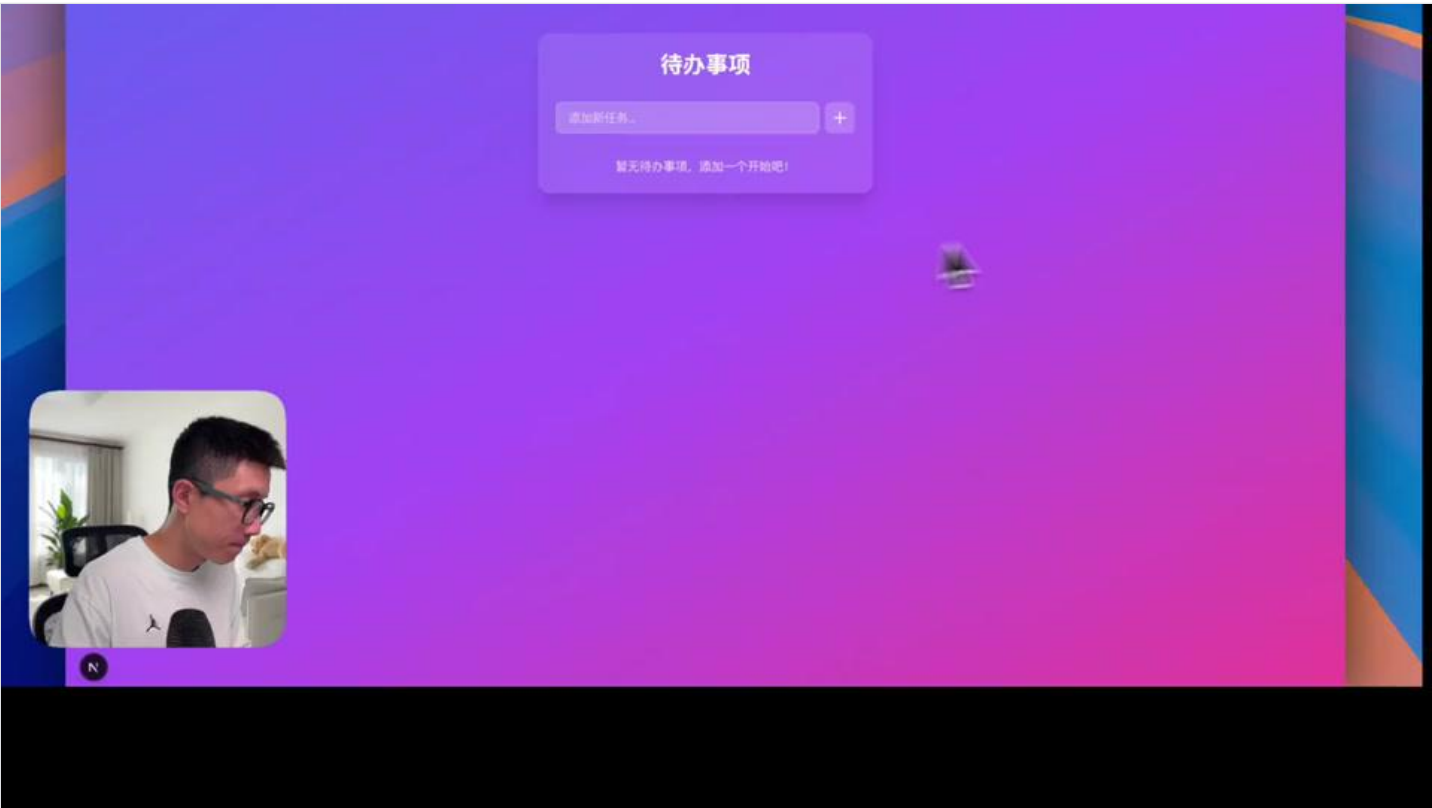
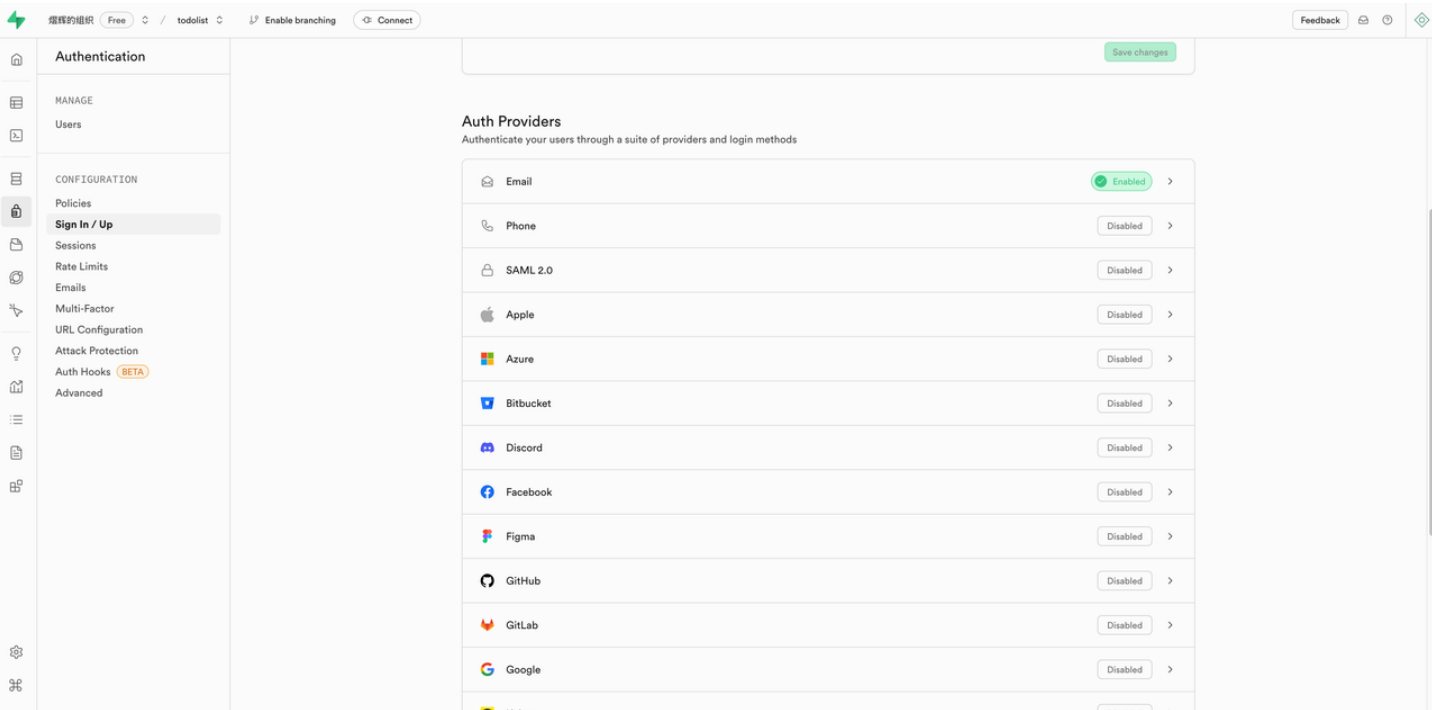


# 轻松接入邮件登陆注册功能（Auth）



## 1. Supabase的登陆注册方案

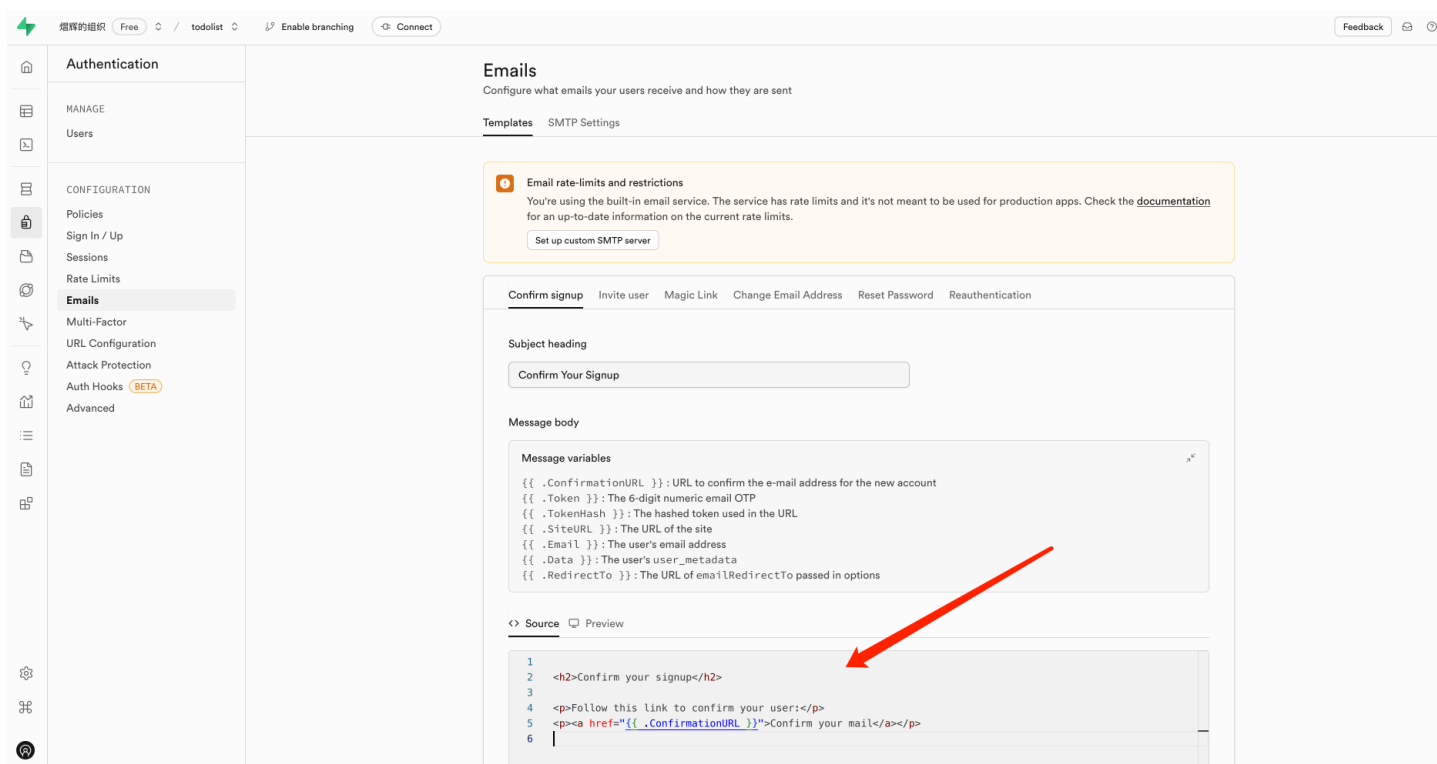
### 1.1 支持的登陆方式



进入到「Authentication」 - 「Sign In/Up」页面，你会看到很多Auth Provider。supabase不仅支持邮件登录，还支持基本市面上所有常见的第三方登陆，比如说Github、Google、Discord Apple。你只需要在supabase的后台启用，并且填入相应平台的key等信息，就能启动第三方平台的登陆注册功能。

## 1.2 邮件登陆

目前邮件注册登录的方式是默认支持的。我们可以点击Emails，去调整邮件的相应模板（Templates）。

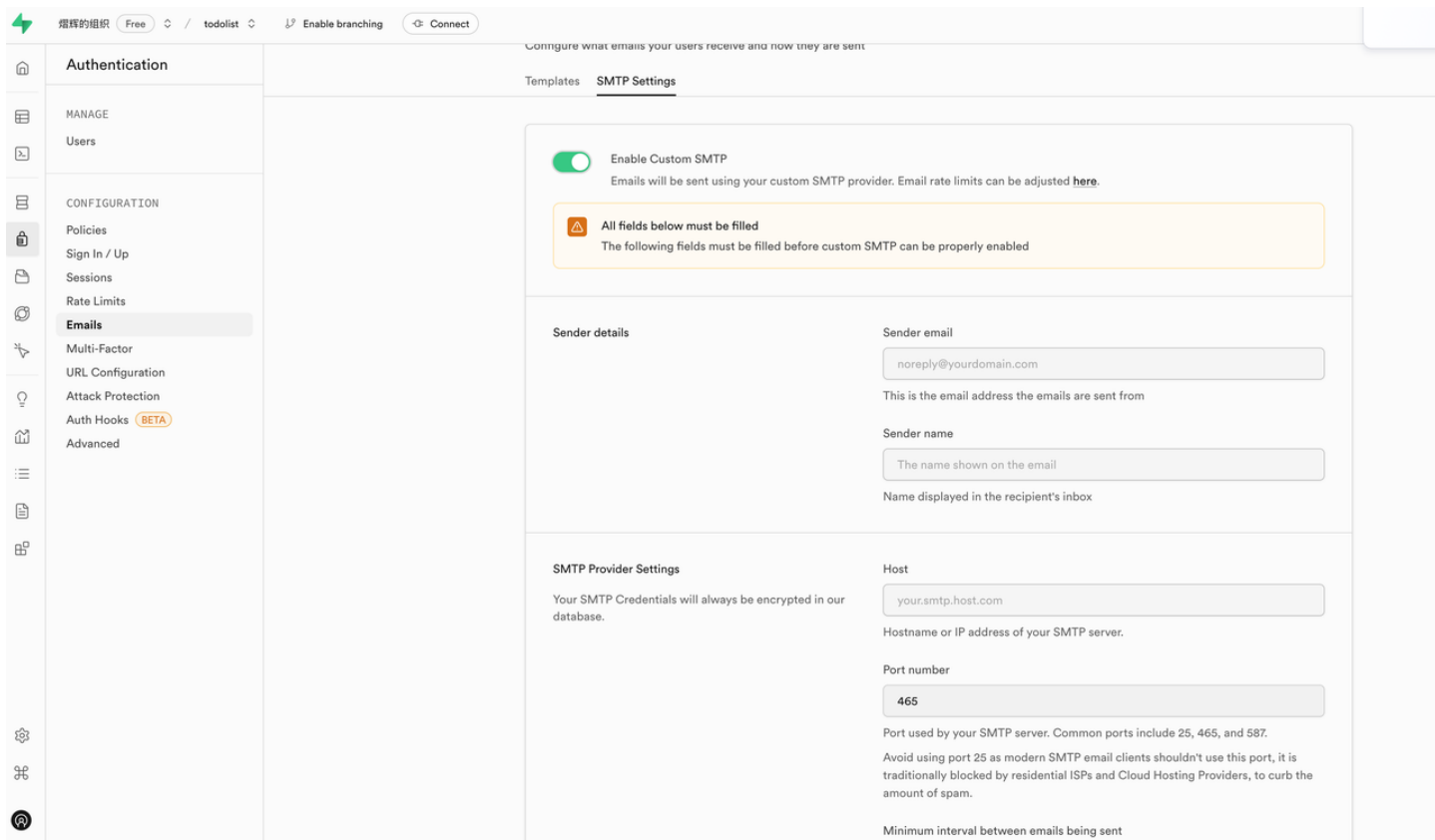


在上线前，需要大家SMTP Settings。调整成自己邮件服务商的地址。



### 提示

目前所使用的邮箱服务是supabase提供的，会有速率限制。上线前请务必调整成自己的邮箱服务，填入相关信息就好。例如[Resend](#)、[163邮箱](#)。



这边我拿163举例，点击「设置」 - 「POP3/SMTP/IMAP」



然后开启服务，并新增授权密码，保存这个授权密码。

常规设置

邮箱密码修改

签名

来信分类

账号与邮箱中心

邮箱安全设置

反垃圾/黑白名单

POP3/SMTP/IMAP

文件夹和标签

多标签窗口

换肤



建议您使用网易邮箱官方客户端，确保邮箱安全  
官方客户端具有邮件加密传输、邮箱登录二次验证安全保障功能，有效避免信息泄露、账号被盗等潜在安全风险。



扫码下载App

POP3/SMTP/IMAP

开启服务：

IMAP/SMTP服务 已开启 | 关闭

POP3/SMTP服务 已开启 | 关闭

POP3/SMTP/IMAP服务能让你在本地客户端上收发邮件，[了解更多 >](#)

温馨提示：在第三方登录网易邮箱，可能存在邮件泄露风险，甚至危害Apple或其他平台账户安全，[立即下载官方客户端 >](#)

收取选项：

☒ 收取最近30天邮件

☐ 收取全部邮件

温馨提示：收取大量邮件，会耗费您更多的流量，建议您选择“收取最近30天邮件”

通知提醒：

☐ 开启客户端删除邮件提醒

当邮件客户端大量删除邮件时，系统会发送提醒信息

授权密码管理：

授权码是用于登录第三方邮件客户端的专用密码。  
适用于登录以下服务：您开启的服务（例如POP3/IMAP/SMTP）、Exchange/CardDAV/CalDAV服务。

使用设备	启用时间	操作
设备1	2024.6.1	删除

新增授权密码 每个账号最多设置2个授权密码，[开通邮箱会员](#)后可设置10个

然后填入你的授权密码和邮箱，就能够支持自己的SMTP邮件服务商了。其他邮件服务商的方式也都类似。

耀辉的组织 Free / todolist Enable branching Connect Feedback

Authentication

MANAGE Users

CONFIGURATION

Polices

Sign In / Up

Sessions

Rate Limits

Emails

Multi-Factor

URL Configuration

Attack Protection

Auth Hooks (BETA)

Advanced

Sender details

Sender email  
kapiai@163.com  
This is the email address the emails are sent from

Sender name  
yihui  
Name displayed in the recipient's inbox

SMTP Provider Settings

Your SMTP Credentials will always be encrypted in our database.

Host  
smtp.163.com  
Hostname or IP address of your SMTP server.

Port number  
465  
Port used by your SMTP server. Common ports include 25, 465, and 587.  
Avoid using port 25 as modern SMTP email clients shouldn't use this port, it is traditionally blocked by residential ISPs and Cloud Hosting Providers, to curb the amount of spam.

Minimum interval between emails being sent  
60 seconds  
How long between each email can a new email be sent via your SMTP server.

Username  
kapiai@163.com

Password  
.....  
For security reasons, the password is write-only. Once saved, it cannot be retrieved or displayed.



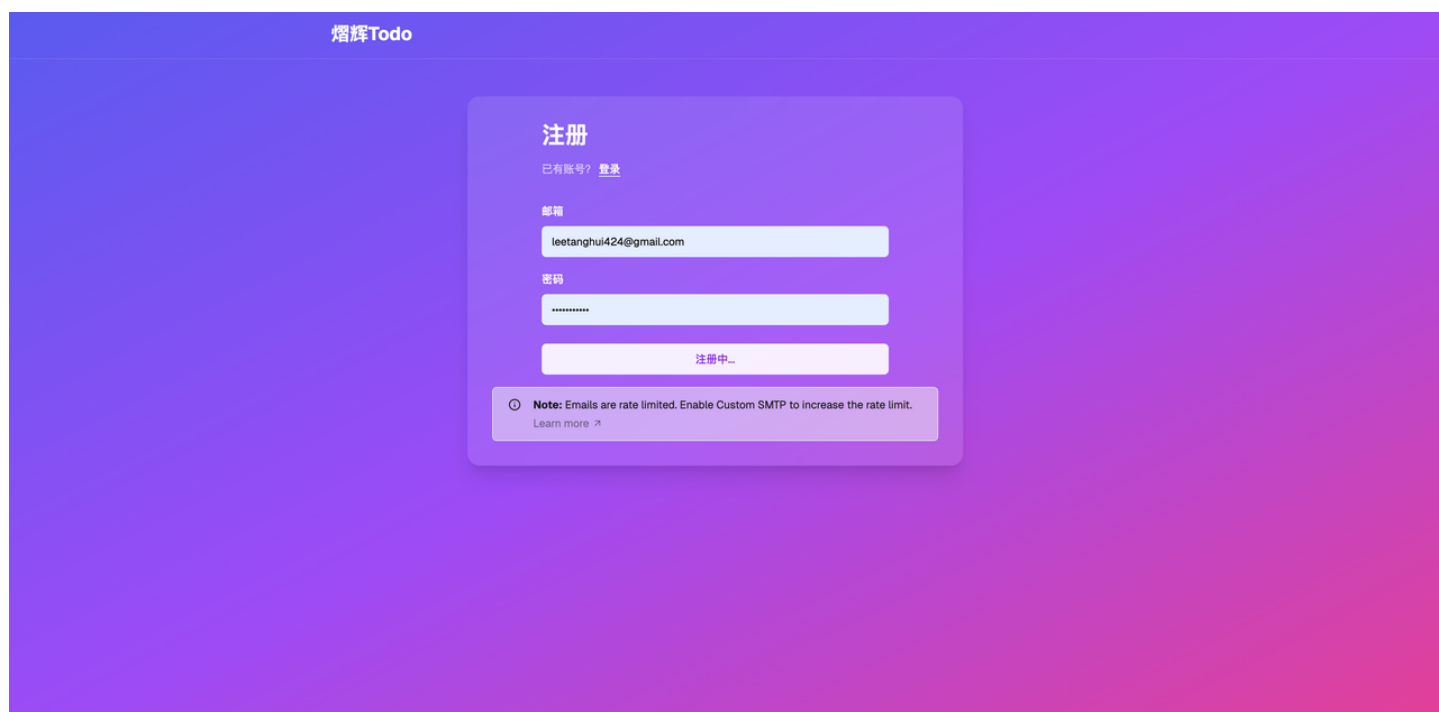
提示：

163等免费SMTP服务会有每日邮件限制（官方并没有明确公开具体的数字）。但是一般非商业化项目注册使用完全够了，如果你有更多需求。建议使用[Resend](#)等专业的邮件服务商。

## 2. 给项目添加登陆/注册/登出

### 1. 注册

这边我们的模板当中是自动帮你集成了邮件登录注册的，所以你如果在上一步的环境变量只要填写正确，其实你无需任何的配置。我们可以直接输入自己的邮箱和密码进行注册。

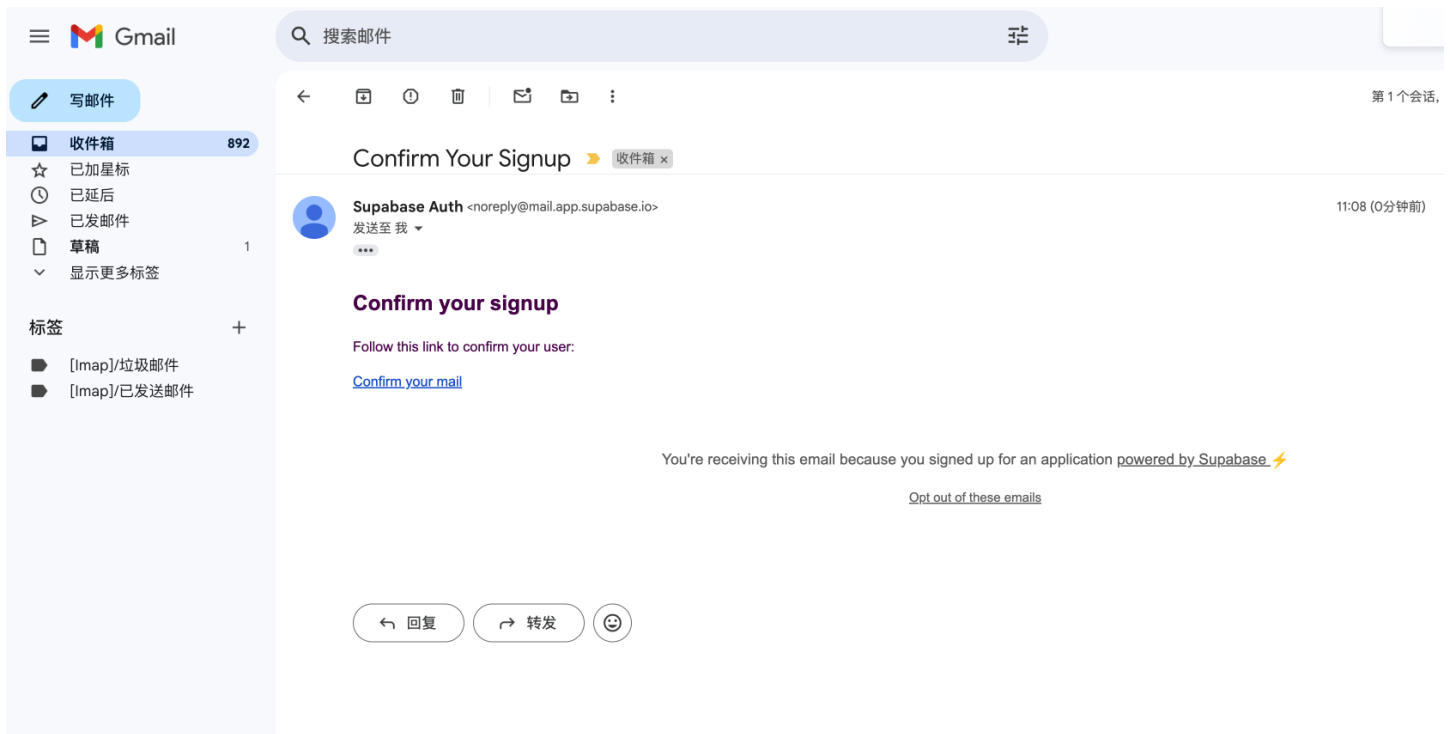
A screenshot of a web application's registration page. The page has a purple gradient background. At the top left, it says "熠辉Todo". In the center, there's a white registration form with the title "注册". Below the title, there's a link "已有账号? 登录". The form has two input fields: "邮箱" (Email) with the value "leetanghui424@gmail.com" and "密码" (Password) with masked characters. Below these is a "注册中..." button. At the bottom of the form, there's a note: "Note: Emails are rate limited. Enable Custom SMTP to increase the rate limit." with a "Learn more" link.

虽然我们的模板自动集成了登录注册功能，但是也不妨我们来看一下sup实现注册有多简单，其实也就是下面这一个方法的事情：

app/actions.ts

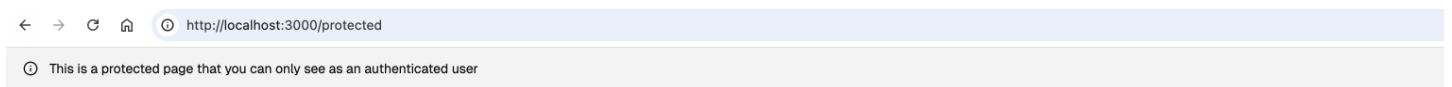
```
1  const { error } = await supabase.auth.signUp({
2    email,
3    password,
4    options: {
5      emailRedirectTo: `${origin}/auth/callback`,
6    },
7  });
```

然后进入到我们的邮箱内，就看到我们邀请注册了。



## 2. 登录

## 3. 登陆后的页面



### Your user details

```
{
  "id": "335bcf54-942b-4081-a103-e75b783fbec8",
  "aud": "authenticated",
  "role": "authenticated",
  "email": "leetanghui424@gmail.com",
  "email_confirmed_at": "2025-02-25T03:10:38.159307Z",
  "phone": ""
}
```

### Next steps

#### ☐ Create some tables and insert some data

Head over to the **Table Editor** for your Supabase project to create a table and insert some example data. If you're stuck for creativity, you can copy and paste the following into the **SQL Editor** and click RUN!

```
create table notes (
  id bigserial primary key,
  title text
);

insert into notes(title)
values
  ('Today I created a Supabase project.'),
  ('I added some data and queried it from Next.js.'),
  ('It was awesome!');
```

#### ☐ Query Supabase data from Next.js

To create a Supabase client and query data from an Async Server Component, create a new page.tsx file at `/app/notes/page.tsx` and add the following.

```
import { createClient } from '@utils/supabase/server'

export default async function Page() {
  const supabase = createClient()
  const { data: notes } = await supabase.from('notes').select()

  return <pre>{JSON.stringify(notes, null, 2)}</pre>
```

登录成功之后呢，你会被重定向到这个 `/protected` 页面。看到他是很多代码别慌！这个页面其实是一个用户后台页面，你可以自行修改这个页面。

很多的网站都是这样的操作，首页其实是落地页，用户没有登录的时候也可以查看，但是一旦用户登录了，就会被重定向到一个后台页面，或者叫做用户仪表盘页面。

## 4. 移除掉重定向

但是我们的落地页不需要这个页面，因为我们的首页就是我们的Todo的产品功能，所以说我们关闭这个重定向功能。

1.找到`utils/supabase/middleware.ts`这个文件，注释掉这几行代码。

utils/supabase/middleware.ts

```
1  //if (request.nextUrl.pathname === "/" && !user.error) {
2  //  return NextResponse.redirect(new URL("/protected", request.url));
3  // }
```

2.在`app/actions.ts`中，将登陆后的重定向路径修改成 `/`

代码块

```
1  export const signInAction = async (formData: FormData) => {
2    const email = formData.get("email") as string;
3    const password = formData.get("password") as string;
4    const supabase = await createClient();
5
6    const { error } = await supabase.auth.signInWithPassword({
7      email,
8      password,
9    });
10
11    if (error) {
12      return encodedRedirect("error", "/sign-in", error.message);
13    }
14
15    //return redirect("/protected");
16    return redirect("/"); // 修改这行成/
17  };
```

3.将`app/auth/callback/route.ts`中的重定向功能修改

代码块

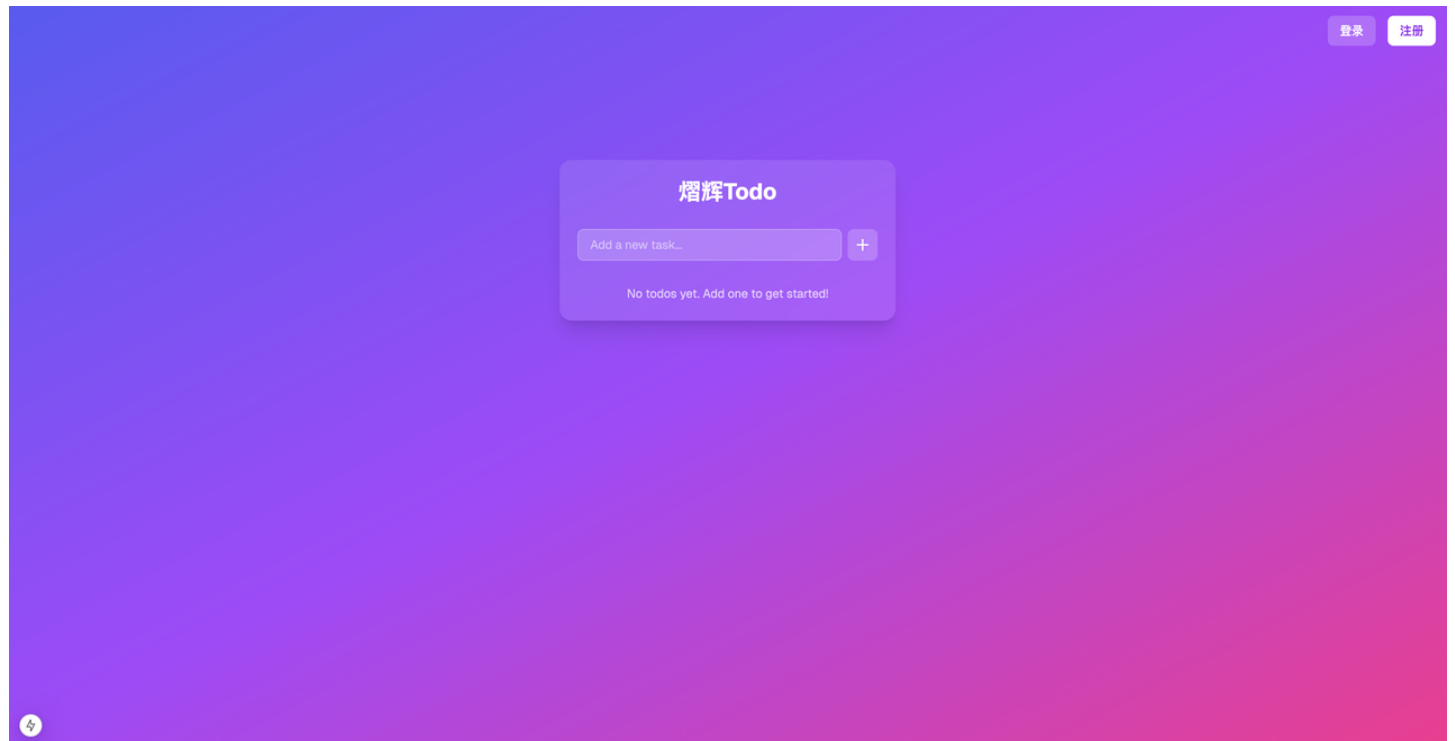
```
1  export async function GET(request: Request) {
2    // The `/auth/callback` route is required for the server-side auth flow
    // implemented
3    // by the SSR package. It exchanges an auth code for the user's session.
```

```

4 // https://supabase.com/docs/guides/auth/server-side/nextjs
5 const requestUrl = new URL(request.url);
6 const code = requestUrl.searchParams.get("code");
7 const origin = requestUrl.origin;
8 const redirectTo = requestUrl.searchParams.get("redirect_to")?.toString();
9
10 if (code) {
11   const supabase = await createClient();
12   await supabase.auth.exchangeCodeForSession(code);
13 }
14
15 if (redirectTo) {
16   return NextResponse.redirect(`${origin}${redirectTo}`);
17 }
18
19 // URL to redirect to after sign up process completes
20 //return NextResponse.redirect(`${origin}/protected`);
21 return NextResponse.redirect(`${origin}/`);
22 }
23

```

这样我们登录之后就回到了首页



## 5. 退出登陆

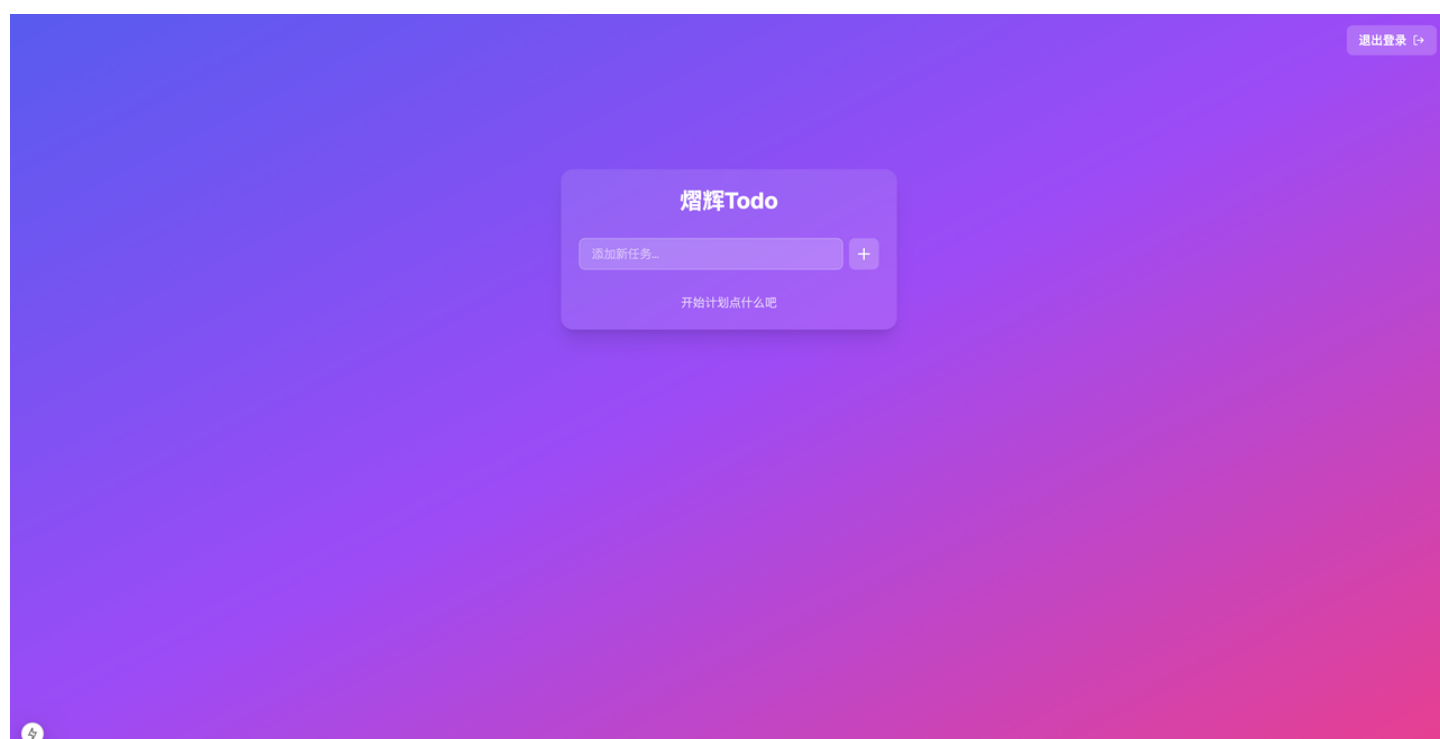


现在你可能会发现在登录态的时候，右上角不应该显示登录和注册按钮。所以我们来修改这个逻辑。在Cursor的Composer中输入以下提示词：

提示词

- 1 修改右上角按钮的显示逻辑，如果用户是登录的状态，那么应该显示退出登录用户点击退出按钮后。调用 `@actions.ts` 中的退出登陆方法。

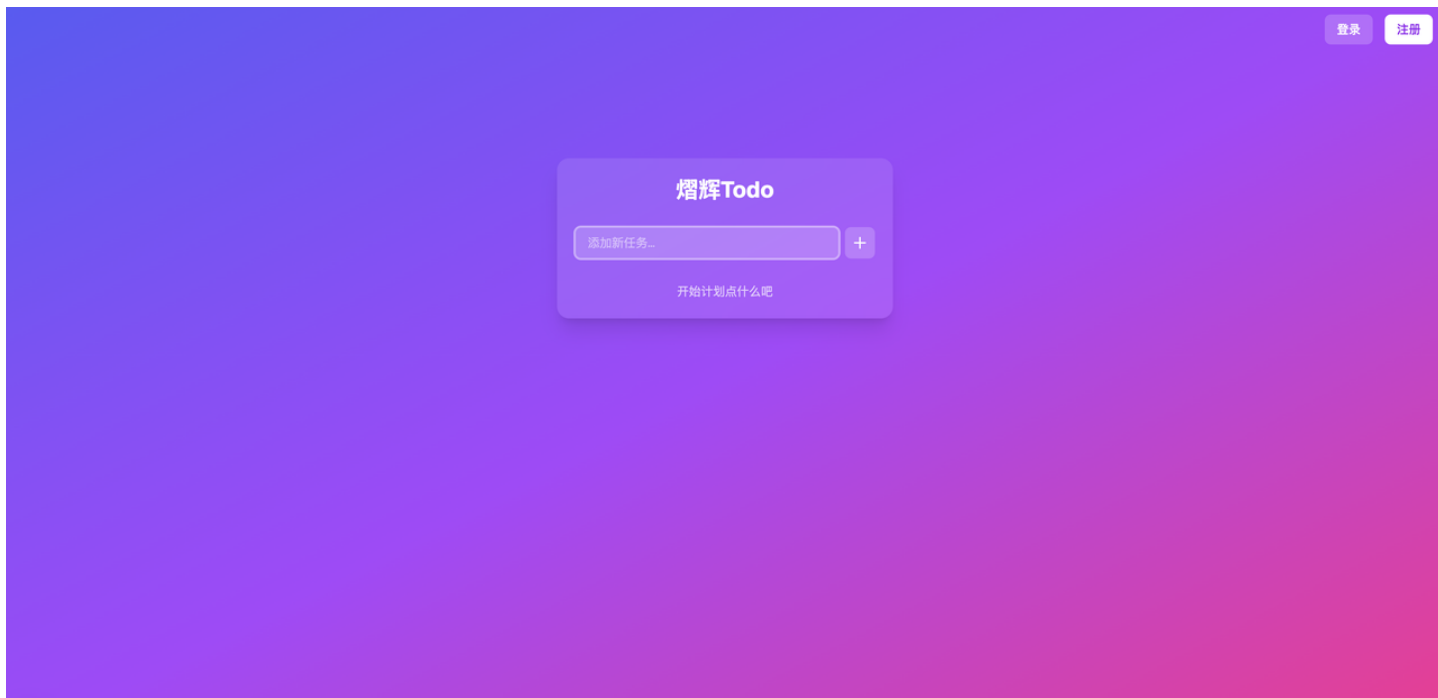
再回到页面检查一下效果，这就完成退出登陆的功能了！



完成项目的登陆注册功能

可以看到基于模板和supabase，完成一个登陆注册功能真的很简单。

### 3. 未登录重定向



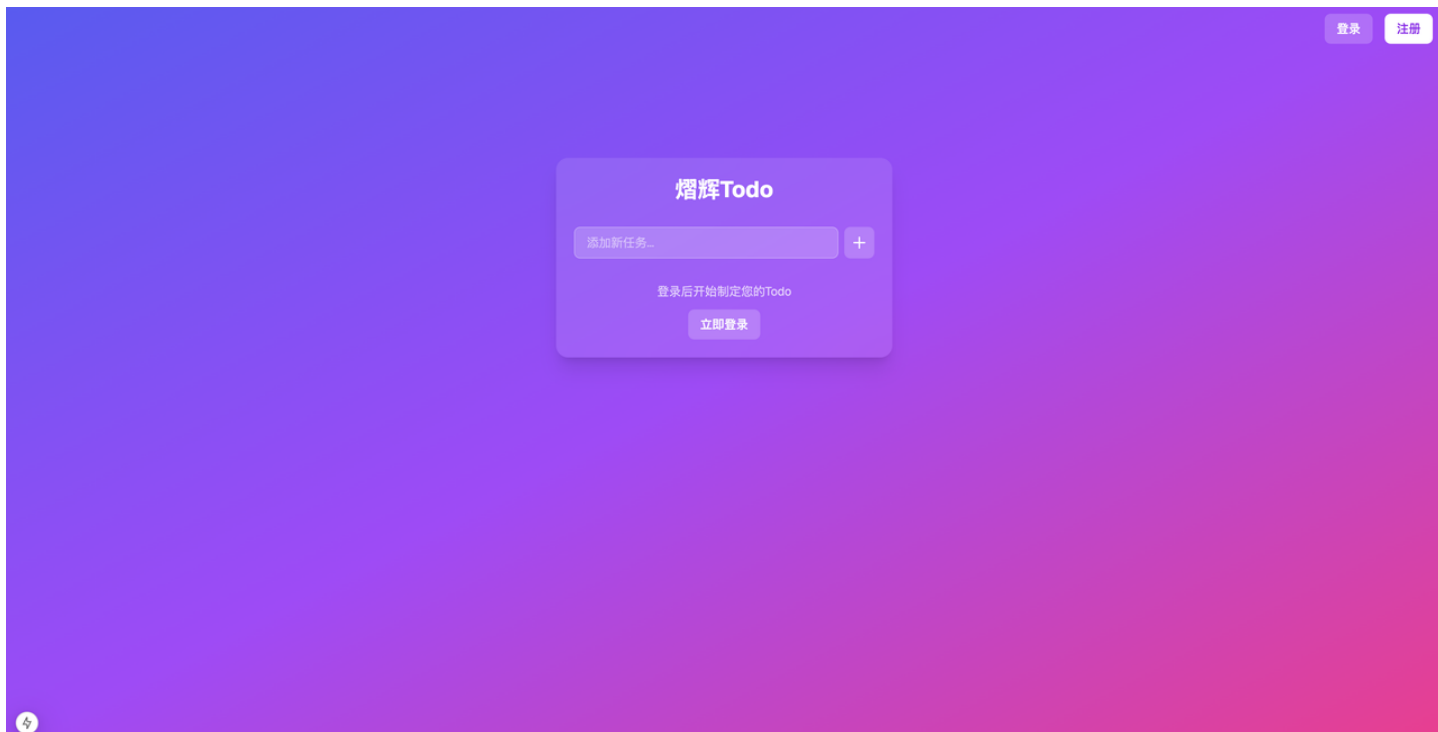
接下来我们给页面新增加一个逻辑，就是如果用户没有登录的话，那么此时用户去新增Todo的时候，应该让他重定向到登录页面，并且修改下方的文案。

在Cursor的Composer中输入以下提示词：

代码块

1. `@page.tsx` 给页面增加一个逻辑，如果用户在添加todo的时候，没有登陆，那么应该重定向到登陆页面
2. 下方文案「开始计划点什么吧」，如果在没有登录的时候，显示登录后制定Todo

之后检查一下，这就达成了我们想要的需求：



## 4. supabase方法总结

虽然这个模板帮我们在登录注册这儿做了非常多的事情，但是我们也可以来学习一下supabase是如何简化登录注册方案的。

模板中封装登录注册的主要逻辑是在 `app/actions.ts` 这个文件里面：

### 4.1 注册

`supabase.auth.signUp()`：该方法是supabase邮件密码注册的方法，只需要传入邮件和密码，并传入注册后的重定向地址就能够实现注册。

app/actions.ts

```
1  const { error } = await supabase.auth.signUp({
2    email,
3    password,
4    options: {
5      emailRedirectTo: `${origin}/auth/callback`,
6    },
7  });
```

### 4.2 登陆

`supabase.auth.signInWithPassword()`：该方法是实现邮件密码登录的方法，参数是 `email` 和 `password`。

app/actions.ts

```
1  const { error } = await supabase.auth.signInWithPassword({
2    email,
3    password,
4  });
```

## 4.3 退出登陆

`supabase.auth.signOut()`：该方法是实现退出登录的方法。

app/actions.ts

```
1  await supabase.auth.signOut();
```



友情提示

记得git commit保存项目代码哦~