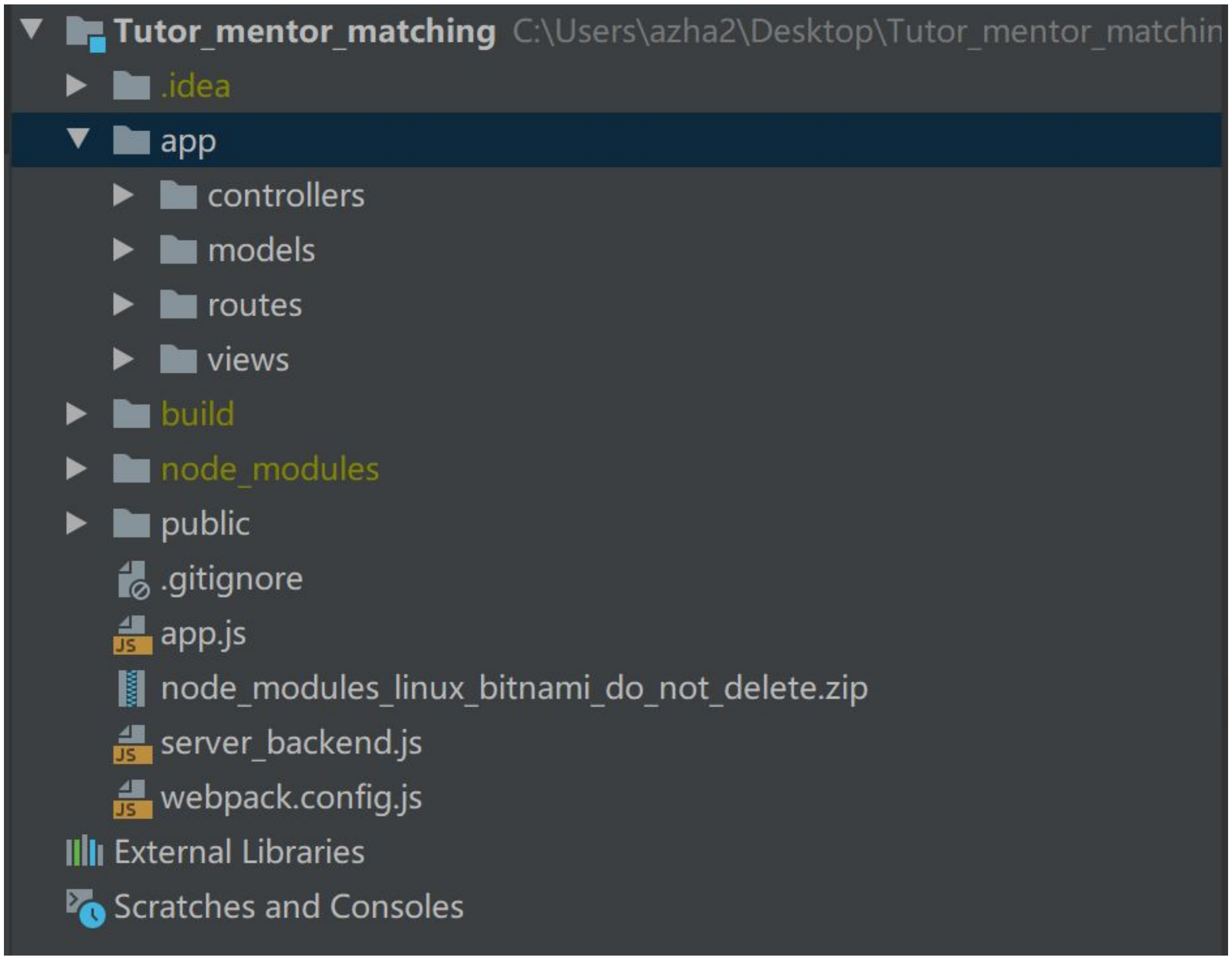
1. Web Site

The structure of the web site is shown as below:



```
▼ Tutor_mentor_matching C:\Users\azha2\Desktop\Tutor_mentor_matchin
  ▶ .idea
  ▼ app
    ▶ controllers
    ▶ models
    ▶ routes
    ▶ views
  ▶ build
  ▶ node_modules
  ▶ public
    .gitignore
    app.js
    node_modules_linux_bitnami_do_not_delete.zip
    server_backend.js
    webpack.config.js
  External Libraries
  Scratches and Consoles
```

1.1 App Entry Points

The primary entry point is the 'app.js' file (set to port 3000). Running this file starts the service.

There is a secondary entry point, the 'server_backend.js' file (set to port 4000), which is a handy console interface used in back-end developers' testing. This console recognizes the commands set in function 'Handler.backendConsole' of the file '/app/controllers/backend.server.controller.js', and displays the exact content that is ought to be sent to the front-end.

1.1.1 Running Prerequisites - Database Credentials

Both the two entry points involves database operations. Thus, we should put the connection credentials to the file '/public/passwords/DBCredentialDetails.txt'. This file is ignored by Git. A sample content (three lines) is listed below:

```
ec2-[redacted].amazonaws.com:27017/DSDB
adm
lkjoi
```

Which represent:

Database address
Account
Password

1.1.2 Running Prerequisites - Libraries

The entire required library folder ('node_modules') should be extracted from the file 'node_modules_linux_bitnami_v_3_0.zip'. The content has been tested under Windows/Mac OS/Linux environments.

The 'package.json' file is ignored by Git. Though it is not required to run the program, a sample content is listed below:

```
{
  "name": "tutor_mentor_matching",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "dependencies": {
    "babel-plugin-transform-runtime": "^6.23.0",
    "babel-preset-env": "^1.7.0",
    "babel-preset-es2015": "^6.24.1",
    "bootstrap-loader": "^3.0.4",
    "ejs": "^2.7.1",
    "express-session": "^1.16.2",
    "firebase": "^7.2.3",
    "intro.js": "^2.9.3",
    "intro.js-react": "^0.2.0",
    "jquery": "^3.4.1",
    "jsx-loader": "^0.13.2",
    "jwt-simple": "^0.5.6",
    "moment": "^2.24.0",
    "node-sass": "^4.13.0",
    "path": "^0.12.7",
    "postcss-loader": "^3.0.0",
    "precss": "^4.0.0",
    "react-datepicker": "^2.9.6",
    "react-bootstrap": "^1.0.0-beta.14",
    "sass-loader": "^8.0.0",
    "webpack-dev-server": "^3.8.1",
    "whatwg-fetch": "^3.0.0",
    "bootstrap": "^4.3.1",
    "webpack": "^4.41.2"
  },
  "devDependencies": {},
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "https://github.sydney.edu.au/yyao3172/Tutor_mentor_matching.git"
  },
  "author": "",
  "license": "ISC"
}
```

1.1.3 Running Prerequisites - webpack

If changes in the front-end are made, we need to run command ‘webpack’ in the root directory, to ‘pack’ the latest changes. Without running ‘webpack’, the front-end appearances will not be updated.

1.2 Front-end

The front-end codes reside in the ‘/public/css’, ‘/public/javascripts’ and ‘/public/scripts’ folders. They follow the React framework conventions.

1.3 Back-end

For the back-end of our system, we followed the MVC structure, and the content reside in the ‘/app’ folder.

1.4 Front-end and Back-end communication

To locate the communications, simply search the '/getData' route in the project. The '/getData' route is the only tunnel used for data communication. One example format of its parameters is listed below:

```
const formData ={
        "funID": "getTutorPopupInfoViaID",
        "paramNum" :1,
        "param1": tutorId
    };
    console.log(tutorId);
    fetch(
        '/getData',{
            method:'POST',
            credentials: 'include',
            body: JSON.stringify(formData),
            headers: {
                'Content-Type': 'application/json',
                'Accept': 'application/json',
                'Access-Token': sessionStorage.getItem('access_token') || ''
            },
        }
    )
```

According to the "funID" property, the request data will be mapped into the function with the exact same name in the backend controller ('app/controllers/backend.server.controller.js
').

1.5 Bot User Generation

To generate bot users for testing, please refer to the file '/app/models/add_data_sample.js'. This file can be run as a stand-alone js program.

1.6 Further Improvements

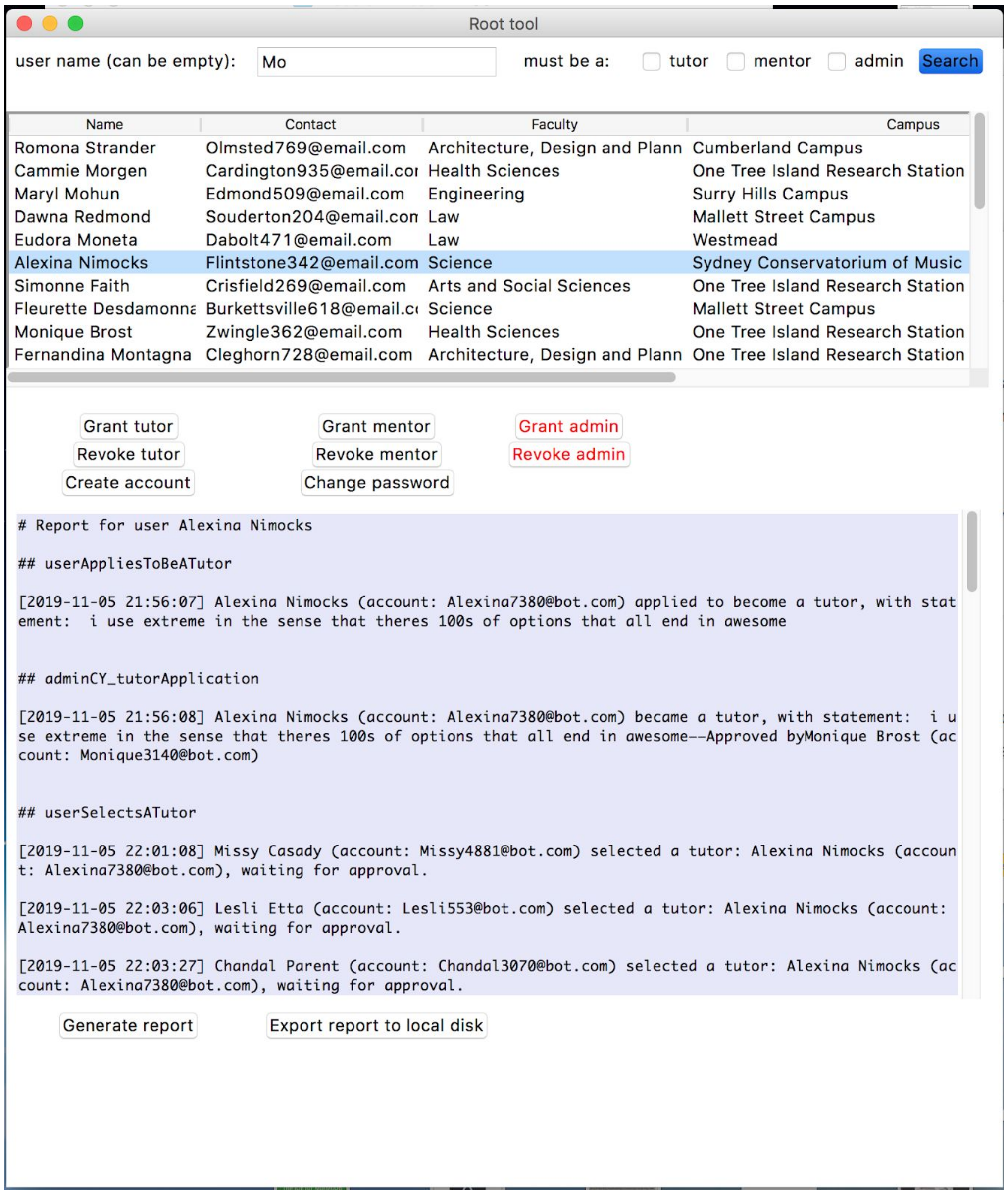
Here we list the features that may be added in the future:

1. Logging in with UniKey (AAF rapid connect).
2. User avatar editing (image upload).

3. User information card. Change the user names on the website from plain string to redirecting links, leading to the corresponding user information. Could be achieved by adding another field in the user’s session, which indicates who he/she is going to query.

2. Root Tool

Considering the security issues, we migrated the root's functionalities from the web interface to a stand-alone desktop tool. Who possess this tool would have the topmost control of the website (i.e. root).



Name	Contact	Faculty	Campus
Romona Strander	Olmsted769@email.com	Architecture, Design and Plann	Cumberland Campus
Cammie Morgen	Cardington935@email.co	Health Sciences	One Tree Island Research Station
Maryl Mohun	Edmond509@email.com	Engineering	Surry Hills Campus
Dawna Redmond	Souderton204@email.con	Law	Mallett Street Campus
Eudora Moneta	Dabolt471@email.com	Law	Westmead
Alexina Nimocks	Flintstone342@email.com	Science	Sydney Conservatorium of Music
Simonne Faith	Crisfield269@email.com	Arts and Social Sciences	One Tree Island Research Station
Fleurette Desdamonna	Burkettsville618@email.c	Science	Mallett Street Campus
Monique Brost	Zwingle362@email.com	Health Sciences	One Tree Island Research Station
Fernandina Montagna	Cleghorn728@email.com	Architecture, Design and Plann	One Tree Island Research Station

```
# Report for user Alexina Nimocks

## userAppliesToBeATutor

[2019-11-05 21:56:07] Alexina Nimocks (account: Alexina7380@bot.com) applied to become a tutor, with stat
ement:  i use extreme in the sense that theres 100s of options that all end in awesome


## adminCY_tutorApplication

[2019-11-05 21:56:08] Alexina Nimocks (account: Alexina7380@bot.com) became a tutor, with statement:  i u
se extreme in the sense that theres 100s of options that all end in awesome--Approved byMonique Brost (ac
count: Monique3140@bot.com)


## userSelectsATutor

[2019-11-05 22:01:08] Missy Casady (account: Missy4881@bot.com) selected a tutor: Alexina Nimocks (accoun
t: Alexina7380@bot.com), waiting for approval.

[2019-11-05 22:03:06] Lesli Etta (account: Lesli553@bot.com) selected a tutor: Alexina Nimocks (account:
Alexina7380@bot.com), waiting for approval.

[2019-11-05 22:03:27] Chandal Parent (account: Chandal3070@bot.com) selected a tutor: Alexina Nimocks (ac
count: Alexina7380@bot.com), waiting for approval.
```

The source code of this tool is in 'root_tool_python' folder.

The tool's functionality can be concluded as:

1. Grant/revoke privilege tags (especially the Admin tag).

2. Create accounts / Change passwords.
3. Generate reports.

The possible further improvements of this root tool are listed below:

1. System compatibility. Though this tool utilizes the tkinter GUI library, we have not verified its compatibility on Windows/Mac OS systems.
2. Remove all the tutor-selections/mentor-pairs (when it is the end of a semester).
3. Overall statistics of the user base (the tutor numbers, the mentor numbers).
4. Time filter for the report. The current implementation sorts the report entries with category, then timestamp in ascending order.
5. Export the report to pdf format. The current txt file is written using MarkDown convention, which could have a better appearance when converted to pdf.