## Version History

<table>
<thead>
<tr>
<th>Date</th>
<th>Author</th>
<th>Version</th>
<th>Changes made</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/22/2020</td>
<td>JZ</td>
<td>1.0</td>
<td>Filled in content</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Table of Contents

1. Introduction
   1.1. System Requirements
   1.2. System Module Design
   1.3. Module Requirements
       1.3.1. Functional Requirements
       1.3.2. Non Functional (Quality of Service) Requirements

2. Installation
   2.1. Frontend Installation
   2.2. Backend Installation

3. Deployment
   3.1. Frontend Deployment
       3.1.1. Local Deployment
       3.1.2. Remote Deployment
   3.2. Backend Deployment
       3.2.1. Local Deployment
       3.2.2. Remote Deployment
   3.3. Common Usage of Firebase
       3.3.1. Authentication
       3.3.2. Firestore
       3.3.3. Storage
       3.3.4. Hosting
       3.3.5. Functions

4. Troubleshooting
   4.1. Common Errors and Solution
1. Introduction

1.1 System requirements

System environments: Windows 10 / MacOS / Linux / Android / IOS
Browser: Chrome, Edge, Firefox, Safari etc.
Developing environment: nodeJs, reactJS etc. (see Module Requirements below)

1.2 System Module Design

The basic layout of a resource page in 1 student at a time site consists of a header component, carousel component, a resource displaying component, and a comment component.

All of the components mentioned above work separately from each other, the same component in different resource pages also works on their own, it does not affect the component in other pages as well.

Pages besides resource pages, for instance, Who We Are page, Donation Page, does not have a comment section component or the resource displaying component or both. Instead, they are using their unique component that is not shared with any other pages.
We are using ReactJS as our frontend framework, thus, almost all content we see in any of the web pages in our app are controlled by states. We are using “react-redux” to control the states (or the content app shows to the viewers).

Besides controlling the states, part of redux also serves as a connection between frontend and backend, and the requests from users will mostly be handled by the action creator portion of redux. Responses from servers will also be processed by it.

Our APIs are written by ExpressJS, and hosted by Firebase API hosting service, it is closely connected with Firebase Firestore, in which we store all of our data except for images, and Firebase Authentication, where we store all of our user information, Firebase storage, where we put the images at.

Some web pages (these pages include sign up / login / reset password / get involved / about us / all resource pages) are protected by google recaptcha service, it protects our web pages from malicious attacks of web crawlers.

### 1.3 Module Requirements

Details of required package names and versions can be seen in the package.json file in backend repo and frontend repo.

#### 1.3.1 Functional Requirements

Functional Requirements are the requirements that must be met if the web app is to function correctly.

For Backend:

<table>
<thead>
<tr>
<th>Package name</th>
<th>Version</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>node</td>
<td>&gt;=10</td>
<td>engine</td>
</tr>
<tr>
<td>body-parser</td>
<td>^1.19.0</td>
<td>To ensure the request body can be correctly parsed into json</td>
</tr>
<tr>
<td>cors</td>
<td>^2.8.5</td>
<td>Allows a server to indicate any other origins (domain, protocol, or port) than its own from which a browser should permit loading of resources.</td>
</tr>
</tbody>
</table>
Express v4.17.1

Framework we are using to write our APIs

Firebase v7.22.1

To initialize firebase services

Firebase Admin v9.3.0

Allow us to make changes to the firestore and so on

Firebase Functions v3.6.1

Allow us to convert our express APIs into firebase functions

Node-fetch v2.6.1

Allow us to fetch data from an URL inside of our APIs. Mainly used to validate captcha in our project

For Frontend:

<table>
<thead>
<tr>
<th>Package name</th>
<th>Version</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>antd</td>
<td>^4.7.3</td>
<td>UI library, we used a lot of its components</td>
</tr>
<tr>
<td>axios</td>
<td>^0.20.0</td>
<td>We use axios to make asynchronous requests to our APIs in action creators</td>
</tr>
<tr>
<td>Library</td>
<td>Version</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>bootstrap</td>
<td>^4.5.3</td>
<td>UI library</td>
</tr>
<tr>
<td>emailjs-com</td>
<td>^2.6.3</td>
<td>We use it to send emails from users to the organization</td>
</tr>
<tr>
<td>firebase</td>
<td>^7.24.0</td>
<td>To initialize firebase services, mostly in the frontend, it is used to store pictures into storage space, and authenticate users</td>
</tr>
<tr>
<td>google-map-react</td>
<td>^2.1.9</td>
<td>Display location of the organization, used in About Us page</td>
</tr>
<tr>
<td>immutable</td>
<td>^4.0.0-rc.12</td>
<td>To ensure the security of react states, protects them from being modified unintentionally by developers</td>
</tr>
<tr>
<td>install</td>
<td>^0.13.0</td>
<td></td>
</tr>
<tr>
<td>moment</td>
<td>^2.29.0</td>
<td>It is used to convert timestamps into human readable times</td>
</tr>
<tr>
<td>react</td>
<td>^16.14.0</td>
<td>Frontend framework</td>
</tr>
<tr>
<td>Library</td>
<td>Version</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>react-bootstrap</td>
<td>^1.4.0</td>
<td>Another UI library, we major UI library we are using in our web app</td>
</tr>
<tr>
<td>react-dom</td>
<td>^16.14.0</td>
<td>Mainly used in App.js</td>
</tr>
<tr>
<td>react-google-recaptcha</td>
<td>^2.1.0</td>
<td>Protect our web app and server being spam attacked by web crawlers</td>
</tr>
<tr>
<td>react-html-parser</td>
<td>^2.0.2</td>
<td>Convert html strings into react components</td>
</tr>
<tr>
<td>react-moment</td>
<td>^0.9.7</td>
<td>Similar as moment, react-moment makes moment easier to use in react</td>
</tr>
<tr>
<td>react-redux</td>
<td>^7.2.1</td>
<td>Manage react states</td>
</tr>
<tr>
<td>Package</td>
<td>Version</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>react-responsive</td>
<td>^8.1.0</td>
<td>Makes the layout responsive to different screen sizes</td>
</tr>
<tr>
<td>react-router-dom</td>
<td>^5.2.0</td>
<td>Build routes for our web app</td>
</tr>
<tr>
<td>react-scripts</td>
<td>^3.4.4</td>
<td></td>
</tr>
<tr>
<td>react-ui-cards</td>
<td>^2.0.0</td>
<td>To properly display resource cards</td>
</tr>
<tr>
<td>react-youtube</td>
<td>^7.13.0</td>
<td>To display youtube videos</td>
</tr>
<tr>
<td>redux</td>
<td>^4.0.5</td>
<td>Dependency of react-redux</td>
</tr>
<tr>
<td>Package name</td>
<td>Version</td>
<td>Reason</td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>redux-immutable</td>
<td>^4.0.0</td>
<td>Dependency of react-redux</td>
</tr>
<tr>
<td>redux-thunk</td>
<td>^2.3.0</td>
<td>Allow action creators to return a function rather than just action objects</td>
</tr>
<tr>
<td>semantic-ui-calendar-react</td>
<td>^0.15.3</td>
<td>Used in Events Page to select dates</td>
</tr>
<tr>
<td>semantic-ui-css</td>
<td>^2.4.1</td>
<td>css sheets that supports semantic UI library</td>
</tr>
<tr>
<td>semantic-ui-react</td>
<td>^1.2.1</td>
<td>Another UI library</td>
</tr>
<tr>
<td>styled-components</td>
<td>^5.2.0</td>
<td>render styles to components</td>
</tr>
</tbody>
</table>

### 1.3.2 Non Functional Requirements

**For Backend:** none.  
**For Frontend:**

<table>
<thead>
<tr>
<th>Package name</th>
<th>Version</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package</td>
<td>Version</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>@ant-design/icons</td>
<td>^4.2.2</td>
<td>It contains icons we are using in the project</td>
</tr>
<tr>
<td>@testing-library/jest-dom</td>
<td>^4.2.4</td>
<td></td>
</tr>
<tr>
<td>@testing-library/react</td>
<td>^9.5.0</td>
<td></td>
</tr>
<tr>
<td>@testing-library/user-event</td>
<td>^7.2.1</td>
<td></td>
</tr>
<tr>
<td>react-flippy</td>
<td>^0.1.5</td>
<td></td>
</tr>
<tr>
<td>react-font</td>
<td>^1.2.1</td>
<td>Apply google font families to the fonts of the content</td>
</tr>
</tbody>
</table>

## 2. Installation

First, go to https://github.com/JackyBreak/1ST
Clone project to your local computer by running:

```
git clone https://github.com/JackyBreak/1ST.git
```

in your console.

Second, install npm.
2.1 Frontend Installation
(Step by step installation guide)

Install firebase service, run:
`npm install -g firebase-tools`

Login into firebase, run:
`firebase login`

In your console, cd into the `frontend` folder, for instance,
`cd /d/myfolder/1ST/1st`

Install dependencies, run:
`npm install`

2.2 Backend Installation

In your console, cd into the `backend` folder, for instance,
`cd /d/myfolder/1ST/Firebase/functions`

Install dependencies, run:
`npm install`

3. Deployment

3.1 Frontend Deployment

3.1.1 Local Deployment

To test your code with fast iteration, you need to deploy/host the web app locally. Assuming you have already installed the project:

To host the web app locally, do the following:

In your console, cd into the `frontend` folder, for instance,
cd /d/myfolder/1ST/1st

Host web app on localhost:3000
npm start Or react-scripts start

It takes some time to set up the website locally, after a while, you should be able to see the website on "localhost:3000 ", if you want to use an alternative port, checkout this thread. Everytime you make some changes on your frontend code, the webpage will refresh itself.

3.2.2 Remote Deployment

After you finished your upgrade to the frontend, and you want to deploy the website in Firebase hosting to make it available to all users, you can follow the procedure below:

To host the web app in Firebase Hosting, do the following:
Change the api url in domain.js file under directory: /d/myfolder/1ST/1st/src, if the localDomain parameter is current pointing to localhost, change it to the alternative url. That url is the address of our APIs.

```javascript
1 // Please replace this long localhost:5000 string with local domain variable, it makes deployment much easier
2 // export const localDomain = "http://localhost:3000/";
3 // Use when deploy
4 export const localDomain = "https://us-central1-student-1-time.cloudfunctions.net/app/";
```

In your console, cd into the frontend folder, for instance, cd /d/myfolder/1ST/1st

Build the react app, if you are using MacOS or Linux, run:

```
GENERATE_SOURCEMAP=false react-scripts build
```

If you are using Windows, run:

```
npm run winBuild
OR
set "GENERATE_SOURCEMAP=false" && react-scripts build
```

Deploy the web app to Firebase Hosting, run:

```
firebase deploy
```

See Troubleshooting section if you encountered errors during deployment.

3.2 Backend Deployment
3.2.1 Local Deployment

To test your apis with fast iteration, you need to host the apis locally. Assuming you have already installed the project:

To host the web app locally, do the following:

In your console, cd into the backend folder, for instance,

cd /d/myfolder/1ST/Firebase/functions

To host APIs on localhost:8000 and 5000,

npm run serve

**Or**

firebase emulators:start --only functions

That way, you can host it locally at port 8000/5000, you can then test the APIs with postman or insomnia.

3.2.2 Remote Deployment

After you finished your upgrade to the APIs and you want to deploy them in Firebase functions to make it available to the website, you can follow the procedure below:

To host the APIs in Firebase functions, do the following:

Clean up the code, open index.js in the folder /d/myfolder/1ST/Firebase/functions, Comment out everything that is related to body-parser and port listening, this step is critical.

In your console, cd into the backend folder, for instance,

cd /d/myfolder/1ST/Firebase/functions

To deploy your APIs, run:

npm run deploy **OR** firebase deploy --only functions

It might take a while, but after the wait, your APIs should be updated and hosted at “https://us-central1-student-at-a-time.cloudfunctions.net/app/”, however you cannot directly testing the APIs by softwares like postman or insomnia for security reasons (otherwise anybody can change our data once they know this url and the route names of our APIs), but you can change the domain to the url and test it locally.

If you encounter any errors during deployment, checkout Troubleshooting Section.
3.3 Common Usage of Firebase

Follow this [LINK](#) to the Firebase console. There are several features that we are using for our website.

### 3.3.1 Authentication

In the Authentication section, you can manually add/delete an user, as well as disable an user and change the password of an user. Under the “Templates” tab you can set up the format of Password resetting emails.

### 3.3.2 Firestore

We use Firestore as our database, general ways to use it is simple and it is mentioned in the User Manual. As an Implementer however, if you ever wanted to retrieve a list of documents in an order from Firestore using some APIs, you need to create an index under the “Index” tab. Normally we need to create a “Single Field” index for a certain field of a collection. For instance, if I want to retrieve a list of housing resources from Firestore, with all documents sorted in a descending order on their creation dates, it won't do it automatically for us even if we correctly wrote our APIs, we can do so by creating a “Single Field” index. To create such an index, click on the “Add exception” button and then follow the instructions.

### 3.3.3 Storage

Firebase Storage is where we store our images, the only thing we need to pay attention to is that we currently do not have a cleaning mechanism, thus, if we don't clean it from time to time, the storage cost might increase.

### 3.3.3 Hosting

Hosting section of the Firebase console is where we host our react app. Currently we have two sites, which corresponds to two different addresses, only the “onestudentatatime” is in use. If you want to add another site for testing purposes, you can deploy it to “student-at-a-time” or you can start a new site by clicking “Add another site”.

### 3.3.3 Functions

Functions section is where we deploy our APIs. By clicking on “View Logs”, you can view all the request logs to the APIs. It is useful when you debug errors of the 1ST website.
### 4. Troubleshooting

<table>
<thead>
<tr>
<th>Error key words and Symptoms</th>
<th>Possible cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>cor, cross-region error, website unable to load data from database, and output error messages in browser console</td>
<td>missing &quot;app.use(cors({origin:true}));&quot; in index.js in functions folder, or the firebase is still trying update the functions you deployed just now</td>
<td>Make sure to add that line of code in index.js. Wait for a minute then refresh the page. Redeploy the backend APIs</td>
</tr>
<tr>
<td>Error during deployment of frontend react app. File uploading process stopped in half way.</td>
<td>firebase deployment bug</td>
<td>delete .firebase folder</td>
</tr>
<tr>
<td>Error during deployment of backend APIs.</td>
<td>probably firebase deployment bug</td>
<td>make sure to comment out anything related to port listenting or body-parser.</td>
</tr>
<tr>
<td>Other bugs</td>
<td>coding mistakes etc.</td>
<td>follow the error message, or search the error message on google for solutions</td>
</tr>
</tbody>
</table>