SWEN 325 Ionic App Report

Zhanghao Kong
300432074

Architecture of the App

(1) Overall Architecture

The app is a todo list app for users to record their schedules. The design purpose of the app is giving users a good support on managing their schedule. At the current stage, the app is just designed to have a functionality to record schedules.

This app is designed to be supported by various device types. When users use the app to record their schedules, the app would just require text typing. So, there is no special requirement for the hardware device. Any normal mobiles which can access the internet is able to use the app. Because I use an ionic capacitor to develop the app, so the app can be easily deployed to either iOS or Android environments. This is also one of the good points of using an ionic framework.

Because I use firebase, users schedules created in the local can be pushed to the remote. But their phones must be connected to the internet. Besides, because it's a thin client app, it is much easier than building a rich client app. A rich client app is much more complex and harder to be maintained.

In my app,it's having 7 different pages with different functions.

pages:

1.event-create:use to create new schedule.

2.event-list:to check all schedule here.

3.home:for user to choose create new schedule or check added schedule

4.login:login page

5.password-reset:reset password page

6.service:event and user services here.

7.sign-up:sign up page

8.profile:recording user's information

(2) Code Architecture

Although I have 8 different pages, only two different types in it. Page service is distinctive because all other pages are used to implement screen only, service is used to provide methods for other pages to use.This structure breaks my app into distinct features and each component of my app only needs to take responsibility for its own specific functionality. So, things are not messed up and the app is easy to be maintained and extended.

The separation of service and page avoids the self repetition of coding and makes the code composition work perfectly. Instead of adding code directly to a particular page, I use a service to save some specific code. The execution of this specific code is independent of the page and can be reused elsewhere. For example, if my application often needs to get data from a remote database and display it on the screen, it's better to use services than encode them on each page.

2. External Component – Firebase

Firebase is a great platform that provides a lot of back-end support to help people make their applications better. In my application, I use firebase to support my application's authentication, cloud storage, and real-time database.

Integrating firebase into my ionic application is mainly through the app.component.ts file and firebase.config.ts file. These files specify the root component of the application, which is the first component loaded by the application runtime. The root is basically an empty shell for other components to load into the firebase.

I first import firebase into the root component file, telling the root user that I need to use firebase for the application. After that, I get a namespace from the import that I can use to access all the core functions in firebase. Next, I went to the firebase website and created a workspace for my application. Connecting my application to the workspace is very simple. All I need to do is pass the credentials generated by firebase web to the firebase initialization function in the root component.

It can take a long time to set up my own authentication system, so I chose firebase as the back end of the application to enable me to set up email and password authentication systems.

The first thing I needed to do was to activate the Email/Password sign-in method on the Firebase website for my app. Then, because there are 4 pages going to use this authentication system, I created a service to handle all the authentication related interactions between the app and Firebase. The 4 pages are login page with logout , sign-up page, reset-password page and profile page with a logout functionality as well. Inside the created service, everything was going to be simple, since all I needed to do was just to call the related function from Firebase and Firebase handles everything behind the scene for me.

Real time databases are mainly used to deal with crud operation in application programs. Crud means create, read, update, and delete. These operations can be used not only to study the modification of events, but also to modify user profiles. Both features of the application must have real-time updates after the user makes changes. By using a real-time database, applications can synchronize changed data and immediately display the changes to users.

3.advantages and disadvantages of ionic framework:

Advantage:

1.Easy to test

As long as the ionic application can only run through the web view, you can use the device's browser to test the application. This is more convenient, because you don't even have to use the test equipment to make sure everything runs smoothly. The same concept applies to all kinds of mobile devices in the modern world.

The browser provides built-in testing and debugging tools to simplify the whole testing process. To test the angular components used in older versions, you can use the angular cli, which is suitable for web component testing. Therefore, only a test device or emulator may be required to test some native functions.

2.simple code structure

With just a simple command run, “ionic start”, Ionic framework provided a concise code structure to me, giving me a very good start on building the app. The auto-generated app project has an app folder containing a home page, an assets folder for holding properties (e.g. icon images) and a theme folder with a SCSS file specifying theme colors that are going to be used by the app. The auto- generated files under the home page folder was really helpful for understanding what the necessary components of an Ionic app are. For the home page, there are 4 files making it up. They are one module.ts file, for locating the page during page navigations; one SCSS file, for making the page view on the screen nicely; one .ts file, for declaring all the functionality the page has and a spec.ts file, for testing purposes.

The file structure for a single page seems complicated. In fact, if we want to generate a new page, we don’t need to create those files one by one instead we just need to type a short command on CLI, “ionic generate page”, then all necessary components for the page are generated. Generating a new service is also the same, with command “ionic generate service”.

Disadvantage:

1.no hot-reloading

I found that there is no hot-reloading for Ionic framework. Hot reloading is a software engineering term, which is saying a technique that allows the app to just refresh the current working file but not the whole app and enables the app to continuously work, implement changes in a live mode. Because Ionic doesn’t have the feature so it makes the testing become complicated. While I was developing the app, each time I made a change, the whole app was refreshed, and the screen would absolutely go back to the home page. So, if I want to see the change I made in my event-list page, I have to navigate back to that page manually. This advantage could possibly cause huge time consuming when the testing is on a heavy application.

2.Plugin-dependent system

When developing applications with ionic, there are too many codes, imports, and plug-ins to write. In this case, coding is the code that needs to be written in HTML, CSS, and JavaScript to build each page; imports is the library of angular; plug-ins are features of Cordova and capacitor. For example, in order to implement a simple table view of the user profile on the profile page, it is necessary to write a lot of code, such as. HTML, introduction. Ts and profile.service.ts Documents. Since the size of the application is still small, this issue doesn't seem to matter, but if the application gets bigger and bigger, it can make the application heavy and difficult to maintain.

4.Usability test

Goals for the usability test:

1.user can create their own account.

2. With users' own account , they can record their schedules and can check their schedules anytime.

3.user can delete their schedule anytime when that schedule passed.

4.user can upload their profile to firebase.

Format of the usability test

Through multiple instance tests (create users, record schedules, use all functions in the application) to prove whether the application meets the target.

Target users(and how many):

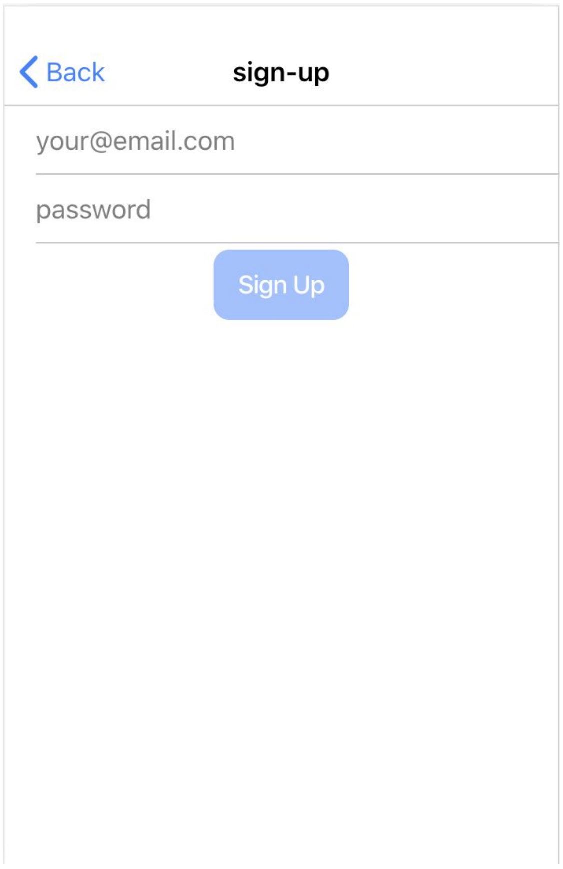
1.students: students have many homework or lectures, so they need to use this app to remind themselves when they need to finish their assignment or go to lecture.

2.office worker: For office worker, they will have many meetings ,dinner or parties. My app can tell them what need to be done or what they have to prepare to do on time.

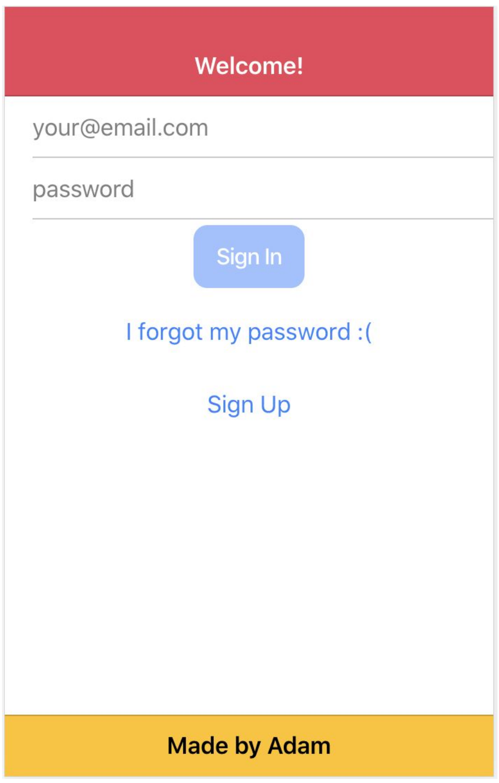
Most of people who like to set up plan are all suitable to use this app, it can help them to establish a sense of time and mention good supervision effect.

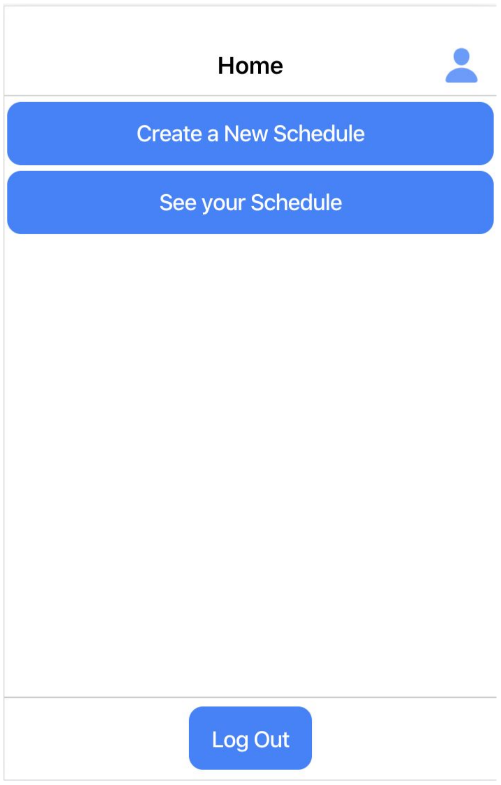
Tasks that users have to perform with the App (including screen shots):

1.user need to sign up a account for use.

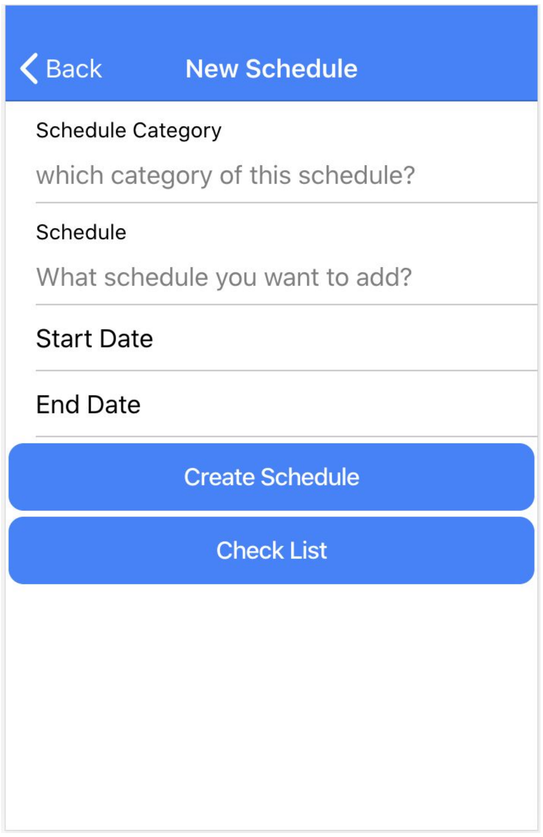


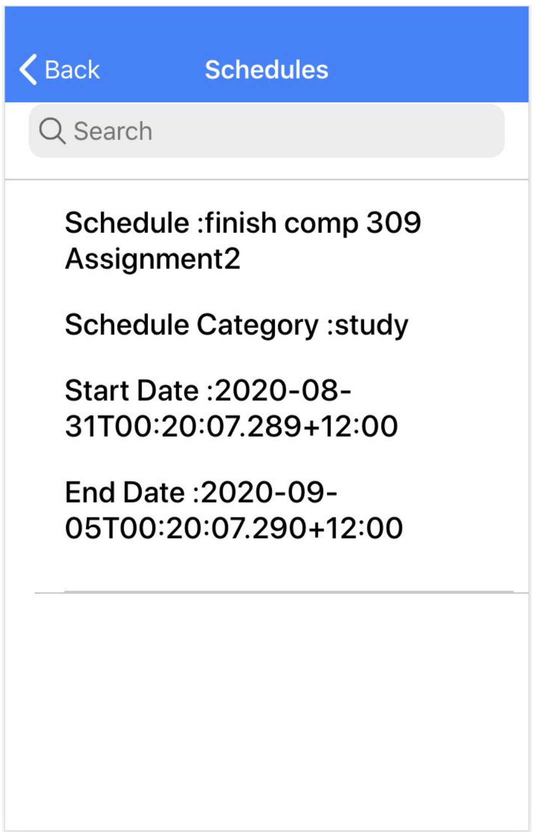
2.after sign up, user can login and create their own schedules.





3.then user can create their own schedule and check them in schedule list.


Back
New Schedule
Schedule Category
which category of this schedule?
Schedule
What schedule you want to add?
Start Date
End Date
Create Schedule
Check List


Back
Schedules
Search

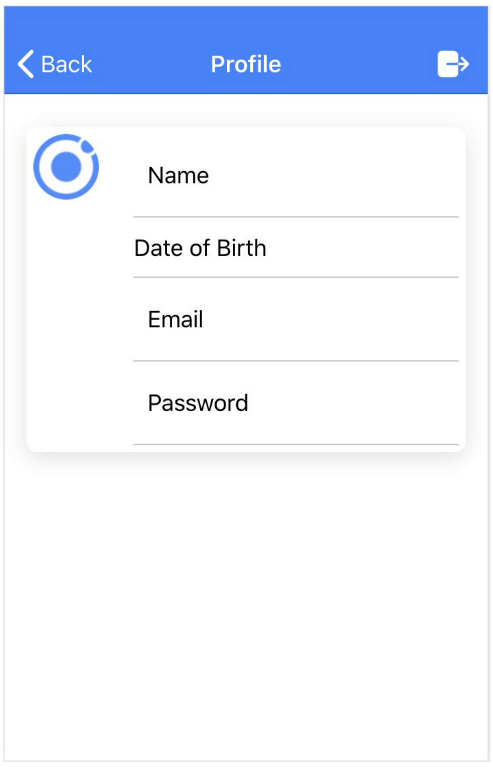
Schedule :finish comp 309 Assignment2

Schedule Category :study

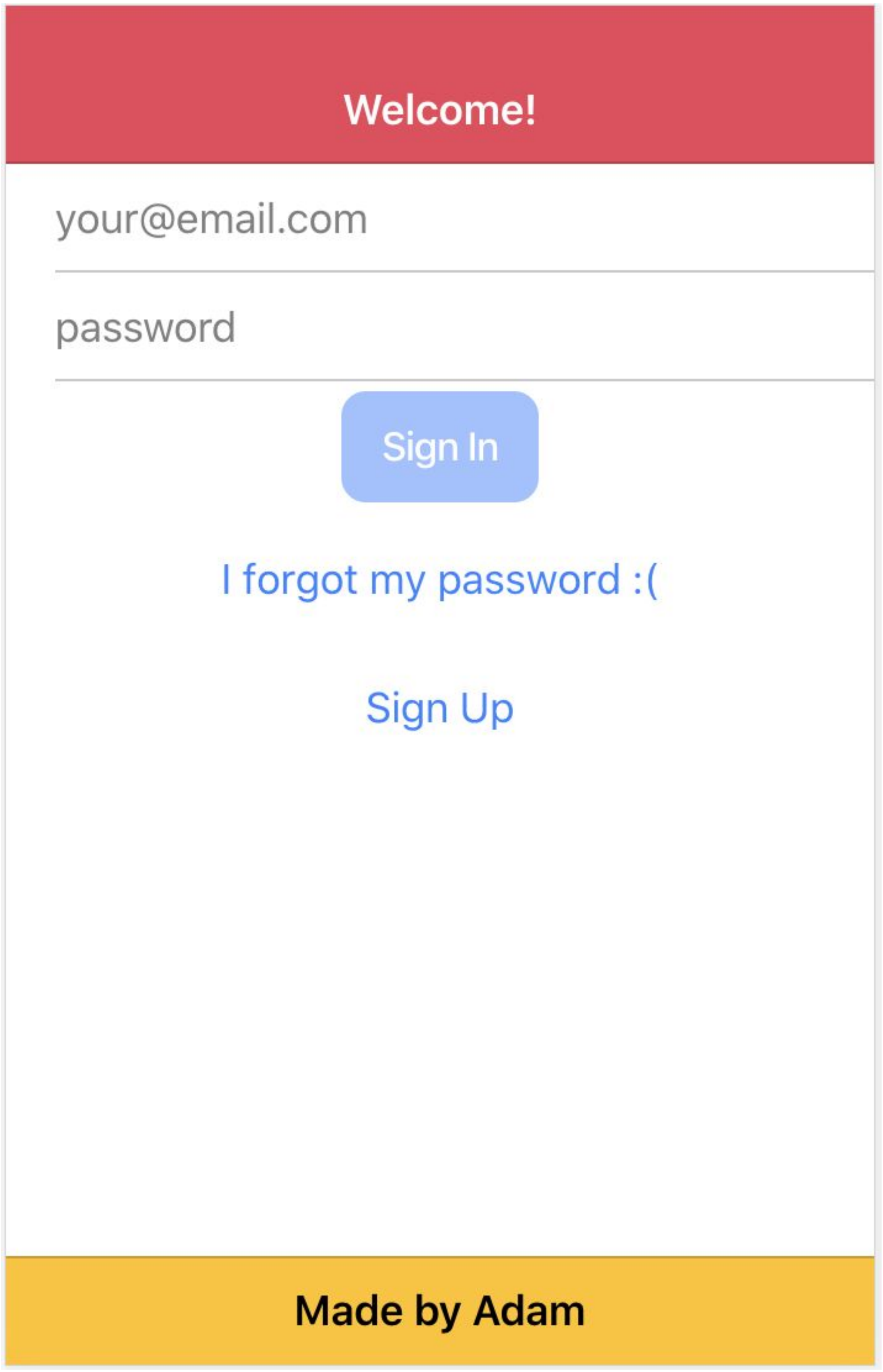
Start Date :2020-08-31T00:20:07.289+12:00

End Date :2020-09-05T00:20:07.290+12:00

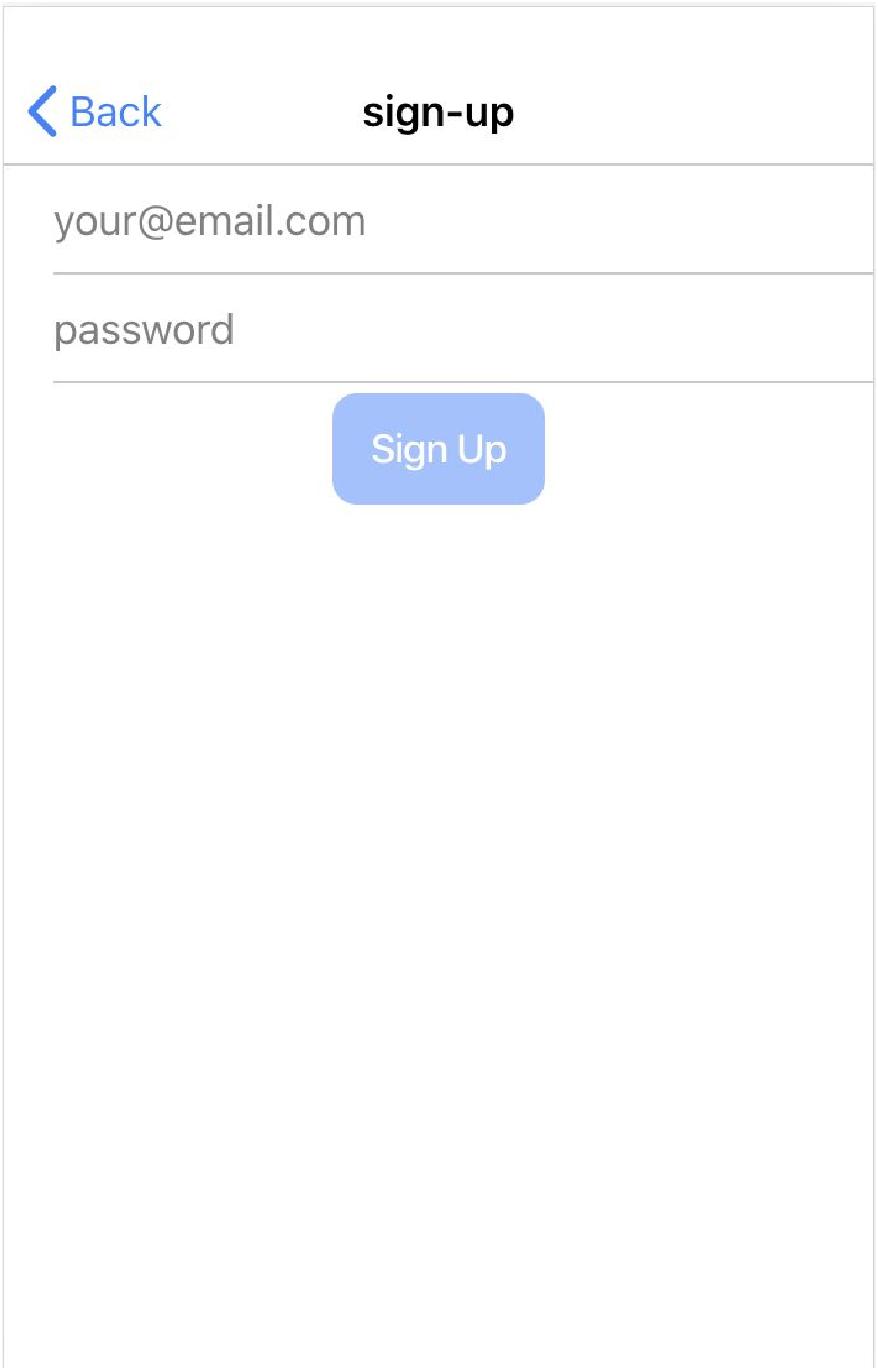
4.User can upload their profile here.


Back
Profile
Name
Date of Birth
Email
Password

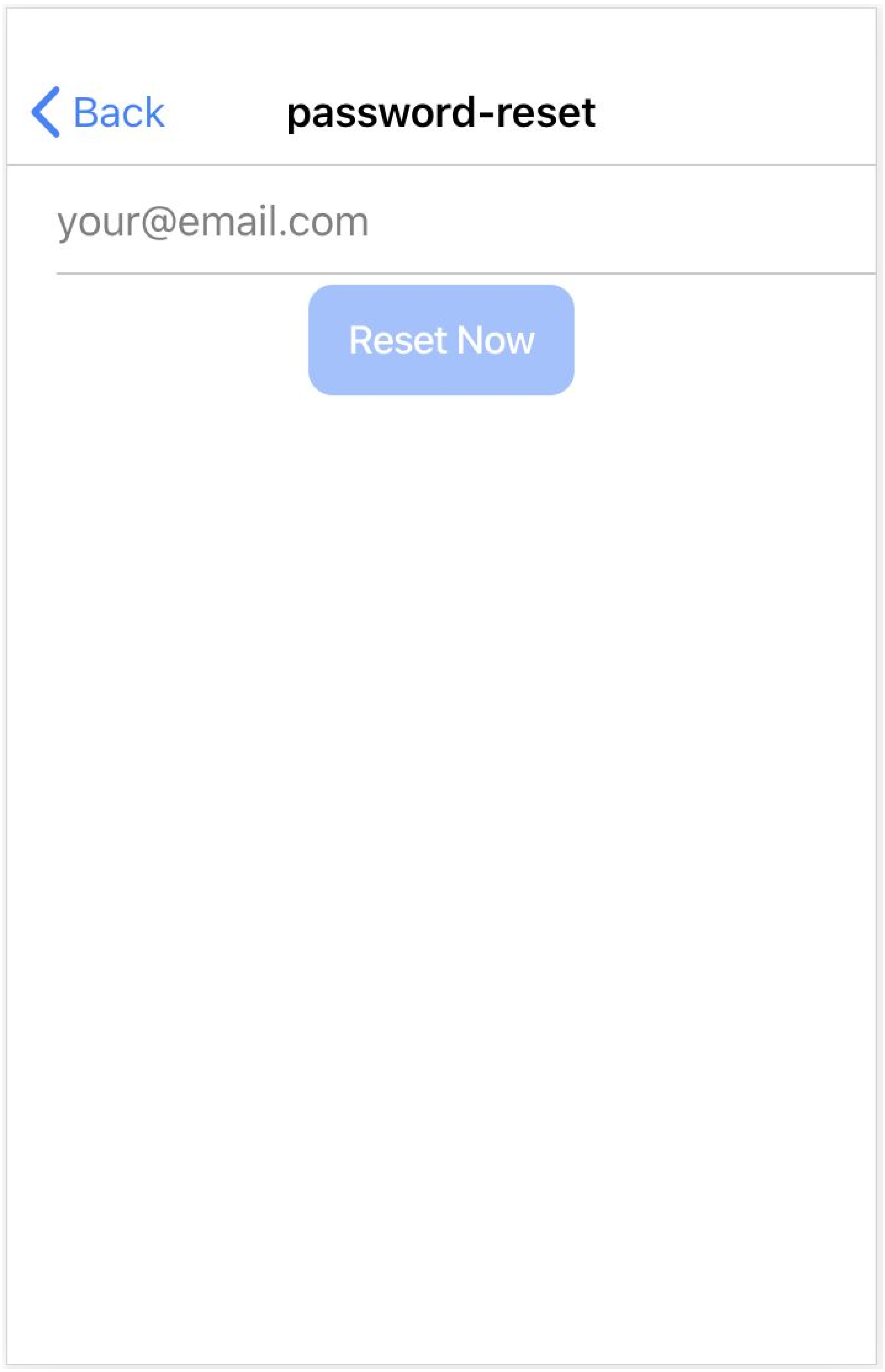
1.Login page: design for let user login in their account or change password.New user also can create a new account here by clicking “sign up” to move to sign up page.



2.Sign up page: User can enter their email and password to create a new account for store their schedules.

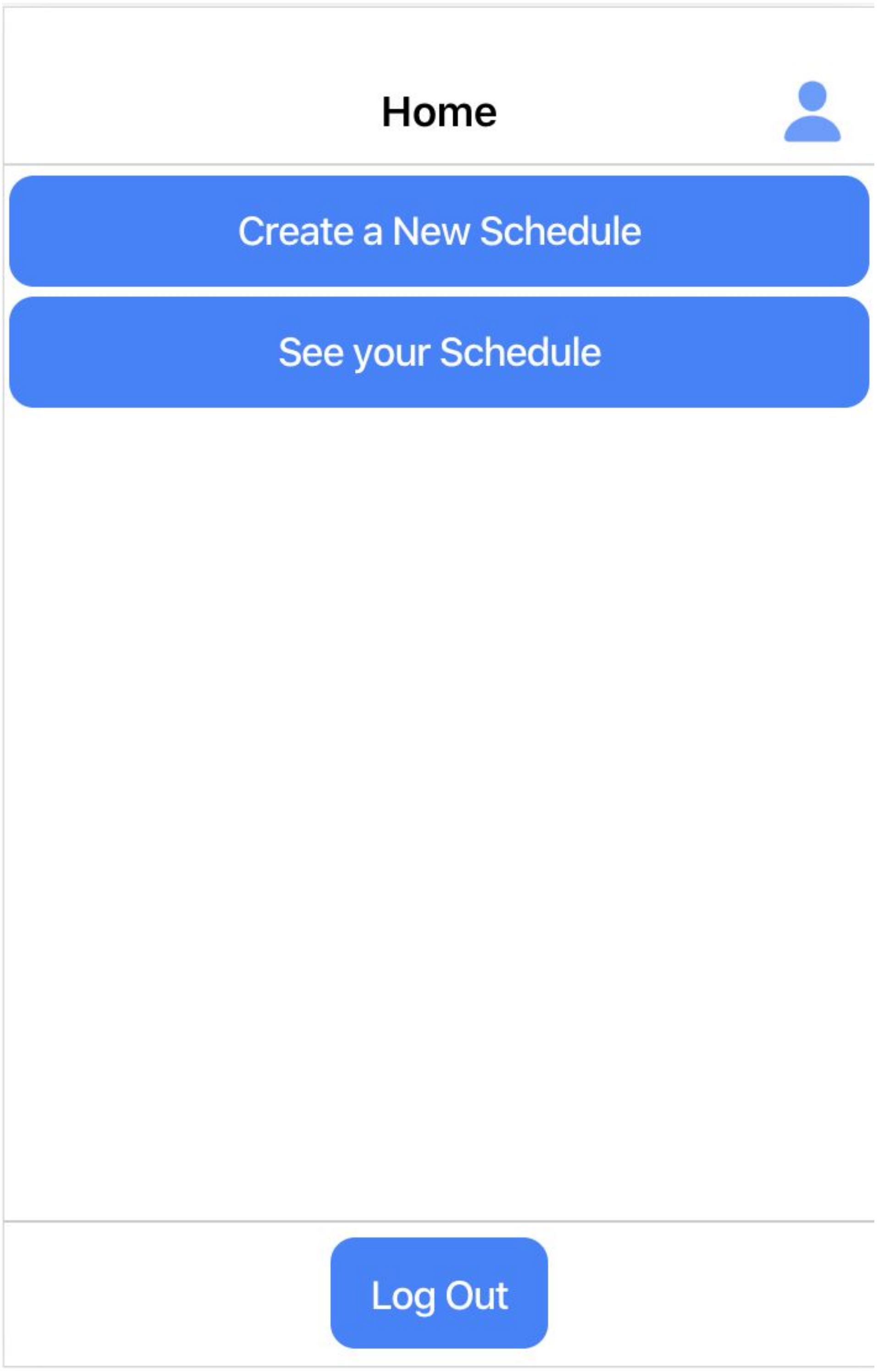


3.Password-reset page:user can change their password in this page with a email which is already sign up.

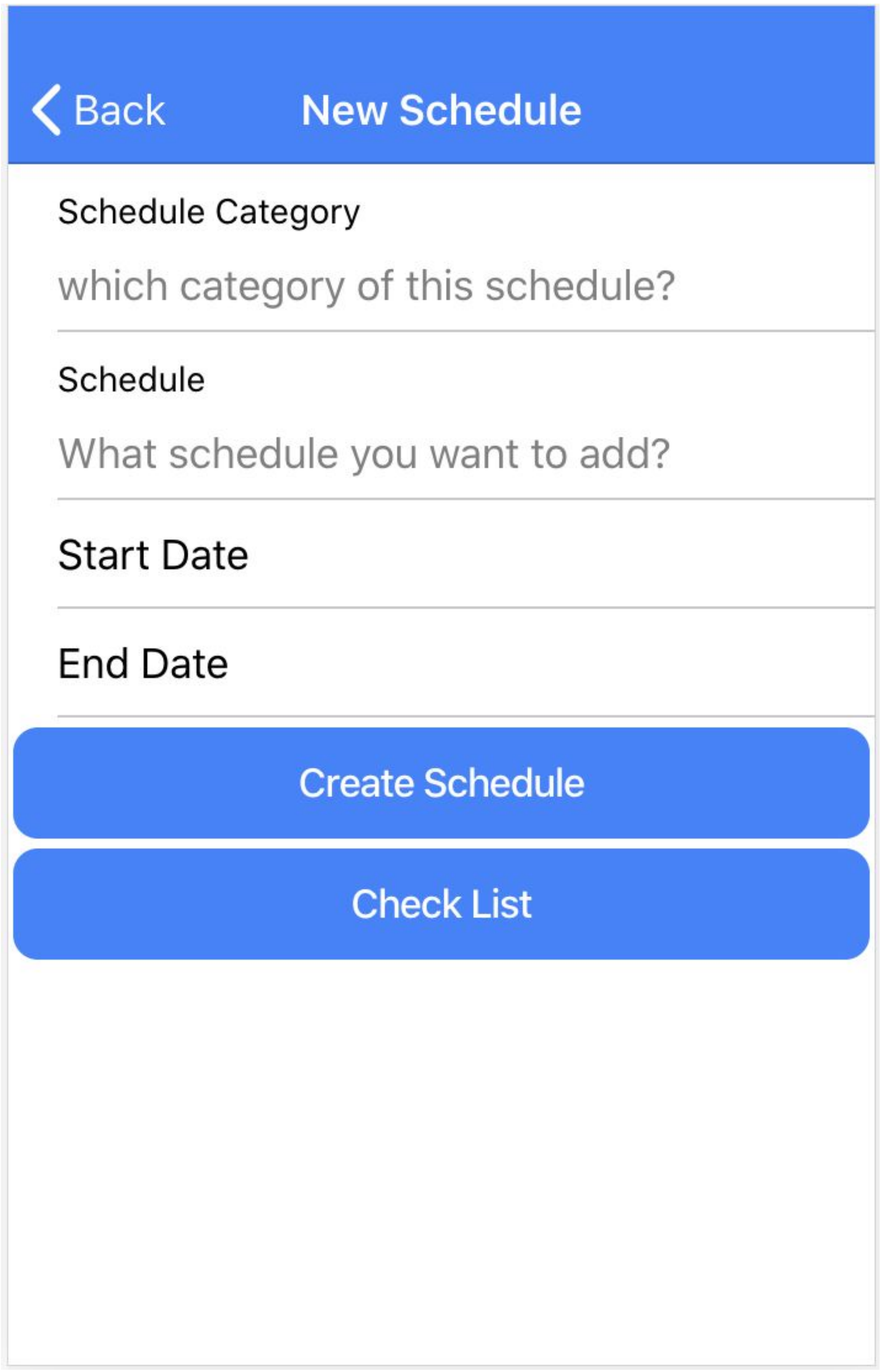


4.Home page: in this page, there are 3 buttons, profile, create new schedule and see added schedule.

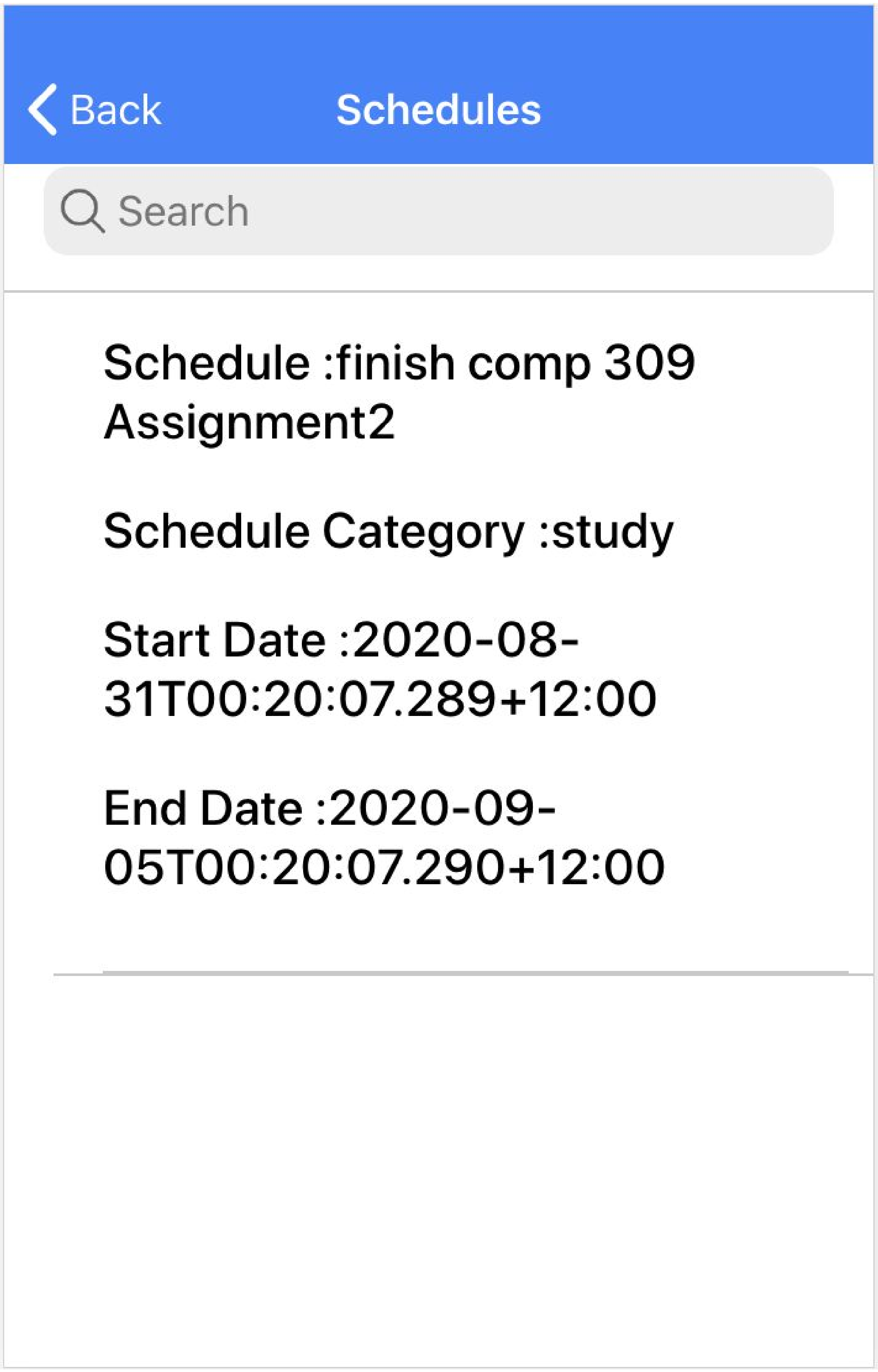
Users can create new schedules or check the schedule list here. Besides, users also can upload profile details by clicking the right-top button to move to the profile page.



5.create event page: in this page, it’s designed to let user to create schedule , user need to enter 4 different attributes to finish adding new schedule. After adding new schedule. Click “check list ” button, page can jump to list page.



6.list page: all added schedules are showing here. This is designed for users to check their schedule. It's the main part of this app. All attributes of each schedule will be shown here.



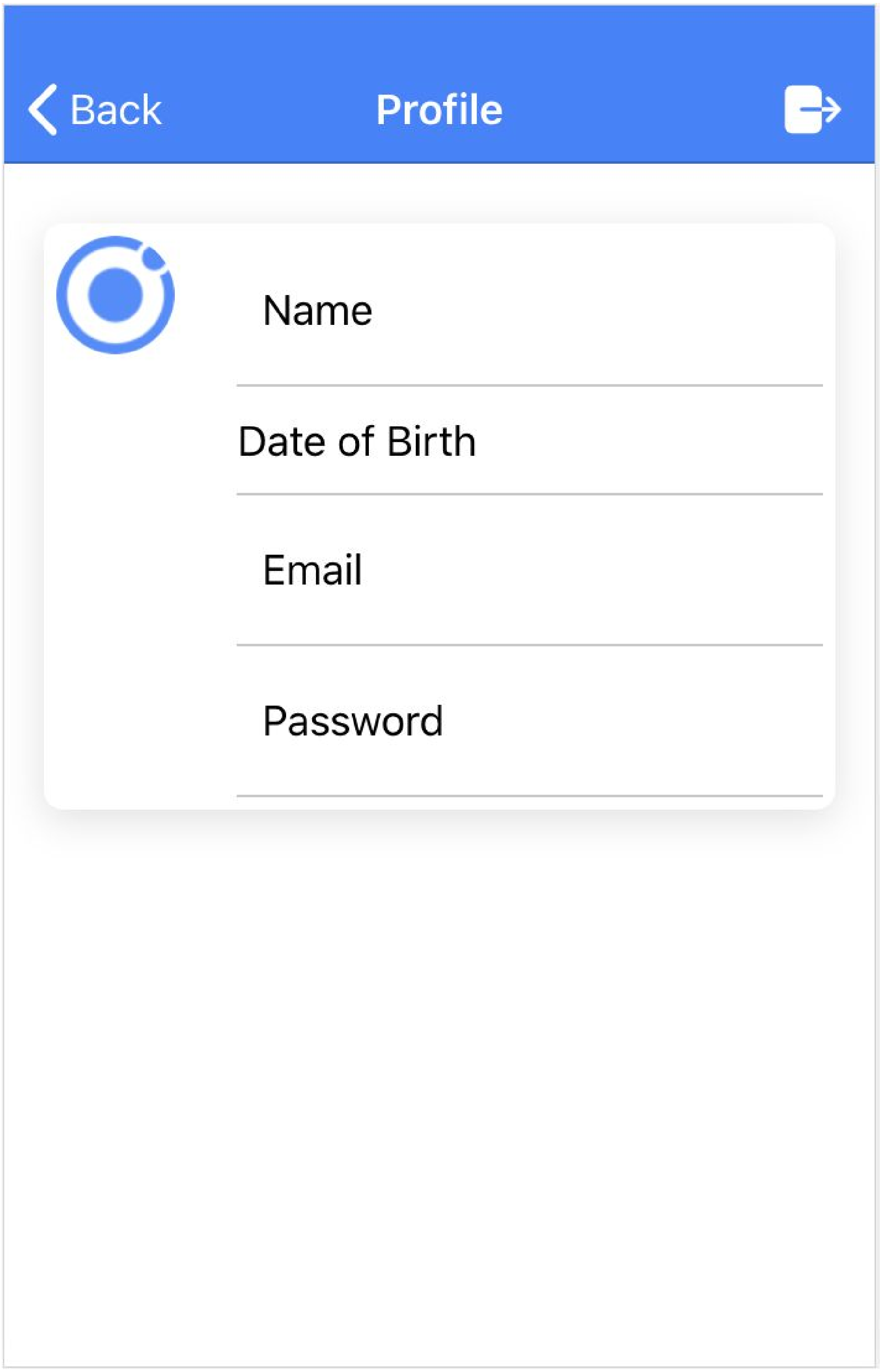
Schedule :finish comp 309 Assignment2

Schedule Category :study

Start Date :2020-08-31T00:20:07.289+12:00

End Date :2020-09-05T00:20:07.290+12:00

7.profile page:this page is design for uploading users details.besides, at right-top, there is a button for log out as well.


Back
Profile
Name
Date of Birth
Email
Password