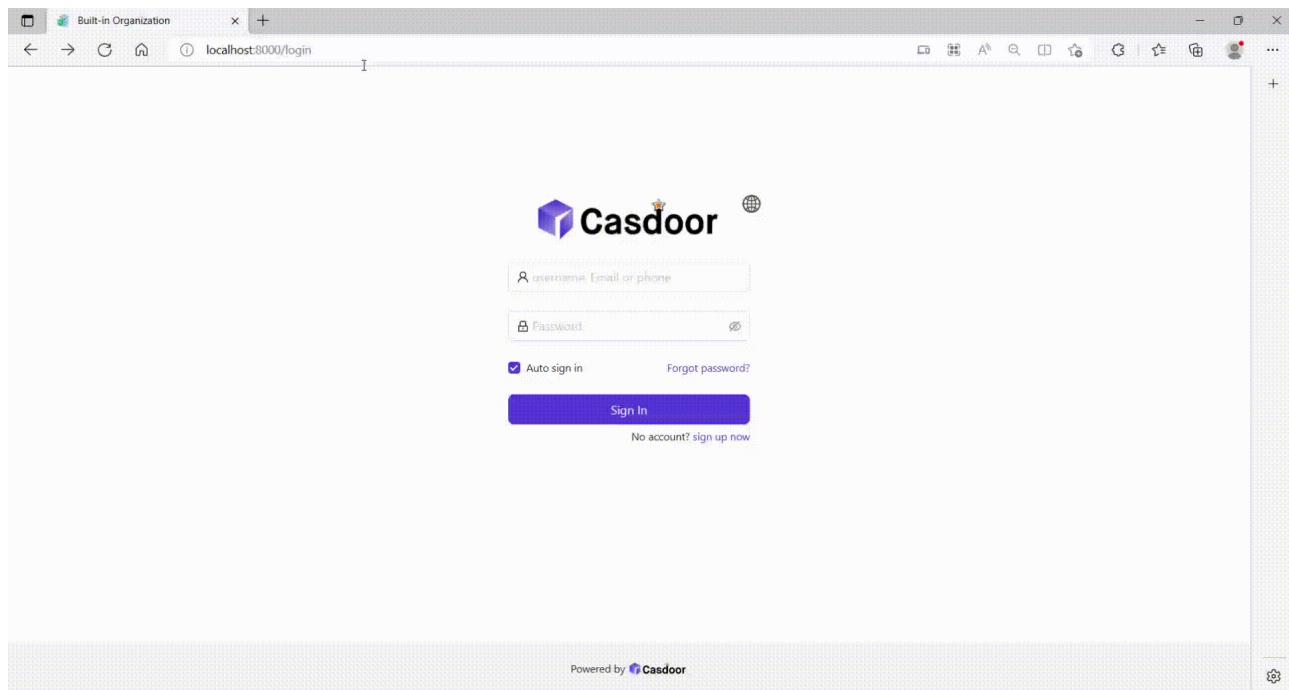
```
curl '<http://localhost/oauth/
authorize?response_type=token&client_id=app&scope=openid&redirect_uri=http%3A%2F%2Flocalhost%3A8080%2Fapp%2F>' 
-i -X GET \
-H 'Accept: application/x-www-form-urlencoded'
```

We have already obtained the client ID and redirect URI before; we input these parameters.

Parameter	Type	Constraints	Description
response_type	String	Required	Space-delimited list of response types. Here, token, i.e. an access token
client_id	String	Required	a unique string representing the registration information provided by the client
scope	String	Optional	requested scopes, space-delimited
redirect_uri	String	Optional	redirection URI to which the authorization server will send the user-agent back once access is granted (or denied), optional if pre-registered by the client

Execute the command, and we can get the result below, which means that we have successfully integrated Casdoor with Cloud Foundry.

```
HTTP/1.1 302 Found
Content-Security-Policy: script-src 'self'
Strict-Transport-Security: max-age=31536000
Set-Cookie: X-Uaa-Csrf=09mMqMDhcwHGLMufnb4YA1; Path=/; Max-Age=86400; Expires=Fri, 5 May 2023 14:53:54 GMT; HttpOnly; SameSite=Lax
Cache-Control: no-store
Content-Language: en
X-XSS-Protection: 1; mode=block
X-Frame-Options: DENY
X-Content-Type-Options: nosniff
Location: http://localhost:8080/app/#token_type=bearer&access_token=eyJhbGciOiJIUzI1NiIsImprdsI6Imh0dHBzO18vbG9jYWxob3N0OjgwODAvdWFhL3Rva
```



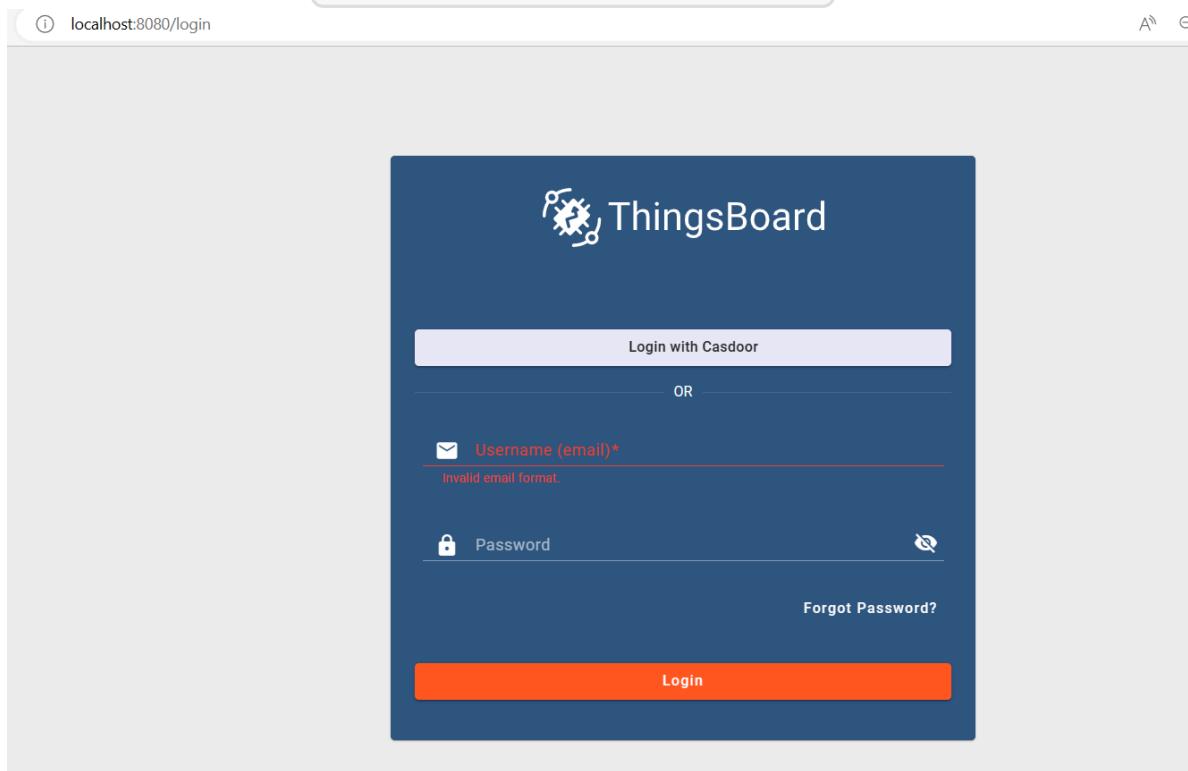
# Thingsboard

Before the integration, we need to deploy Casdoor locally.

Then, we can quickly implement a Casdoor-based login page in our own app by following these steps.

## Step 1: Configure Casdoor application

1. Create a new Casdoor application or use an existing one.
2. Add a redirect URL: `http://CASDOOR_HOSTNAME/login`



3. Copy the client ID and client secret. We will need them in the following steps.

## Step 2: Add a user in Casdoor

Now that you have the application, you need to create a user and assign a role.

Go to the "Users" page and click on "Add user" in the top right corner. This will open a new page where you can add the new user.

Save the user after adding a username and selecting the organization "Thingsboard" (other details are optional).

Next, you need to set up a password for the user. You can do this by clicking on "Manage your password".

Choose a password for the user and confirm it.

## Step 3: Prerequisites and Build Thingsboard App

First of all, Thingsboard only supports Java 11 (OpenJDK).

You can download it from the following link:

[JDK Download Page](#)

To start Thingsboard, follow these steps (for Windows system):

- Download and extract the package. [Download the package](#)
- Configure Thingsboard in \thingsboard\conf\thingsboard.yml according to your preferences, including the configuration of Kafka, PostgreSQL, and others.

- Run `install.bat -loadDemo` in the command line in the Thingsboard folder to install and add demo data.

```
C:\Program Files (x86)\thingsboard>install.bat --loadDemo  
Detecting Java version installed.  
CurrentVersion 110  
Java 11 found!  
Installing thingsboard ...  
...  
ThingsBoard installed successfully!
```

- Run `net start thingsboard` in the command line to start Thingsboard. You should see the following output:

```
The ThingsBoard Server Application service is starting.  
The ThingsBoard Server Application service was started successfully.
```

---

## Step 4: Integrate Casdoor

Now open <http://localhost:8080/> and log in to the admin account:

Account: [sysadmin@thingsboard.org](mailto:sysadmin@thingsboard.org) / Password: sysadmin

After successfully logging in, click the OAuth2 button at the bottom left of the page.

The screenshot shows the ThingsBoard Home dashboard. On the left, a sidebar menu includes: Home, Tenants, Tenant profiles, Resources (with a dropdown arrow), Notification center, Settings, Security (with a dropdown arrow), General, Two-factor authentication, and OAuth2.

The main dashboard area displays the following information:

- Tenants**: 2 (with a + button)
- Tenant profiles**: 2 (with a + button)
- CPU**: 15% | 8 cores
- Devices**: 9
- Assets**: 2
- Users**: 8
- Customers**: 3
- Realtime - last h**: A chart showing CPU usage over time, ranging from 0% to 100% with a peak around 75% at 13:40.
- Documentation**: Includes links to Getting started, Tenant profiles, API, and Widgets Library.
- Configured features**: Includes Email, SMS, Slack, OAuth 2, and 2FA.
- Transport messages**: History - last 30 days, showing 0k messages from May 02 to May 05.

Fill in the blanks as follows:

## Providers

Custom

Login provider*	Custom	Allowed platforms	Web, Android, iOS
Client ID*	e324f9a3f55e1adac4ef	Client secret*	28b3f98c1f55c1cc57f74b9b1a68b5d2e79

General

Access token URI*	http://localhost:8000/api/login/oauth/access_token	Authorization URI*	http://localhost:8000/login/oauth/authorize
JSON Web Key URI	http://localhost:8000/.well-known/jwks	User info URI	http://localhost:8000/api/userinfo

Mapper

Client authentication method*	POST
-------------------------------	------

Provider label\*

Casdoor	Login button icon
---------	-------------------

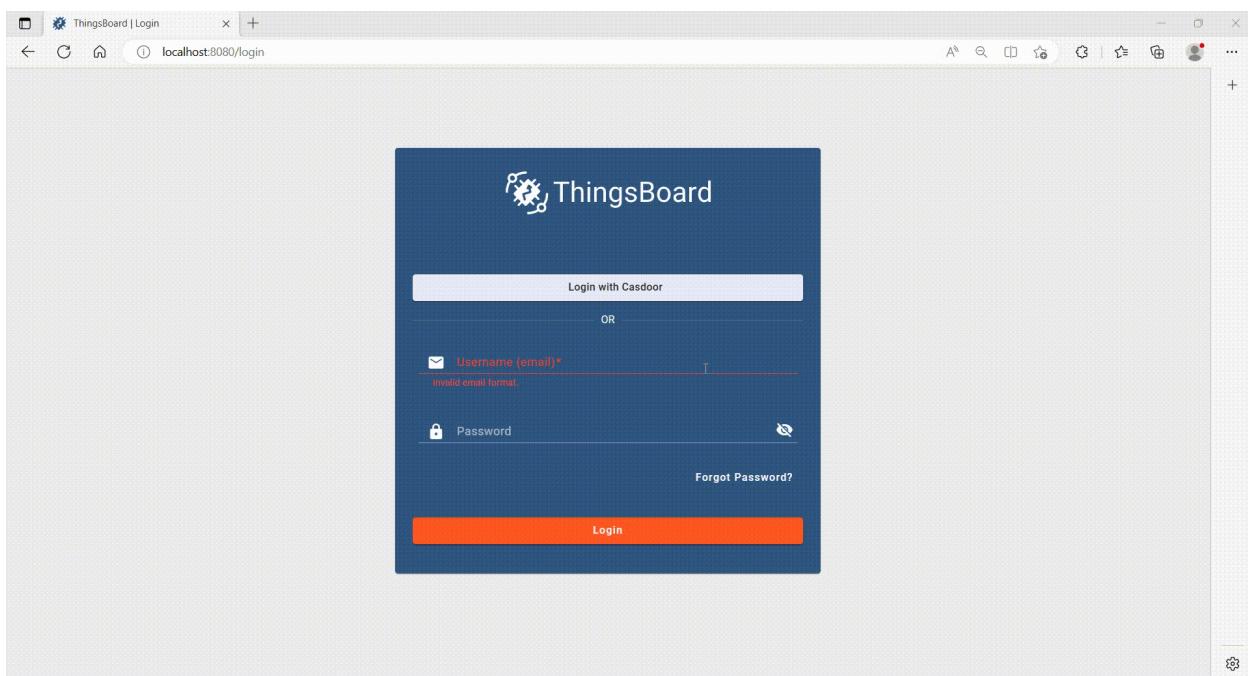
Allow user creation

You can get these values from the following link: [OIDC discovery URL](#)

```
{  
  "issuer": "https://door.casdoor.com",  
  "authorization_endpoint": "https://door.casdoor.com/login/oauth/authorize",  
  "token_endpoint": "https://door.casdoor.com/api/login/oauth/access_token",  
  "userinfo_endpoint": "https://door.casdoor.com/api/userinfo",  
  "jwks_uri": "https://door.casdoor.com/.well-known/jwks",  
  "introspection_endpoint": "https://door.casdoor.com/api/login/oauth/introspect",  
  "response_types_supported": ["code"]}
```

After filling in these blanks, you have successfully integrated Casdoor with Thingsboard. When you log in to <http://localhost:8080/>, you should see the

following:



# JavaScript

## Firebase

Firebase project using Casdoor as Identity Provider

## 微信小程序

在 微信小程序中使用 Casdoor

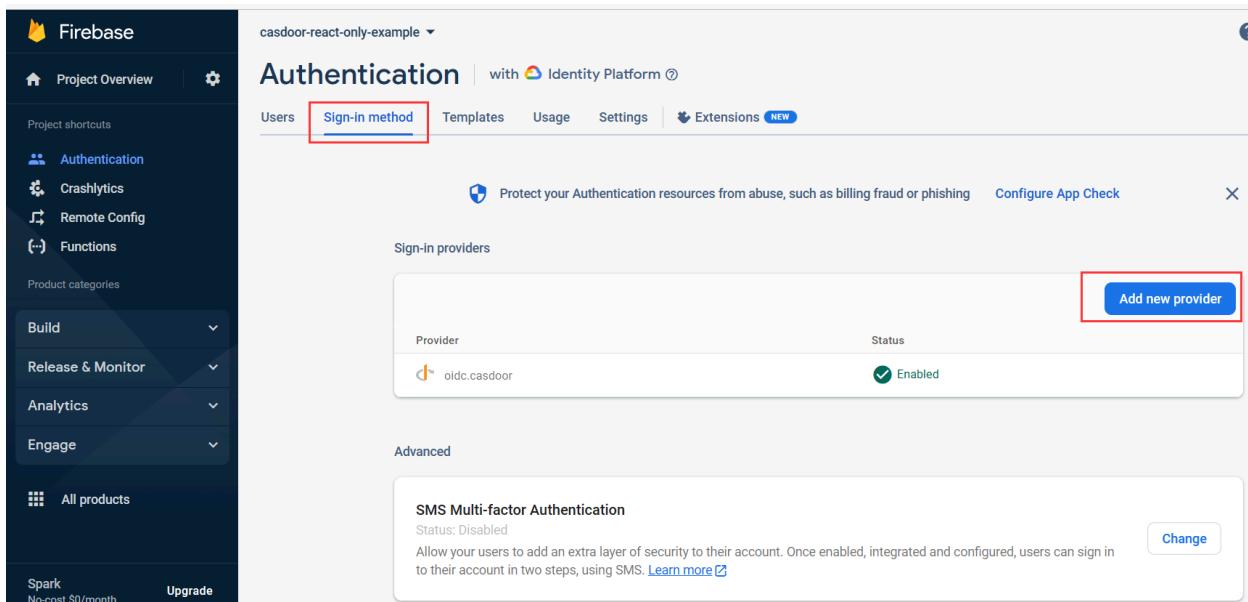
# Firebase

Firebase supports OIDC as an Identity Provider, you can use Casdoor as an OIDC provider for Firebase web app.

## 1. Create a Firebase project

Go to [Firebase Console](#) to create a project.

### 1.1 Add Casdoor as provider



The screenshot shows the Firebase Authentication console for a project named "casdoor-react-only-example". The "Sign-in method" tab is selected. In the "Sign-in providers" section, there is a table with one row:

Provider	Status
oldc.casdoor	Enabled

A red box highlights the "Add new provider" button at the bottom right of the "Sign-in providers" section. Other tabs visible include "Users", "Templates", "Usage", "Settings", and "Extensions".

You need to enable "Identity Platform" feature first to enable OIDC integration on Firebase.

Select [OpenID Connect](#) in Custom providers, fill in the following information:

Name (in order)	Description	Example value
Name	Any be any string you would like	casdoor
Client ID	Client ID for the Casdoor application	294b09fbc17f95daf2fe
Issuer (URL)	Casdoor server URL	<a href="https://door.casdoor.com">https://door.casdoor.com</a>
Client Secret	Client secret for Casdoor application	dd8982f7046ccba1bbd7851d5c1ece4e52bf039d

door.casdoor.com/applications/casbin/app-vue-python-example

**Casdoor** Home User Management Identity Authorization Logging & Auditing Business & Payments Admin

Edit Application Save Save & Exit

Name ⓘ: app-vue-python-example

Display name ⓘ: Casdoor Vue Python Example

Logo ⓘ: URL ⓘ: https://cdn.casbin.org/img/casdoor-logo\_1185x256.png

Preview: 

Home ⓘ: https://demo.gsoc.com.cn

Description ⓘ:

Organization ⓘ: casbin

Tags ⓘ:

Client ID ⓘ: 294b09fbc17f95daf2fe

Client secret ⓘ: dd8982f7046ccba1bbd7851d5c1ece4e52bf039d

Cert ⓘ: cert-built-in

The above examples values can be retrieved from Casdoor demo site: [app-vue-python-example](#)



Enable

### 1 Define new OIDC provider

Grant type

Code flow     Implicit flow (id\_token)

Name

casdoor

Provider ID: oidc.casdoor

Client ID

294b09fbc17f95daf2fe

Issuer (URL)

https://door.casdoor.com

Client secret

dd8982f7046ccba1bbd7851d5c1ece4e52bf039d

**Next**



Configure OIDC integration

## 1.2 Add callback url

Add Callback URL to Casdoor application Redirect URLs:

 OpenID Connect  Enable

 Define new OIDC provider  
Name: casdoor, Provider ID: oidc.casdoor, Client ID: 76dfa0e75796f8443b5e, Issuer (URL): h...

**2 Configure OIDC integration**

When completing set up, add this authorization callback URL to your OIDC app configuration.

Callback URL

 To complete set up, follow the steps for your platform

[Apple ↗](#) [Android ↗](#) [Web ↗](#)

 Delete provider Cancel

← → C [door.casdoor.com/applications/casbin/app-vue-python-example](https://door.casdoor.com/applications/casbin/app-vue-python-example)

Tags  :

Client ID  : 294b09fb17f95daf2fe

Client secret  : dd8982f7046ccba1bbd7851d5c1ece4e52bf039d

Cert  : cert-built-in

**Redirect URLs  :**  Redirect URLs

**Redirect URL**

Token format  : JWT

Token expire  : 168 Hours

Refresh token expire  : 0 Hours

Enable password  :

Enable signup  :

Signin session  :

Here we provide an example [casdoor-firebase-example](#) for you to use Casdoor

authentication in your app. To see more details for how to use Casdoor authentication in your app, please refer to [Firebase document](#).

# 微信小程序

## ① 信息

Casdoor now supports WeChat Mini Program starting from version 1.41.0.

## 介绍

Since WeChat Mini Program does not support standardized OAuth, it cannot redirect to the self-hosted Casdoor webpage for login. Therefore, the process of using Casdoor for WeChat Mini Program is different from that of regular programs.

This document will explain how to integrate Casdoor into WeChat Mini Program. You can find an example for this integration on GitHub here: [casdoor-wechat-miniprogram-example](#). For more detailed information, please refer to the WeChat Mini Program [login document](#).

The configuration includes the following names:

`CASDOOR_HOSTNAME`: The domain name or IP address where the Casdoor server is deployed, e.g., <https://door.casbin.com>.

## Step 1: Deploy Casdoor

Firstly, the [Casdoor server](#) should be deployed.

After successfully deploying Casdoor, you need to ensure:

1. Casdoor can be accessed and used normally.
2. 将 Casdoor 的 `origin` 值 (conf/app.conf) 设置为 `CASDOOR_HOSTNAME`。

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 origin = "http://10.144.1.2:8000"|
          CASDOOR_HOSTNAME
```

## Step 2: Configure Casdoor Application

1. Create a WeChat IDP in Casdoor and provide the `APPID` and `APPSECRET` given to you by the WeChat Mini Program development platform.

New Provider [Save](#) [Save & Exit](#) [Cancel](#)

Name [?](#) : provider\_Mini Program

Display name [?](#) : Mini Program

Category [?](#) : OAuth

Type [?](#) : WeChat Mini Program

Client ID [?](#) : \*\*\*

Client secret [?](#) : \*\*\*

Provider URL [?](#) : <https://github.com/organizations/xxx/settings/applications/1234567>

[Save](#) [Save & Exit](#) [Cancel](#)

2. Create a new Casdoor application or use an existing one.
3. Add the IDP created in the previous step to the application you want to use.

**!** TIPS

For convenience, Casdoor will treat the first WeChat type IDP in the application as the WeChat Mini Program IDP by default.

Therefore, if you want to use WeChat Mini Program in this app, do not add multiple WeChat type IDPs in one app.

# Step 3: Write WeChat MiniProgram Code

WeChat Mini Program provides an API to internally log in and obtain the code. The code should then be sent to Casdoor. Casdoor will use this code to retrieve information (such as OpenID and SessionKey) from the WeChat server.

The following code demonstrates how to accomplish the above process:

```
// Login in mini program
wx.login({
  success: res => {
    // This is the login code that needs to be sent to Casdoor
    console.log(res.code)

    wx.request({
      url: `${CASDOOR_HOSTNAME}/api/login/oauth/access_token`,
      method: "POST",
      data: {
        "tag": "wechat_miniprogram", // Required
        "client_id": "6825f4f0af45554c8952",
        "code": res.code,
        "username": this.data.userInfo.nickName, // Update user
        profile when you log in.
        "avatar": this.data.userInfo.avatarUrl,
      },
      header: {
        "content-type": "application/x-www-form-urlencoded",
      },
      success: res => {
        console.log(res)
        this.globalData.accessToken = res.data.access_token // Get
        Casdoor's access token
      }
    })
  }
})
```

It is important to note that the `tag` parameter is mandatory to inform Casdoor that this is a request from the WeChat Mini Program.

The above code includes the username and avatar URL of the WeChat Mini Program user during login. You can choose to pass these two parameters separately and then pass them to Casdoor after a successful login and obtaining the access token:

```
wx.getUserProfile({
  desc: 'share your info to Casdoor',
  success: (res) => {
    this.setData({
      userInfo: res.userInfo,
      hasUserInfo: true
    })
    console.log(app.globalData.accessToken)
    wx.request({
      url: `${CASDOOR_HOSTNAME}/api/update-user`, // Casdoor URL
      method: "POST",
      data: {
        "owner": "test",
        "name": "wechat-oGk3T5tIiMFo3SazC075f0HEiE7Q",
        "displayName": this.data.userInfo.nickName,
        "avatar": this.data.userInfo.avatarUrl
      },
      header: {
        "Authorization": "Bearer " + app.globalData.accessToken,
        // Bearer token
        "content-type": "application/json"
      },
      success: (res) => {
        console.log(res)
      }
    })
  }
})
```

Additionally, you can use the access token as a bearer token for any Casdoor operation you require.

 TIPS

Currently, Casdoor is unable to bind existing accounts to WeChat Mini Program users. After Casdoor retrieves the OpenID from WeChat, it will either create a new user if the ID does not exist, or use the existing user if it does.



> 集成 >

Lua

# Lua



APISIX

在 APISIX 中使用 Casdoor

# APISIX

Currently, there are 2 methods to use Casdoor to connect to APISIX via APISIX plugins and protect the APIs behind APISIX: using APISIX's Casdoor plugin or using APISIX's OIDC plugin.

## 通过 APISIX 的 Casdoor 插件连接 Casdoor

This plugin, `authz-casdoor`, can protect APIs behind APISIX, forcing every single request to get authenticated without modifying the code of the API.

### 如何启用它？

You need to specify this plugin when creating the route and provide all the required fields. 参见下面的示例。

```
curl "http://127.0.0.1:9180/apisix/admin/routes/1" -H "X-API-KEY: edd1c9f034335f136f87ad84b625c8f1" -X PUT -d '
{
  "methods": ["GET"],
  "uri": "/anything/*",
  "plugins": {
    "authz-casdoor": {
      "endpoint_addr": "http://localhost:8000",
      "callback_url": "http://localhost:9080/anything/callback",
      "client_id": "7ceb9b7fda4a9061ec1c",
      "client_secret": "3416238e1edf915eac08b8fe345b2b95cdba7e04"
    }
  }
}'
```

In this example, we created a route "/anything/\*" pointed to "httpbin.org:80" using APISIX's admin API, with the "authz-casdoor" plugin enabled. This route is now under the authentication protection of Casdoor.

## 属性

名称	类型	申请标准	默认	有效期	描述
endpoint_addr	字符串	必填			The URL of Casdoor.
client_id	字符串	必填			The client ID in Casdoor.
client_secret	字符串	必填			The client secret in Casdoor.
callback_url	字符串	必填			The callback URL which is used to receive state and code.

*endpoint\_addr* 和 *callback\_url* 不应以"/"结尾

In the configuration of the "authz-casdoor" plugin, we can see four parameters.

第一个是“callback\_url”。 This is the callback URL in OAuth2. It should be emphasized that this callback URL must belong to the "uri" you specified for the route. For example, in this example, <http://localhost:9080/anything/callback> obviously belongs to "/anything/\*". Only by this way, the visit toward the callback\_url can be intercepted and utilized by the plugin (so that the plugin can get the code and state in OAuth2). The logic of the callback\_url is implemented completely by the plugin, so there is no need to modify the server to implement this callback.

The second parameter "endpoint\_addr" is obviously the URL of Casdoor. 第三个和第四个参数是“client\_id”和“client\_secret”，您可以在注册应用程序时从Casdoor获取这些参数。

## 它是如何工作的？

Suppose a new user who has never visited this route before is going to visit it (<http://localhost:9080/anything/d?param1=foo&param2=bar>). Considering that "authz-casdoor" is enabled, this visit would be processed by the "authz-casdoor" plugin first. After checking the session and confirming that this user hasn't been authenticated, the visit will be intercepted. With the original URL the user wants to visit kept, they will be redirected to the login page of Casdoor.

After successfully logging in with a username and password (or whatever method they use), Casdoor will redirect this user to the "callback\_url" with GET parameters "code" and "state" specified. Because the "callback\_url" is known by the plugin, when the visit toward the "callback\_url" is intercepted this time, the logic of the "Authorization code Grant Flow" in OAuth2 will be triggered. This means that the plugin will request the access token to confirm whether this user is really logged in. After this confirmation, the plugin will redirect this user to the original URL they want to visit, which was kept by us previously. The logged-in status will also be kept in the session.

Next time this user wants to visit the URL behind this route (for example, <http://localhost:9080/anything/d>), after discovering that this user has been authenticated previously, this plugin won't redirect this user anymore. This way, the user can visit whatever they want under this route without being interfered.

## 通过 APISIX 的 OIDC 插件连接 Casdoor

Casdoor can use the OIDC protocol to connect to APISIX, and this document will show you how to do it.

The following are some of the names used in the configuration:

`CASDOOR_HOSTNAME`: Domain name or IP where the Casdoor server is deployed.

`APISIX_HOSTNAME`: 部署 APISIX 的域名或 IP。

## Step 1: Deploy Casdoor and APISIX

Firstly, deploy [Casdoor](#) and [APISIX](#).

在成功部署后，您需要确保：

1. 可以登录并正常使用Casdoor。
2. 将Casdoor的 `origin` 值 (conf/app.conf) 设置为 `CASDOOR_HOSTNAME`。

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 origin = "http://10.144.1.2:8000"| CASDOOR_HOSTNAME
```

## Step 2: Configure Casdoor application

1. Create a new Casdoor application or use an existing one.
2. Add a redirect URL: `http://APISIX_HOSTNAME/REDIRECTWHATYOUWANT`, and replace `REDIRECTWHATYOUWANT` with the desired redirect URL.
3. Select "JWT-Empty" for the Token format option.
4. Add the desired provider and configure other settings.

Client ID ② :	07860a229bd0b162cdfa						
Client secret ② :	ea02l...9373fe3e						
Redirect URLs ② :	<table border="1"><tr><td>Redirect URLs</td><td>Add</td></tr><tr><td>Redirect URL</td><td><a href="http://localhost:9000/callback">http://localhost:9000/callback</a></td></tr><tr><td></td><td>Add Redirect URL you want</td></tr></table>	Redirect URLs	Add	Redirect URL	<a href="http://localhost:9000/callback">http://localhost:9000/callback</a>		Add Redirect URL you want
Redirect URLs	Add						
Redirect URL	<a href="http://localhost:9000/callback">http://localhost:9000/callback</a>						
	Add Redirect URL you want						
Token format ② :	JWT-Empty						
	Select JWT-Empty						

On the application settings page, you will find the `Client ID` and `Client Secret` values as shown in the picture above. We will use them in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`, where you will find the OIDC configuration of Casdoor.

## Step 3: Configure APISIX

APISIX 拥有官方 OIDC 支持，由 [lua-resety-openidc](#) 实现。

You can customize the settings according to the APISIX OIDC documentation. The following routing settings will be used:

```
# Use your own X-Api-Key
$ curl -X POST APISIX_HOSTNAME/apisix/admin/routes -H "X-Api-Key:edd1c9f034335f136f87ad84b625c8f1" -d '{
  "uri": "/get",
  "name": "apisix_casdoor_test",
  "plugins": {
    "openid-connect": {
      "client_id": "Client ID",
      "client_secret": "Client Secret",
      "discovery": "http://CASDOOR_HOSTNAME/.well-known/openid-configuration",
      "introspection_endpoint_auth_method": "client_secret_basic",
      "logout_path": "/logout",
      "realm": "master",
      "redirect_uri": "http://APISIX_HOSTNAME/REDIRECTWHATYOUWANT",
      "bearer_only": false,
      "set_id_token_header": false,
      "access_token_in_authorization_header": true,
      "set_access_token_header": true,
      "set_userinfo_header": false,
      "realm": "master"
    }
  },
  "upstream": {
    "type": "roundrobin",
    "nodes": {
      "httpbin.org:80": 1
    }
  }
}'
```

Now, visit `http://APISIX_HOSTNAME/get`, and the browser will redirect you to the Casdoor login page. After successfully logging in, you will see that a request has been sent to `httpbin.org` as shown in the screenshot below.

# PHP

## 禅道

在禅道中使用 Casdoor 进行身份验证

## Using Casdoor as an OAuth2 Server in ShowDoc

Using Casdoor as an OAuth2 server in ShowDoc

## Flarum

Using OAuth2 to connect various applications, like Flarum

## Moodle

Using OAuth to connect Moodle

# 禅道

Zentao is an agile (scrum) project management system/tool, but it does not support OIDC itself. To integrate Zentao with Casdoor SSO, we need to use a 3rd-party OIDC module called [zentao-oidc](#), and this document will show you how to do it.

## Step 1: Deploy Casdoor and Zentao

Firstly, deploy [Casdoor](#) and [Zentao](#). After a successful deployment, make sure:

1. Casdoor 可以正常登录使用。
2. You can successfully log in and use Zentao.

## Step 2: Integrate Zentao OIDC third-party module

Install [zentao-oidc](#) by running the following command:

```
git clone https://github.com/casdoor/zentao-oidc.git
```

Alternatively, you can download the ZIP and unzip it.

This module is used to integrate Zentao with SSO for OpenId. Here's how to use it:

1. Copy the entire `oidc` directory to the module of Zentao and use it as a

module of Zentao. Rename the downloaded package to "oidc".

## 2. Configure the filter.

Since the Zentao framework filters the parameters in the URL and does not allow spaces, you need to put the following code at the end of `/config/my.php`.

```
$filter->oidc = new stdclass();
$filter->oidc->index = new stdclass();
$filter->oidc->index->paramValue['scope'] = 'reg::any';
```

## 3. Modify `/module/common/model.php`.

Add 'oidc' to the anonymous access list and add a line to the `isOpenMethod` method of `model.php`.

```
public function isOpenMethod($module, $method)
{
    if ($module == 'oidc' and $method == 'index') {
        return true;
    }
}
```

## 4. If you don't want the Zentao login screen to appear, go directly to the Casdoor login screen.

Modify the last line of code in `public function checkPriv()` in `/module/common/model.php`.

```
//return print(js::locate(helper::createLink('user', 'login',
```

5. Modify the `setSuperVars()` method inside `framework/base/router.class.php` and comment out the following statements.

```
public function setSuperVars()  
// unset($_REQUEST);
```

## Step 3: Configure Casdoor Application

1. Create a new Casdoor application or use an existing one.
2. Add your redirect URL.

Client ID <small>?</small> :	d8d7715e24f077066a20						
Client secret <small>?</small> :	[REDACTED]						
Cert <small>?</small> :	cert-built-in						
Redirect URLs <small>?</small> :	<table border="1"><tr><td>Redirect URLs</td><td>Add</td></tr><tr><td colspan="2">Redirect URL</td></tr><tr><td colspan="2">🔗 http://127.0.0.1/zentao/oidc-index.html</td></tr></table>	Redirect URLs	Add	Redirect URL		🔗 http://127.0.0.1/zentao/oidc-index.html	
Redirect URLs	Add						
Redirect URL							
🔗 http://127.0.0.1/zentao/oidc-index.html							

3. Add the provider you want and fill in other required settings.

## Step 4: Configure Zentao

Configure the `config.php` file in the `oidc` directory.

```
$config->oidc->clientId = "<Your ClientId>";
```

Set your redirect URL in `module/oidc` in the `public function index()` method.

```
$oidc->setRedirectURL($path."/zentao/oidc-index.html");
```

① 备注

这里的URL是指调用 'index' 方法在 'oidc' 模块中。 You also need to set a variable separator. By default, the framework uses a dash ("‐"). Please refer to the official Zentao framework for more details. "[zentaoPHP框架](#)"

# Using Casdoor as an OAuth2 Server in ShowDoc

## Using Casdoor for Authentication in ShowDoc

ShowDoc is an online API documentation and technical documentation tool that is perfect for IT teams. ShowDoc makes it easy to use Markdown syntax to write beautiful API documents, data dictionary documents, technical documents, online Excel documents, and more.

ShowDoc supports 3rd-party authentication, including OAuth2. Here is a tutorial for achieving this.

### Step 1: Create a Casdoor Application

Go to your Casdoor and add a new application called ShowDoc. Here is an example of creating the ShowDoc application in Casdoor.

[Edit Application](#)[Save](#)[Save & Exit](#)Name [?](#) :

myApplication

Display name [?](#) :

myApplication

Logo [?](#) :URL [?](#) :[https://cdn.casdoor.com/logo/casdoor-logo\\_1185x256.png](https://cdn.casdoor.com/logo/casdoor-logo_1185x256.png)

Preview:

Home [?](#) :Description [?](#) :Organization [?](#) :

built-in

Client ID [?](#) :

208d745196c23df9fd5b

Client secret [?](#) :

4c89f447af77bc276431ab885463ebcb8d6efc3c

Cert [?](#) :

cert-built-in

Please remember the `client ID` and `client Secret` for the next step.

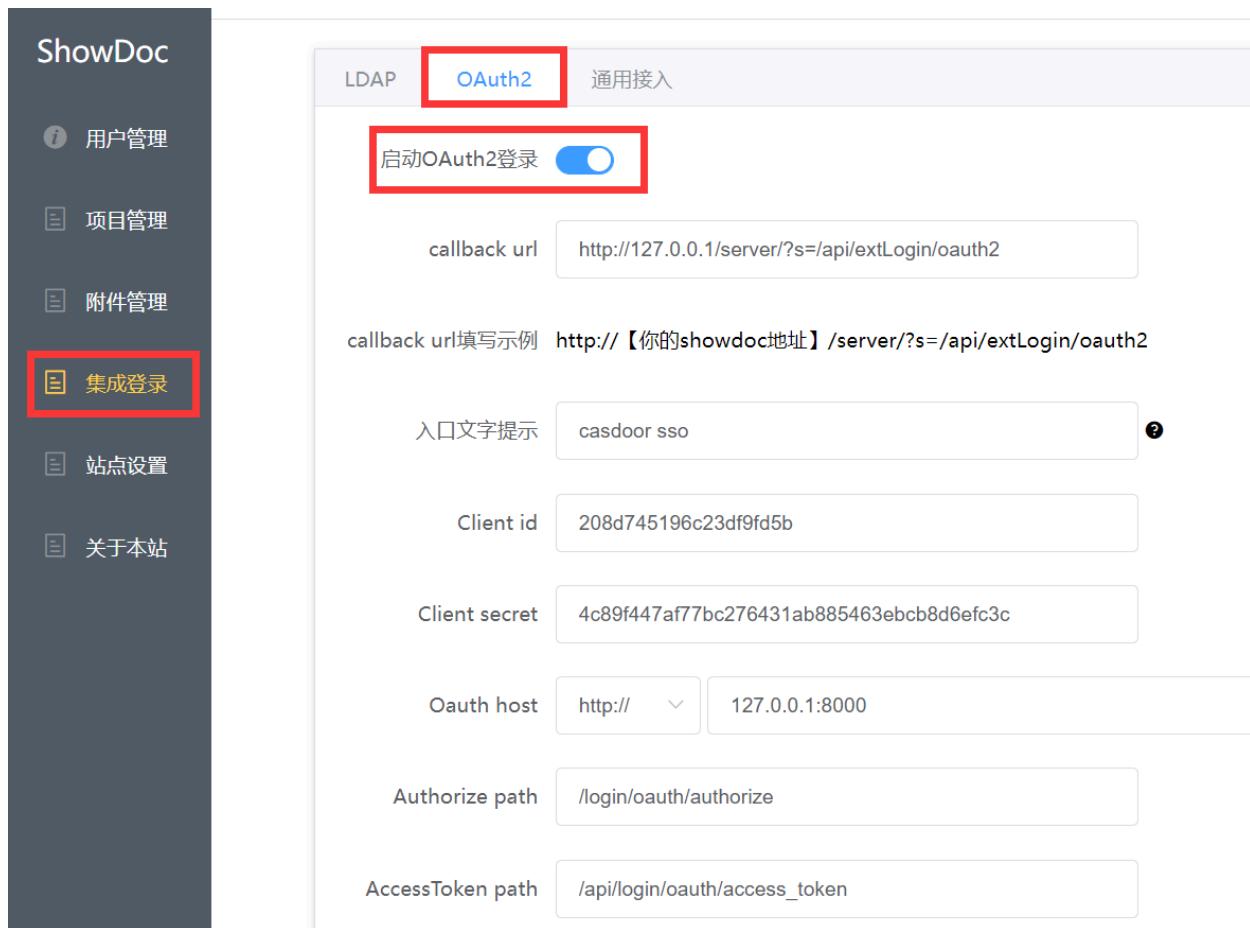
:::信息

Please don't fill in the **callback URL** in this step. The URL depends on the configurations on ShowDoc in the next step. We will come back to set a correct callback URL later.

:::

## Step 2: Configure ShowDoc

First, enable the OAuth2 login button. Then, fill in the `callback URL` as shown in the example. Fill in the `client ID` and `client secret` that were remembered in the previous step.



`Authorize path`, `AccessToken path`, and `User info path` are required. You can fill them in as shown below.

Authorize path: /login/oauth/authorize

## Step 3: Configure the Callback URL in Casdoor

Go back to the application edit page in step 1 and add the `callback URL` that you filled in ShowDoc.



The screenshot shows a user interface for managing redirect URLs. At the top, there is a header with the text "Redirect URLs ?". Below the header, there is a "Redirect URLs" button and an "Add" button. Underneath these buttons, there is a section titled "Redirect URL" containing a single entry: "🔗 http://127.0.0.1/server/?s=/api/extLogin/oauth2".

## Step 4: Have a Try on ShowDoc

You should see the following on the login page:

# 登录

 用户名/邮箱 密码 验证码[注册新账号](#)[casdoor sso](#)

恭喜您！ 您已完成所有步骤 Press the 'Casdoor SSO' button, and you will be redirected to the Casdoor login page.

# Flarum

Casdoor can use OAuth2 to connect various applications. In this example, we will show you how to use OAuth2 to connect Flarum to your applications.

Here are some configuration names you will need:

`CASDOOR_HOSTNAME`: The domain name or IP where the Casdoor server is deployed.

`Flarum_HOSTNAME`: The domain name or IP where Flarum is deployed.

## Step 1: Deploy Casdoor and Flarum

First, deploy [Casdoor](#) and [Flarum](#).

After a successful deployment, make sure:

1. You have downloaded the Flarum plugin [FoF Passport](#).
2. Casdoor can be logged in and used normally.
3. You can set `CASDOOR_HOSTNAME = http://localhost:8000` when deploying Casdoor in `prod` mode. See [production mode](#).

## Step 2: Configure Casdoor application

1. Create a new Casdoor application or use an existing one.
2. Find the redirect URL: `<CASDOOR_HOSTNAME>/auth/passport`.
3. Add the redirect URL to the Casdoor application:

The screenshot shows the 'Application Settings' section of the Casdoor interface. It includes fields for 'Client ID' (014ae4bd048734ca2dea), 'Client secret' (f26a4115725867b7bb7b668c81e1f8ffae1544d), 'Cert' (cert-built-in), and 'Redirect URLs'. The 'Redirect URLs' field contains '<your flarum install>/auth/passport' with an 'Add' button and an 'Action' dropdown menu.

On the application settings page, you will find two values: `Client ID` and `Client secret`. We will use these values in the next step.

Open your favorite browser and visit: `http://CASDOOR_HOSTNAME/.well-known/openid-configuration`. You will see the OIDC configuration of Casdoor.

## Step 3: Configure Flarum

1. Install the plugin [FoF Passport](#).
2. Configure the app:

The screenshot shows the configuration page for the FoF Passport extension in Casdoor. The top navigation bar has tabs for 'Home', 'Applications', 'Users', 'Logs', and 'Help'. Below the navigation, there's a sidebar with 'FoF Passport' and a note: 'The OAuth2 (and Laravel passport) compatible oauth extension'. A green toggle switch is labeled 'Enabled'. The main content area contains several configuration fields:

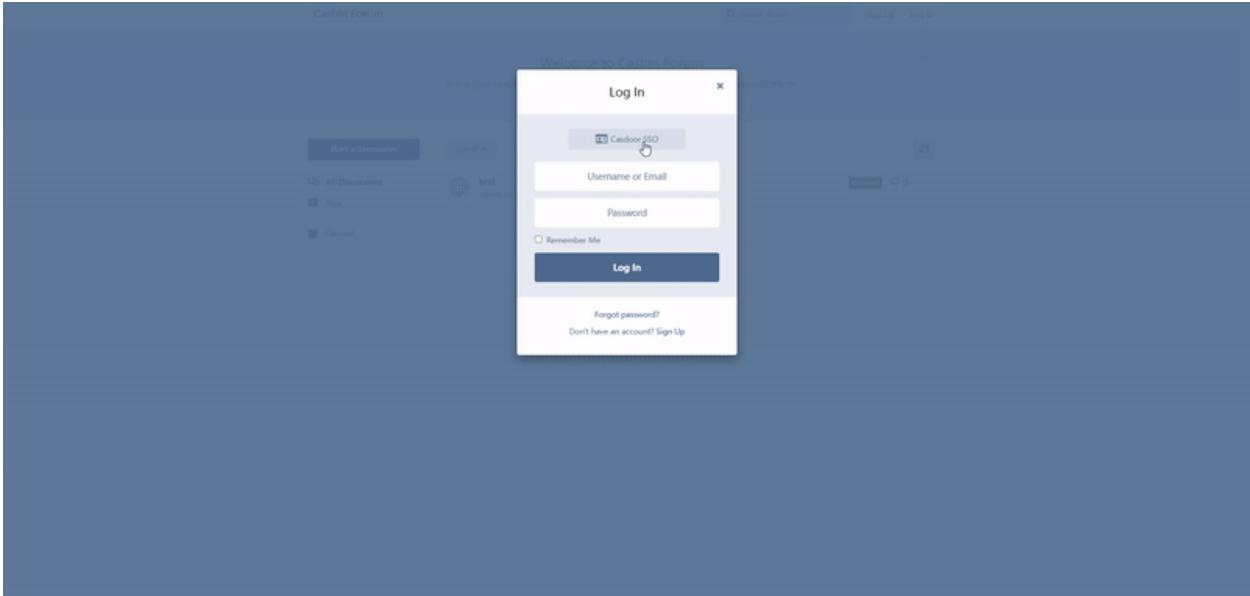
- OAuth authorization URL:** https://door.casdoor.com/login/oauth/authorize
- OAuth token URL:** https://door.casdoor.com/api/login/oauth/access\_token
- Api URL providing user details when authenticated:** https://door.casdoor.com/api/user
- OAuth application ID:** 014ae4bd048734ca2dea
- OAuth application secret:** f26a4115725867b7bb7b668c81e1f8f7fae1544d
- OAuth scopes to request:** openid profile email
- Label for login button:** Casdoor SSO
- Icon for login button:** far fa-id-card

A blue 'Save Changes' button is located at the bottom left.

3. Find the Client ID and Client Secret in the Casdoor application page.

- Token server URL: http://**CASDOOR\_HOSTNAME**/api/login/oauth/access\_token
- Authorization server URL: http://**CASDOOR\_HOSTNAME**/login/oauth/authorize
- UserInfo server URL: http://**CASDOOR\_HOSTNAME**/api/get-account
- Scopes: address phone openid profile offline\_access email

Log out of Flarum and test SSO.



# Moodle

Casdoor can be used to connect Moodle using OAuth.

The following are some configuration settings:

- `CASDOOR_HOSTNAME`: The domain name or IP where the Casdoor server is deployed.
- `Moodle_HOSTNAME`: The domain name or IP where Moodle is deployed.

## Step 1: Deploy Casdoor and Moodle

First, deploy Casdoor and Moodle.

After successful deployment, ensure the following:

1. Casdoor can be logged in and used without issues.
2. You can set `CASDOOR_HOSTNAME` as `http://localhost:8000` when deploying Casdoor in `prod` mode. See [production mode](#).

## Step 2: Configure Casdoor Application

1. Create a new Casdoor application or use an existing one.
2. Find the redirect URL: `Moodle_HOSTNAME/admin/oauth2callback.php`.
3. Add the redirect URL to the Casdoor application.

For more information on OAuth, refer to [OAuth](#).

# Step 3: Configure Moodle

## 1. Locate OAuth

MyMoodle Home Dashboard My courses Site administration AU Edit mode

Site administration

General Users Courses Grades Plugins Appearance Server Reports Development

Your site is not yet registered. [Register your site](#)

**Server**

- System paths
- Support contact
- Session handling
- HTTP
- Maintenance mode
- Cleanup
- Environment
- PHP info
- Performance
- Update notifications
- File types
- OAuth 2 services**

## 2. Configure this application

MyMoodle Home Dashboard My courses Site administration AU Edit mode

Detailed instructions on configuring the common OAuth 2 services

Name: Casdoor

Client ID: 154fb67917b18c0a1850 → **your Client ID**

Client secret: 380a93b5717ab0f8545fbc → **your Client secret**

Service base URL: https://demo.casdoor.com → **your Casdoor Home**

Logo URL: https://cdn.casdoor.com/s

This service will be used: Login page and internal services

Name displayed on the login page:

Scopes included in a login request: openid profile email

Scopes included in a login request for offline access: openid profile email

Additional parameters included in a login request:

Additional parameters included in a login request for offline access:

Login domains:

Require email verification

I understand that disabling email verification can be a security issue.

**Save changes** Cancel

## 3. Configure this mapping

## User field mappings for issuer: Casdoor

External field name	Internal field name	Edit
address	address	
email	email	
name	firstname	
phone	phone1	
picture	picture	
preferred_username	username	

Create new user field mapping for issuer "Casdoor"

## 4. Locate the OAuth2 plugin

General    Users    Courses    Grades    Plugins    Appearance    Server    Reports    Development

Your site is not yet registered. [Register your site](#)

**Plugins** [Install plugins](#) [Plugins overview](#)

**Activity modules**

- [Manage activities](#)
- [Common activity settings](#)
- [Assignment](#)
- [Assignment settings](#)
- [Submission plugins](#)
- [Manage assignment submission plugins](#)
- [File submissions](#)
- [Online text submissions](#)
- [Feedback plugins](#)
- [Manage assignment feedback plugins](#)
- [Feedback comments](#)
- [Annotate PDF](#)
- [File feedback](#)
- [Offline grading worksheet](#)
- [Book](#)
- [Chat](#)
- [Database](#)
- [External tool](#)
- [Manage tools](#)
- [Feedback](#)
- [File](#)
- [Folder](#)
- [Forum](#)
- [Glossary](#)
- [HSP](#)
- [IMS content package](#)
- [Lesson](#)
- [Page](#)
- [Quiz](#)
- [General settings](#)
- [Safe Exam Browser templates](#)
- [Safe Exam Browser access rules](#)
- [SCORM package](#)
- [Text and media area](#)
- [URL](#)
- [Workshop](#)

**Admin tools**

- [Manage admin tools](#)
- [Accessibility](#)
- [Brickfield registration](#)
- [Accessibility toolkit settings](#)
- [Reports](#)
- [Recycle bin](#)

**Antivirus plugins** [Manage antivirus plugins](#)

**Authentication**

- [Manage authentication](#)
- [Email-based self-registration](#)
- [Manual accounts](#)
- [OAuth 2](#)

## 5. Enable the OAuth2 plugin

### Manage authentication

#### Available authentication plugins

Name	Users	Enable	Up/Down	Settings	Test settings	Uninstall
Manual accounts	2			<a href="#">Settings</a>		
No login	0					
Email-based self-registration	0			<a href="#">Settings</a>		<a href="#">Uninstall</a>
OAuth 2	8			<a href="#">Settings</a>	<a href="#">Test settings</a>	

## 6. If you want to prevent the editing of Casdoor's email

### Lock user fields

You can lock user data fields. This is useful for sites where the user data is maintained by the administrators manually by editing user records or uploading using the 'Upload users' facility. If you are locking fields that are required by Moodle, make sure that you provide that data when creating user accounts or the accounts will be unusable.

Consider setting the lock mode to 'Unlocked if empty' to avoid this problem.

Lock value (First name) auth_oauth2   field_lock_firstname	<input type="button" value="Unlocked"/> Default: Unlocked
Lock value (Last name) auth_oauth2   field_lock_lastname	<input type="button" value="Unlocked"/> Default: Unlocked
Lock value (Email address) auth_oauth2   field_lock_email	<input type="button" value="Locked"/> Default: Unlocked
Lock value (City/town) auth_oauth2   field_lock_city	<input type="button" value="Unlocked"/> Default: Unlocked
Lock value (Country) auth_oauth2   field_lock_country	<input type="button" value="Unlocked"/> Default: Unlocked
Lock value (Language) auth_oauth2   field_lock_lang	<input type="button" value="Unlocked"/> Default: Unlocked

here is switch to lock email

For more information on Moodle, refer to [Moodle](#) and [Fields mapping](#).

Log out of Moodle and test SSO.

## MyMoodle Website





&gt;

集成

&gt;

Ruby

# Ruby

**GitLab**

Using Casdoor for authentication in a self-developed GitLab server

# GitLab

Casdoor can use the OIDC protocol to link to a self-deployed GitLab server, and this document will show you how to do it.

## ⚠ 注意事项

As the [GitLab docs](#) state, GitLab only works with OpenID providers that use HTTPS, so you need to deploy Casdoor with HTTPS, such as putting Casdoor behind an NGINX reverse proxy with an SSL certificate setup. Casdoor itself only listens on port 8000 by default via HTTP and has no HTTPS-related functionality.

The following are some of the names mentioned in the configuration:

`CASDOOR_HOSTNAME`: The domain name or IP where the Casdoor server is deployed, e.g., `https://door.casbin.com`.

`GITLAB_HOSTNAME`: The domain name or IP where GitLab is deployed, e.g., `https://gitlab.com`.

## Step 1: Deploy Casdoor and GitLab

Firstly, Casdoor and GitLab should be deployed.

After a successful deployment, you need to ensure:

1. Casdoor can be logged into and used normally.
2. 将 Casdoor 的 `origin` 值 (`conf/app.conf`) 设置为 `CASDOOR_HOSTNAME`。

```
conf > ⚙ app.conf
 8  dbName = casdoor
 9  redisEndpoint =
10 defaultStorageProvider =
11 isCloudIntranet = false
12 authState = "casdoor"
13 httpProxy = "127.0.0.1:10808"
14 verificationCodeTimeout = 10
15 initScore = 2000
16 logPostOnly = true
17 origin = "http://10.144.1.2:8000"| CASDOOR_HOSTNAME
```

## Step 2: Configure Casdoor application

1. 创建或使用现有的 Casdoor 应用程序。
2. Add a redirect URL: [http://GITLAB\\_HOSTNAME/users/auth/openid\\_connect/callback](http://GITLAB_HOSTNAME/users/auth/openid_connect/callback).
3. Add the provider you want and supplement other settings.

Description <a href="#">?</a> :	GitLab	
Organization <a href="#">?</a> :	built-in	
Client ID <a href="#">?</a> :	eab9...35b6	Client ID
Client secret <a href="#">?</a> :	95e7...b3a0188a5	Client secret
Redirect URLs <a href="#">?</a> :	<a href="#">Add</a>	
<a href="#">Redirect URL</a>		
<a href="http://GITLAB_HOSTNAME/users/auth/openid_connect/callback">http://GITLAB_HOSTNAME/users/auth/openid_connect/callback</a>		GitLab redirect url

Notably, you can get two values on the application settings page: [Client ID](#) and [client secret](#) (see the picture above), and we will use them in the next step.

Open your favorite browser and visit: [http://\*\*CASDOOR\\_HOSTNAME\*\*.well-known/openid-configuration](http://<CASDOOR_HOSTNAME>.well-known/openid-configuration), where you will see the OIDC configuration of Casdoor.

## Step 3: Configure GitLab

You can follow the steps below to set this up, or make custom changes according to [this document](#) (e.g., if you are installing GitLab using source code rather than the Omnibus).

1. 在 GitLab 服务器上，打开配置文件。

```
sudo editor /etc/gitlab/gitlab.rb
```

2. 添加提供商配置。 (The HOSTNAME URL should include http or https)

```
gitlab_rails['omniauth_providers'] = [
  {
    name: "openid_connect",
    label: "Casdoor", # optional label for the login
button, defaults to "Openid Connect"
    args: {
      name: "openid_connect",
      scope: ["openid", "profile", "email"],
      response_type: "code",
      issuer: "<CASDOOR_HOSTNAME>",
      client_auth_method: "query",
      discovery: true,
      uid_field: "preferred_username",
      client_options: {
        identifier: "<YOUR CLIENT ID>",
        secret: "<YOUR CLIENT SECRET>",
        redirect_uri: "<GITLAB_HOSTNAME>/users/auth/
openid_connect/callback"
    }
  }
]
```

3. 重新启动 GitLab 服务器。
4. Each registered user can open `GITLAB_HOSTNAME/-/profile/account` and connect the Casdoor account.

The screenshot shows the 'User Settings > Account' page in GitLab. On the left, there's a sidebar with options like Profile, Account (which is selected), Applications, Chat, Access Tokens, Emails, Password, Notifications, SSH Keys, GPG Keys, and Preferences. The main content area has a 'Two-Factor Authentication' section with a status of 'Disabled' and a 'Enable two-factor authentication' button. Below it is a 'Social sign-in' section with a sub-section for 'Connected Accounts'. A red box highlights the 'Connect Casdoor' button next to its icon.

5. 完成！ Now, you can log in to your own GitLab using Casdoor.

The screenshot shows the GitLab login page. It features the GitLab logo and the text 'A complete DevOps platform'. Below that, it says 'GitLab is a single application for the entire software development lifecycle. From project planning and source code management to CI/CD, monitoring, and security.' A note indicates 'This is a self-managed instance of GitLab.' To the right is the login form with fields for 'Username or email' and 'Password', and checkboxes for 'Remember me' and 'Forgot your password?'. Below the form is a link 'Don't have an account yet? Register now'. At the bottom is a 'Sign in with' section containing a red box around the 'Casdoor' button and its icon. There's also a 'Remember me' checkbox in this section.



> 集成 >

Haskell

# Haskell

## Hasura

Before the integration, we need to deploy Casdoor locally.

# Hasura

Before the integration, we need to deploy Casdoor locally.

Then we can quickly implement a Casdoor-based login page in our own app with the following steps.

## Configure Casdoor application

1. Create or use an existing Casdoor application.
2. Add a redirect URL: `http://CASDOOR_HOSTNAME/login`



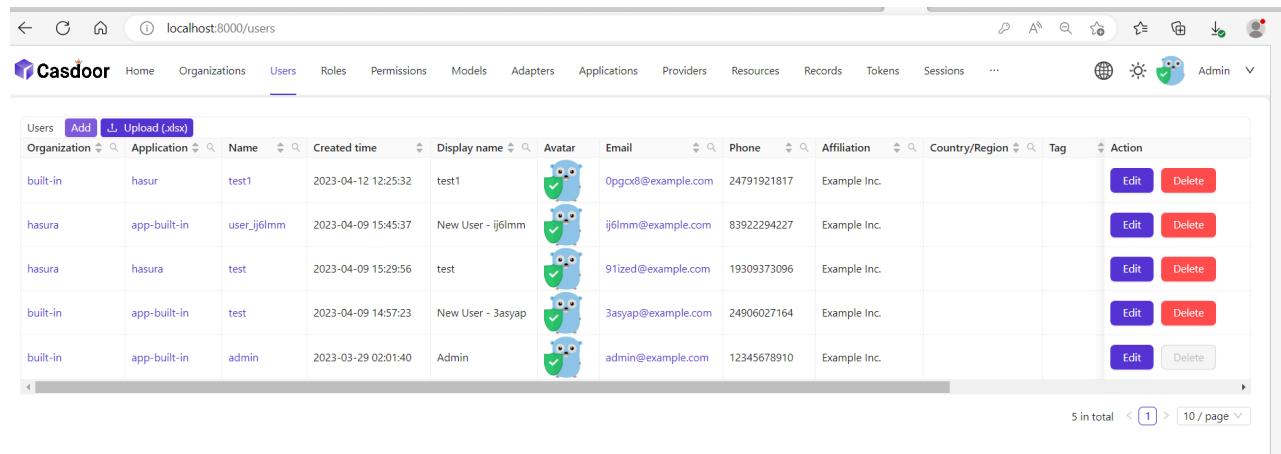
The screenshot shows a table titled "Redirect URLs" with one row. The "Redirect URL" column contains "`http://localhost:8080/login`". The "Action" column has three icons: a blue square with an upward arrow, a green circle with a checkmark, and a red square with a minus sign.

3. Copy the client ID; we will need it in the following steps.

## Add a user in Casdoor

Now that you have the application, but not a user. That means you need to create a user and assign the role.

Go to the "Users" page and click on "Add user" in the top right corner. That opens a new page where you can add the new user.



Organization	Application	Name	Created time	Display name	Avatar	Email	Phone	Affiliation	Country/Region	Tag	Action
built-in	hasur	test1	2023-04-12 12:25:32	test1		0pgcx8@example.com	24791921817	Example Inc.			<button>Edit</button> <button>Delete</button>
hasura	app-built-in	user_ij6lmm	2023-04-09 15:45:37	New User - ij6lmm		ij6lmm@example.com	83922294227	Example Inc.			<button>Edit</button> <button>Delete</button>
hasura	hasura	test	2023-04-09 15:29:56	test		91lized@example.com	19309373096	Example Inc.			<button>Edit</button> <button>Delete</button>
built-in	app-built-in	test	2023-04-09 14:57:23	New User - 3asyap		3asyap@example.com	24906027164	Example Inc.			<button>Edit</button> <button>Delete</button>
built-in	app-built-in	admin	2023-03-29 02:01:40	Admin		admin@example.com	12345678910	Example Inc.			<button>Edit</button> <button>Delete</button>

5 in total < 1 > [ 10 / page ]

Save the user after adding a username and adding the organization Hasura (other details are optional).

Now you need to set up a password for your user, which you can do by clicking "manage your password."

Choose a password for your user and confirm it.

# Build the Hasura App

Start the Hasura by Docker or Hasura Cloud.

Now create a `users` table with the following columns:

- `id` of type Text (Primary Key)
- `username` of type Text

Refer to the image below for reference.

The screenshot shows the Hasura Data Manager interface. On the left, there's a sidebar with 'Data Manager' and 'Databases (1)'. Under databases, it lists 'default' and 'public'. The 'SQL' tab is selected. In the main area, the title is 'Add a New Table'. The 'Table Name' field contains 'users'. The 'Table Comment' field contains 'comment'. Under 'CONFIGURE FIELDS', there's a section for 'Columns' with three rows. Each row has a column name ('id', 'username', 'column\_name'), a type ('Text'), a dropdown for 'default\_value', and checkboxes for 'Nullable' and 'Unique'. The 'Primary Key' section shows 'id' selected. There's also a 'Foreign Keys' section. A yellow circular icon with a hand icon is in the bottom right corner.

The next step is to create a `user` role for the app. Users should be able to see only their records but not other people's records.

Configure the `user` role as shown in the image below. For more information, read about [configuring permission rules in Hasura](#).

The screenshot shows the Hasura Cloud interface with the 'DATA' tab selected. A table lists roles and their permissions:

Role	insert	select	update	delete
admin	✓	✓	✓	✓
user	✗	✗	✗	✗

A modal window is open for the 'user' role's 'select' permission. It shows a custom check for row-level security:

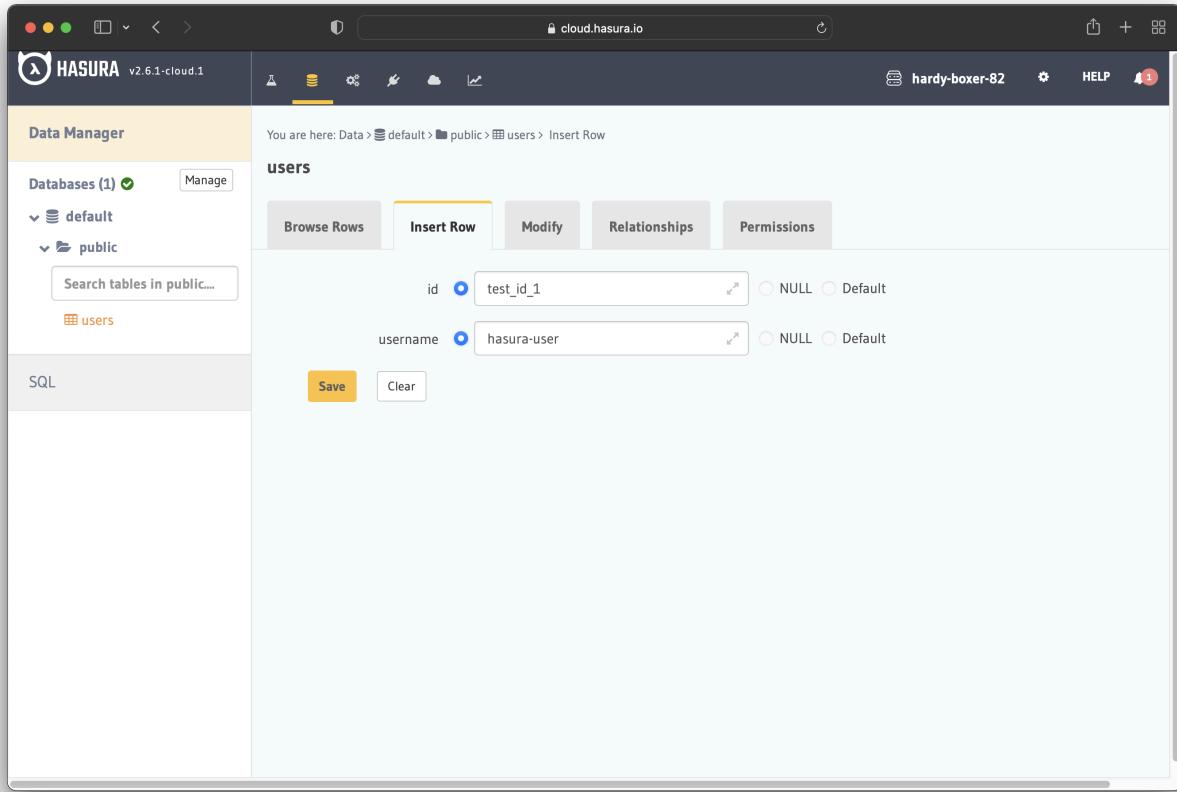
```
1  {"id": {"_eq": "X-Hasura-User-Id"}}

{
  "id": {
    "eq": "X-Hasura-User-Id"
  }
}
```

Below the modal, under 'Column select permissions', 'id' and 'username' are checked.

This way, users cannot read other people's records. They can only access theirs.

For testing purposes, add a dummy user. This is to ensure that when you use the JWT token, you only see your user's details and not other users' details.



Now you need to set the `JWT_SECRET` in Hasura.

## Configure Hasura with Casdoor

In this step, you need to add the `HASURA_GRAPHQL_JWT_SECRET` to Hasura.

To do so, go to the Hasura docker-compose.yaml and then add the new `HASURA_GRAPHQL_JWT_SECRET` as below.

The `HASURA_GRAPHQL_JWT_SECRET` should be in the following format. Remember to change `<Casdoor endpoint>` to your own Casdoor's URL (like <https://door.casdoor.com>)

```
HASURA_GRAPHQL_JWT_SECRET: '{"claims_map": {  
    "x-hasura-allowed-roles": {"path": "$.roles"},  
    "x-hasura-default-role": {"path": "$.roles[0]"},  
    "x-hasura-user-id": {"path": "$.id"}  
}, "jwk_url": "<Casdoor endpoint>/.well-known/jwks"}'
```

Save the change and reload the docker.

```
## enable debugging mode. It is recommended to disable this in production
HASURA_GRAPHQL_DEV_MODE: "true"
HASURA_GRAPHQL_ENABLED_LOG_TYPES: startup, http-log, webhook-log, websocket-log, query-log
HASURA_GRAPHQL_ADMIN_SECRET: myadminsecretkey
HASURA_GRAPHQL_JWT_SECRET: '{"claims_map": {
  "x-hasura-allowed-roles": ["user", "editor"],
  "x-hasura-default-role": "user",
  "x-hasura-user-id": "4ec7ccee-ec7b-4191-a78d-e11f50686f8b"
}, "jwk_url": "https://door.casdoor.com/.well-known/jwks"}'
```

## Retrieve the JWT Token

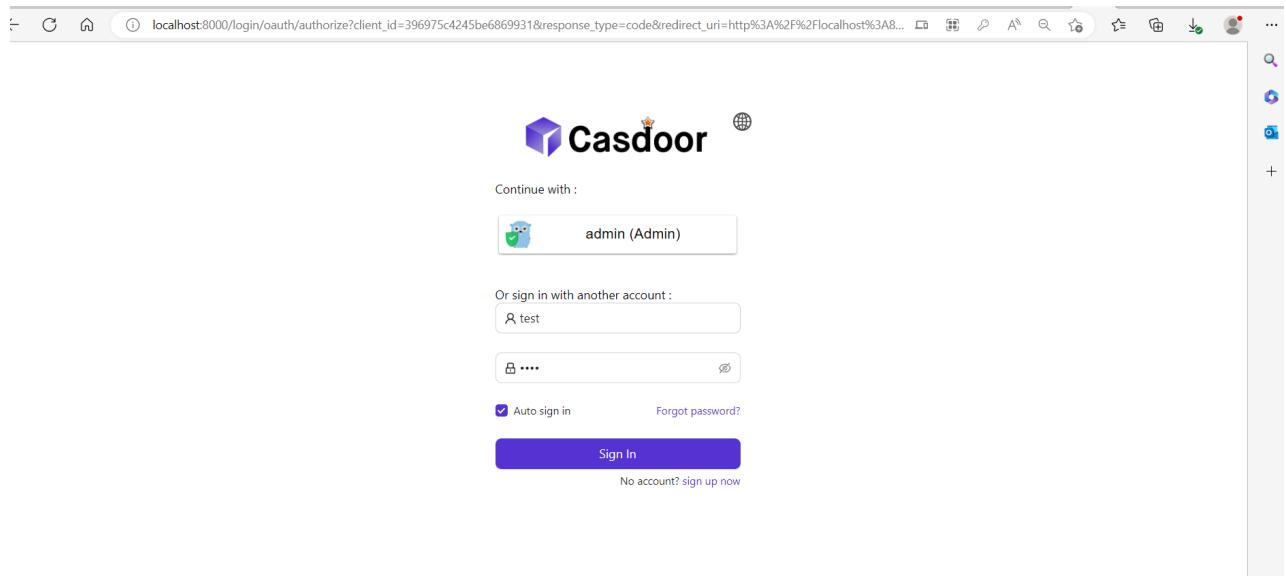
Since there is no client implementation, you can get your access token by making a request by the below URL:

```
http://localhost:8000/login/oauth/authorize?client_id=<client
ID>&response_type=code&redirect_uri=http%3A%2F%2Flocalhost%3A8080%2Flogin&scope=read&state=app-
built-in<public certificate>>
```

Change `client ID` to the ID you copied before and input the public certificate of Casdoor, which you can find in Casdoor's Certs page.

Then input the username and password you created for Hasura before.

Click "Sign in"



Go back to the Casdoor/Token page.

Tokens	Add	Name	Created time	Application	Organization	User	Authorization code	Access token	Action
b6ea3e35-abcdef-41d8-a1a2-01f00fd8264b	2023-04-12 13:06:53	hasura	hasura	test	433b504b4f6a593e4a11	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
16024557-df21-4779-bfb9-959e5dae078c	2023-04-12 12:51:47	hasur	built-in	test1	2879fb282019cf723c	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
f3cb1070-c2d4-40f0-8bc0-59919d26d162	2023-04-11 15:04:00	hasura	hasura	test	2a370971798d403fc6ef	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
64993582-2322-4df7-ab20-cb23201bc77b	2023-04-11 00:37:22	springboot	built-in	admin	a2396037c3ba4fd9221e	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
f65a3813-a655-4f70-9c9a-f08ce4607815	2023-04-11 00:31:37	springboot	built-in	admin	d048c79cd1469fd829d	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
5828069e-15eb-4c92-933c-fecda8ed621c	2023-04-11 00:06:54	springboot	built-in	admin	7cc27dc752cc4188ac8d	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
2277e0f2-7e78-462f-a654-3c53759784af	2023-04-11 00:05:17	springboot	built-in	admin	56141e709a06931b7faa	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
55bd324a-6039-40f6-b707-2a55d78ae911	2023-04-11 00:05:07	springboot	built-in	admin	9a1413bc172591a64353	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
4b30acbe-fa22-4387-8098-9a46e70f6972	2023-04-10 23:59:19	springboot	built-in	admin	88b0997b675917f0fdc	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	
1a1a-5a7a-47a1-77a2-111111111111	2023-04-10 23:59:19	springboot	built-in	admin	1a1a-5a7a-47a1-77a2-111111111111	eyJhbGciOiJSUz1NilsImtpZC16ImNlcnQtYnVpbHQtaW4iLCj0e		<button>Edit</button> <button>Delete</button>	

Find the Username you input before, then click "edit"

### Copy the Access Token

Token details

Edit Token	Save	Save & Exit
Name:	b6ea3e35-abcf-41d8-a1a2-01f00fd8264b	
Application:	hasura	
Organization:	hasura	
User:	test	
Authorization code:	433b504b4f6a593e4a11	
Access token:	eyJhbGciOiJSUzI1NiIsImtpZCI6ImNlcnQtYnVpbHQtaW4iLCJ0eXAiOiKV1QifQ.eyJvd25lci6Imhhc3VySislm5hbWUiOiJ0ZXN0liwiY3JlYXRIZFRpbWUiOilyMDIzMzTA0LTA1VDE1OjI5OjU2KzA4OjAwIiwidXbkyXRIZFRpbWUiOiiLCjpZC16ijRy	
Expires in:	604800	
Scope:	read	
Token type:	Bearer	

Save Save & Exit

Now you can use the access token to make the authenticated request. Hasura returned the appropriate user rather than returning all the users from the database.

HASURA v2.22.0 API DATA ACTIONS REMOTE SCHEMAS EVENTS SETTINGS HELP Allow List

GraphQL REST

> GraphQL Endpoint  
Request Headers

ENABLE	KEY	VALUE
<input type="checkbox"/>	Hasura-Client-Name	casdoor
<input checked="" type="checkbox"/>	content-type	application/json
<input type="checkbox"/>	x-hasura-admin-secret	*****
<input checked="" type="checkbox"/>	Authorization	Bearer eyJhbGciOiJSUzI1NiIsImtpZCI6ImNlcnQtYnVpbHQtaW4iLCJ0eXAiOiJV1QfQeyJvd25ci6lmhhc3VyYSlsIm5hbWL
Enter Key		Enter Value

Explorer X GraphiQL ▶ Prettify History Explorer Code Exporter REST Derive action Analyze ◀ Docs

query MyQuery {  
 users {  
 id  
 username  
 }  
}

1  
2  
3  
4  
5  
6  
7

{  
 "data": {  
 "users": [  
 {  
 "id": "4ec7ccce-ec7b-4191-a78d-e11f50686f8b",  
 "username": "test"  
 }  
 ]  
 }  
}

QUERY VARIABLES



> 集成 >

Python

# Python



JumpServer

Using CAS to connect JumpServer

# JumpServer

[Casdoor](#) can be used to connect [JumpServer](#).

The following are some of the names in the configuration:

`CASDOOR_HOSTNAME`: The domain name or IP where Casdoor server is deployed.

`JumpServer_HOSTNAME`: The domain name or IP where JumpServer is deployed.

## Step 1: Deploy Casdoor and JumpServer

Firstly, deploy [Casdoor](#) and [JumpServer](#).

After successful deployment, ensure the following:

1. Casdoor can be logged in and used normally.
2. You can set `CASDOOR_HOSTNAME` to `http://localhost:8000` when deploying Casdoor in `prod` mode. See [production mode](#).

## Step 2: Configure Casdoor application

1. Create a new Casdoor application or use an existing one.
2. Find a redirect URL: `CASDOOR_HOSTNAME/cas/your_organization/your_application/login`.
3. Add your redirect URL to the Casdoor application: `JumpServer_HOSTNAME`.

For more information about [CAS](#), refer to the documentation.

# Step 3: Configure JumpServer

## 1. Find Auth:

JumpServer

Administrator

Settings

Basic

Email

Auth

Message

Terminal

Applets

Security

Period clean

Tools

Tasks

Other

License

Auth

Basic

CAS

Enable CAS Auth

Server url

Proxy server url

Version

Logout completely

User attr map

```
1 = [  
2   "uid": "username"  
3 ]
```

Create user if not

Reset

Submit

## 2. Configure this app:

JumpServer

Administrator

Settings

Basic

Email

Auth

Message

Terminal

Applets

Security

Period clean

Tools

Tasks

Other

License

Auth

Basic

CAS

Enable CAS Auth

your casdoor

your orgnazition

your application

Server url

Proxy server url

Version

Logout completely

User attr map

```
1 = [  
2   "uid": "username"  
3 ]
```

Create user if not

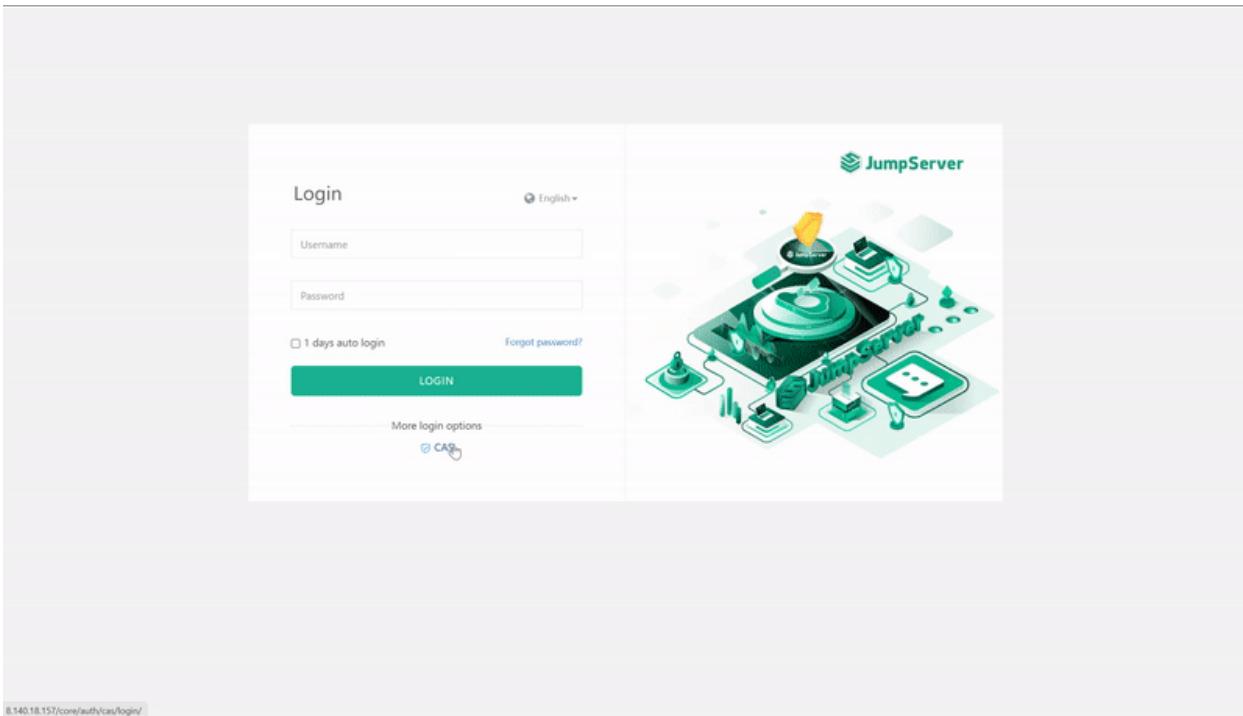
Reset

Submit

- `/login` endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/login>.
- `/logout` endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/logout>.
- `/serviceValidate` endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/serviceValidate>.
- `/proxyValidate` endpoint: <https://door.casdoor.com/cas/casbin/cas-java-app/proxyValidate>.

For more information about [CAS](#) and [JumpServer](#), refer to the documentation.

Log out of JumpServer and test SSO:





&gt;

监控

# 监控



## Web UI

Monitor runtime information on the Casdoor web page



## Prometheus

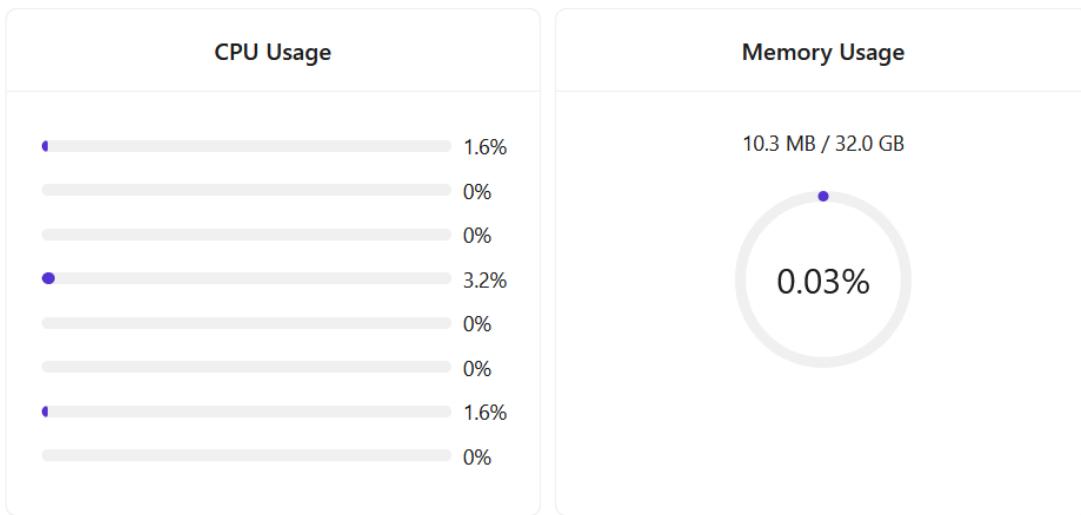
Use Prometheus to collect information about running Casdoor.

# Web UI

You can monitor the runtime information of Casdoor on the [Casdoor web page](#), including CPU Usage, Memory Usage, API Latency, and API Throughput.

On the UI, you can view the following information:

- CPU Usage and Memory Usage



- API Latency, including count times and average latency

API Latency				
Method	Endpoint	Latency (ms)	Avg Latency (ms)	Throughput (req/s)
GET	/api/get-cert	3	0.667	
GET	/api/get-certs	27	1.333	
GET	/api/get-chats	27	1.519	
GET	/api/get-default-application	3	5.333	
GET	/api/get-email-and-phone	1	1.000	
GET	/api/get-global-providers	58	1.202	

- API Throughput, including total throughput and throughput per API

API Throughput			
Total Throughput: 2			
Name	Method	Throughput	
/api/get-prometheus-info	GET	1	
/api/get-system-info	GET	1	

# Prometheus

To collect Casdoor's runtime metrics, such as API Throughput, API Latency, CPU Usage, Memory Usage, and more, you need to configure your Prometheus profile.

```
global:  
  scrape_interval: 10s # The time interval for fetching metrics  
  
scrape_configs:  
  - job_name: 'prometheus'  
    static_configs:  
      - targets: ['localhost:9090']  
  - job_name: 'casdoor' # Name of the application to be monitored  
    static_configs:  
      - targets: ['localhost:8000'] # Back-end address of Casdoor deployment  
    metrics_path: '/api/metrics' # Path for collecting indicators
```

After successful configuration, you will find the following information in Prometheus:





&gt;

国际化

# 国际化

Casdoor支持多种语言. 通过部署[Crowdin](#) 翻译, 我们可以提供 西班牙语、法语、德语、中文、印尼语、日语、韩语等等的支持.

Casdoor使用官方的 Crowdin cli 来同步Crowdin 的翻译。 如果您想要添加对其他语言的支持, 请提交您在 [社区](#) 中的提案。 此外, 如果您想要帮助我们加快翻译工作, 请考虑帮助我们翻译 [Crowdin](#)。



&gt;

Contributor Guide

# Contributor Guide

欢迎使用 Casdoor！ This document serves as a guideline on how to contribute to Casdoor.

If you find any incorrect or missing information, please leave your comments or suggestions.

## Get Involved

有许多方式可以为Casdoor做贡献。 以下列出了一些贡献方式：

- Use Casdoor and report issues. When using Casdoor, report any issues - whether they're bugs or proposals - on [GitHub Discussions](#) or on [Discord](#) before filing an issue on GitHub.

### ① 信息

Please use English to describe the details of your problem when reporting an issue.

- Help with documentation. Starting your contribution work with docs is a good choice.
- Help solve issues. We have a table that contains easy tasks suitable for beginners under [Casdoor Easy Tasks](#), with different levels of challenges labeled with different tags.

# 贡献

If you are ready to create a PR, here is the workflow for contributors:

1. Fork to your own repository.
2. Clone your fork to a local repository.
3. Create a new branch and work on it.
4. Keep your branch in sync.
5. Commit your changes. Make sure your commit message is concise.
6. Push your commits to your forked repository.
7. 创建从您的分支到我们的**master**分支的合并请求。

# 合并请求

## Before You Get Started

Casdoor uses GitHub as its development platform, and pull requests are the primary source of contributions.

Here are some things you need to know before opening a pull request:

- You need to sign the CLA when you first create a pull request.
- Explain why you are submitting the pull request and what it will do to the repo.

- Only one commit is allowed. If the PR does more than one thing, please split it.
- If there are any newly added files, please include the Casdoor license at the top of the new file(s).

```
// Copyright 2022 The Casdoor Authors. All Rights Reserved.  
//  
// Licensed under the Apache License, Version 2.0 (the "License");  
// you may not use this file except in compliance with the License.  
// You may obtain a copy of the License at  
//  
//     http://www.apache.org/licenses/LICENSE-2.0  
//  
// Unless required by applicable law or agreed to in writing,  
// software  
// distributed under the License is distributed on an "AS IS"  
// BASIS,  
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or  
// implied.  
// See the License for the specific language governing permissions  
// and  
// limitations under the License.
```

## Semantic PRs

您的合并请求应遵循常规承诺的样本。 基本要求是有PR 标题或至少一个 提交消息。 例如，三个常用的PR标题如下：



### 注意事项

PR titles must be in lowercase.

1. fix: a commit of the type `fix` patches a bug in your codebase.

fix: prevent racing of requests

2. **feat**: a commit of the type `feat` introduces a new feature to the codebase.

feat: allow provided config object to extend other configs

3. **docs**: a commit of the type `docs` adds or improves documentation.

docs: correct spelling of CHANGELOG

For more details, please refer to the [Conventional Commits](#) page.

## Linking PRs with Issues

You can link a pull request to an issue to show a fix is in progress and to automatically close the issue when the pull request is merged.

### Linking a Pull Request to an Issue Using a Keyword

您可以通过在拉取请求说明或提交消息中使用支持的关键词将拉取请求链接到议题。 拉取请求**必须**在默认分支上。

- close
- fix
- resolve

An issue in the same repository, for instance:

Fix: #902

For more details, please refer to [Linking a Pull Request to an Issue](#).

## Modifying PRs

Your PR may need revision. Please modify the same PR when the code needs changes; don't close the PR and open a new one. Here is an example:

- Modify the code on your local.
- Modify your commit.

```
git commit --amend
```

- 推送代码到远程仓库

```
git push --force
```

Then, you will have successfully modified the PR!

## 相关代码

Some Principles:

- Readability: important code should be well-documented. Code style should comply with the existing one.

## Naming Convention

For instance, `signupUrl` for variable names, `Signup URL` for UI.

## How to Update i18n Data?

Please note that we use [Crowdin](#) as a translating platform and i18next as a translating tool. When you add strings using i18next in the `web/` directory, you can run the `i18n/generate_test.go` to auto-generate `web/src/locales/**/data.json`.

Run `i18n/generate_test.go`:

```
cd i18n && go test
```

默认情况下，所有语言都以英文填写。After your PR has been merged, you are encouraged to help translate the newly added strings in `web/src/locales/zh/data.json` by [Crowdin](#).

### ⚠ 注意事项

If you are not familiar with a language, please do not translate it; keep the file as it is.

## 许可证书

By contributing to Casdoor, you agree that your contributions will be licensed under the Apache License.