AFFORDABLE

Developers Guide

Avishek Banerjee, Rupam Kundu,
Sangram Grewal, Steve Kwon
The Ohio State University

Sponsor

Warren Campbell

Sean Pannella

Contents

Chapter 1

Setup

The first step is to download and install the tools that are required to develop the software. The step by step details are given in this section.

1.1 Environment Setup

1.1.1 MAC OS

1.1.1.1 React.js and Node.js

- Get latest Node.js from `https://nodejs.org/en/download/`
- npm i -g typescript
- npm i -g yarn
- npm i -g serve
- npm i -g create-react-app
- Follow:
  `https://levelup.gitconnected.com/typescript-and-react-using-create-react-app-a-step-by-step-guide-to-setting-up-your-first-app-6deda70843a4`
- If the dummy application mentioned in the previous link runs, react setup is complete.

1.1.1.2 Mysql

- Install MySQL following `https://dev.mysql.com/doc/refman/5.6/en/osx-installation-pkg.html`

- Open a terminal
- Type: mysql -u root -p
- Then you will be prompted to enter a password (generally the password of your machine). Alter the root@localhost's password to "password" using:

  mysql>ALTER USER 'root'@'localhost' IDENTIFIED BY 'password'

  Note: Refer to Section 2.3, below the table in case the above command is used with a different password.
- Create a database named Affordable_database:
  mysql>CREATE DATABASE Affordable_database;
- mysql>USE Affordable_database;
- mysql>CREATE TABLE customers1 (Username VARCHAR(255), Password VARCHAR(255), Email VARCHAR(255), Question1 VARCHAR(255), Answer1 VARCHAR(255), Question2 VARCHAR(255), Answer2 VARCHAR(255), Question3 VARCHAR(255), Answer3 VARCHAR(255), TwoFactor VARCHAR(255), TwoFactorCode VARCHAR(255));
- Check whether the table is created, type: mysql> show tables;

1.1.2 Linux

To install and use ReactJS, we need two things: Node.js and NPM. Node.js is a Javascript run-time environment that allow us to execute Javascript code like working on a server (or a local server, if were running it in our computers). In the other hand NPM is a package manager for Javascript, that is, NPM allows to install Javascript libraries to make our experience even more richer by expanding the basic functionalities. Following are the steps that needs to be followed for the Linux systems[1] https://codeburst.io/installing-reactjs-and-creating-your-first-application-d437706498ed

1.1.2.1 React.js and Node.js

Open a terminal and write the following commands.

- sudo apt-get install curl
- curl -sL `https:deb.nodesource.com/setup_6.x` — sudo -E bash -

- sudo apt-get install -y nodejs
- Check if Node.js and NVM are installed
  node -v
  npm -v
- sudo apt-get install -y build-essential
- sudo npm install -g create-react-app
- create-react-app hello-world
  This creates and sets up the trail application
- cd/hello-world
- npm start
  This command will start a local server and will open our application in the browser.
- More packages are needed to be installed once the whole thing is setup

1.1.2.2 Mysql

Open terminal and type in the following

- sudo apt-get update
- sudo apt-get install mysql-server
- Type: mysql -u root -p
- Then you will be prompted to enter a password (generally the password of your machine). Alter the root@localhost's password to a "password" using:

  mysql>ALTER USER 'root'@'localhost' IDENTIFIED BY 'password'

  Note: Refer to Section 2.3 (below the table) for Password Update Options in the code if the above command is executed with a different password.
- Create a database named Affordable_database:
  mysql>CREATE DATABASE Affordable_database;
- mysql>USE Affordable_database;

- mysql>CREATE TABLE customers1 (Username VARCHAR(255), Password VARCHAR(255), Email VARCHAR(255), Question1 VARCHAR(255), Answer1 VARCHAR(255), Question2 VARCHAR(255), Answer2 VARCHAR(255), Question3 VARCHAR(255), Answer3 VARCHAR(255), TwoFactor VARCHAR(255), TwoFactorCode VARCHAR(255));
- Check whether the table is created, type: mysql > show tables;

1.2 Git Clone

- Go to: `https://www.x3xoj4kpf0ql1dwzact9uu8v01ic81pd.net:5443/sean/AffordableServerOSU`
- Create an account and Sean will add you to the project once you create an account.
- Open a terminal and type in the commands mentioned below.
- git clone https://www.x3xoj4kpf0ql1dwzact9uu8v01ic81pd.net:5443 /sean/AffordableServerOSU.git
- cd AffordableServerOSU
- There are two folders here: *affordable* and *server*.
- Inside *server*, run "npm ls" and make sure the following packages are installed as shown:

  body-parser@1.18.2 cookie-parser@1.4.3 cors@2.8.4 debug@2.6.9 express@4.16.3 express-fileupload@0.4.0 fast-sha256@1.1.0 image-data-uri@2.0.0 jade@1.11.0 morgan@1.9.0 mysql@2.16.0 node-mandrill@1.0.1 nodemailer@4.6.8 nodemailer-smtp-pool@2.8.3 openssl-wrapper@0.3.4 qrcode@1.3.0 serve-favicon@2.5.0 speakeasy@2.0.0
- Inside *affordable*, run "npm ls" and make sure the following packages are installed as shown:

  express@4.16.4 fast-sha256@1.1.0 nodemailer@4.6.8 nodemailer-smtp-pool@2.8.3 qrcode.react@0.8.0 react@16.5.0 react-dom@16.5.0 react-router-dom@4.3.1 react-scripts@1.1.5 react-select@2.0.0 requests@0.2.2

Note: In case, any of the packages are missing, manually install it using "npm install $< package - name >$". For example, "npm install nodemailer".

Chapter 2

Code Database

2.1 Folder Architecture

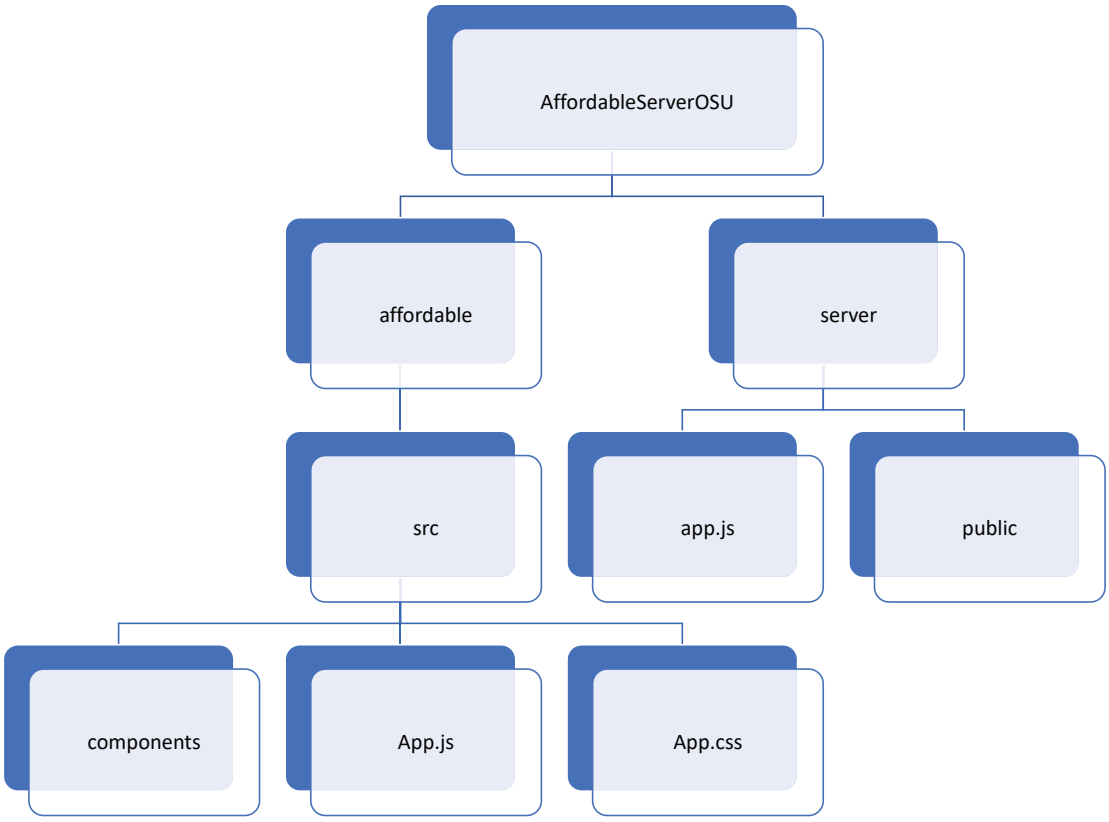

Figure 2.1: Folder Architecture

2.2 Front-End codes : REACT

Table 2.1: Codes inside Components folder inside affordable, which are used in REACT platform

Component Name	**Description**
app.css	CSS styles that are used for various component design in the Front-end.

app.js	Main Component for Front-End, listing out all the submodules.
	
	class App extends Component {
	render() {
	return (...)
	}
	}

Navigation.js	Creating Navigation links for all pages.
header.js	Basic header in all pages with the AFFORDABLE logo.
Main.js	First page containing login and register buttons.

Login.js	Login page with username/password/two-factor-authentication and also forget username-password option.
	
	class Login extends React.Component {
	handle_validate_two_factor()
	handle_validate_login_credentials()
	render() {
	return(...)
	}

Table 2.2: Codes inside Components folder inside affordable, which are used in REACT platform

Component Name	**Description**

openaccnt2factor.js	Registration Page
	
	class openaccnt2factor extends React.Component {
	handle_is_username_in_database()
	handle_confirm_password()
	handle_check_password_format()
	showAlert_on_validate_email()
	two_factor_checkbox_toggle()
	handle_two_factor_code_validation()
	handleUploadRegistrationForm()
	render() {
	return(...)
	}

Email_Verify.js	Email Validation Page sent out as an embedded link via email to validate new user's email

retrieve.js	Retrieving account via username/email, followed by two factor/security question verification.
	
	class retrieve extends React.Component {
	togglefunc_Google_Auth_checkbox()
	togglefunc_Raw_Code_checkbox()
	handle_security_qs_answers()
	handle_validate_code()
	showAlert_on_retreive_status()
	render() {
	return(...)
	}

reset_password.js	On validation from retrieve.js, password resetting options are given by this code.
	
	class reset_pass extends React.Component {
	handle_confirm_password()
	updatepasswd()
	handle_check_password_format()
	render() {
	return(...)
	}

dashboard.js/app_apply.js	Upon successful login credential validation, these codes constitute the main application page.

2.3 Back-End codes : NodeJS

Table 2.3: Codes inside app.js under server folder, which are used in NodeJS

POST Methods	**Description**
username_check_already_in_database	Checks whether the username typed by the user already in database
recovery_email	Sends recovery Email
is_email_in_database	Checks whether user's email is already in database
generate_QR_Image	Generates QR code and image for users requesting two factor authentication
twofactorlogin_email	Validates two factor code from Retrieve Password page based on email
twofactorlogin_username	Validates two factor code from login page as well as Retrieve Password page based on username
validate_two_factor_code	Validates two factor code from registration page
validate_login	Validates user credentials for login
update_password	Updates user password
check_username	Check whether user's username exists in database for Password/Username recovery
check_email	Check whether user's email exists in database for Password/Username recovery
upload_form_to_database	Upload inputs from registration for new user to database

Note:

1. In reference to Section 1.1, if mySQL is setup with a password other than "password", make changes in app.js and update the line - const password = "new_password" in Line #56.

2. All the "console.log()" are commented in app.js. Uncomment it to debug any specific module. It will get printed in the terminal running the server (Shown in Section 3.3).

3. The QR Image is generated in the backend and saved inside a folder called "public" under "server".

Chapter 3

Running the Application

3.1 MySQL

- Open a Terminal and follow the steps as shown in the Figure below and do not close this terminal.

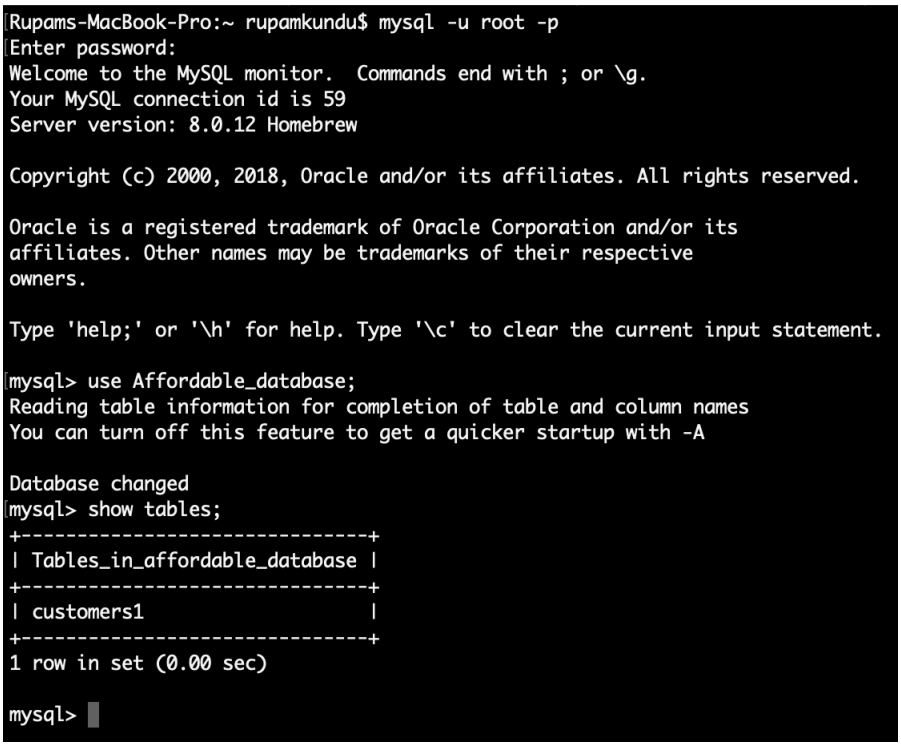

Figure 3.1: Running MySQL database

3.2 Front-end : REACT

- In a different terminal, navigate to the folder "affordable", and type "yarn start" which yields the following output, followed by starting the Application at localhost.

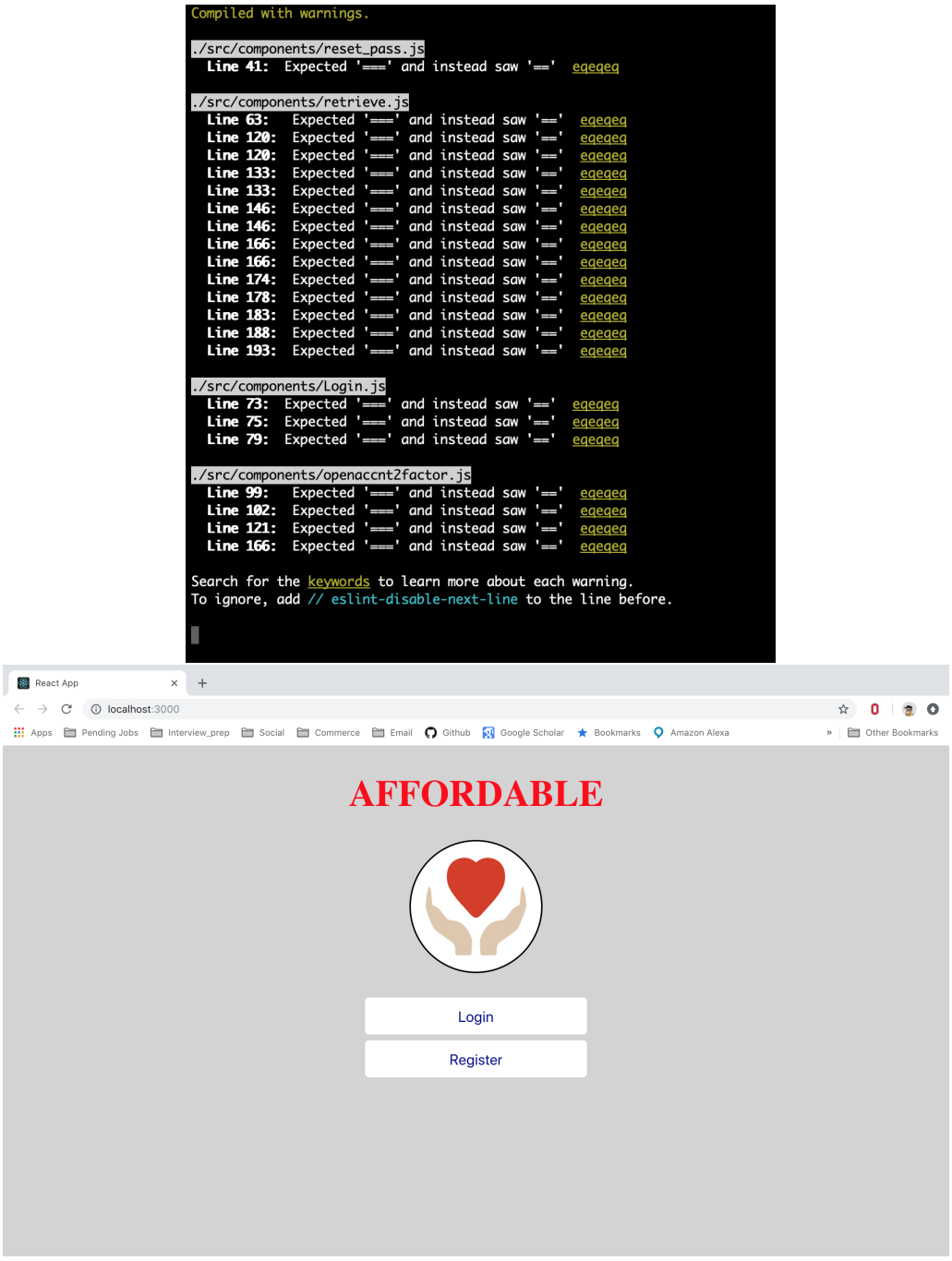

Figure 3.2: Running REACT for the front-end

3.3 Back-end : NodeJS

- In another terminal, navigate to the folder "server", and type "npm start" as show below.

```
[Rupams-MacBook-Pro:server rupamkundu$ npm start

> server@0.0.0 start /Users/rupamkundu/Capstone_Affordable/AffordableServerOSU_BACKUP/server
> node ./bin/www
```

Figure 3.3: Running NodeJS

- Subsequently, all frontend interactions with backend will be reflected in the terminal running the server as shown below.

```
Rupams-MacBook-Pro:server rupamkundu$ npm start

> server@0.0.0 start /Users/rupamkundu/Capstone_Affordable/AffordableServerOSU_BACKUP/server
> node ./bin/www

./public/filename1.png
POST /generate_QR_Image 200 72.202 ms - 40
GET /public/filename1.png 200 5.921 ms - 1684
./public/filename2.png
POST /generate_QR_Image 200 24.722 ms - 40
Database is connected ...


POST /username_check_already_in_database 200 80.925 ms - 43
Database is connected ...


POST /username_check_already_in_database 200 5.470 ms - 53
Database is connected ...


POST /is_email_in_database 200 6.202 ms - 50
Database is connected ...


Records inserted
POST /upload_form_to_database - - ms - -
rupamk
Database is connected ...


POST /check_username 200 4.645 ms - 628
Database is connected ...


POST /recovery_email 200 10.301 ms - 40
```

Figure 3.4: Sample communication between front-end and back-end

Chapter 4

Conclusion

Immediate Scope for future improvements.

1. The email contents for “Validate Email” and “Register Now” for authenticated users are not organized properly.
2. “Register Later” button has no functionality in the current implementation.
3. The current implementation allows user to use previously used password while resetting password which is typically not allowed in the current login systems.
4. For authenticated users, on clicking “Register Now” gets an email with the raw two factor code. During resetting password if the user opts for providing that Raw Auth code, the two factor should be disabled immediately. Also, there should be options to re-enable the Two Factor Authentication.
5. Security questions list should be expanded.
6. Enable Two-factor Authentication twice for each user.
7. Also, enable an option during login for two factor authenticated user to accept the raw authentication code as done in retrieve password page.