

Profile Builder

Submitted By -

1. Ms. Ashwini Gore
2. Ms. Nandita Dixit
3. Mr. Viplava Bharadwaj
4. Mr. Yugeshwar Devtare

Guided By –

1. Mr. Ashutosh Mishra
2. Mr. Sachin Pande

Index

1. Introduction	1
a. Overview	
b. Objectives	
c. Roles & Responsibility	
2. Implementation Methodology	2
a. Process Flow	
b. Technologies	
c. Database Architecture	
d. Flow between NodeJs and ReactJs	
e. NodeJs Architecture	
f. ReactJs Architecture	
3. Timeline	7
4. Output	8
5. Installation	13
6. Conclusion	14
7. References	15

1. Introduction

a. Overview

This is a project document for the '**Profile Builder**' a web application. Profile builder is a web application for **Skillship Foundation** for assigning tasks to different roles. In this portal there are roles such as Super Admin, Region Head, City Head, and Campus CEO who can assign tasks to each other and keep an eye on completion status. Also there is a leaderboard where the users are ranked according to their tasks completion. This document describes the objectives and goal of the system. This document models the functional requirements with process flow, interaction diagrams, designs, implementation, output and installation.

b. Objectives

- Four different users –
 - Superadmin
 - Region Heads
 - City Heads
 - Campus CEOs
- Superadmin can track every Region Heads, City Heads and Campus CEOs for tasks and their respective users from the dashboard.
- Each users have their own dashboard to manage tasks and their respective users.
- Superadmin can assign tasks and create different users.
- Region Heads can assign tasks and create City Heads.
- City Heads can assign tasks and create Campus CEO.
- Maintained Leader Boards according to tasks completion.
- Maintained profile page for each users.

c. Roles & Responsibility

- Frontend
 - 1. Ashwini Gore
 - 2. Yugeshwar Devtare
- Backend
 - 1. Nandita Dixit
 - 2. Viplava Bhardwaj

2. Implementation Methodology

a. Technologies

Frontend

- ReactJs
- react-router-dom
- axios
- react-validation
- Bootstrap
- validator

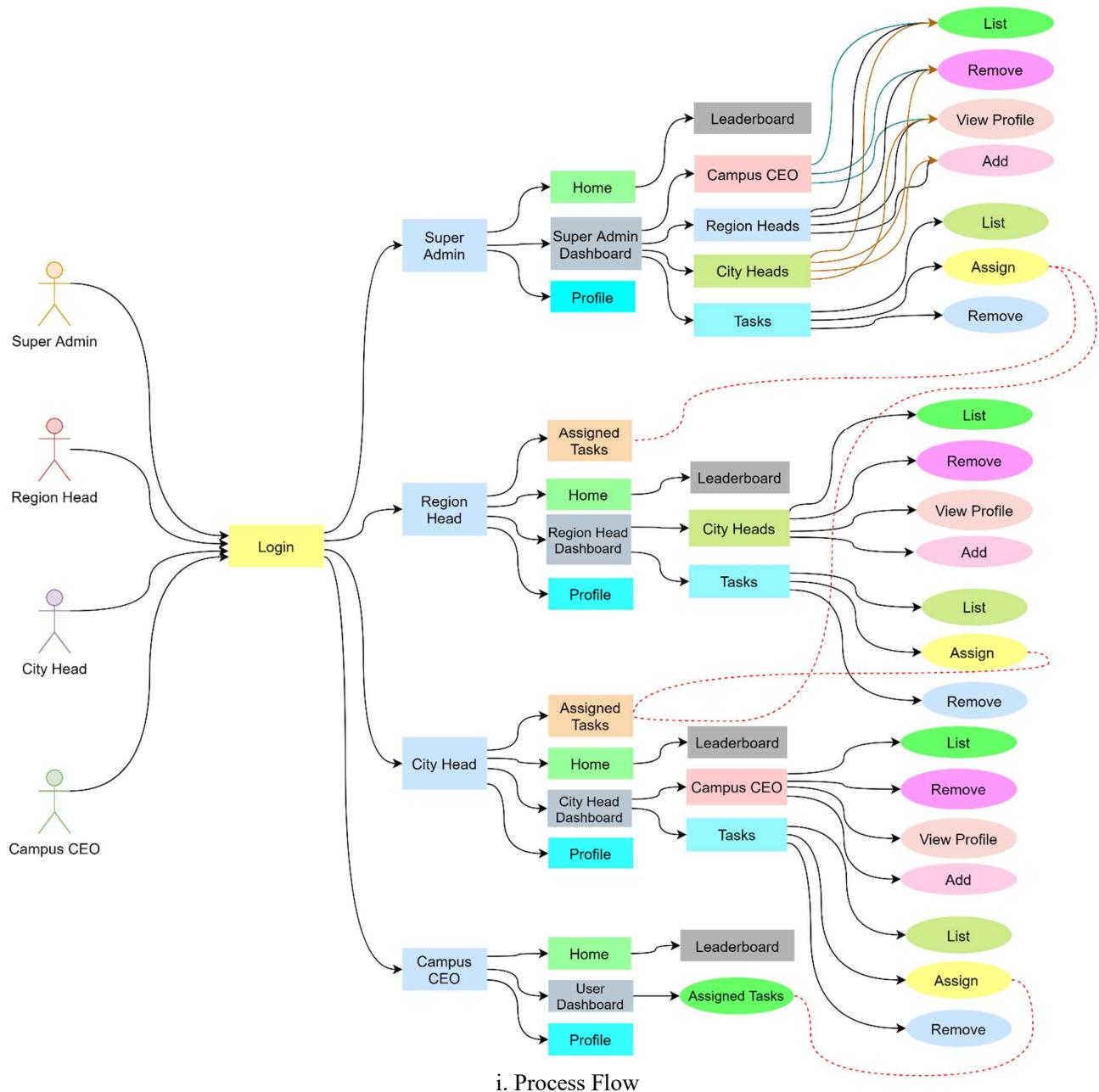
Backend

- NodeJs
- Express
- bcryptjs
- jsonwebtoken
- Sequelize

Database

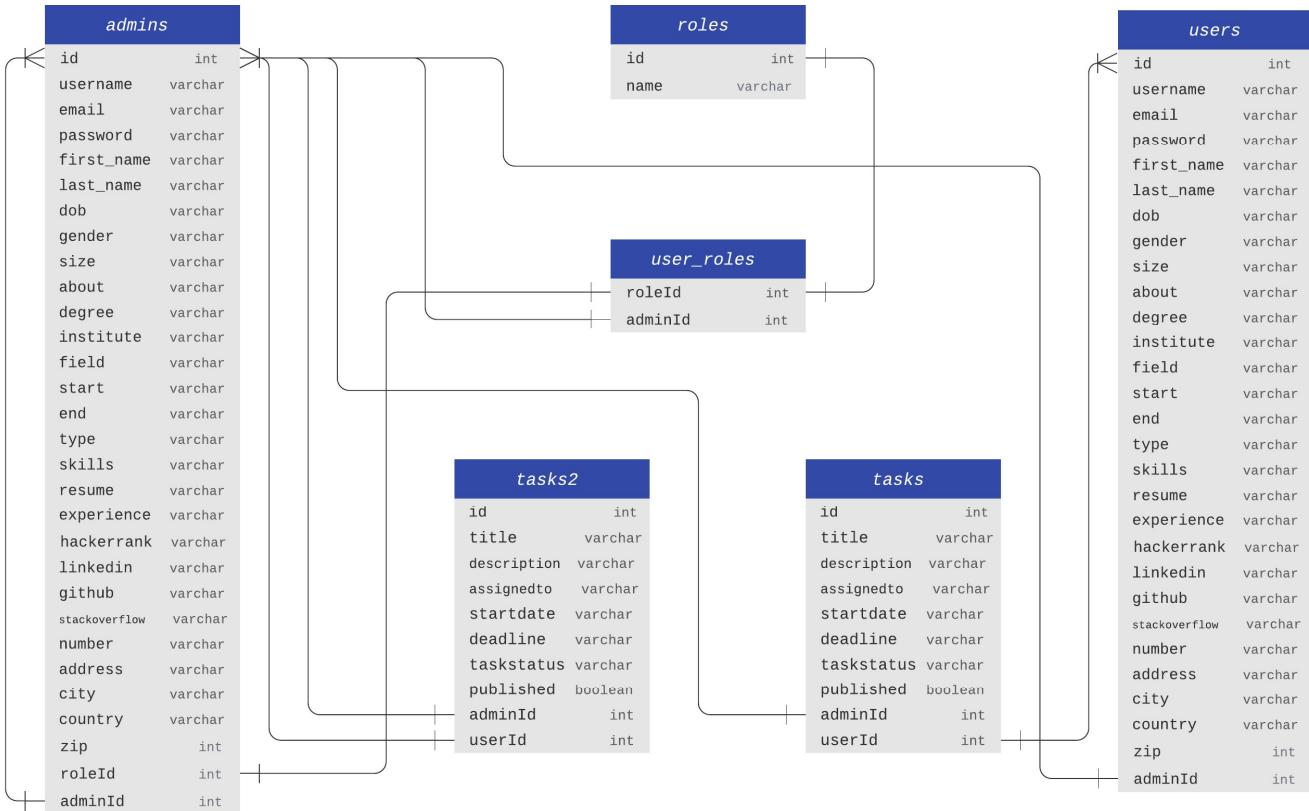
- MySQL

b. Process Flow



c. Database Architecture

2

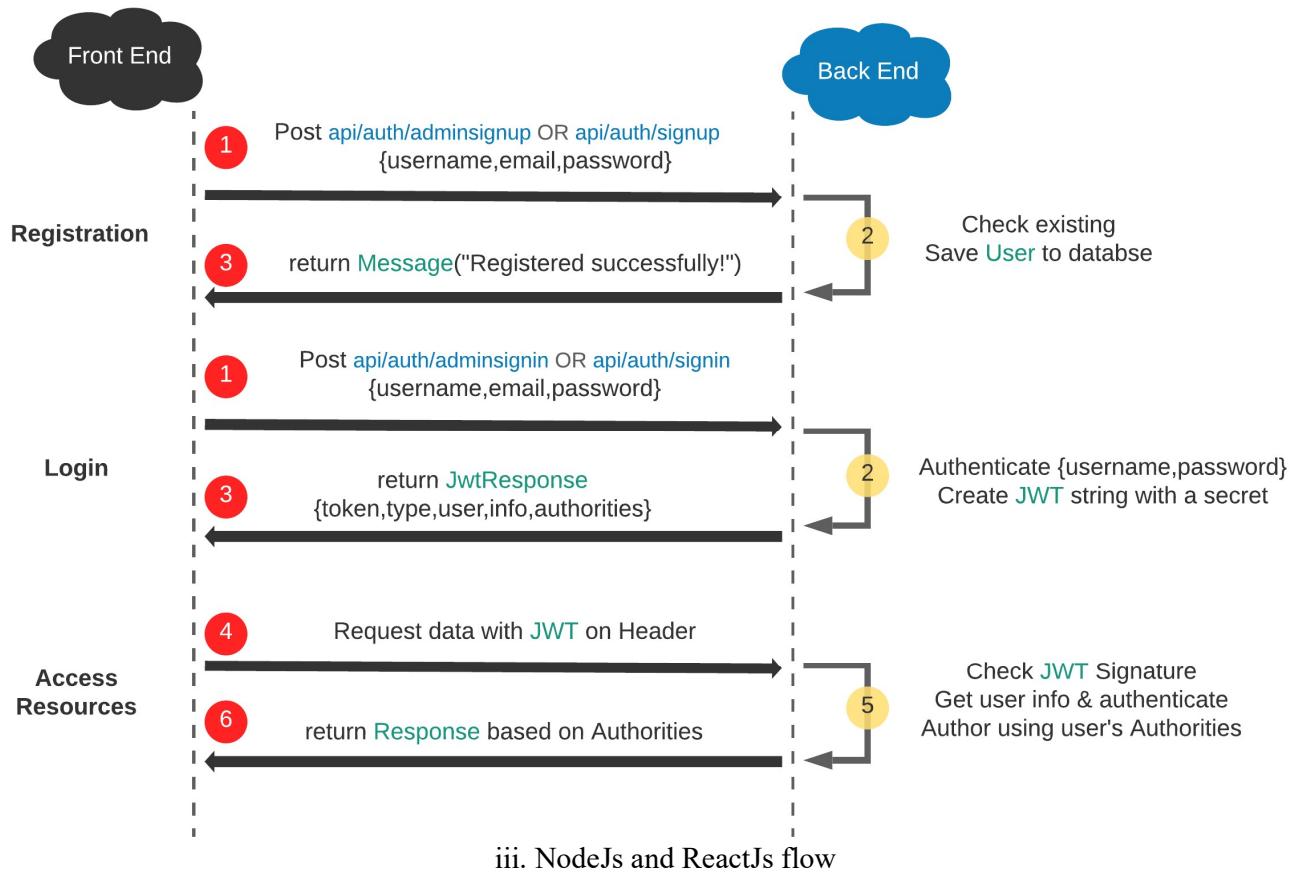


ii. Database Architecture

Created six databases in MySQL –

1. **admins** – Used for storing three admins
 - Superadmin
 - Region Heads
 - City Heads
2. **users** – Used for storing Campus CEOs.
3. **roles** – Used for storing roles with respective ids.
 - superadmin
 - admin (Region Heads)
 - admin2 (City Heads)
4. **user_roles** – Used for storing the roleId and adminId.
5. **tasks** – Used for storing tasks given to Campus CEOs.
6. **tasks2** – Used for storing tasks given to Region Heads & City Heads.

d. Flow between NodeJs and ReactJs



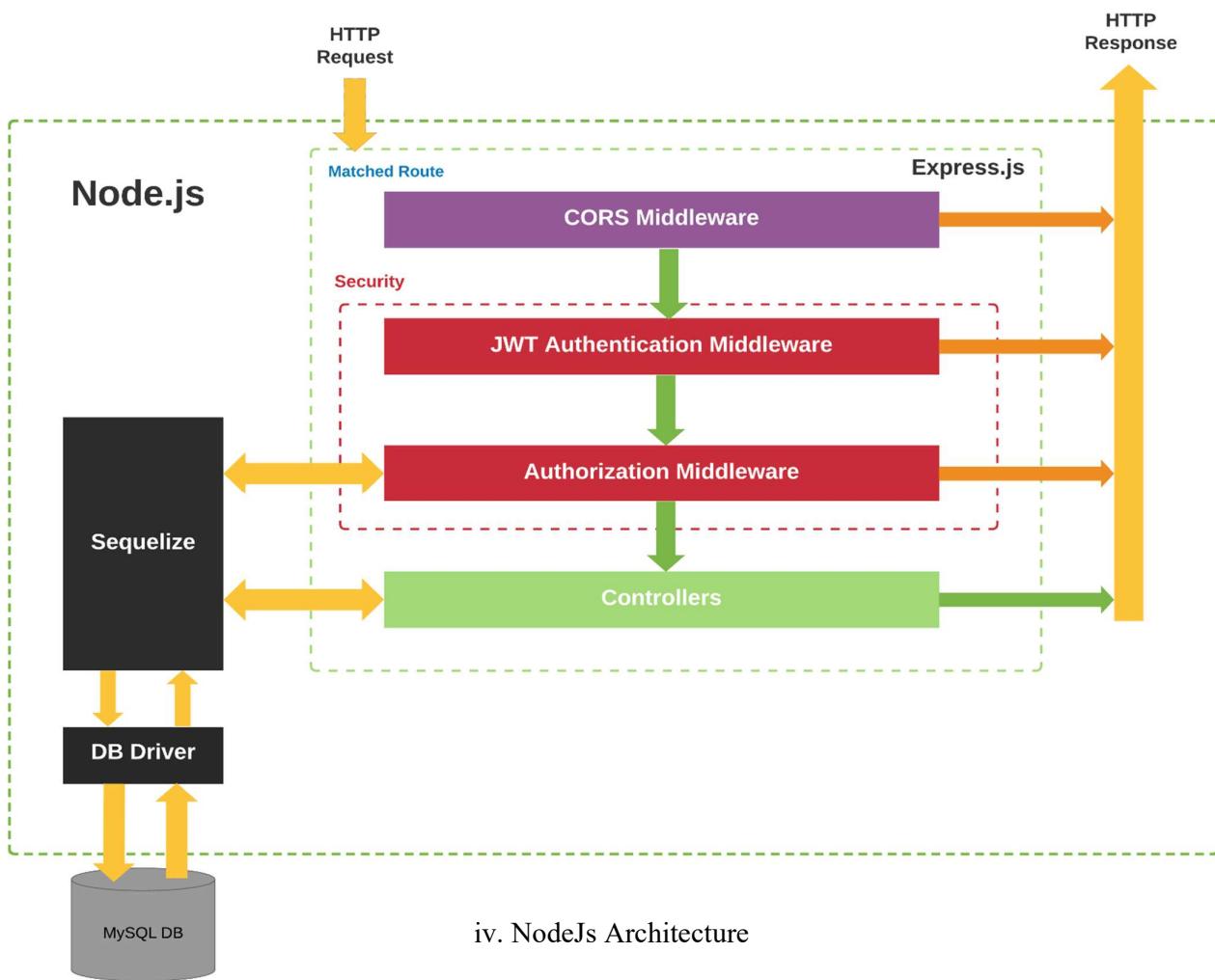
The diagram shows the flow of Registration, Login and Authorization process.

There are 2-2 endpoints for authentication admins & users:

1. Admins
 - `api/auth/adminsignup` for Registration
 - `api/auth/adminsignin` for Login
2. Users
 - `api/auth/signup` for Registration
 - `api/auth/signin` for Login

When request is send from frontend to protected data/endpoints, it will add legal JWT to HTTP `x-access-token` Header.

e. NodeJs Architecture



With the help of Express routes, **HTTP request** that matches a route will be checked by **CORS Middleware** before coming to **Security layer**.

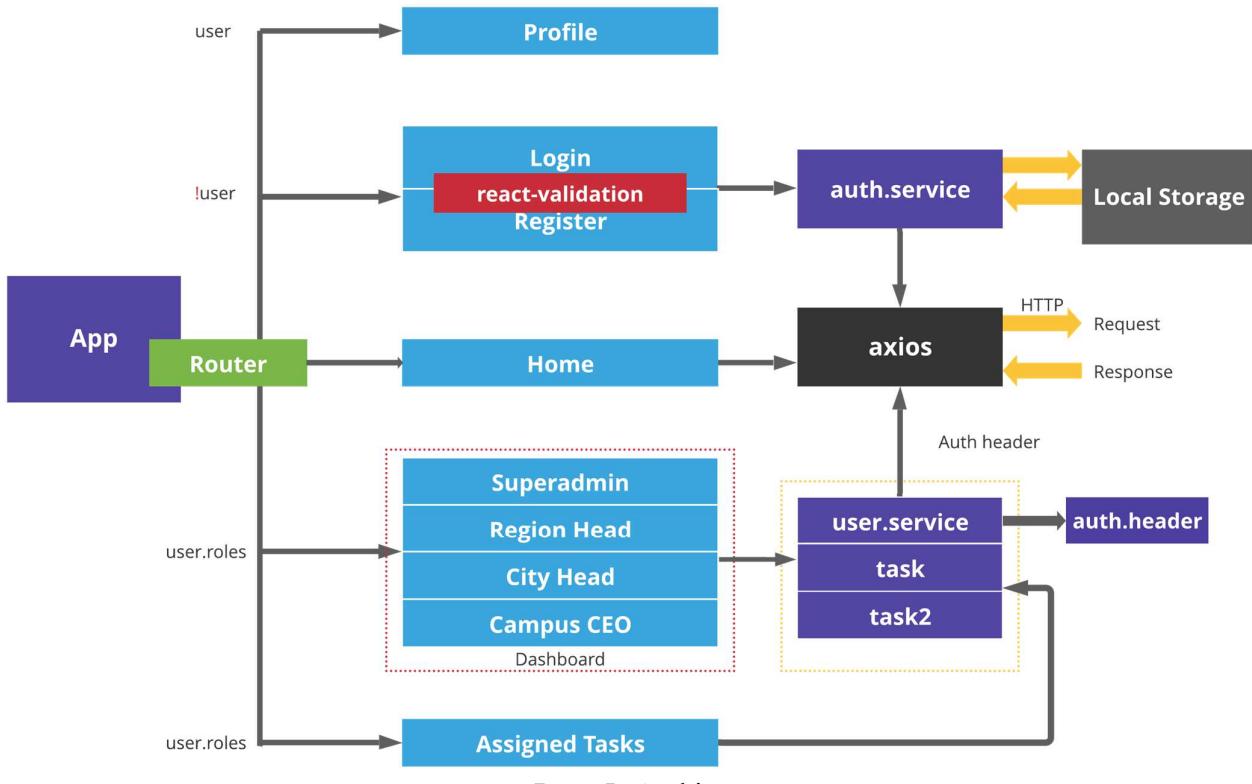
Security layer includes:

- JWT Authentication Middleware: verify SignUp, verify token
- Authorization Middleware: check Admin's roles & Users with record in database

If these **middlewares** throw any error, a message will be sent as **HTTP response**.

Controllers interact with MySQL Database via **Sequelize** and send **HTTP response** (token, user information, tasks, data based on roles...) to client.

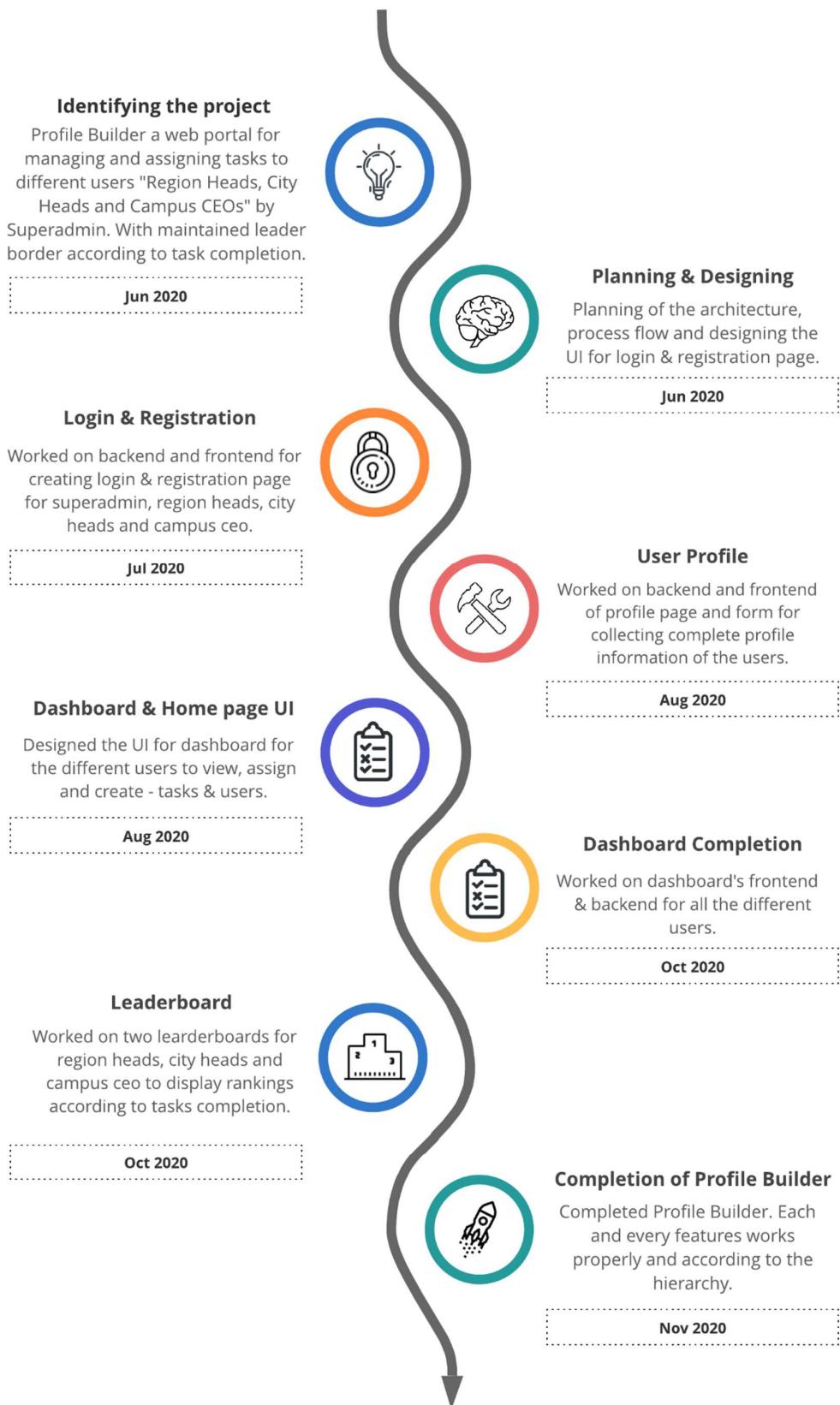
f. ReactJs Architecture



v. ReactJs Architecture

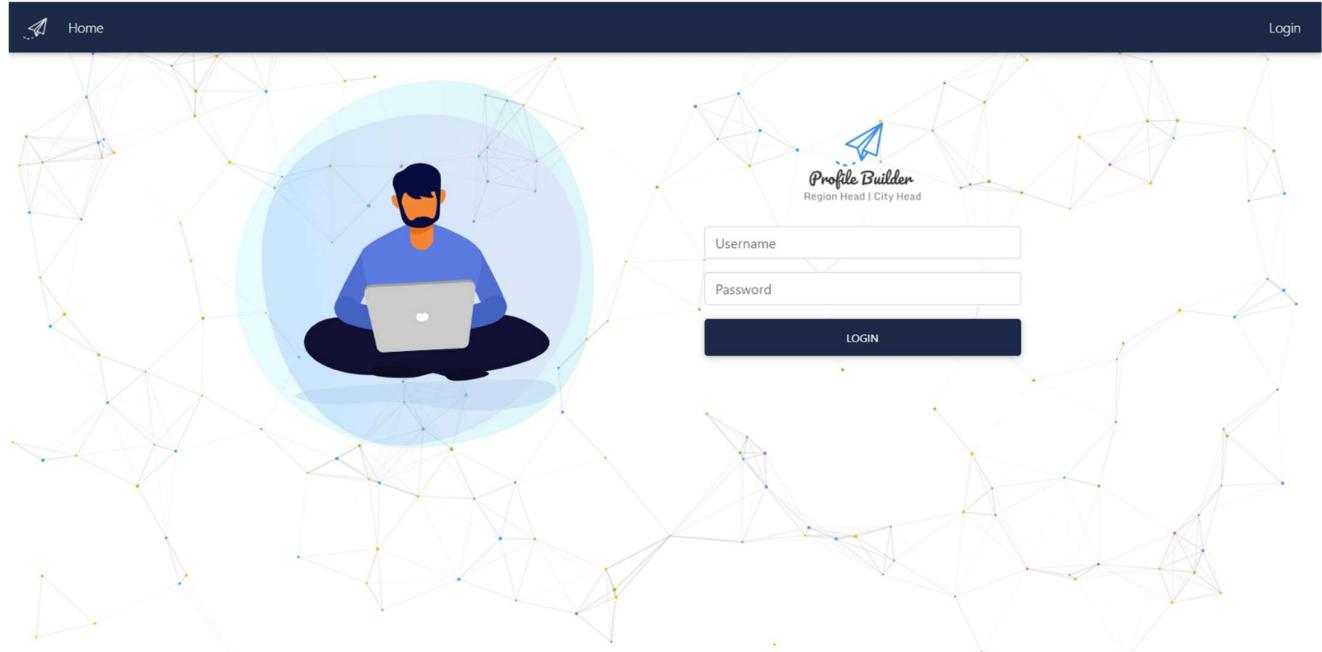
- The **App** component is a container with React Router (**BrowserRouter**). Based on the state, the navbar can display its items.
- **Login & Register** components have form for data submission (with support of **react-validation** library). They call methods from **auth.service** to make login/register request.
- **auth.service** methods use **axios** to make HTTP requests. It also store or get **JWT** from Browser Local Storage inside these methods.
- **Home** component is public for all visitor.
- **Profile** component displays user information after the login action is successful.
- **Superadmin, Region Head, City Head & Campus CEO** components will be displayed by state **user.roles**. In these components, we use **user.service**, **task** & **task2** to access protected resources from Web API.
- **user.service**, **task** & **task2** uses **auth-header()** helper function to add JWT to HTTP header. **auth-header()** returns an object containing the JWT of the currently logged in user from Local Storage.
- **Assigned Tasks** component will be displayed by state **user.roles**. In these components, we use **user.service**, **task** & **task2** to access protected resources from Web API to show assigned tasks.

3. Timeline

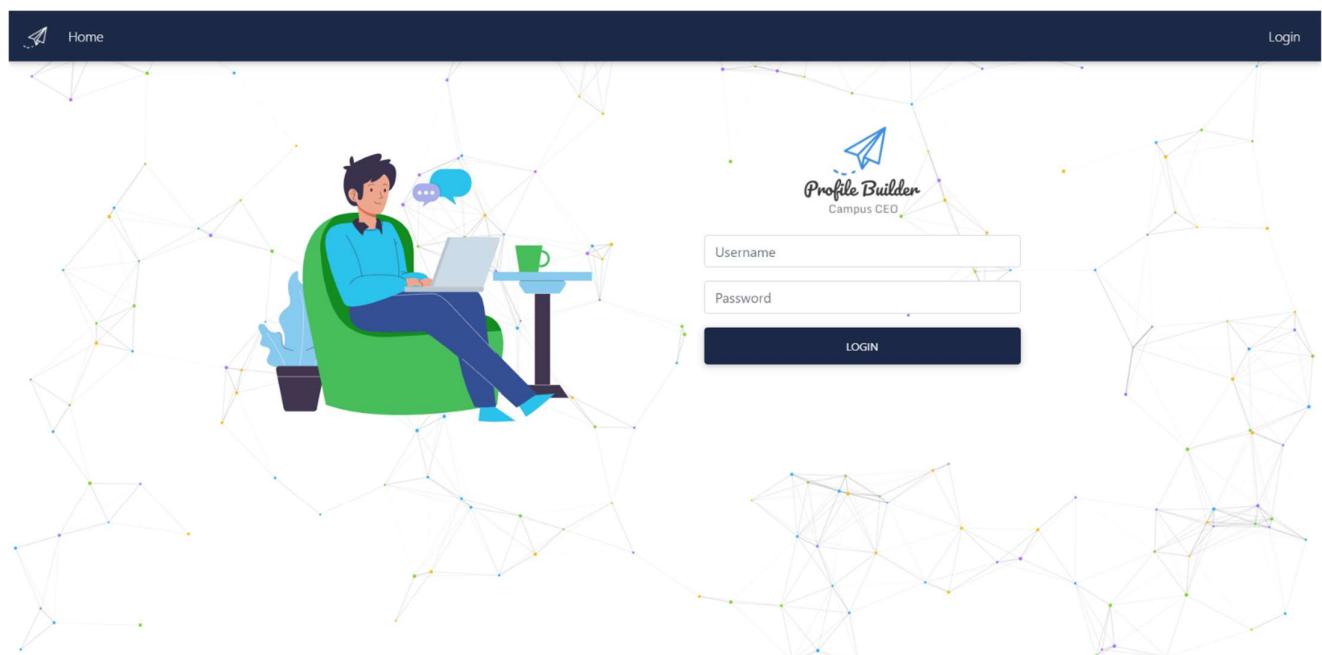


4. Output

Below are some screenshots of Profile Builder.



vii. Admin Login



vii. Campus CEO Login

The Home page features a top navigation bar with a Home icon and a Login link. Below the navigation is a 3D-style illustration of two people working at a desk with various charts and graphs on their screens. To the right of the illustration, the title "Leader Boards" is displayed. Under "Leader Boards", there are two sections: "Region/City Head Board" and "Campus CEO Board", each containing a table with user data.

Username	Position	Task Done	Score	Rank
admin	Region Head	3	15	1
admin2	City Head	2	10	2
mod	City Head	1	5	3

Username	Task Done	Score	Rank
yugesh	2	10	1
admin2w	1	5	2

vii. Home Page

The Profile page is centered around a user profile card for "Yugesh Devtare". The card includes the user's name, title ("Full-Stack Developer"), email ("yugesh227@gmail.com"), and a "UPDATE PROFILE" button. Below the card is a navigation menu with tabs: Basic (selected), Educational, About, Social, and Contact. The "Basic Information" section displays the following details:

Username	superadmin
Name	Yugesh Devtare
Date of Birth	1999-10-14
Gender	Male
T-shirt Size	M

vii. Profile Page

Home Assigned Tasks

yugesh227@gmail.com LogOut

1 Basic 2 Education 3 Describe 4 Social 5 Contact

First Name Last Name

Date of Birth Gender T-Shirt Size

About Me

NEXT

viii. Profile Form

Home Dashboard

yugesh227@gmail.com LogOut

Tasks List

Region Heads
City Heads
Campus CEO

Search Task SEARCH

	ASSIGN TO REGION HEAD	ASSIGN TO CITY HEAD	6
Login for Admin & SuperAdmin			
Login for Admin & SuperAdminfgh			
3			
Login for Admin & SuperAdmin			
Login for Admin & SuperAdmin			
3			

< 1 >

localhost:8081/superadmin#

ix. Superadmin Dashboard

The screenshot shows a web application interface titled "React App" running on "localhost:8081/admin2". The top navigation bar includes links for "Home", "Dashboard", and "Assigned Tasks", along with user information "1231@123.com" and a "LogOut" button. On the left, a sidebar titled "Task List" has a "Campus CEO" section selected. The main content area displays a list titled "Campus CEO" with five entries: "yugesh", "admin2w", "admin2w2", "admin22w2", and "ad2min22w2". Below the list is a "ADD CAMPUS CEO" button. To the right of the list is a decorative illustration of four people working together at a desk with laptops and charts.

x. Region Head Dashboard

The screenshot shows a web application interface titled "React App" running on "localhost:8081/admin". The top navigation bar includes links for "Home", "Dashboard", and "Assigned Tasks", along with user information "yuges2h227@gmail.com" and a "LogOut" button. On the left, a sidebar titled "Tasks List" has a "City Heads" section selected. The main content area displays a list titled "City Heads" with two entries: "admin2" (highlighted in blue) and "mod". Above the list is a "Search User" input field and a "SEARCH" button. To the right of the list is a detailed user profile form with fields for "USERNAME" (admin2), "FIRST NAME", "LAST NAME", "EMAIL" (1231@123.com), "GENDER", and "DOB". Below the form are buttons for "CAMPUS CEOS", "ASSIGN TASK", "VIEW PROFILE", and a delete icon. The bottom of the screen shows a Windows taskbar with various pinned icons.

xi. City Head Dashboard

The screenshot shows a task management interface. At the top, there is a navigation bar with icons for Home and Dashboard, and user information (yugesh227@gmail.com) and LogOut. The main area features a large, abstract network graph composed of colored dots (blue, green, yellow) connected by lines. A central modal window is open, containing fields for Title, Description, Assigned To, Start Date, Deadline, Task Status, and buttons for UPDATE, DELETE, and GO BACK.

Title: Login for Admin & SuperAdmin

Description:

Assigned To: 3

Start Date: dd-mm-yyyy

Deadline: dd-mm-yyyy

Task Status: Status

Buttons: UPDATE, DELETE, GO BACK

xii. Assign Tasks

The screenshot shows a user profile creation interface. At the top, there is a navigation bar with icons for Home and Dashboard, and user information (yugesh227@gmail.com) and LogOut. The main area features a large, abstract network graph. A central modal window is open, featuring a profile icon and the name 'Profile Builder' (Region Head | City Head). It contains fields for Role, Username, Email, and Password, along with ADD and GO BACK buttons.

Role:

Username:

Email:

Password:

Buttons: ADD, GO BACK

xii. Create Users

5. Installation

Below is the installation process for profile builder.

1. Clone the repository and download to your system.
2. In the project directory –
 - o For client (Frontend)
 - run **npm install** or **yarn install** to install client, then
 - run **npm start** or **yarn start** to start client.
 - Open <http://localhost:8081> to view it in the browser.
 - o For server (Backend)
 - run **npm install** or **yarn install** to install client, then
 - create a database in MySQL with the name **testdb**, then
 - run **node server.js** to start client.
3. After successfully running the client & server, open the browser and navigate to <http://localhost:8081/addsuperadmin> for creating a **superadmin**. Fill up the form with a role, username, email, and a password.
4. After successfully creating a **superadmin**, you will be redirected to admin login page.
5. Log in with the username and password.
6. Now you can access all features available for **superadmin**.
7. Superadmin can now create new region heads, city heads and assign them tasks.

Some important *urls*= <http://localhost:8081> are

- *url+/addsuperadmin* for creating a superadmin.
- *url+/adminlogin* for **superadmin, region heads, city heads** login.
- *url+/login* for campus ceo's login.
- *url+/home* for home page.
- *url+/profile* for profile page.

6. Conclusion

Now we have an overview of Profile Builder along with flow of the project. The purpose of this project was to develop a web application to assign and track tasks. Manage different set of users which are going to use the web application. The project Profile Builder has been developed as per the requirement specification. It has been developed using ReactJs, NodeJs and MySQL. The complete system is thoroughly tested, every design procedure and diagrams are present in this project report. Designs and diagrams are easy to understand that any new modules can be incorporated easily.

7. References

- i. ReactJs - <https://reactjs.org/>
- ii. react-router-dom - <https://www.npmjs.com/package/react-router-dom>
- iii. axios - <https://www.npmjs.com/package/axios>
- iv. react-validation - <https://www.npmjs.com/package/react-validation>
- v. bootstrap - <https://getbootstrap.com/>
- vi. validator - <https://validator.w3.org/>
- vii. nodejs - <https://nodejs.org/en/>
- viii. sequelize - <https://sequelize.org/>
- ix. mysql - <https://www.mysql.com/>
- x. expressjs - <https://expressjs.com/>