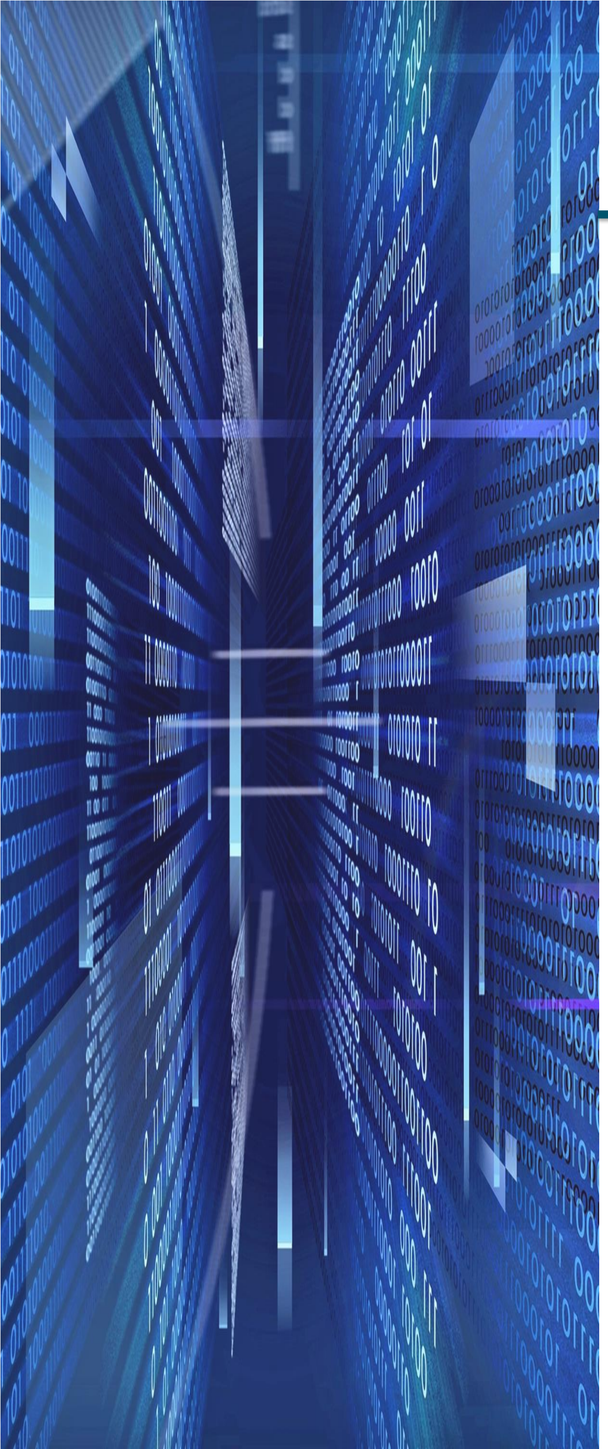


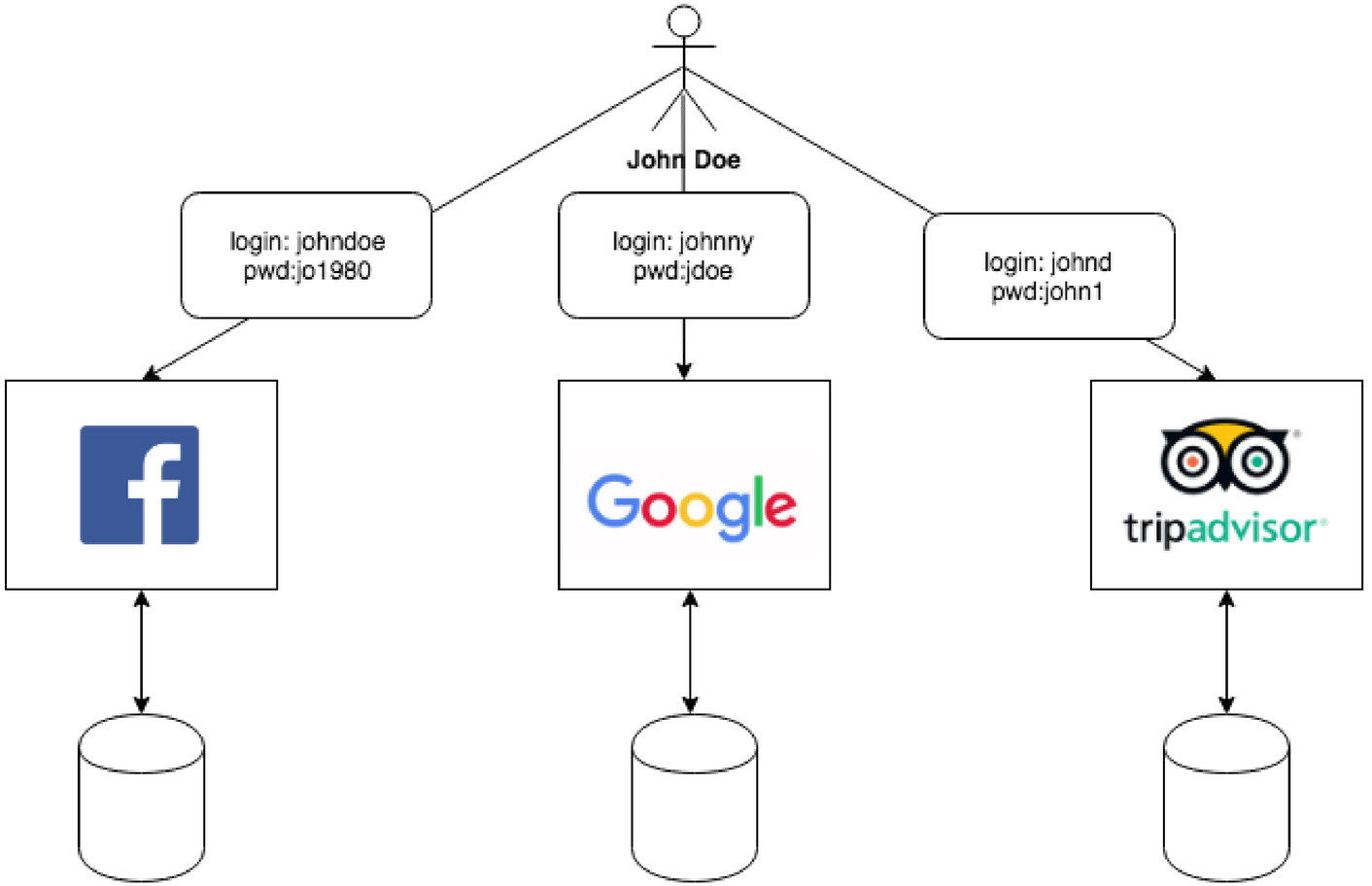
Digital Identity Management

Prof Federica Paci

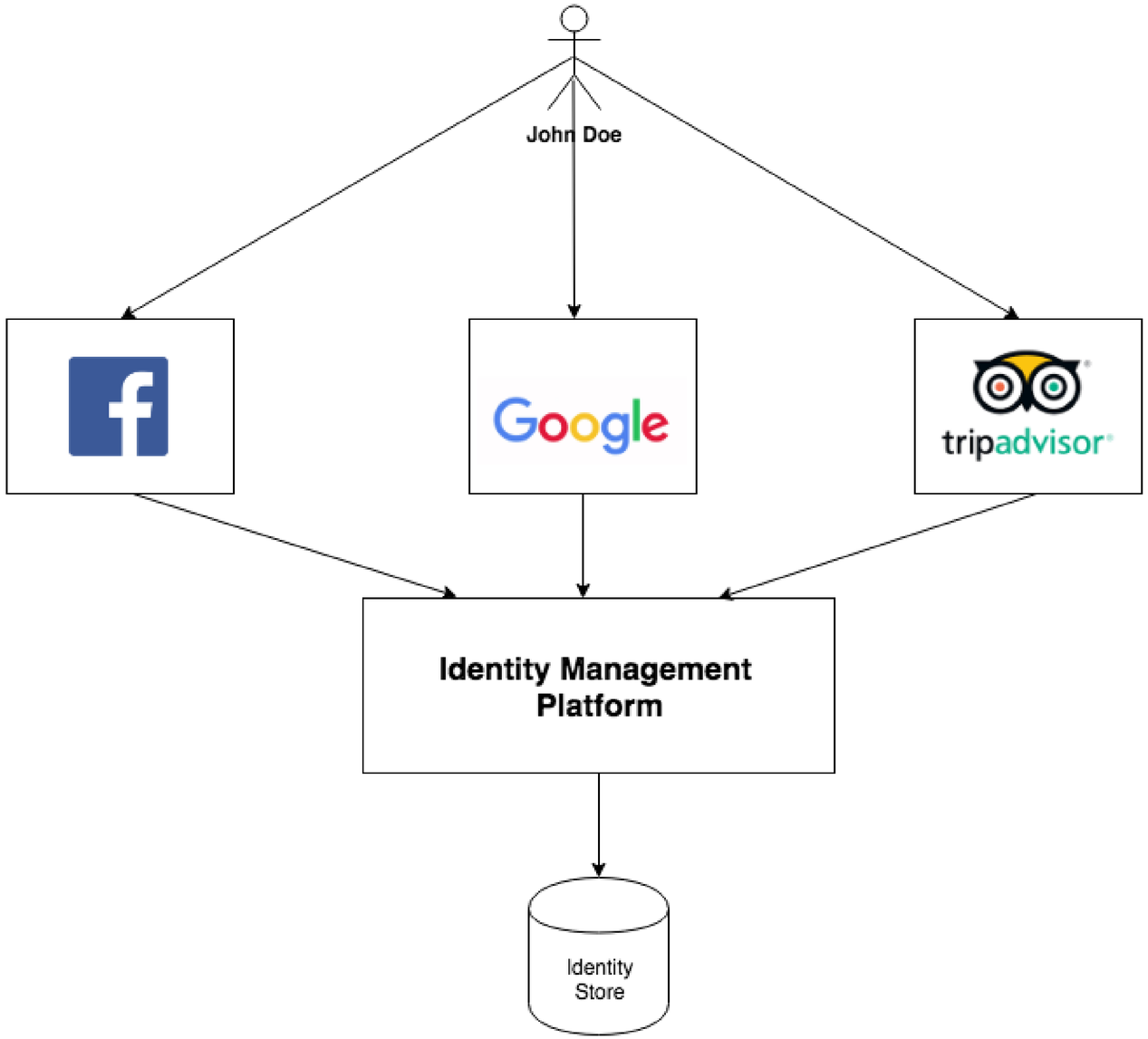
Lecture Outline

- Digital identity management
  - Digital identity
  - Single sign on
  - Federated Identities
  - Standards and Protocols
    - SAML
    - Shibboleth
    - OAuth and OpenID Connect

The problem


John Doe
login: johndoe
pwd:jo1980
login: johnny
pwd:jdoe
login: johnd
pwd:john1
Google
tripadvisor

The solution



Digital Identity Management

- A **digital identity** is the digital representation of the information know about a specific individual
  - Name and last name
  - National insurance number
  - Home address
  - Job title
  - User id and password
- A digital identity management system provides a centralized solution that manages users' digital identities and user access to resources/services.
  - Maintain identity of the user and associates attributes to this identity
  - Verify identity of the user based on his/her identity attributes

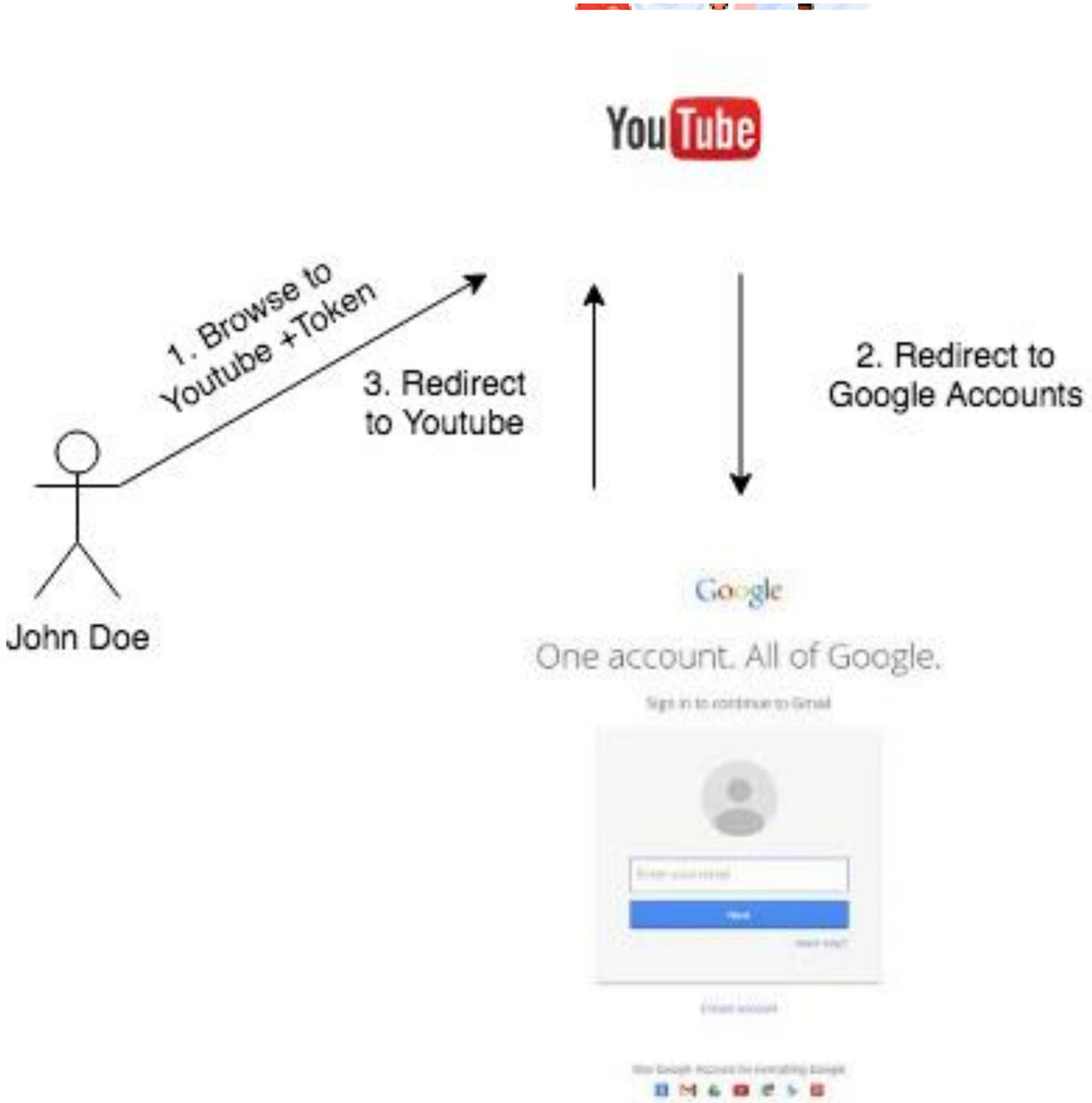
Main Players

- **Subject or User**
  - System entity about which something can be asserted
- **Asserting Party or Identity Provider**
  - System entity that creates assertions about a subject
- **Relying Party or Service Provider**
  - System entity that consumes assertions about a subject

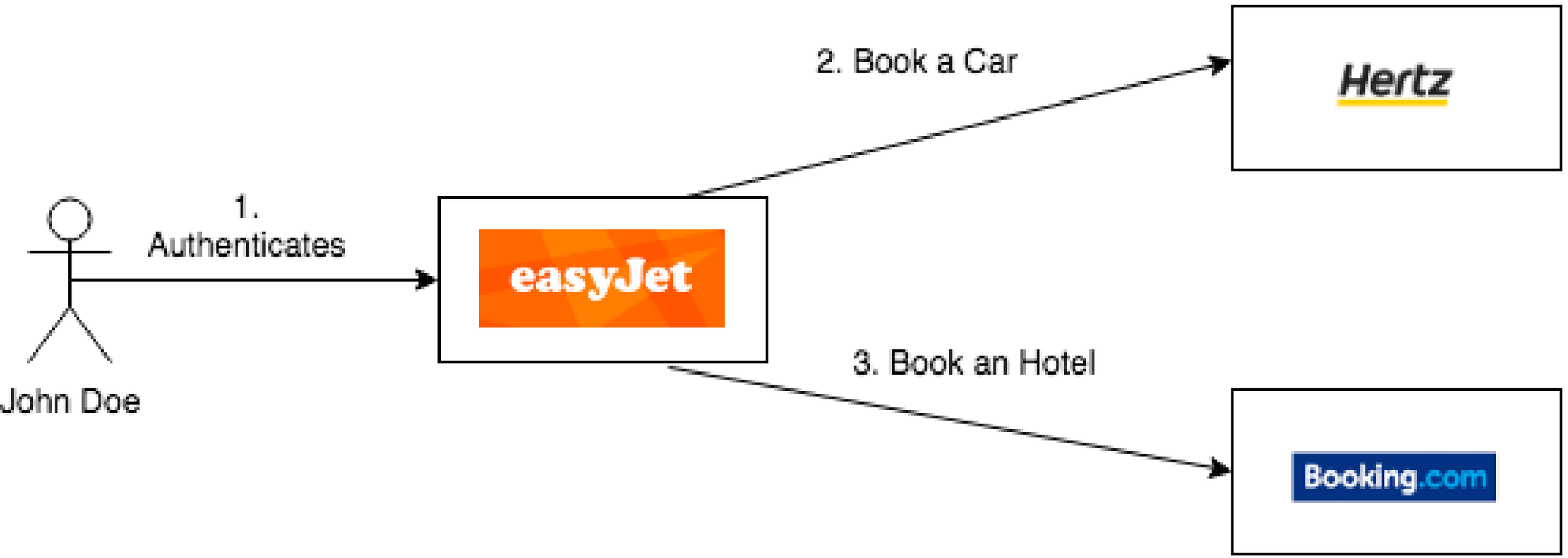
What is Single-Sign On

- User authenticates once and then access all the resources the user is authorized to use
- Authentication to the individual resources is handled by the Identity Provider in a manner that is transparent to the user
- Identity Provider maintains the identity information of the user
- When the user has to authenticate again for a resource, the Identity Provider does the job for the user

An Example of SSO



An Example of Cross Domain SSO



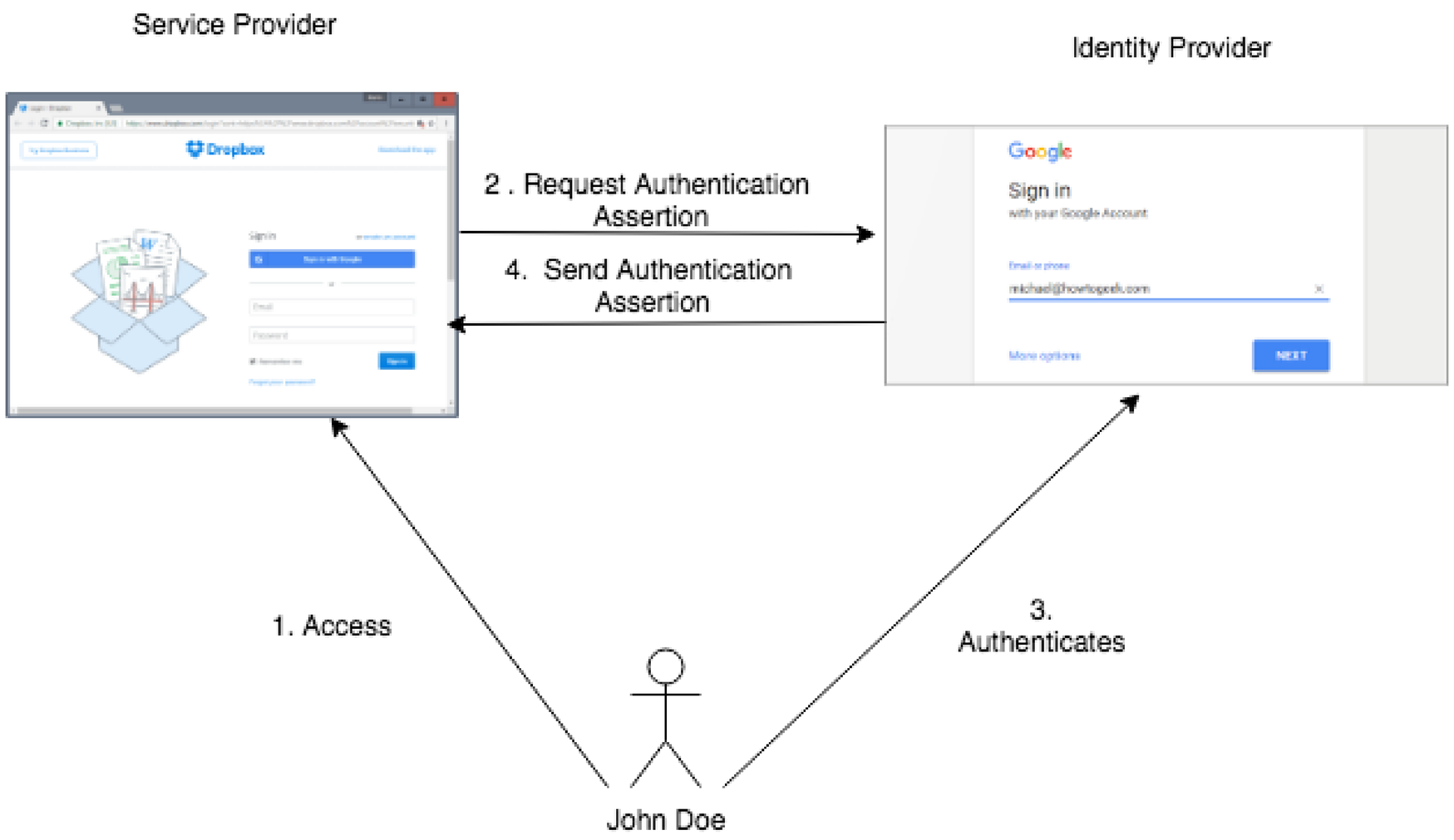
Federated Identity

- Organizations reach an agreement and establish a common, shared identifier to refer to a subject
- Facilitate sharing on identity information across different organisations
- Facilitates SSO
  - An end-user that "logs into" any member of the federation has effectively logged into all of the members
- Reduces costs of maintaining and managing identities
  - No need independently and maintain collect identity information

SAML

- SAML enables SSO and Identity Federation by providing a standard representation for attribute assertions and authentication assertion
- SAML Asserting Party/Identity Provider verifies identity of a user and issues an authentication assertion
- The user can present to a Service Provider the authentication assertion without authenticating again

SAML Use Cases – SP Initiated SSO



SAML Authentication Request <AuthnRequest>

- The following attributes and elements have to be present:
  - **ID** Newly generated number for identification
  - **Version**, SAML current version which is 2.0
  - **IssueInstant** the time when the request was issued UTC format
  - **AssertionConsumerServiceURL** The SAML URL interface of the service provider, where the Identity provider sends the authentication token
  - **<Subject>** to specify the user to be authenticated
  - **<Issuer>** to indicate the unique identifier of the Service Provider who generated the request
  - **<NameIDPolicy>** it indicates the level of security required for the user authentication

SAML <Response>

- It must contain the following attributes and elements
  - **ID** unique identifier of the response
  - **Version**, SAML version
  - **IssueInstant** the time the response has be issued in UTC format;
  - **InResponseTo**, the ID of the authentication request;
  - **Destination**, the URI of the Service provider
  - **<Status>** indicates whether the authentication was successfull or not
  - **<Issuer>** the identifier of the Identity Provider
  - **<Assertion>** contains information about the authentication method and eventually identity attributes of the user
  - **<Signature>** contains the digital signature of the Identity Provider

SAML Authentication Assertion

- It has to contain contain the following attributes and elements:
  - **ID** unique identifier
  - **Version,** SAML version
  - **IssueInstant** issuance time in UTC format
  - **<Subject>** indicates the user to be authenticated
  - **<Issuer>** indicates the Identity Provider
  - **<Conditions>** specifies the time interval when the assertion is valid
  - **<AudienceRestriction>** specifies the Service Provider that is intended to consume the assertion
  - l'elemento **<AuthStatement>** specifies the authentication context
  - **<AttributeStatement>** lists the identity attributes certified by the Identity Provider
  - **<Signature>** contains the signature of the Identity Provider

SAML Assertions

- Declaration of facts about a subject that an asserting party claims to be true
- Three types of assertions
  - **Authentication statements**
    - describe the means used to authenticate a subject
  - **Attribute statements**
    - list attributes that a subject has
  - **Authorization statements**
    - Define subject's permissions

SPID


INPS
L'Istituto
Dati, ricerche e bilanci
Avvisi, bandi e fatturazione
INPS Comunica
Prestazioni e servizi
Amministrazione trasparente
Assistenza
Contatti
Dichiarazioni di accessibilità
Indietro
Vai a MyINPS
Cerca
/ Prestazioni e Servizi / Autenticazione
Autenticazione
PIN
SPID
CIE
CNS
SPID è il sistema di accesso che consente di utilizzare, con un'identità digitale unica, i servizi online della Pubblica Amministrazione e dei privati accreditati. Se sei già in possesso di un'identità digitale, accedi con le credenziali del tuo gestore. Se non hai ancora un'identità digitale, richiedila ad uno dei gestori.
Maggiori informazioni su SPID
Entra con SPID
spid
AgID Agenzia per l'Italia Digitale

Main Players

- **Agenzia per l’Italia Digitale (AgID):** the entity that monitors and authorizes entities to issuing SPID
- **Identity Provider:** private and public entity certified by AgID which has to verify the identity of user and issue SPID
- **Service Provider**: private or public entity which offers online services
- **Attribute Provider:** entity which issues to users the qualifying attributes
- **User:** owner of SPID who uses it to access online services

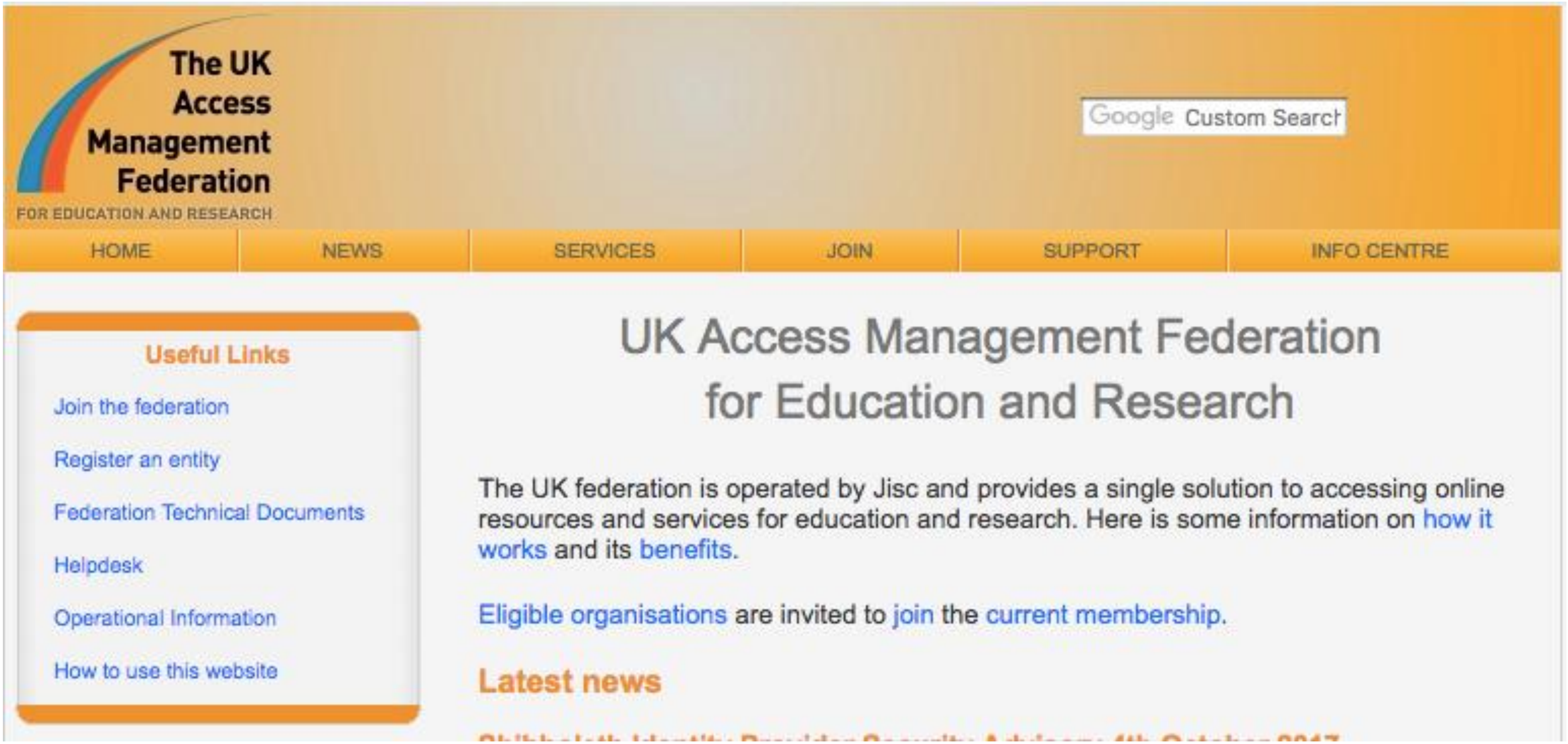
SPID levels of secure authentication

- **level 1**, allows access with username and password;
- **level 2**, allows access with SPID level 1 credentials and the generation of a temporary OPT access code (one time password) or the use of an APP accessible from smartphone or tablet;
- **level 3**, allows access with SPID credentials and the use of additional security solutions and any physical devices (e.g. smart card) that are granted by the identity provider

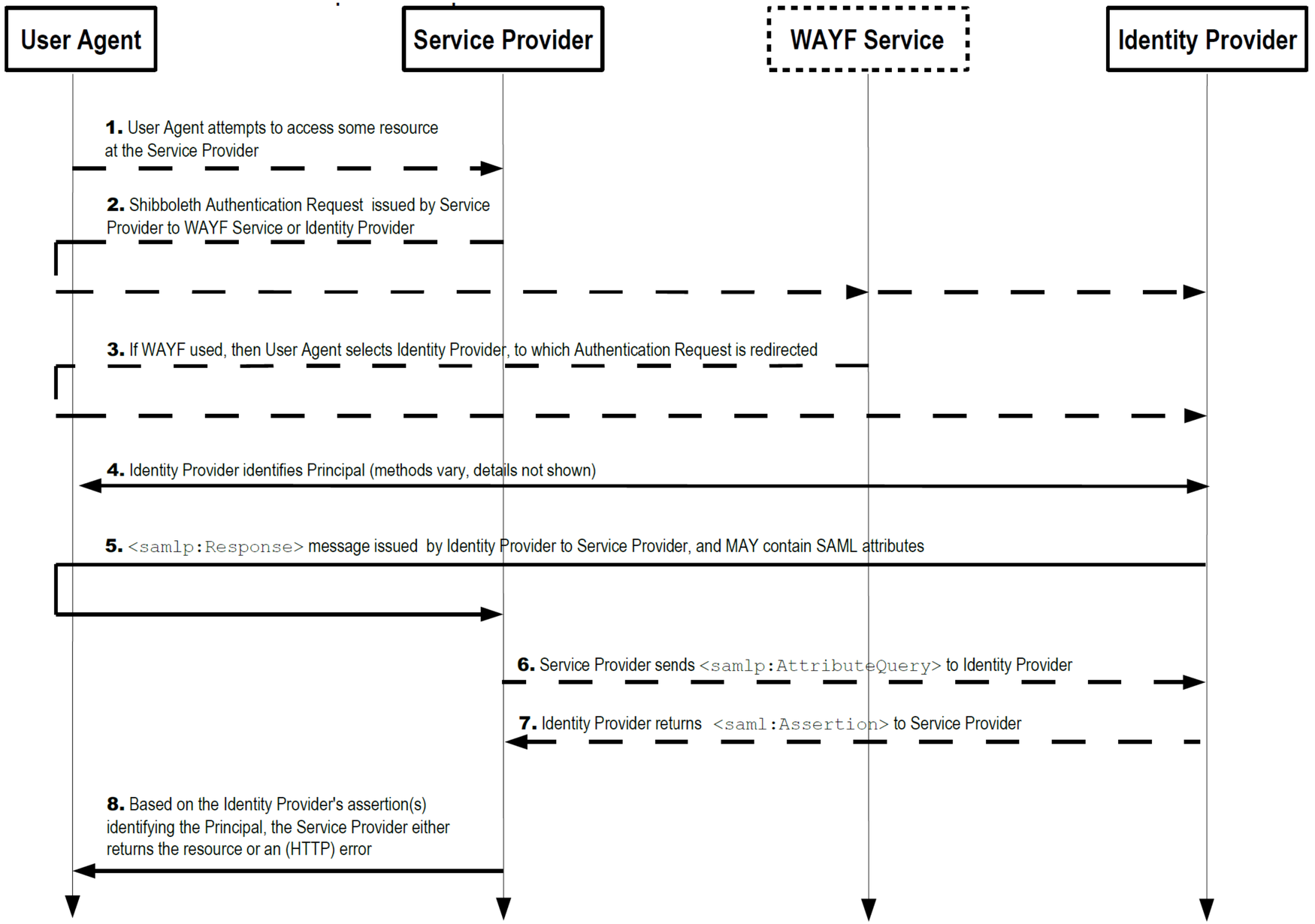
Shibboleth

- Shibboleth is an Internet2 consortium project that enables universities to share resources and research across institutional boundaries.
- Shibboleth enable students, faculty, and staff to access resources at partner institutions without needing to create separate, local IDs and passwords for each one

An example of federation



Shibboleth authentication flow



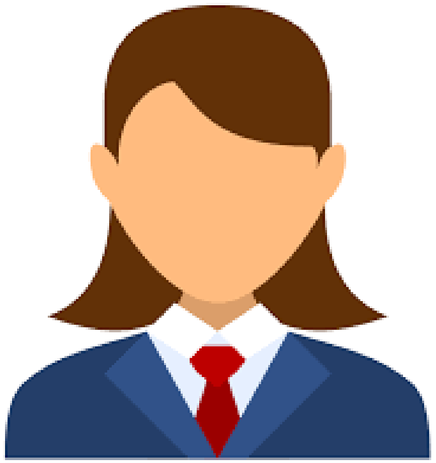
Summary

- SSO simplifies user authentication across different web sites
  - User login once at a web site and then can access resources to other web sites
- Federated SSO allows users to access websites from other service providers
- Federated SSO is implemented through standardization efforts
  - SAML: XML-based framework
  - Shibboleth: SAML-based protocol for educational institutions
  - OpenId Connect: single sign on protocol for web, mobile and cloud apps

OpenID Connect

- OpenID Connect (OIDC) is an open authentication protocol that profiles and extends OAuth 2.0 to add an identity layer.
- OIDC allows clients to confirm an end user’s identity using authentication by an authorization server.
- Implementing OIDC on top of OAuth 2.0 creates a single framework that promises to secure APIs, mobile native applications and browser applications in a single, cohesive architecture.

A simple scenario



Resource Owner

Client Application

Authorization Server/
Resource Server

Protected Resources

What is OAuth?

- It is a standard **authorization protocol** that allows a third-party application to access protected resource hosted on a HTTP server
- It requests an **access token** from the Authorization Server
- Then the third-party application uses the access token to request access to the protected resources

Actors

- **Resource Owner:** entity capable of granting access to a protected resource
- **Resource Server**: the server that stores that resource owner resources
- **Authorization Server:** the server issuing access token to the client after authenticating the resource owner and obtaining its authorization
- **Client:** a third-party application that requests access to protected resources on-behalf of resource owner and with its approval

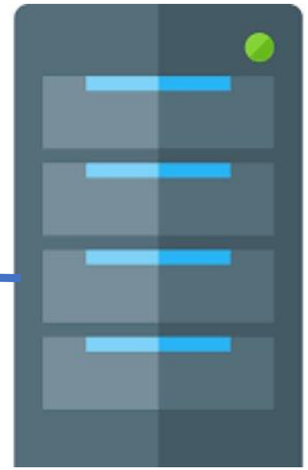
Authorization Flows

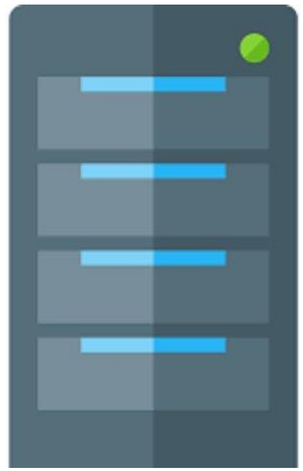
- Authorization Code Grant Flow
- Authorization Code Grant Flow with PCKE
- Resource Owner Password
- Client Credential
- Device Flow

OAuth -The Authorization Code Flow

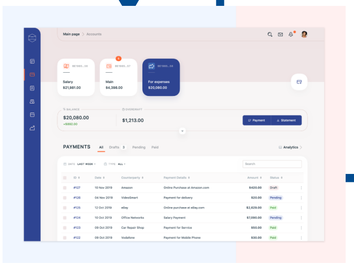
Alice – Resource Owner

Credit Union Bank - Authorization Server





https://example.com/oauth2/register?
application_name=Mybank&
redirect_uri=https://example.com/callback&
grant_type=code



client_id & client_secret

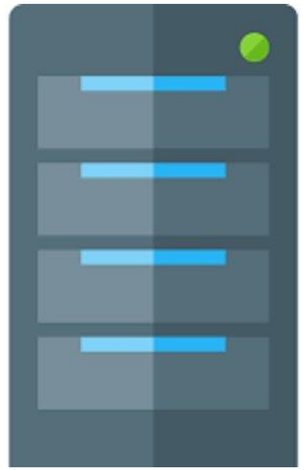
My Bank App – Client Application

Credit Union Bank - Resource Server

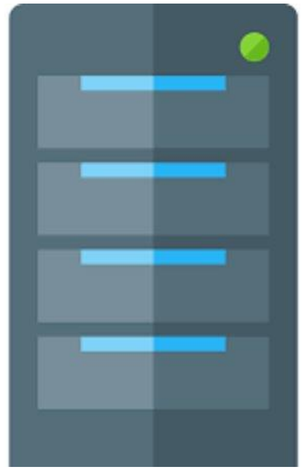
OAuth - The Authorization Code Flow

Alice – Resource Owner

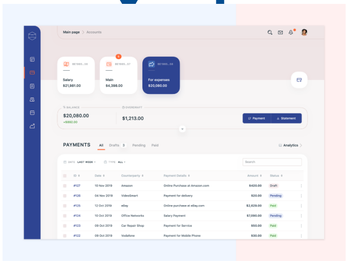
Credit Union Bank - Authorization Server



Alice Authenticates



https://example.com/oauth2/authorize?
response_type=code&
client_id=<client id>&
redirect_uri=https://example.com/callback&
scope=bankaccount&
state=123xyz



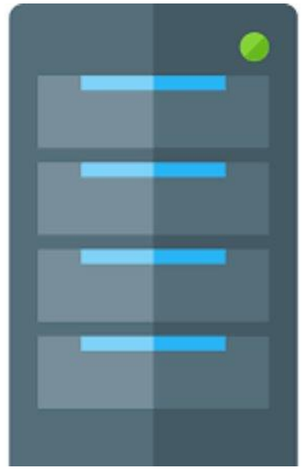
My Bank App – Client Application

Credit Union Bank - Resource Server

OAuth -The Authorization Code Flow

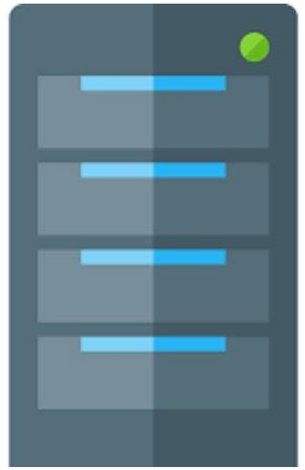
Alice – Resource Owner

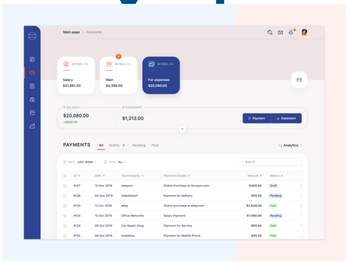
Credit Union Bank - Authorization Server



Authorization Code

https://example.com/oauth2/authorize?code=8042d4cad58c66d0a5edfad8952192&state=123xyz



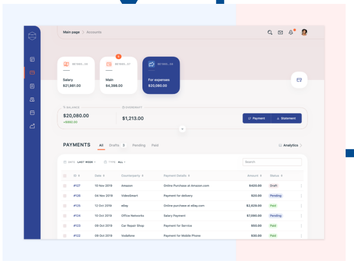


My Bank App – Client Application

Credit Union Bank - Resource Server

OAuth - The Authorization Code Flow

Alice – Resource Owner

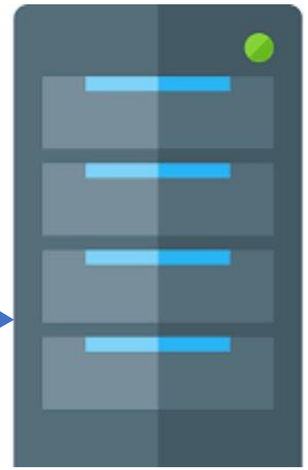


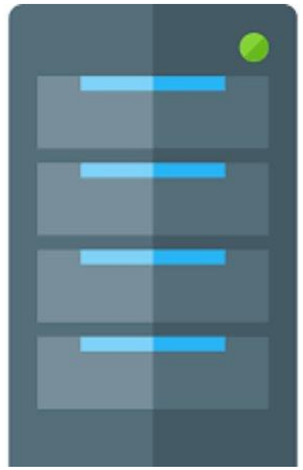
My Bank App – Client Application

POST https://example.com/oauth2/token
grant_type=authorization_code&
code=8042d4cad58c66d0a5edfad8952192&
redirect_uri=https://example.com/callback&
client_id=<client id>&
client_secret=<client secret>

Authorization Code & Redirection URI

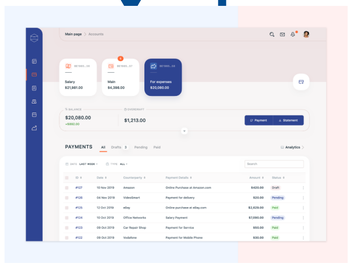
Credit Union Bank - Authorization Server





Credit Union Bank - Resource Server

OAuth -The Authorization Code Grant Flow



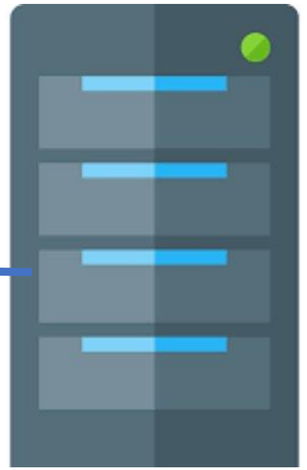
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8

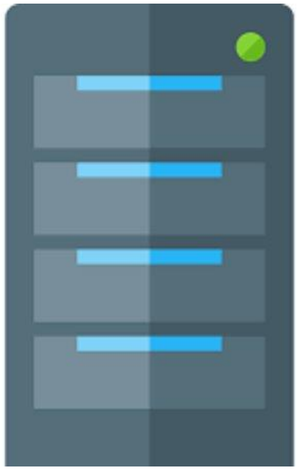
```
{
  "access_token":"2YotnFZFEjr1zCsicMWpAA",
  "token_type":”bearer",
  "expires_in":3600,
  "refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA”
}
```

Access token & Refresh token

My Bank App –
Client Application

Credit Union Bank -
Authorization Server

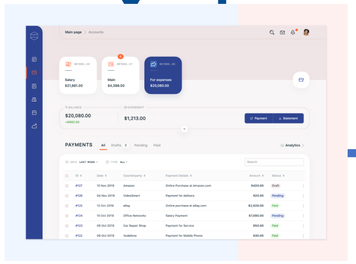




Credit Union Bank -
Rsource Server

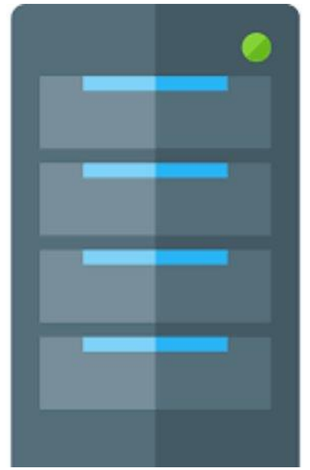
OAuth - The Authorization Code Flow

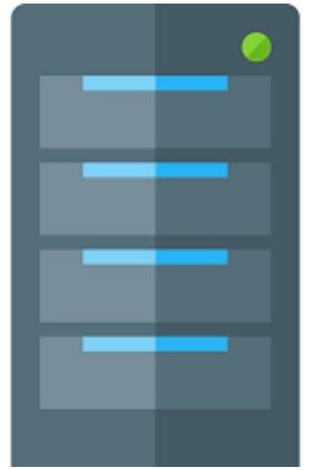
Alice – Resource Owner



Access token

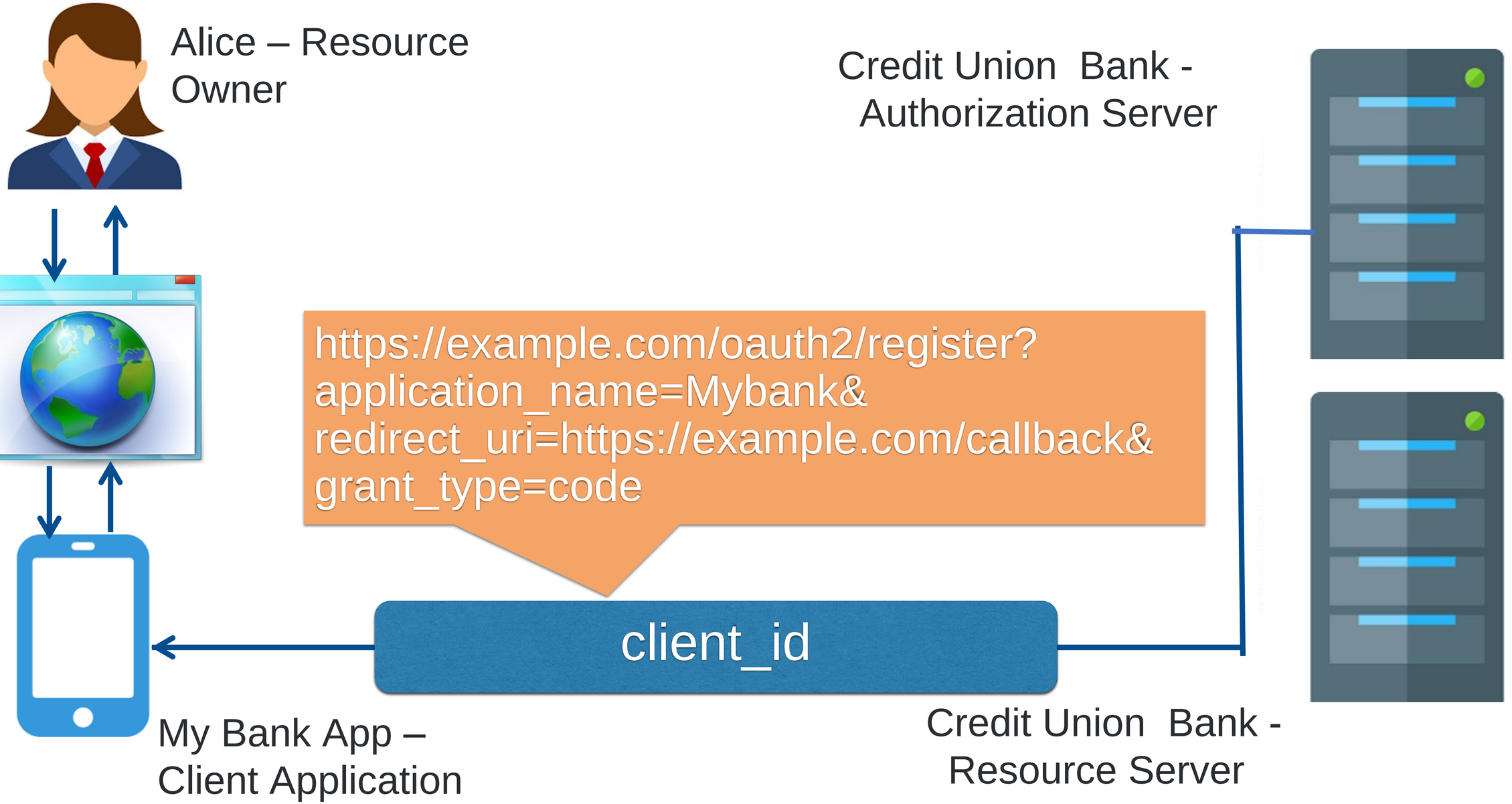
Credit Union Bank - Authorization Server





Credit Union Bank - Resource Server

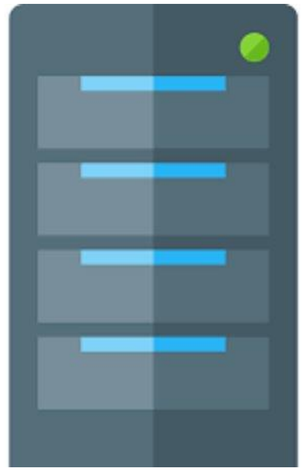
OAuth -The Authorization Code Flow with PCKE

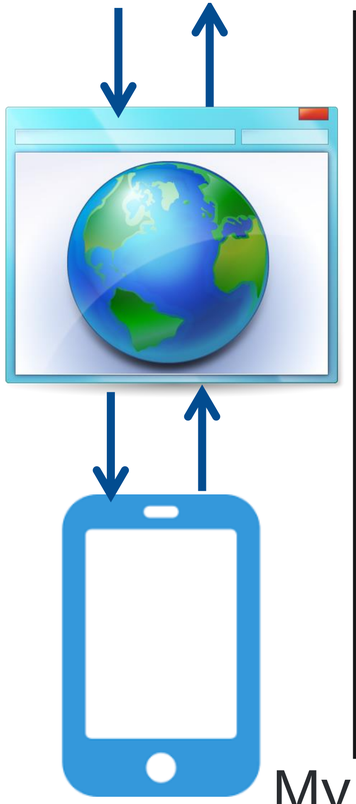


OAuth -The Authorization Code Flow with PCKE

Alice – Resource Owner

Credit Union Bank - Authorization Server



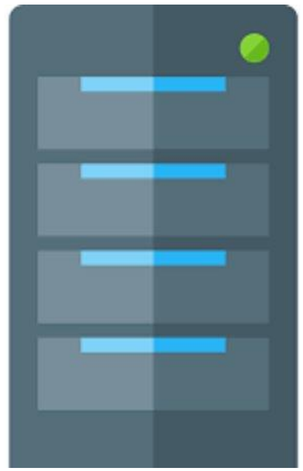


The Client generates a PCKE code verifier, a random string 43-128 characters

5d2309e5bb73b864f989753887fe52f79ce5270395e25862da6940d5

- The Client generates the PCKE Code Challenge

base64url(sha256(code verifier)



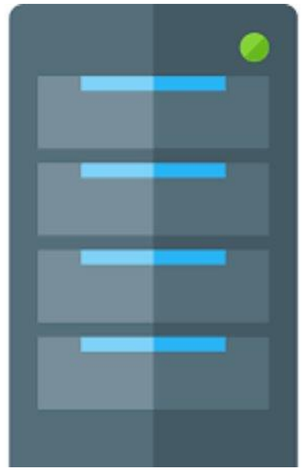
My Bank App – Client Application

Credit Union Bank - Resource Server

OAuth - The Authorization Code Flow

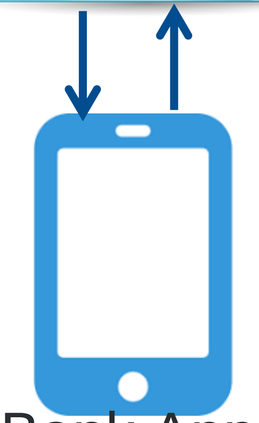
Alice – Resource Owner

Credit Union Bank - Authorization Server

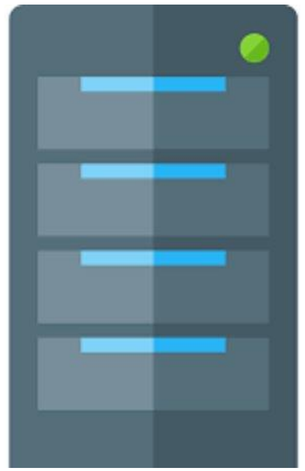


Alice Authenticates

https://example.com/oauth2/authorize?
response_type=code&
client_id=<client id>&
redirect_uri=https://example.com/callback&
scope=bankaccount&
state=123xyz
code_challenge=CODE_CHALLENGE
code_challenge_method=S256



My Bank App – Client Application

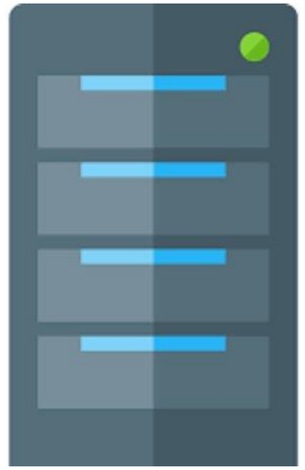


Credit Union Bank - Resource Server

OAuth -The Authorization Code Flow

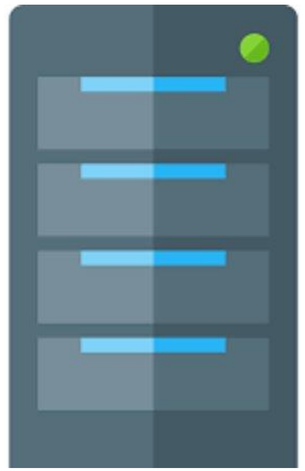
Alice – Resource Owner

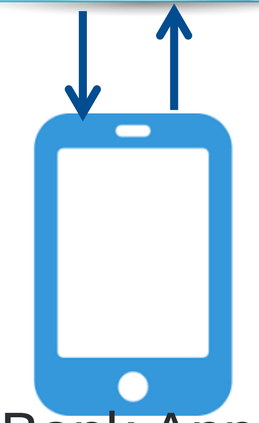
Credit Union Bank - Authorization Server



Authorization Code

https://example.com/oauth2/authorize?
code=8042d4cad58c66d0a5edfad8952192&
state=123xyz





My Bank App – Client Application

Credit Union Bank - Resource Server

OAuth - The Authorization Code Flow

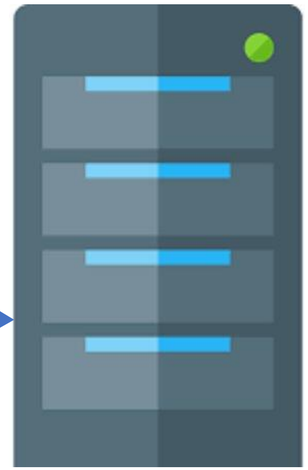
Alice – Resource Owner

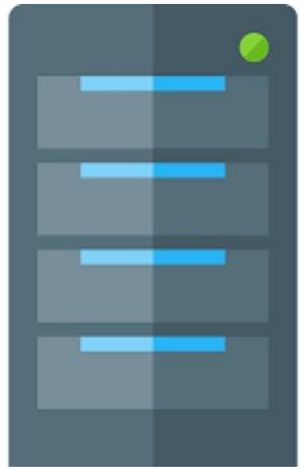
My Bank App – Client Application

POST https://example.com/oauth2/token
grant_type=authorization_code&
code=8042d4cad58c66d0a5edfad8952192&
redirect_uri=https://example.com/callback&
client_id=<client id>&
code_verifier=<PCKE code verifier>

Authorization Code & Code Verifier

Credit Union Bank - Authorization Server





Credit Union Bank - Resource Server

OAuth -The Authorization Code Grant Flow

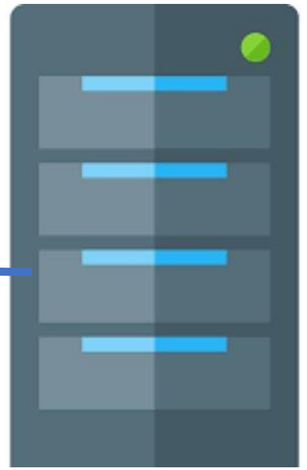
My Bank App –
Client Application

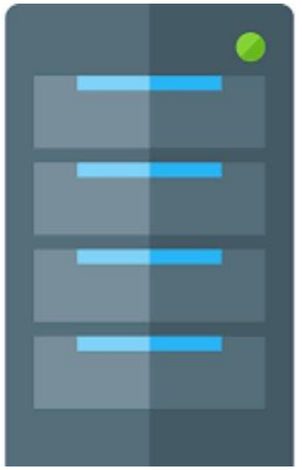
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8

```
{
  "access_token":"2YotnFZFEjr1zCsicMWpAA",
  "token_type":”bearer",
  "expires_in":3600,
  "refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA”
}
```

Access token & Refresh token

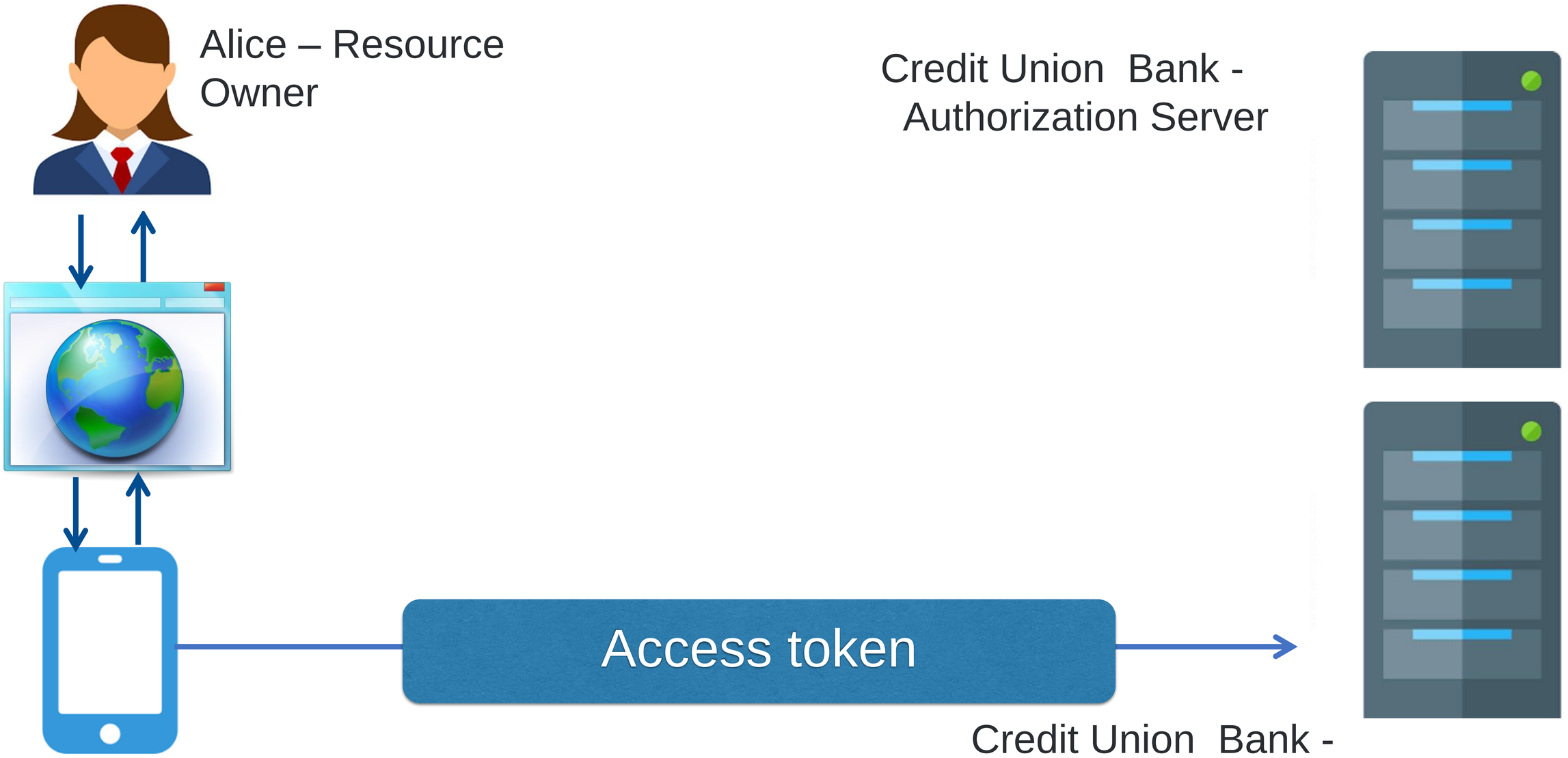
Credit Union Bank -
Authorization Server





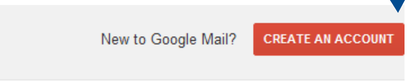
Credit Union Bank -
Rsource Server

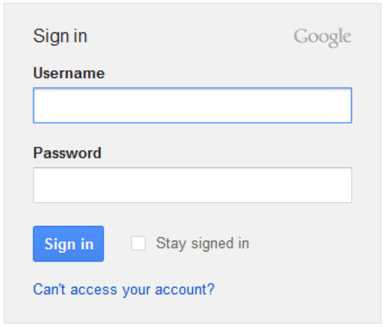
OAuth - The Authorization Code Flow



Resource Owner Password Grant Flow

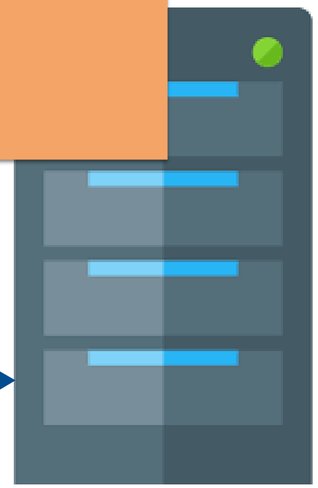
Google login a





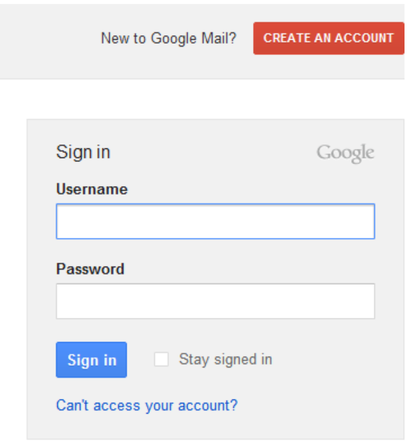
POST https://google.com/oauth2/token
grant_type=password&
client_id=<client id>&
username=johhdoe&
password=A4dm1kl
scope=email

Google login and password

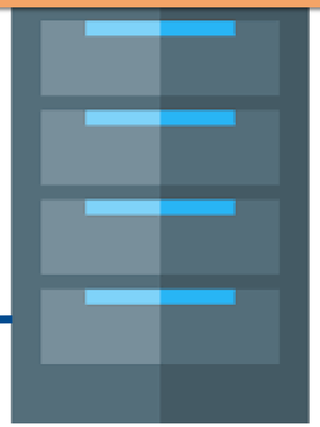


Resource Owner Password Grant Flow

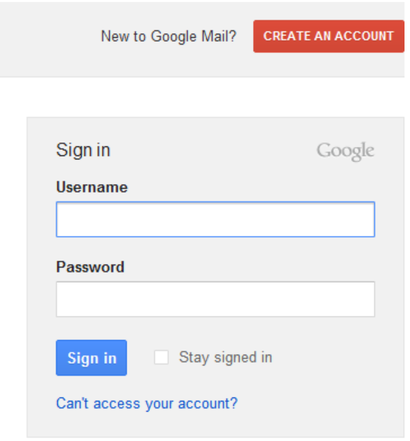
```
HTTP/1.1 200 OK
 Content-Type: application/json;charset=UTF-8
     {
       "access_token":"2YotnFZFEjr1zCsicMWpAA",
       "token_type":”bearer",
       "expires_in":3600,
       "refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA”
      }
```



Access Token

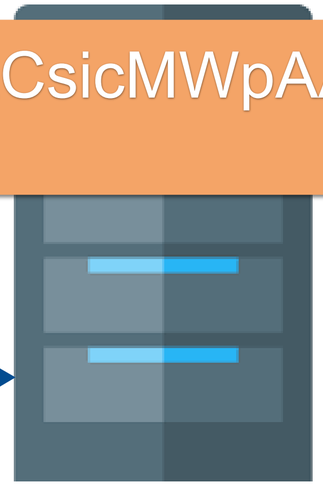


Resource Owner Password Grant Flow

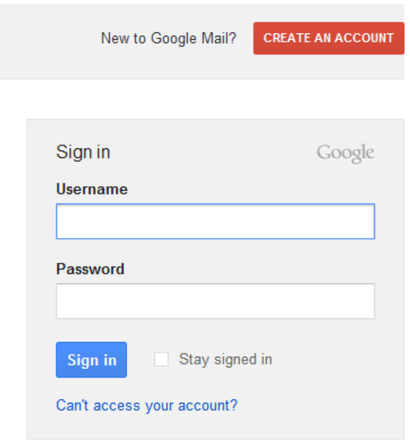


curl –H “Authorization: bearer 2YotnFZFEjr1zCsicMWpAA” \
https://google.com/gmail

Access Email Request &
Access Token

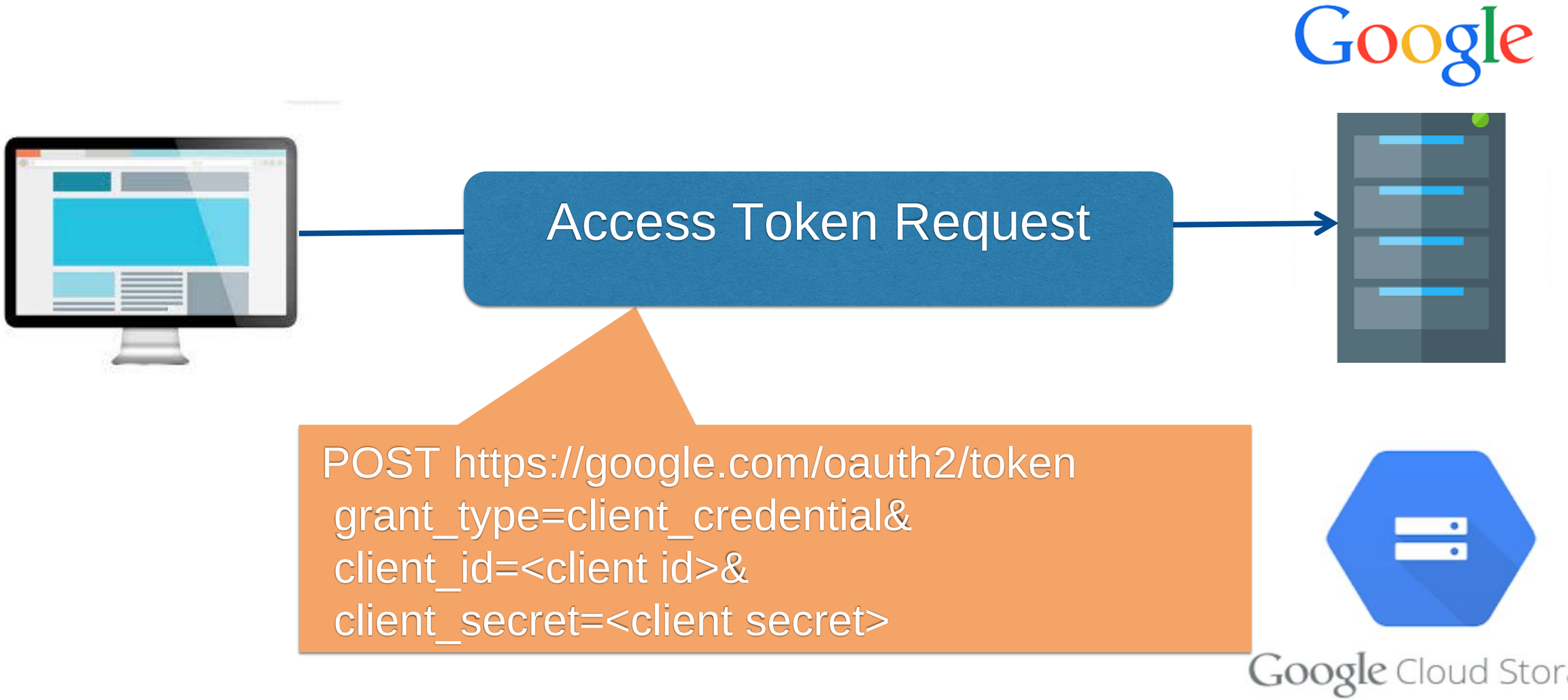


Resource Owner Password Grant Flow



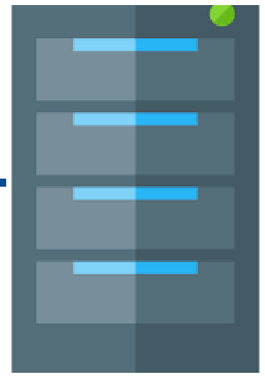
Google

Client Credential Grant Flow



Client Credential Grant Flow

Access Token

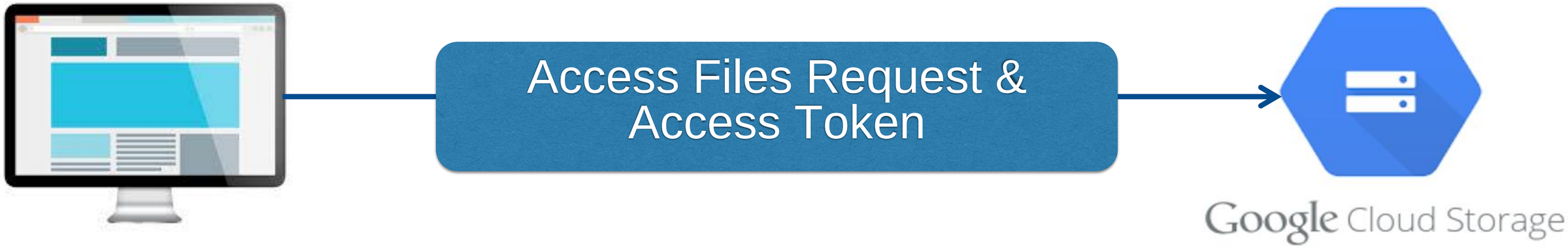


Google

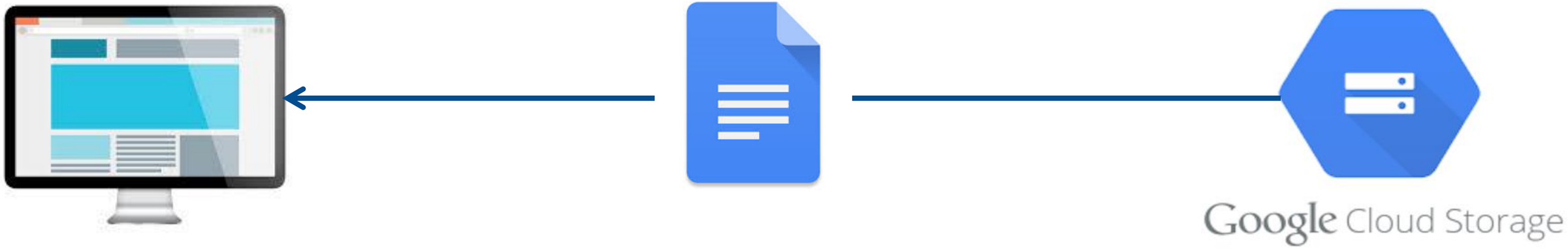
```
HTTP/1.1 200 OK
 Content-Type: application/json;charset=UTF-8
     {
       "access_token":"2YotnFZFEjr1zCsicMWpAA",
       "token_type":”bearer",
       "expires_in":3600,
     }
```

ogle Cloud Storage

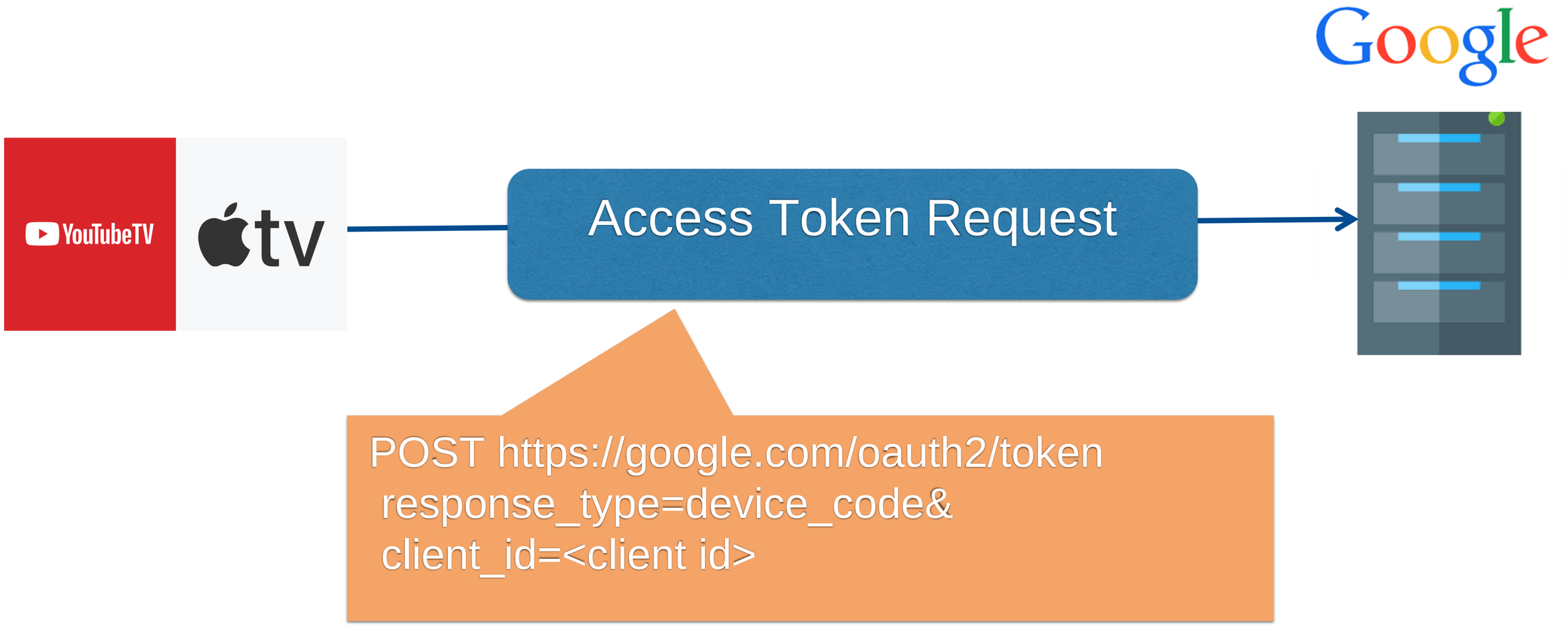
Client Credential Grant Flow



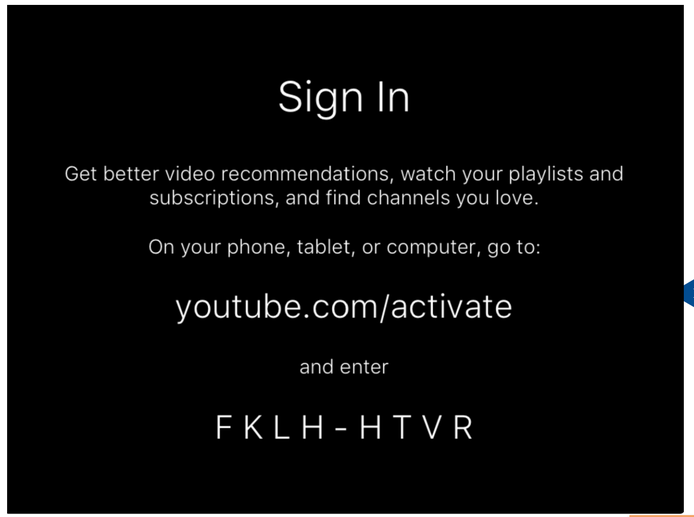
Client Credential Grant Flow



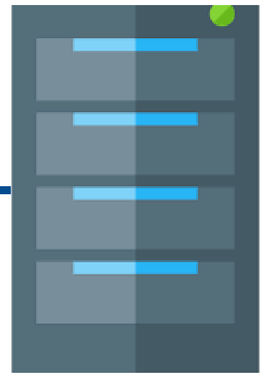
Device Flow


Google
YouTubeTV
tv
Access Token Request
POST https://google.com/oauth2/token
response_type=device_code&
client_id=<client id>

Device Flow



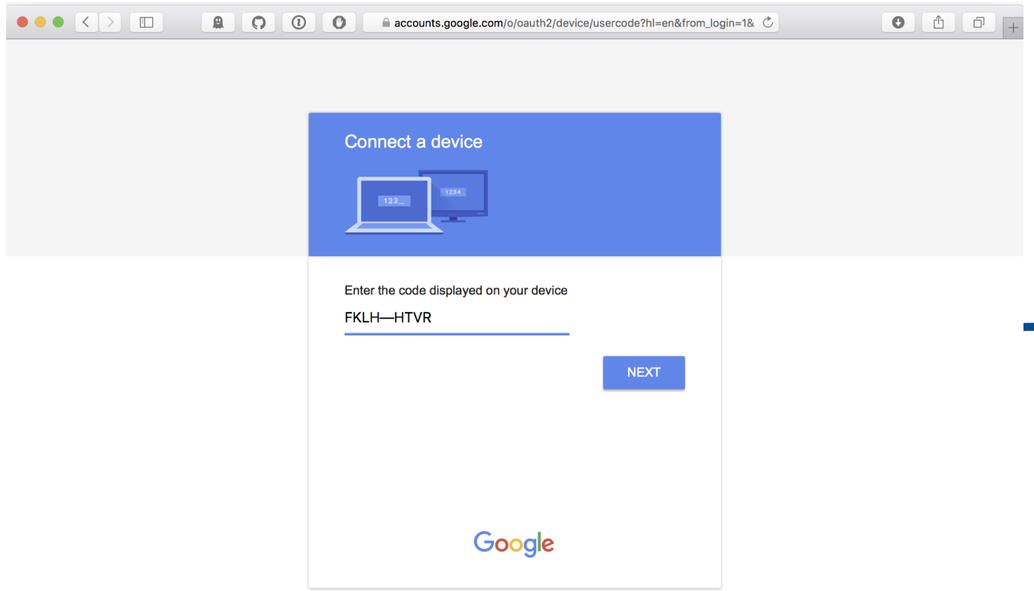
Google



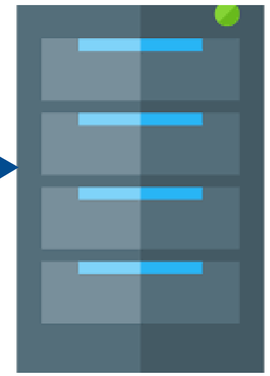
Login Info


```
{
"verification_uri": "youtube.com/activate",
    "user_code": "FKLH-HTVR",
    "device_code": "74tq5miHKB",
    "interval": 5
}
```

Device Flow

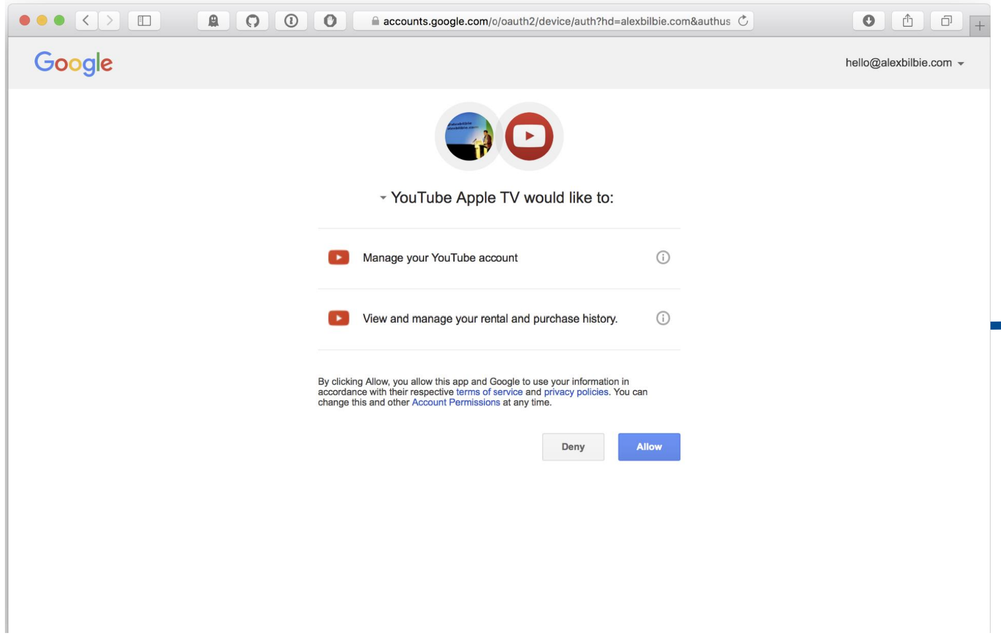


User Code



Google

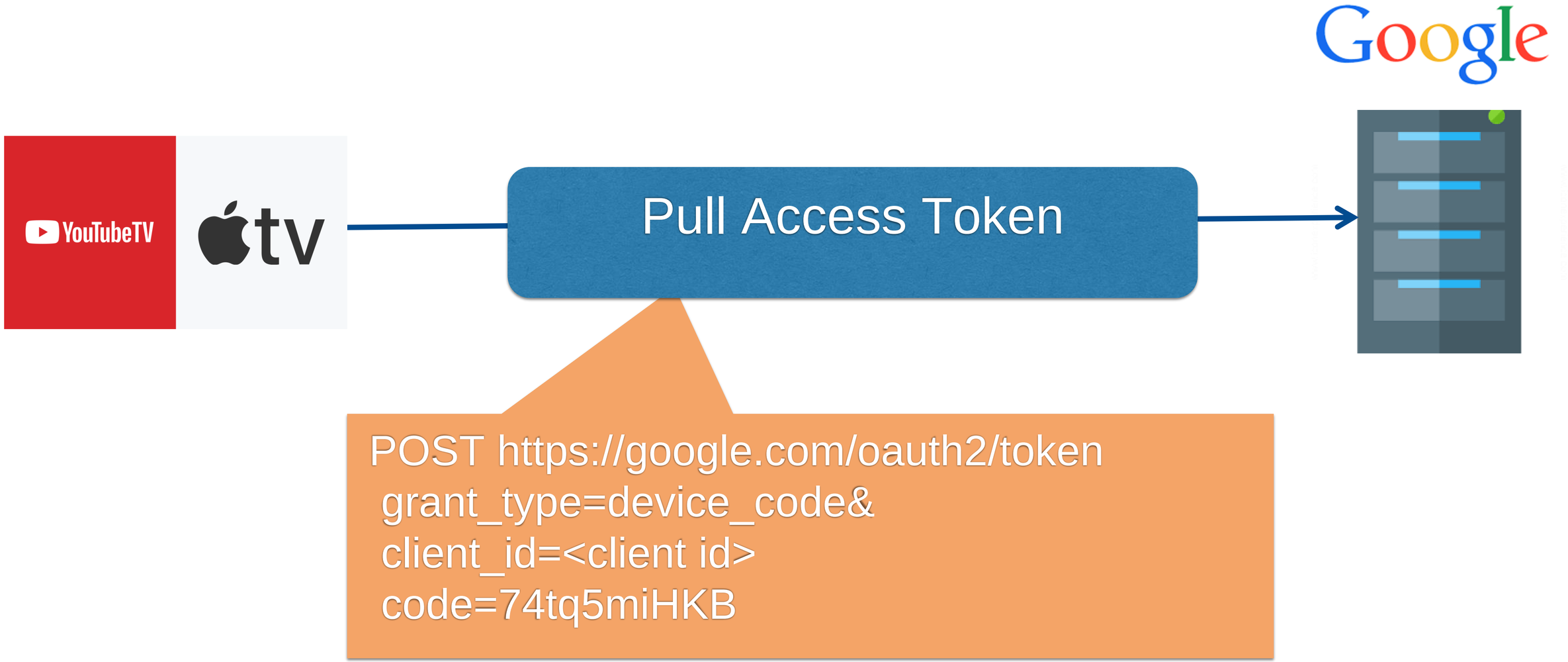
Device Flow



Google

Authorization

Device Flow


Google
YouTubeTV
tv
Pull Access Token
POST https://google.com/oauth2/token
grant_type=device_code&
client_id=<client id>
code=74tq5miHKB

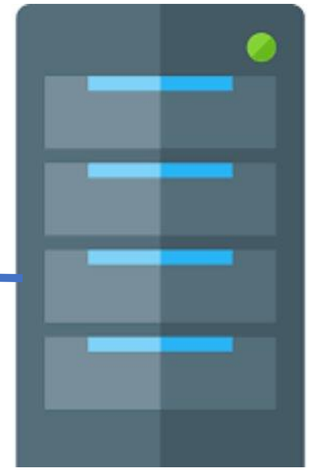
Device Flow

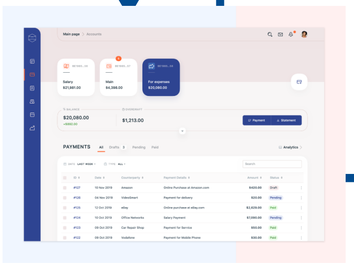

Google
YouTubeTV
tv
Access Token
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
{
"access_token":"2YotnFZFEjr1zCsicMWpAA",
"token_type":”bearer",
"expires_in":3600,
"refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA”
}

OpenID Connect-The Authorization Code Flow

Alice – Resource Owner

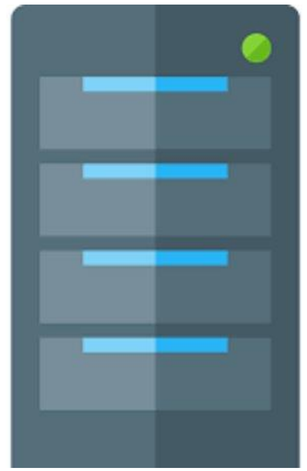
Credit Union Bank - Authorization Server





https://example.com/oauth2/register?
application_name=Mybank&
redirect_uri=https://example.com/callback&
grant_type=code

client_id & client_secret



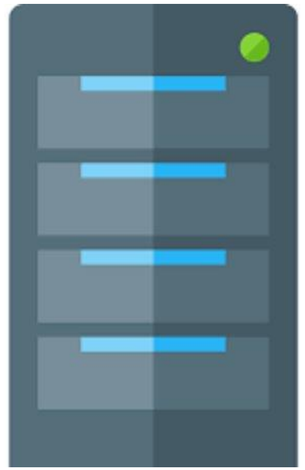
My Bank App – Client Application

Credit Union Bank - Resource Server

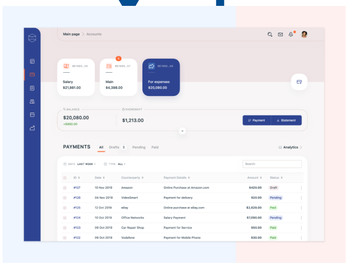
OpenID Connect - The Authorization Code Flow

Alice – Resource Owner

Credit Union Bank - Authorization Server

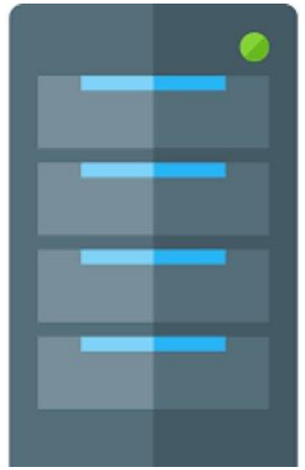


Alice Authenticates



https://example.com/oauth2/authorize?
response_type=code&
client_id=<client id>&
redirect_uri=https://example.com/callback&
scope=open-id&profile&email
state=123xyz

My Bank App – Client Application

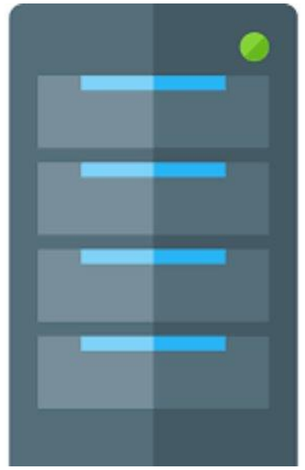


Credit Union Bank - Resource Server

OpenID Connect -The Authorization Code Flow

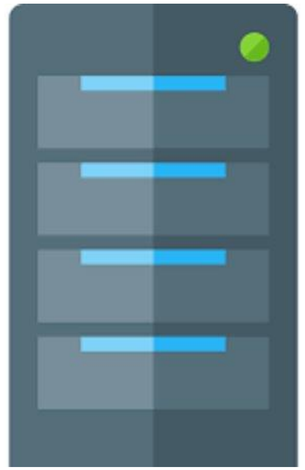
Alice – Resource Owner

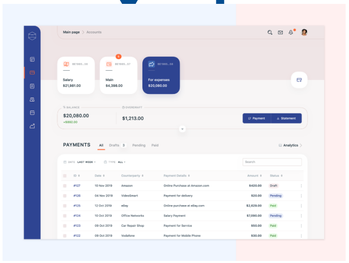
Credit Union Bank - Authorization Server



Authorization Code

https://example.com/oauth2/authorize?code=8042d4cad58c66d0a5edfad8952192&state=123xyz



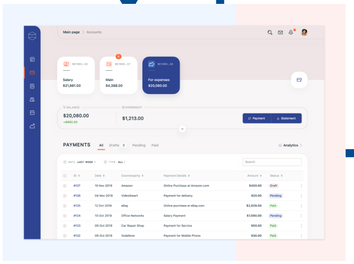


My Bank App – Client Application

Credit Union Bank - Resource Server

OpenID Connect - The Authorization Code Flow

Alice – Resource Owner

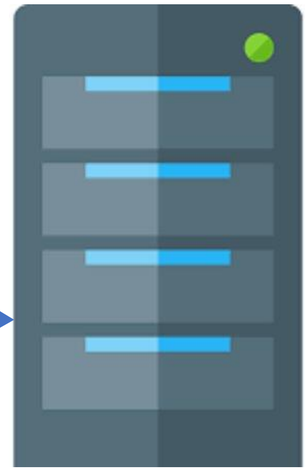


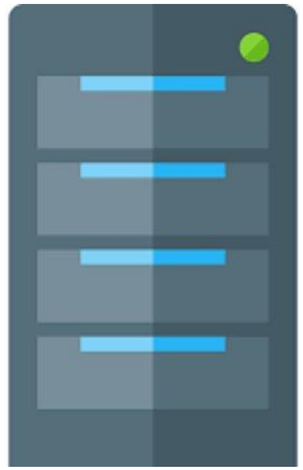
My Bank App – Client Application

POST https://example.com/oauth2/token
grant_type=authorization_code&
code=8042d4cad58c66d0a5edfad8952192&
redirect_uri=https://example.com/callback&
client_id=<client id>&
client_secret=<client secret>

Authorization Code & Redirection URI

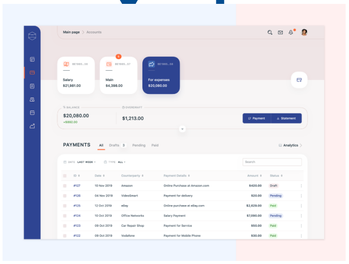
Credit Union Bank - Authorization Server





Credit Union Bank - Resource Server

OpenID Connect -The Authorization Code Grant Flow

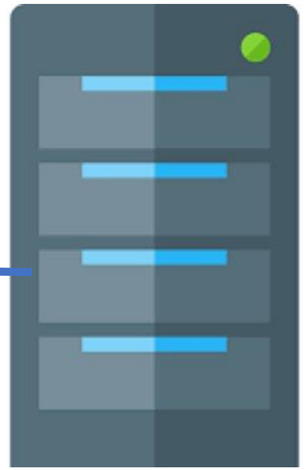


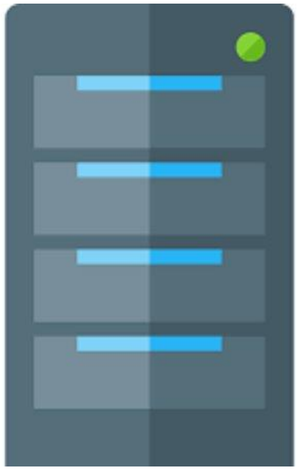
My Bank App –
Client Application

```
HTTP/1.1 200 OK
 Content-Type: application/json;charset=UTF-8
     {
       "access_token":"2YotnFZFEjr1zCsicMWpAA",
        id_token: ":"KLotnFZFEjr1zCsicMWpAA"
      “token_type":”bearer",
      "expires_in":3600,
           }
```

Access token & IDToken

Credit Union Bank -
Authorization Server

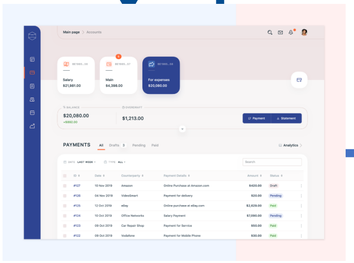




Credit Union Bank -
Rsource Server

OpenID Connect - The Authorization Code Flow

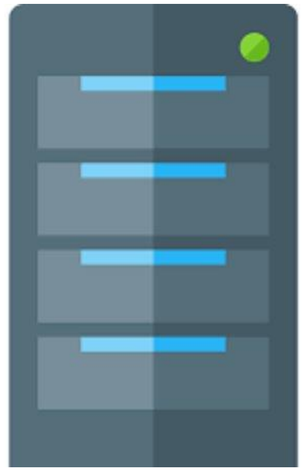
Alice – Resource Owner

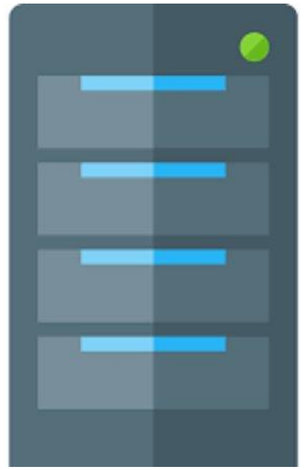


My Bank App – Client Application

Access token

Credit Union Bank - Authorization Server





Credit Union Bank - Resource Server

Id Token


```
{
  "sub"       : "alice",
  "iss"       : "https://openid.c2id.com",
  "aud"       : "client-id",
  "nonce"     : "n-0S6_WzA2Mj",
  "auth_time" : 1311280969,
  "acr"       : "c2id.loa.hisec",
  "iat"       : 1311280970,
  "exp"       : 1311281970
}
```

Exercise 2 – Smart Home

- Alice lives in a smart home with a smart lighting system
- Alice can control her smart light system with MyBulb mobile app
- The smart light systems exposes APIs to switch/on and off the smart bulbs that are part of the system
- Think about:
  - Who are the actors?
  - What are the protected resources?
  - What grant flow would you use?
- Time: 5 min

Summary

- OAuth is an authorization protocol that allows an end-user (the resource owner) to grant access to a client application to protected resources with the end-user consent
- OAuth supports three main grant flows
  - Authorization code grant flow
    a. the client application is a third party application
    b. access is requested on behalf of the end user
  - Resource owner password grant flow
    a. the client application is a a first party application
    b. access is requested on behalf of the end user
  - Client credential grant flow
    a. access is requested on behalf of the client application itself

Resources

- **OAuth Web site**
  - https://oauth.net/2/
- **OAuth specification**
  - https://tools.ietf.org/html/rfc6749
- **OAuth simplified explanation**
  - https://alexbilbie.com/guide-to-oauth-2-grants/
  - http://www.bubblecode.net/en/2016/01/22/understanding-oauth2/
  - https://aaronparecki.com/oauth-2-simplified/
- **OAuth security**
  - https://tools.ietf.org/html/rfc6819