|  |
| --- |
| به‌نام خداوند بخشنده و مهربان  **تمرین دوم:** توسعه اپلیکیشن **استاد:** محمد‌علی نعمت‌بخش  **درس:** پایگاه‌داده پیشرفته دستیار: امیر سرتیپی  **نام و نام‌خانوادگی:** سید عمید اسدالهی مجد  **شماره دانشجویی:** 4003614004 |

* لینک گیت‌هاب: <https://github.com/amidmajd/user-auth-exercise>

## مقدمه

در این تمرین در قسمت بک-اند از فریم‌ورک Django و Django Rest و در فرانت-اند از Bootstrap در CSS استفاده شد. در بک-اند API هایی به منظور ثبت‌نام، احراز هویت[[1]](#footnote-1) و اصالت کاربر[[2]](#footnote-2) تعریف شدند. در فرانت-اند با استفاده از Javascript از این API‌ها استفاده شد. به منظور احراز هویت و اصالت کاربر از توکن JWT و روش ذخیره سازی در کوکی‌ها بهره گرفته شد. این توکن هنگام احراز هویت و اصالت برای هر کاربر منحصرا تولید میشود و در قسمت کوکی‌های پاسخ تحت عنوان JWT و به صورت رمز‌گذاری شده ارسال می‌شود. با استفاده از این توکن می‌توان مطمئن بود کاربر معتبر است و شناسه (آیدی) کاربر چیست و در هر درخواست دیگر نیازی به بررسی کلمه عبور کاربر نیست.

## راهنمای اجرا

نصب پیش‌نیاز‌های Back-end با دستور pip install -r requirements.txt و ساخت پایگاه داده جنگو با اجرای دستورات python manage.py makemigrations و python manage.py makemigrations user و python manage.py migrate و اجرای سرور بک-اند با دستور python manage.py runserver

پس از اجرای سرور جنگو، میتوان فایل index.html را با مرورگر باز نمود و فرانت‌-اند نیاز به دستور خاصی برای اجرا ندارد.

## بک-اند

در بخش مدل‌های جنگو مدل جدیدی با توسعه مدل پیشفرض کاربر پیاده سازی شد که شامل ویژگی‌های یادشده در تمرین میباشد و برای ثبت نام از ایمیل کاربر استفاده میشود. بنابراین ایمیل‌ها در سیستم یکتا می‌باشند. در فریم‌ورک Django Rest به منظور ارتباط با پایگاه داده و قالب‌بندی داده‌ها از Serializer‌ها استفاده میشود که با توجه به تمرین دو Serializer تعریف شدند. در ادامه API های پیاده سازی شده در بک-اند معرفی میشوند.

### ثبت‌نام (RegisterView)

* **آدرس:** /user/register
* **متد:** POST
* **شرح وظیفه:** دریافت اطلاعات مورد نیاز در درخواست از کاربر و بررسی درست بودن و فرمت مناسب‌ داده‌ها و سپس ذخیره در پایگاه داده و ثبت نام کاربر.
* **ورودی:** ایمیل، رمز عبور، نام، نام خانوادگی، تاریخ تولد، شماره تماس، آدرس و جنسیت
* **خروجی:** 
  + پیام موفقیت و اطلاعات ذخیره شده در پایگاه داده
  + خطاهایی نظیر وجود کاربر قبلی با ایمیل وارد شده و ...

### ورود (LoginView)

* **آدرس:** /user/login
* **متد:** POST
* **شرح وظیفه:** دریافت اطلاعات مورد نیاز و احراز هویت و اصالت کاربر. در صورت صحت اطلاعات توکن JWT را مختص کاربر با اعتبار ۶ ساعت تولید می‌کند.
* **ورودی:** ایمیل و رمز عبور
* **خروجی:** 
  + پیام موفقیت و آیدی کاربر
  + توکن JWT تولید شده در کوکی‌های پاسخ
  + خطاهایی نظیر عدم وجود کاربر با آدرس ایمیل ورودی، اشتباه بودن رمز عبور و ...

### خروج (LogoutView)

* **آدرس:** /user/logout
* **متد:** POST
* **شرح وظیفه:** بررسی درخواست و حذف توکن JWT کاربر از کوکی‌ها
* **ورودی:** خالی به همراه کوکی‌ها در درخوسات با توکن JWT از پیش تنظیم شده هنگام ورود کاربر
* **خروجی:** پیام موفقیت

### اطلاعات کاربر (UserView)

* **آدرس:** /user/user
* **متد:** GET
* **شرح وظیفه:** بررسی درخواست و توکن JWT موجود در کوکی‌های درخواست و احراز هویت و اصالت کاربر از این طریق. در صورت عبور موفقیت آمیز از بررسی‌ها اقدام به دریافت اطلاعات همان کاربر از پایگاه داده و ارسال آن‌ها میکند.
* **ورودی:** خالی - کوکی با توکن JWT از پیش تنظیم شده هنگام ورود کاربر
* **خروجی:** 
  + پیام موفقیت و اطلاعات کامل کاربر
  + خطاهایی نظیر کاربر احراز هویت نشده و ...

### اطلاعات کاربران (UsersView)

* **آدرس:** /user/users
* **متد:** GET
* **شرح وظیفه:** بررسی درخواست و توکن JWT موجود در کوکی‌های درخواست و احراز هویت و اصالت کاربر از این طریق. در واقع اگر کاربر وارد شده باشد می‌تواند با این API اطلاعات مختصری از تمام کاربران ثبت‌نام شده را مشاهده نماید. پس از عبور موفقیت‌‌آمیز از بررسی‌ها اقدام به دریافت اطلاعات مختصر همه کاربر از پایگاه داده و ارسال آن‌ها میکند.
* **ورودی:** خالی - کوکی با توکن JWT از پیش تنظیم شده هنگام ورود کاربر
* **خروجی:** 
  + پیام موفقیت و اطلاعات مختصر همه کاربران (شناسه، ایمیل، نام، نام خانوادگی، شماره تماس و جنسیت)
  + خطاهایی نظیر کاربر احراز هویت نشده و ...

### ویرایش اطلاعات کاربر (UpdateView)

* **آدرس:** /user/update/<user\_id>
* **متد:** PUT
* **شرح وظیفه:** بررسی درخواست و توکن JWT موجود در کوکی‌های درخواست و احراز هویت و اصالت کاربر از این طریق و بررسی اینکه آیا کاربر می‌خواهد اطلاعات خود را ویرایش کند یا خیر. در صورت عبور موفقیت‌آمیز از بررسی‌ها اطلاعات ورودی را بررسی کرده و اگر در فرمت مناسب بودند اقدام به ذخیره اطلاعات جدید در پایگاه داده می‌نماید.
* **ورودی:** خالی - کوکی با توکن JWT از پیش تنظیم شده هنگام ورود کاربر
* **خروجی:** 
  + پیام موفقیت و اطلاعات جدید ذخیره شده در پایگاه داده
  + خطاهایی نظیر کاربر احراز هویت نشده یا کاربر احراز اصالت نشده و قصد ویرایش اطلاعات کاربر دیگری را دارد و ...

## فرانت-اند

با بهره‌گیری از Javascript تغییر محتویات صفحه و دریافت اطلاعات از سرور انجام می‌شود. فرانت-اند ابتدا در حالت صفحه خانه قرار دارد و می‌توان با کلیک بر روی هرکدام از منو‌ها به صفحه مورد نظر رفت. همچنین در صورت نیاز پیامی کنار صفحه به منظور اطلاع رسانی به کاربر نشان داده خواهد شد.

با کلیک بر روی هر منو آن منو در بالای صفحه رنگ روشن‌تری میگیرد و نشان می‌دهد کاربر در آن صفحه قرار دارد. همچنین محتوای صفحه با توجه به آن منو تغییر میکند. در واقع صفحه عوض نمی‌شود ولی محتوا عوض می‌شود و یک طراحی SPA[[3]](#footnote-3) داریم.

یک متغییر تحت عنوان BASE\_URL در ابتدای فایل جاوااسکریپت تعریف شده است که آدرس پایه سرور است. این متغییر در حالت توسعه مقدار <http://localhost:8000> را دارد که آدرس سرور محلی بک-اند (جنگو) است.

با بهره‌گیری از سرور درخواست های مورد نیاز انجام می‌شود و پاسخ در فرمت مناسب به کاربر نمایش داده می‌شوند. تمام درخواست‌ها با بهره‌گیری از قابلیت AJAX جاوااسکریپت (که non-blocking هستند) و async-await آن انجام می‌شوند. برای دسترسی به بخش‌هایی نظیر اطلاعات کاربر نیاز به ورود می‌باشد که پس از ورود توکن JWT در کوکی‌ها ذخیره می‌شود و شناسه کاربر نیز در جاوااسکریپت نگه‌داری می‌شود. در درخواست ها با استفاده از پارامتر  
credentials: "include" کوکی‌ها نیز همراه درخواست ارسال می‌شوند. این کوکی‌ها شامل توکن JWT هستند که سرور بتواند با بررسی آن صحت کاربر را تشخیص داده و نیاز به ارسال ایمیل و رمز عبور در هر درخواست نباشد.

بخشی نیز برای مدریت خطا‌ها و نمایش آن‌ها در فرمت و پیام مناسب به کاربر نیز درنظر گرفته‌ شده است.

### مثال: بررسی نحوه کار بخش ویرایش اطلاعات کاربر کنونی

در ابتدا با کلیک بر روی منوی مربوطه، این بخش فعال می‌شود و محتواهای دیگر از صفحه پاک می‌شوند. سپس درخواستی به همراه کوکی شامل توکن JWT به سرور (آدرس /user/user) به منظور دریافت اطلاعات کاربر کنونی ارسال می‌شود.

سرور درخواست و توکن ارسالی موجود در کوکی درخواست را بررسی می‌کند و اگر معتبر بود اطلاعات کاربر را از پایگاه داده دریافت و به عنوان پاسخ به همراه پیام موفق ارسال می‌نماید.

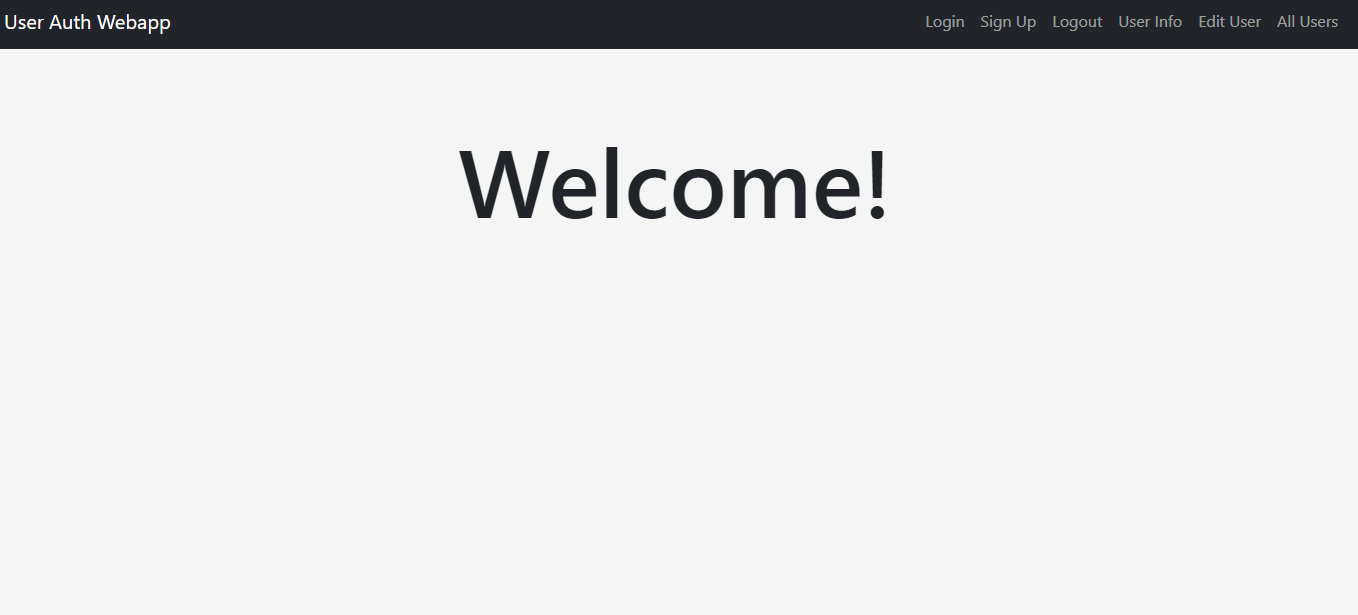
پس از دریافت اطلاعات از سرور ابتدا تابع مدریت خطا صدا زده می‌شود تا اگر خطایی در پاسخ وجود دارد به کاربر نمایش داده شود. در صورت عدم وجود خطا از سمت سرور، اطلاعات در فرم ویرایش قرار داده می‌شوند.

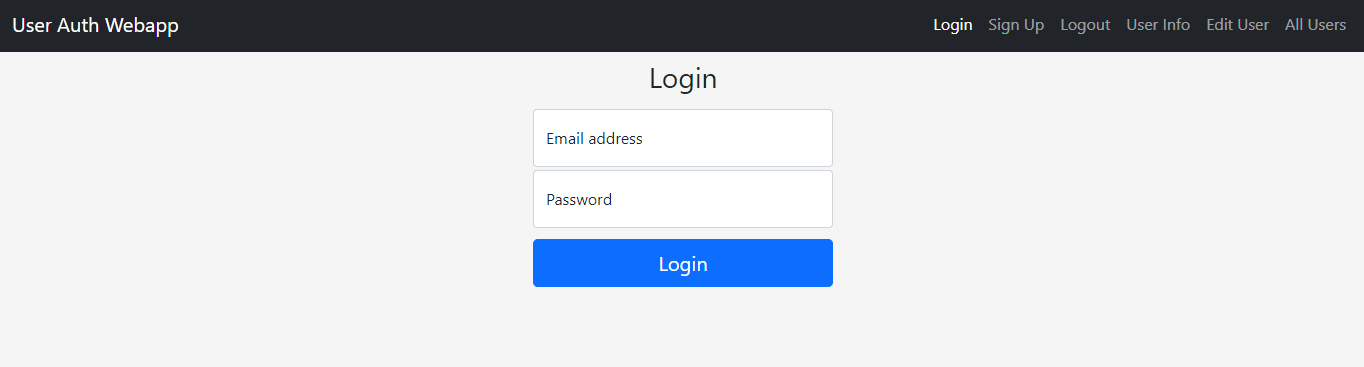
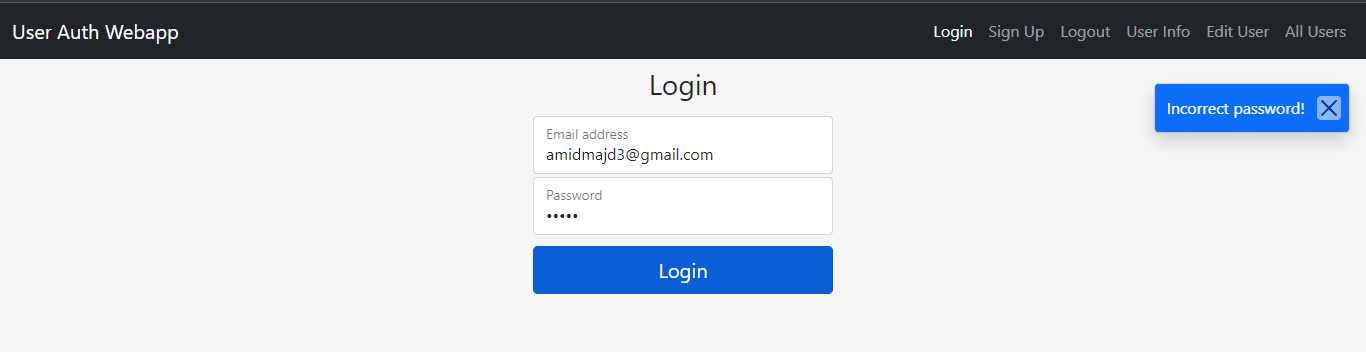
پس از ویرایش اطلاعات و کلیک کاربر روی دکمه ذخیره، درخواست دیگری به سرور (آدرس /user/update/<user\_id>) ارسال می‌شود که شامل اطلاعات جدید است و توکن JWT در بخش کوکی است.

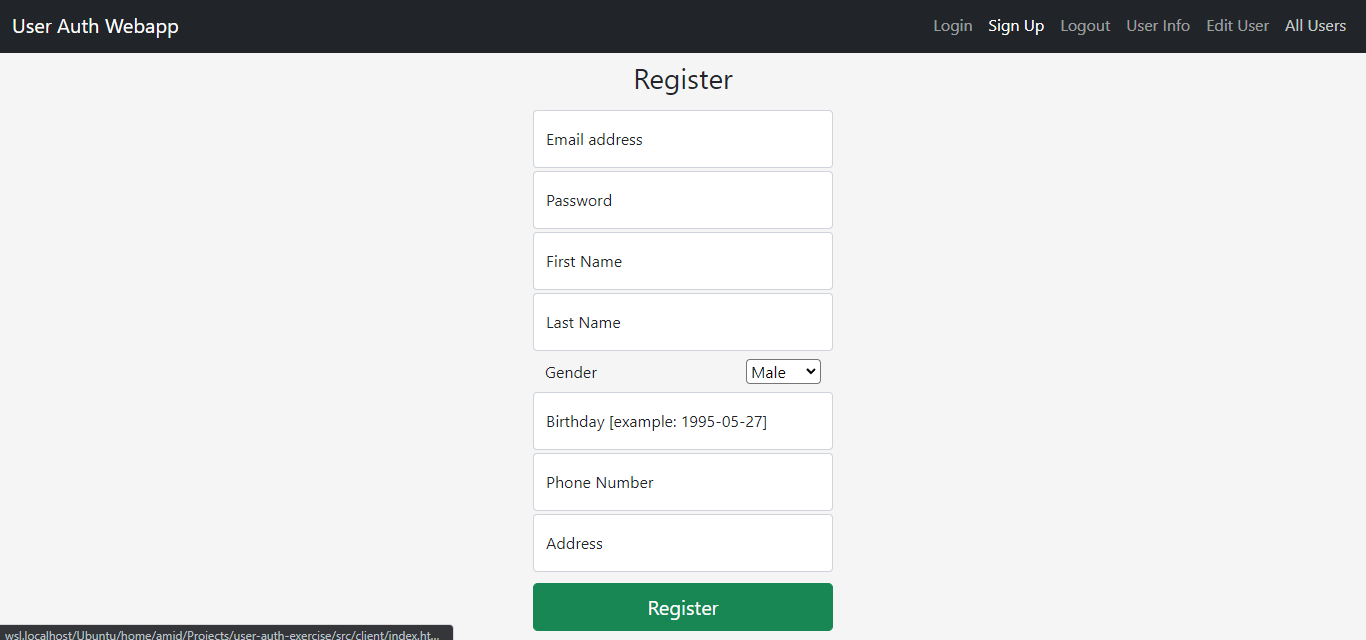
سرور درخواست و توکن ارسالی موجود در کوکی درخواست را بررسی می‌کند و اگر معتبر بود و همچنین اگر اطلاعات ورودی معتبر بود اقدام به ذخیره آن‌ها در پایگاه‌داده می‌نماید و پیام موفقیت آمیز بودن درخواست را ارسال می‌نماید.

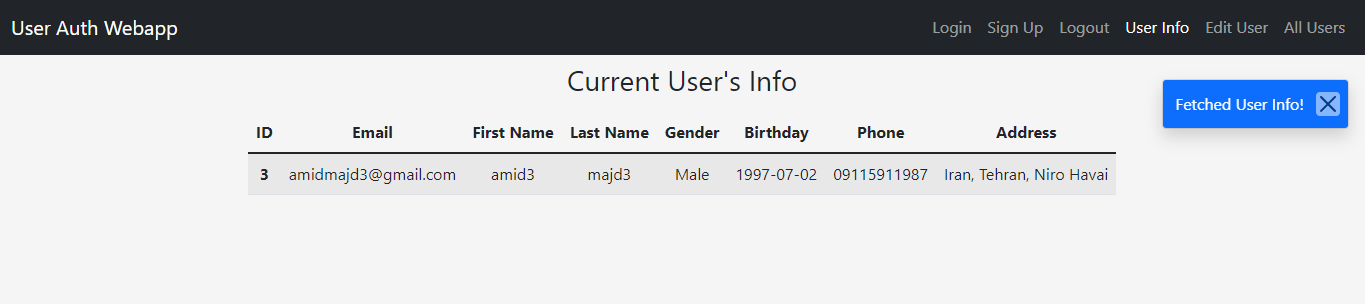
فرانت-اند این پاسخ موفقیت آمیز را دریافت می‌کند و پیامی به کاربر به معنای موفقیت آمیز بودن درخواست نشان می‌دهد.

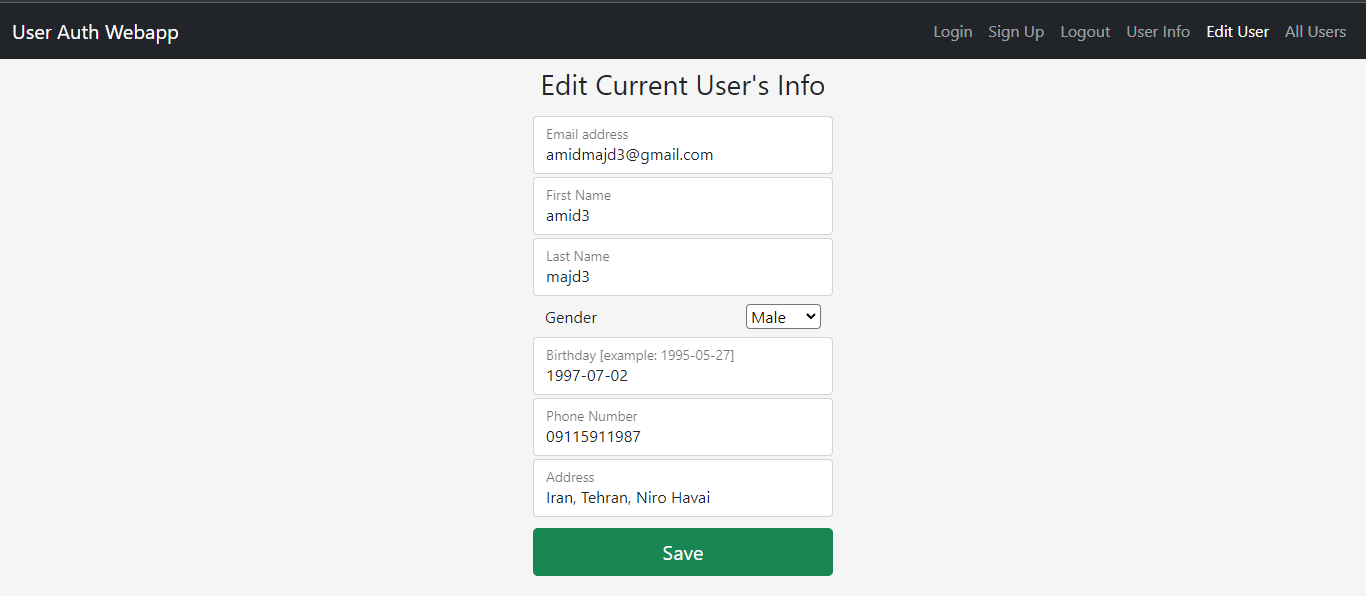
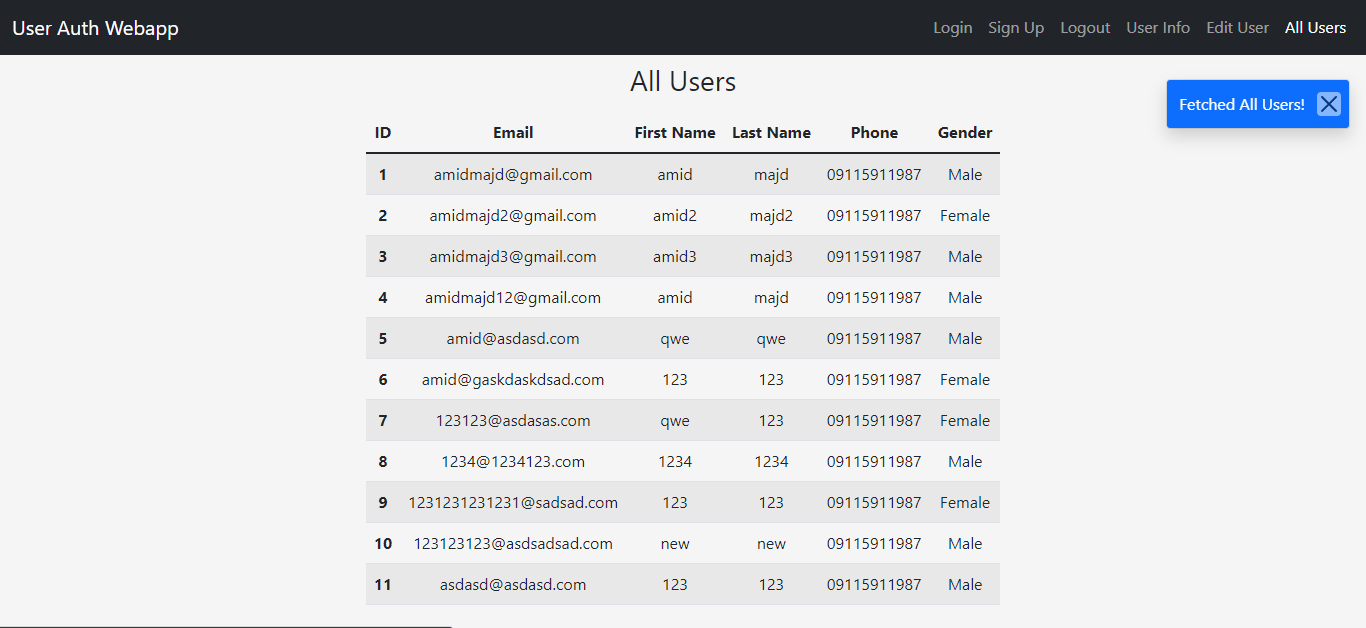
## نمونه تصاویر از فرانت-اند







1. Authentication [↑](#footnote-ref-1)
2. Authorization [↑](#footnote-ref-2)
3. Single Page Application [↑](#footnote-ref-3)