**1. Introduction**

* **Project Title:** Fundtastic
* **Team Members :**

R. LOVA RAJU(frontend)

K. LAXMAN (backend)

P. ANJALI (API)

D. PRAHARSHA (documentation)

**2. Project Overview**

* **Purpose:**

These an Fundraiser apps streamline online fundraising, empowering individuals and organizations to easily create and manage campaigns. They provide tools for secure donation processing, event organization, and performance tracking, all while prioritizing user privacy. Mobile accessibility ensures seamless engagement with donors on the go, and features like social sharing amplify campaign reach. Ultimately, these platforms foster a supportive environment for impactful causes to garner vital support.

* **Features:**
* **Secure Donation Processing:** Safe and reliable methods for handling online donations.
* **Fundraising Resources:** Access to guides and tips for successful fundraising.
* **Campaign Creation and Customization:** Tools to build and personalize fundraising campaigns.
* **Social Sharing Integration:** Features to expand reach through social media.

**3. Architecture**

* **Frontend:**

The frontend is represented by the “Frontend” section, including user interface components such as authentication, create New Fundriser, My Fundriser.

1. **Login/register**

* Create a Component Which contains a form for taking the email and password.
* If the given input matches the data of user or admin then navigate it to their respective home pages.

1. **Home**

* We provided these pages so you can check all of the fundraiser cards and see the available and collected amounts.
* The page allows you to donate money, see recent donations, and see the applicant's name, email, phone number, fundraiser cause, as well as donate money.
* In the page available to donate money, in donars and see the applicant’s name, email, phone number, fundriser cause also and donate money and see the recent donars deatails.

1. **New Fundriser(user):**

* In frontend, we implemented creating a fundriser card to allow users to provide information like fundriser cause, applicant name, phone number, etc.
* On the home page, you can see the fundriser card for available and collected amounts.
* In frontend,we implemented the all the user are create a new fundriser card to give their basic details like fundriser cause, applicant name,phone number etc..

1. **My fundrisers**

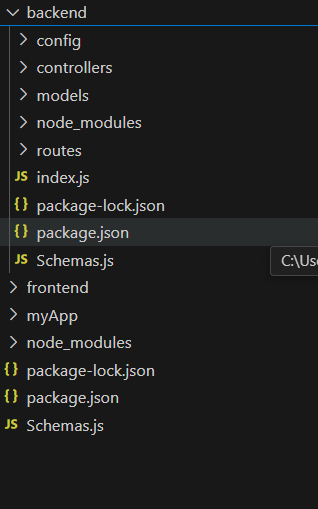
* In these page we provide to see the users their fundriser card,to collected amount.

1. **Admin**

* On the admin dashboard, the admin can view all user details, fundraiser cards, total donations, total funds raised, and all fundrisers.



**Backend:** Outline the backend architecture using Node.js and Express.js.



1. **Database Configuration**

* Set up a MongoDB database either locally or using a clou-based MongoDB service like MongoDB Atlas or use locally with MongoDB compass.
* Create a database and define the necessary collection for fundrisers,users,New fundriser details and other recent data.

1. **Create Express.js Server:**

* Set up an Express.js server to handle HTTP requests and server API endpoints.
* Configure middleware such as body-parser for parsing request bodies and cors for handling cross-origin requests.

1. **Define API Routes:**

* Create separate route files for different API functionalities such as New fundrisers,update fundrisers, users and authentication.
* Define the necessary routes for listing fundrisers,handling user registration and login managing etc.
* Implement route handles using Express.js to handle requests and interact with the database.

1. **Implement Data Models:**

* CRUD operations (create, read, update, delete) for each model to perform database operations.

1. **User Authentication:**

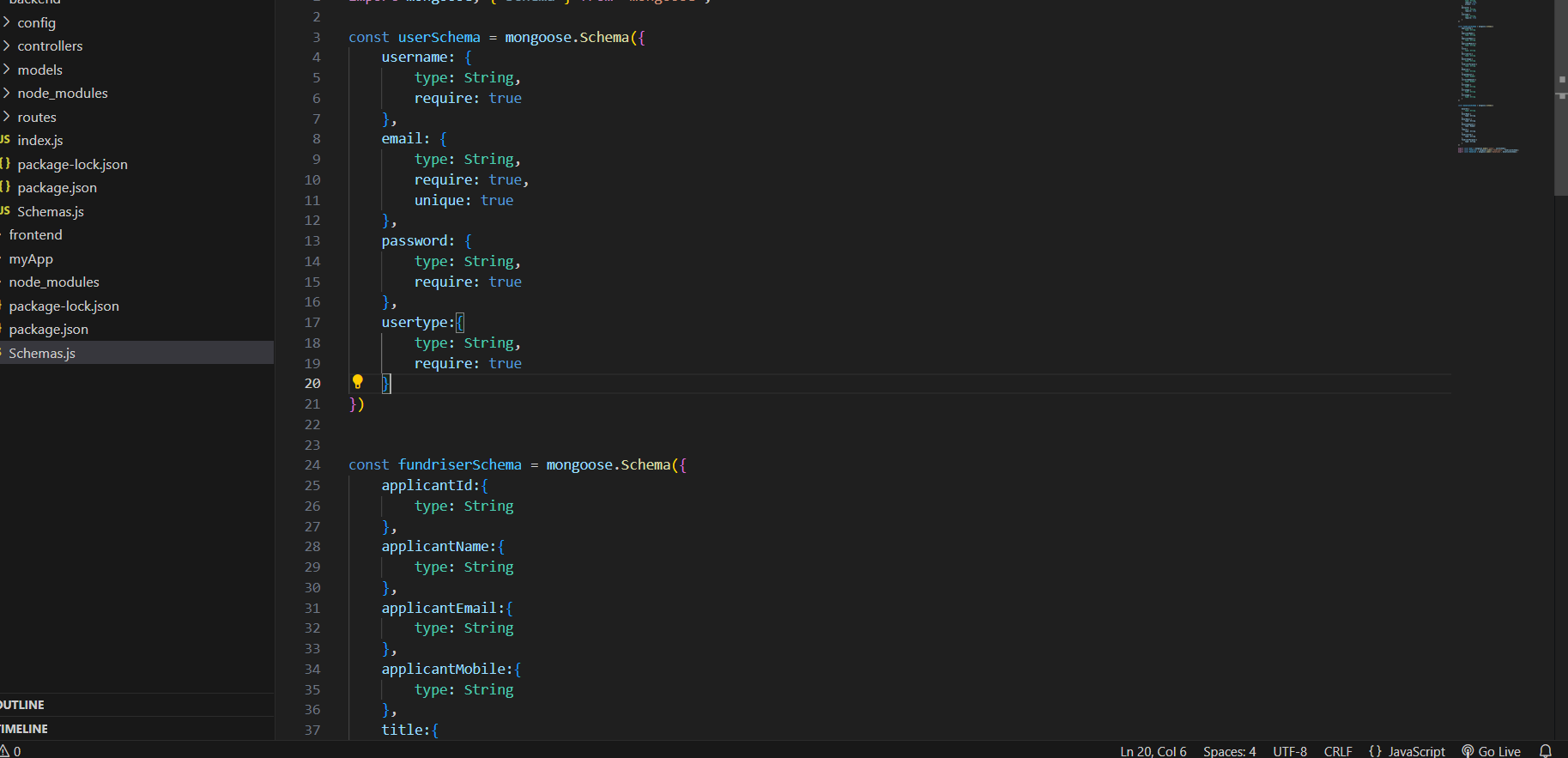
* Create routes and middleware for user registration, login and logout.
* Set up authentication middleware to protect routes that require user authentication.

**Database:** Details the database schema and interactions with MongoDB.

The Database section represents the database that stores collections for Users, Fundriser and My fundrisers.

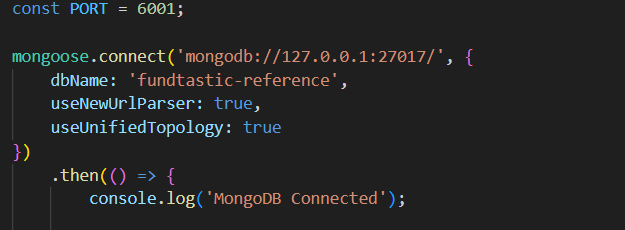
* **Configure schema**

Firstly, configure the Schemas for MongoDB database, to store the data in such a pattern. use the data from the ER diagram to create the schemas.the Schemas for this application look a like to the one provided below.



* **Connect database to backend**

Now, make sure the database is connected before performing any of the actions through the backend. The connections code looks similar to the one provided below.

****

**4. Setup Instructions**

* **Prerequisites:** List software dependencies (e.g.Node.js, MongoDB).
* To develop a full-stack fundriser app using React JS, Node js, and MongoDB, there are several prerequisites you should consider. Here are the key prerequisites for developing such an application:

**Node js and npm**: Install Node js, which includes ppm (Node Package Manager), on your development machine. Node, js is required to run JavaScript on the server side.

* Download: <https://nodejs.org/en/download/>
* Installation instructions: https://nodejs.org/en/download/package-managerl/

MongoDB: Set up a MongoDB database to store hotel and booking information. Install MongoDB locally using a cloud-based MongoDB service.

* Download: <https://www.mongodb.com/try/download/community>
* Installation instructions: htttps://docs.mongodb.com/manual/installation/

**Express.js:** Express.js is a web application framework for Node js. Install Express.js to handle server-side routing. middleware, and API

* Installation: Open your command prompt or terminal and run the following command: **npm install express**

**React.js**: React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications. To install React js, a JavaScript library for building user interfaces, follow the installation guide: <https://reactis.org/docs/create-a-new-react-app.html/>

**HTML, CSS, and JavaScript:** Basic knowledge of HTML, for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

**Database Connectivity:** Use a MongoDB driver or an Object-Document Mapping (ODM)library like Mongoose to connect your Node js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations.

**Front-end Framework:** Utilize Angular to build the user-facing part of the application, including product listings, booking forms, and user interfaces for the admin dashboard.

**Version Control:** Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

* Git: Download and installation instructions can be found at <https://gitscm.com/downloads/>

**Development Environment:** Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

* Visual Studio Code: Download from <https://code.visualstudio.com/downlgad/>
* Sublime Text: Download from <https://www.sublimetext.com/download/>
* WebStorm: Download from <https://www.jetbrains.com/webstorm/download/>

**To Connect the Database with Node JS go through the below provided link:**

* <https://www.section.io/engineering-education/nodejs-mongoosejs-mongodb/>

**Install Dependencies:**

Navigate into the cloned repository directory:

* Install the required dependencies by running the following command:

**npm install**

**Start the Development Server:**

* To start the development server, execute the following command:

**npm run dev or npm run start**

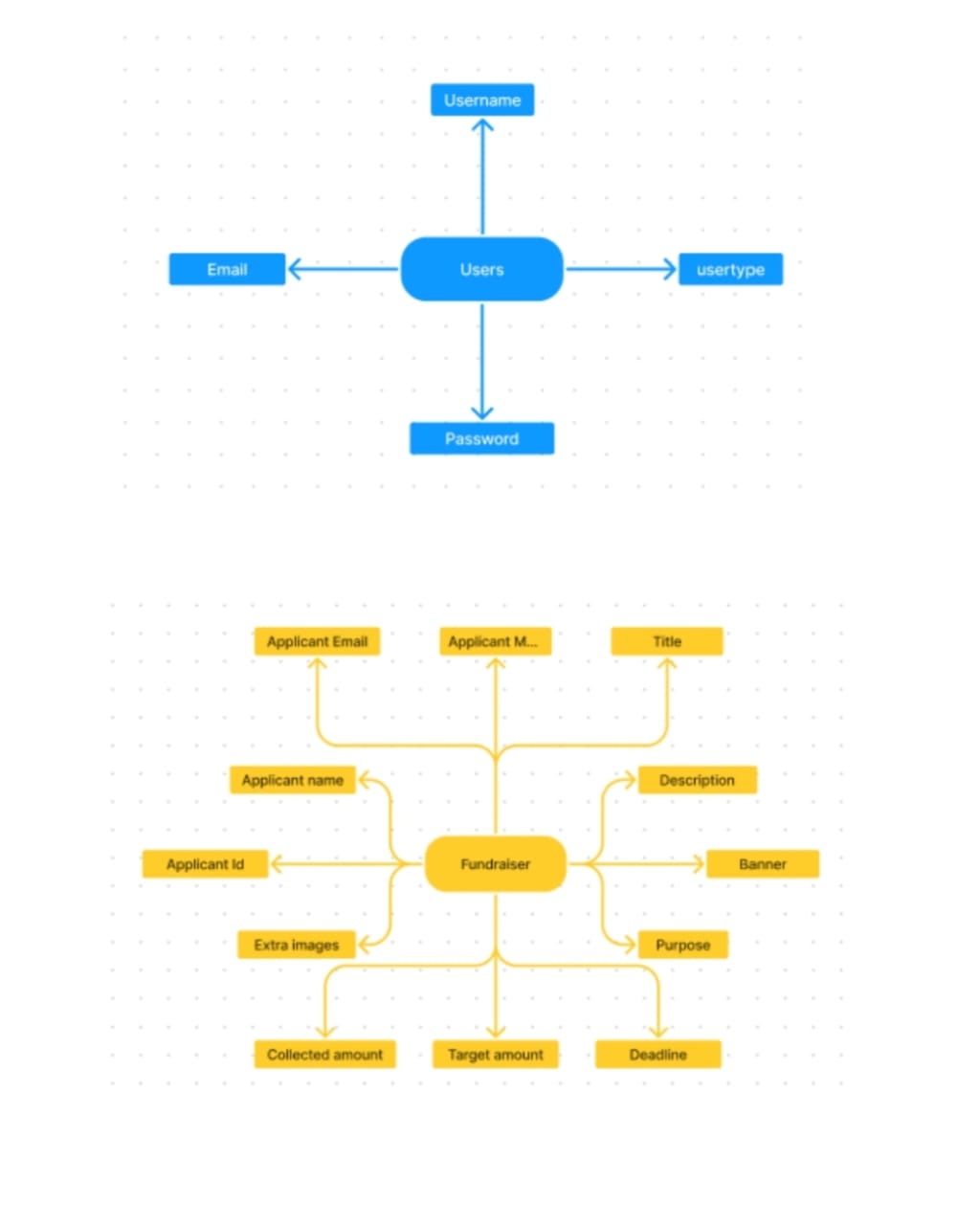
* The e-commerce app will be accessible at http://localhost:3000/ by default. You can change the port configuration in the any file if needed.

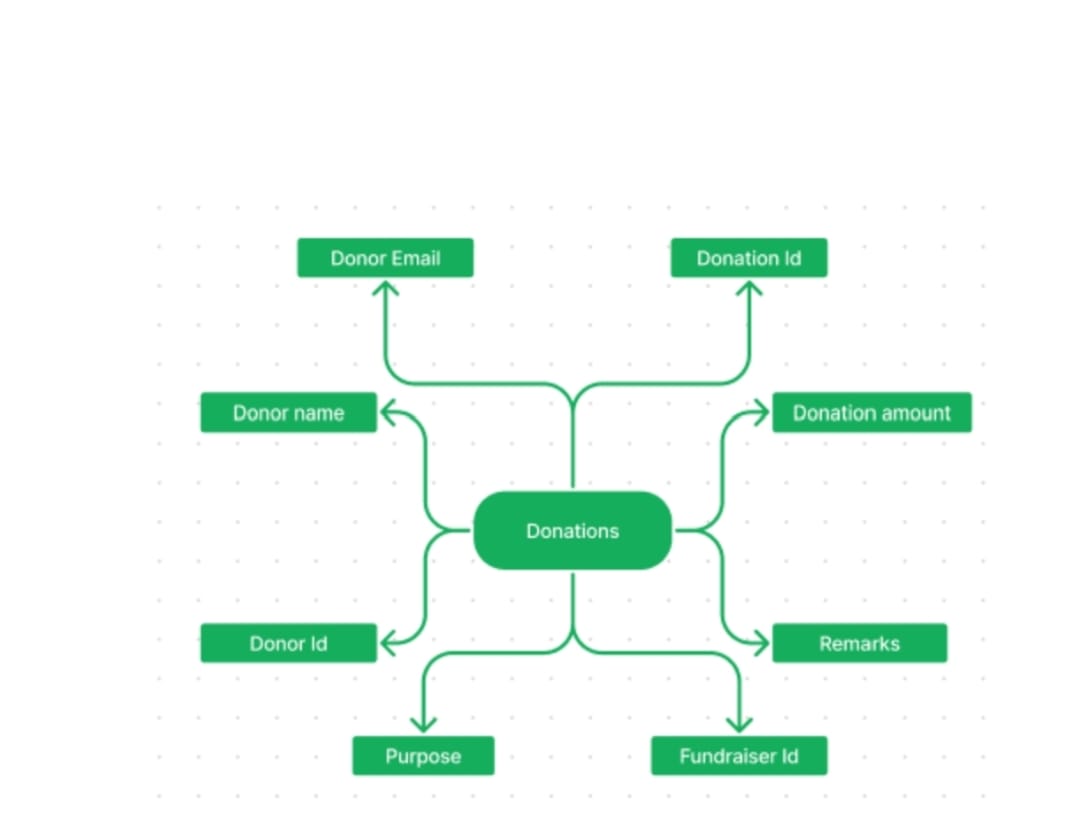
**Access the App:**

* Open your web browser and navigate to http://localhost:3000/
* You should see the fundrising app's homepage, indicating that the installation and the setup was successful.

You have successfully installed and set up the fundriser app on your local machine, you can now proceed with further customization, development, and testing as needed.

**ER DIAGRAM:**

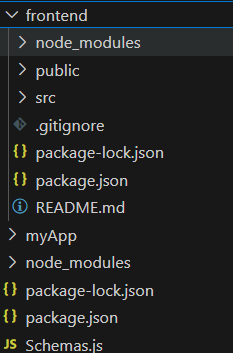




**5. Folder Structure**

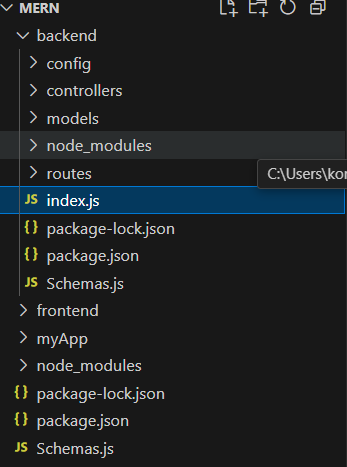
* **Client:**

The below directory structure represents the directories and files in the client folder(front end) where, react.js is used along with API’s.



* **Server:**

The below directory structure represents the directories and files in the server folder (back end) where, node.js, and MongoDB are used along with API.



**6. Running the Application**

* Provide commands to start the frontend and backend servers locally.
  + **Frontend:** npm start in the frontend directory.
  + **Backend:** npm start in the backend directory.

1. **API Documentation**

* POST/api/register – User registration
* POST/api/login - user login
* POST/api/new-fundriser – new fundriser
* POST/api/update-fundriser – update fundriser
* GET/api/fetch fundriser – fetch fundriser
* GET/api/fetch donations – fetch donations
* GET/api/make donations – make donations
* GET/api/fetch users – fetch users

1. **Authentication**

Authentication is handled using JWT tokens.

Tokens are stored in HTTP-only cookies.

**Example of token verification middleware in Node.js:**

const jwt = require(jsonwebtoken);

const auth = (reg, res, next) => !const token - reg.cookies.token;

if (!token) return res.status(401). json(f message: Unauthorized' ));

jwt.verify(token, process.env.JWT\_SECRET, (err, user)=>1if(err) return res.status(403). json(( message:Forbidden'));req.user = user;

next();

});

**8.User Interface**

1. Home page – Here see the all users to fundriser cards and donates.

2. My fundrises – These page a user see to their fundriser card.

3. New fundriser – These page users create a new fundriser card to gives their details.

4. Admin Dashboard - In admin dashboard view all the users, total donations,total fundrised,and all fundrisers.

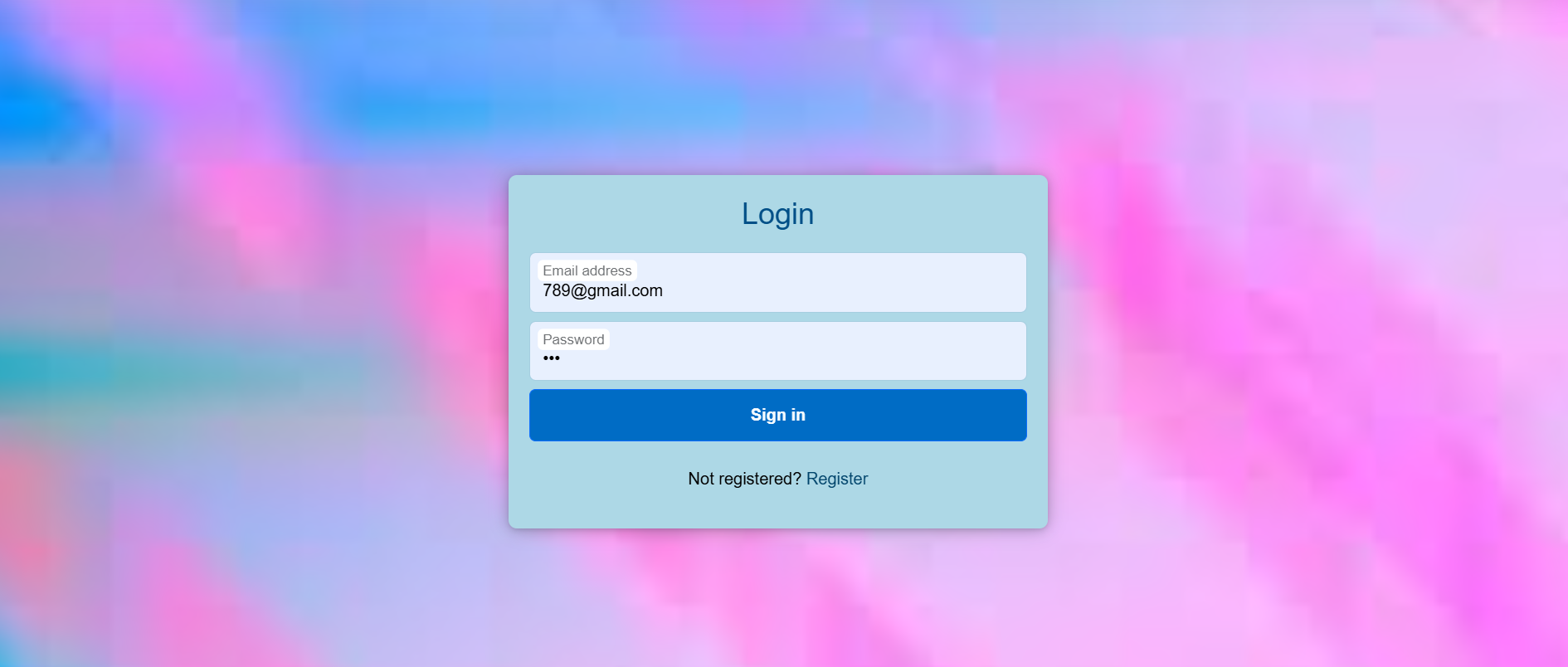
1. **Testing**

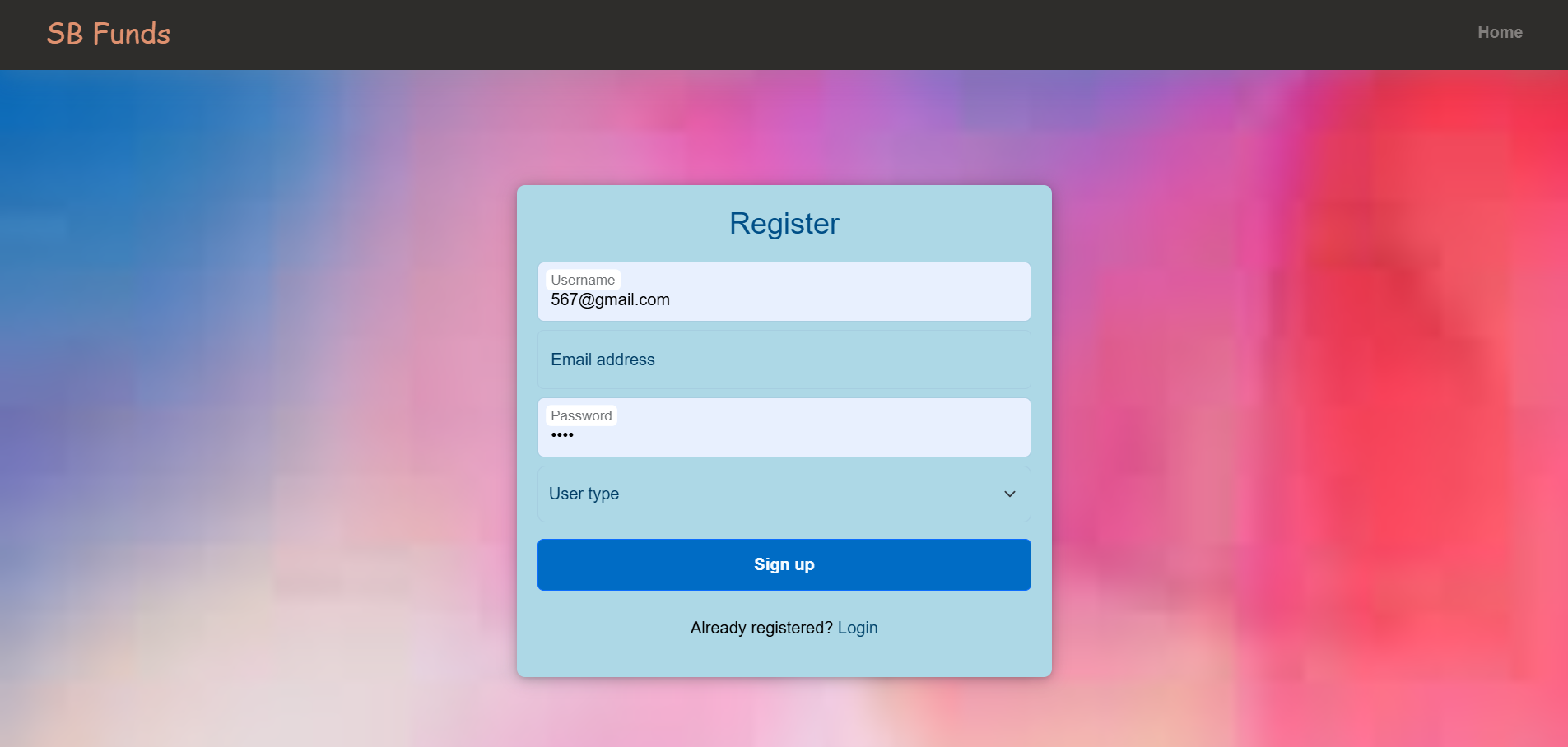
Backend **:** using post man and Thunder client for API testing.

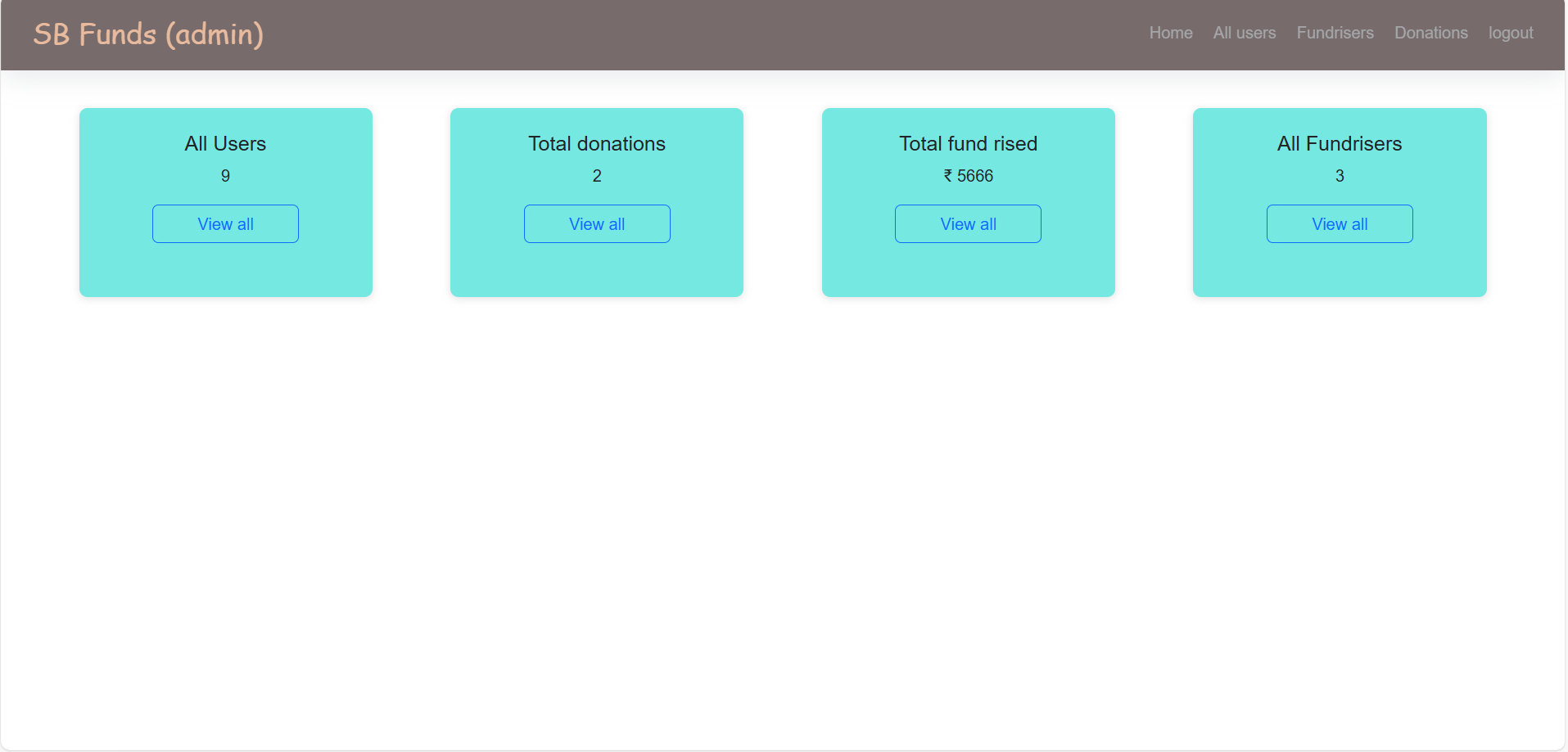
Frontend : Using React Testing Library.

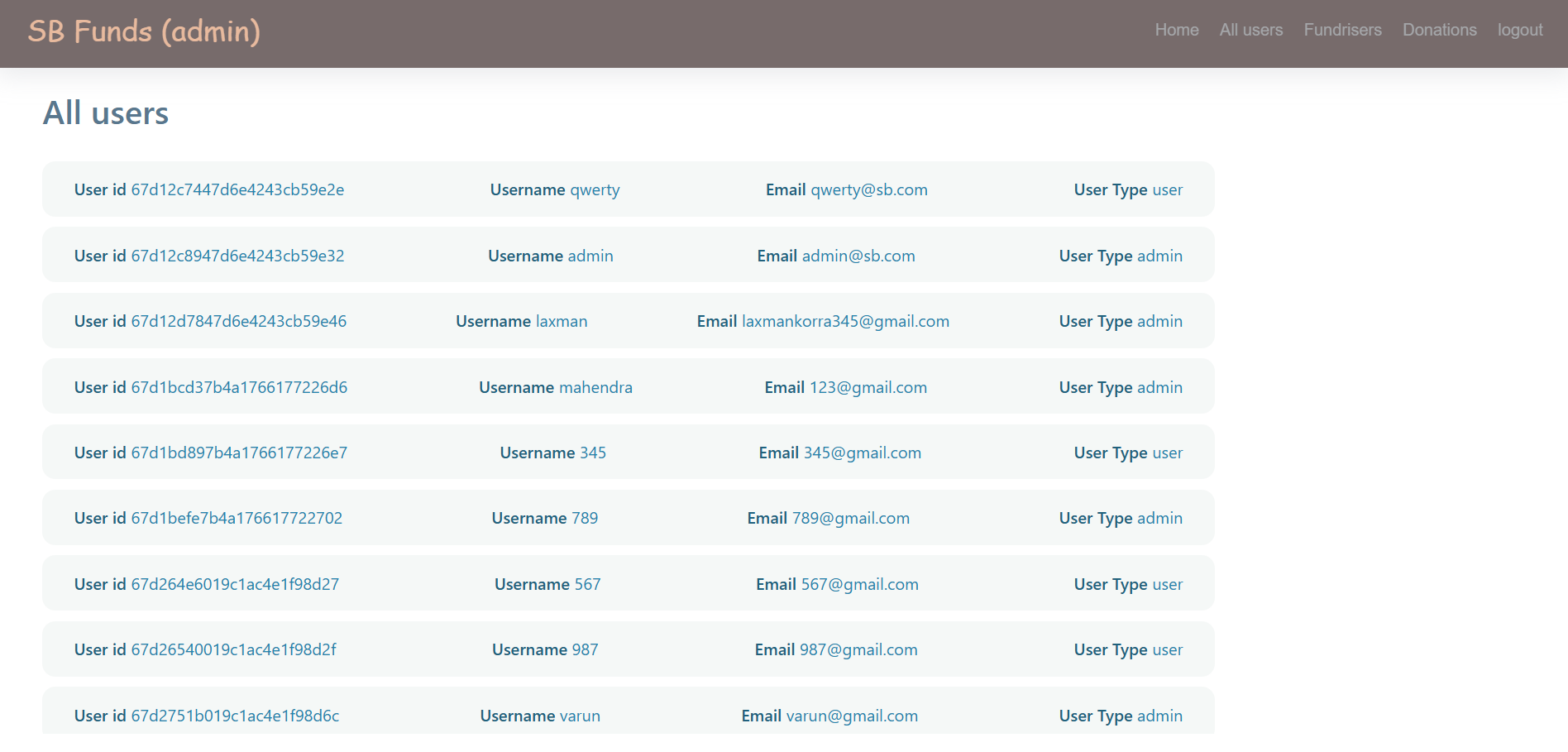
1. **Screenshots**

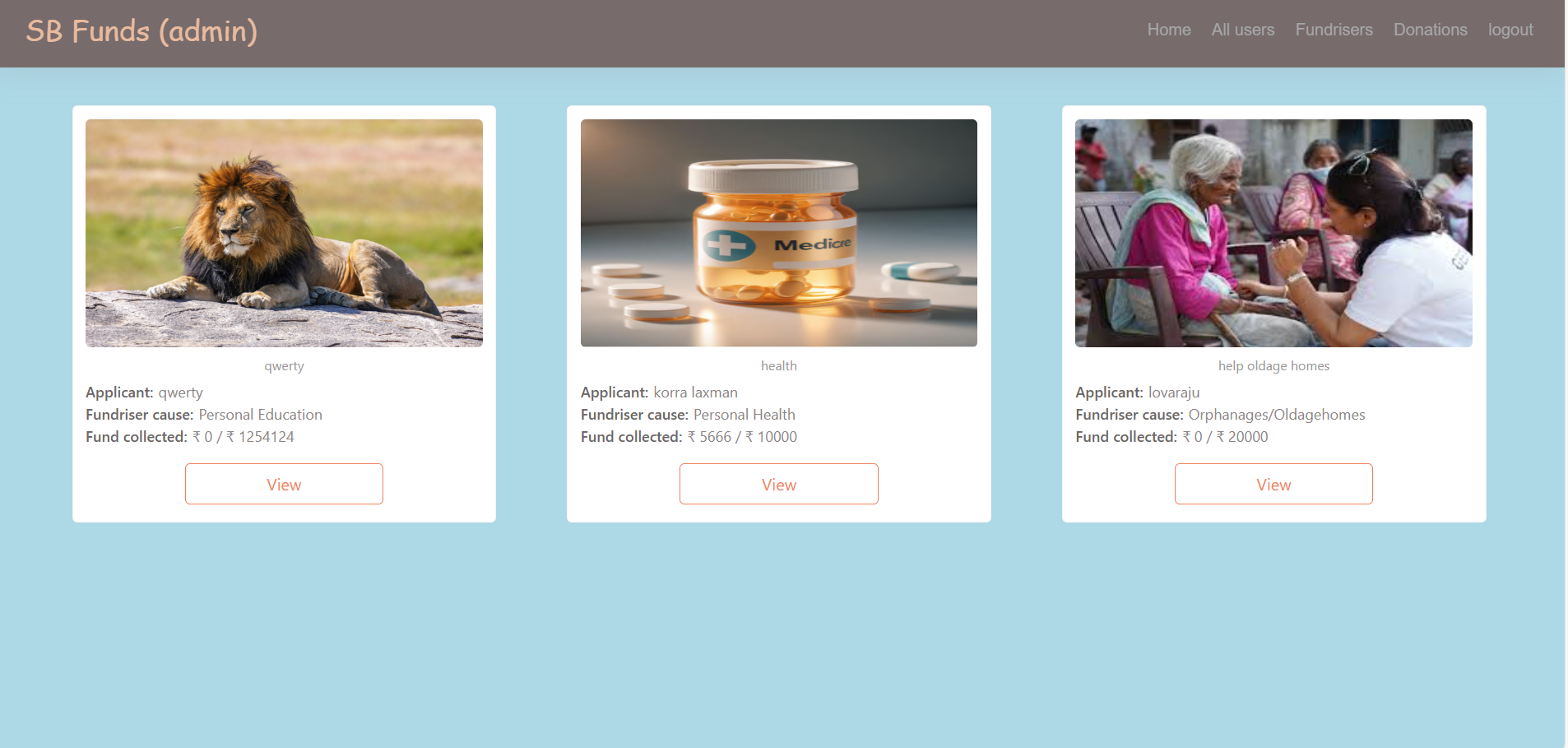
****

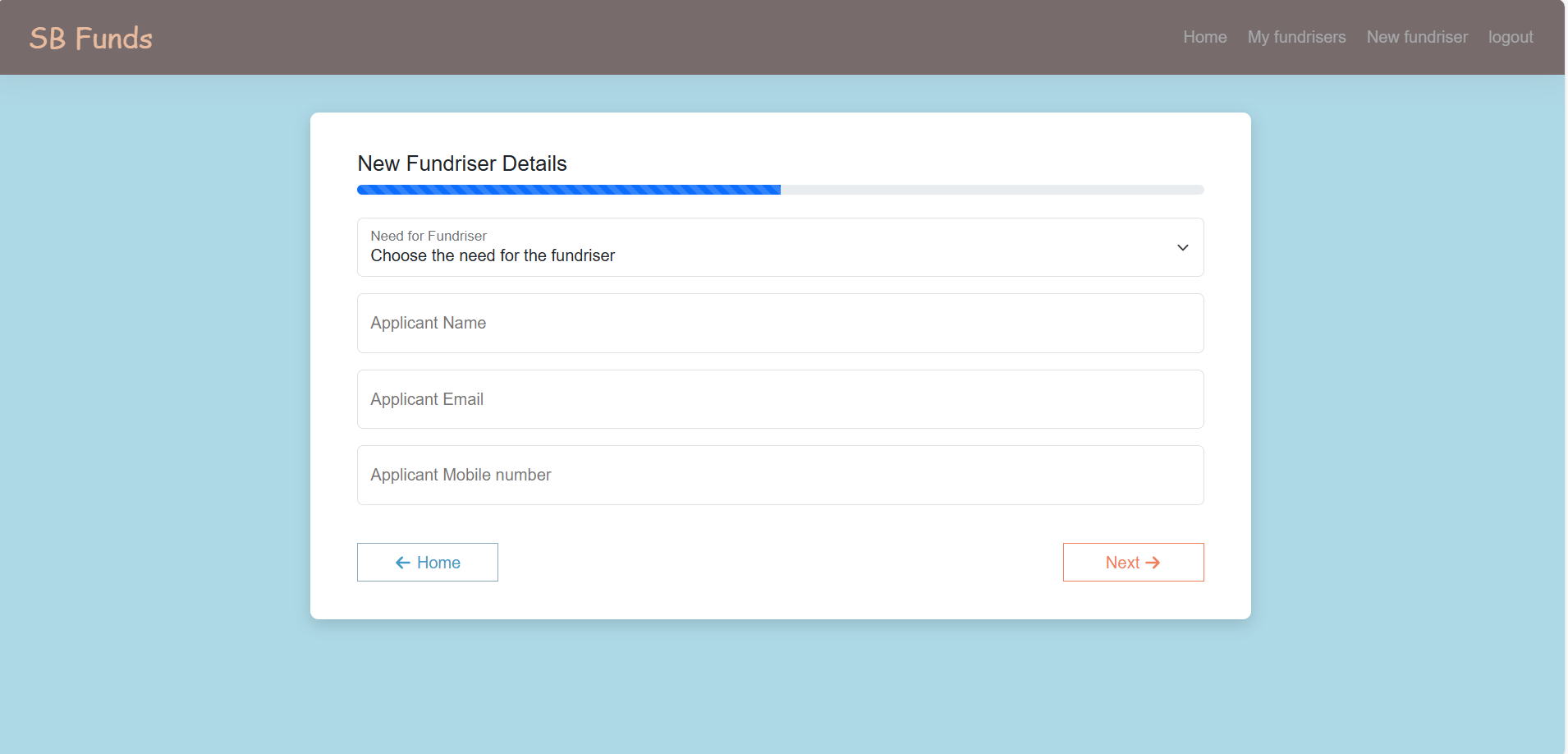
****

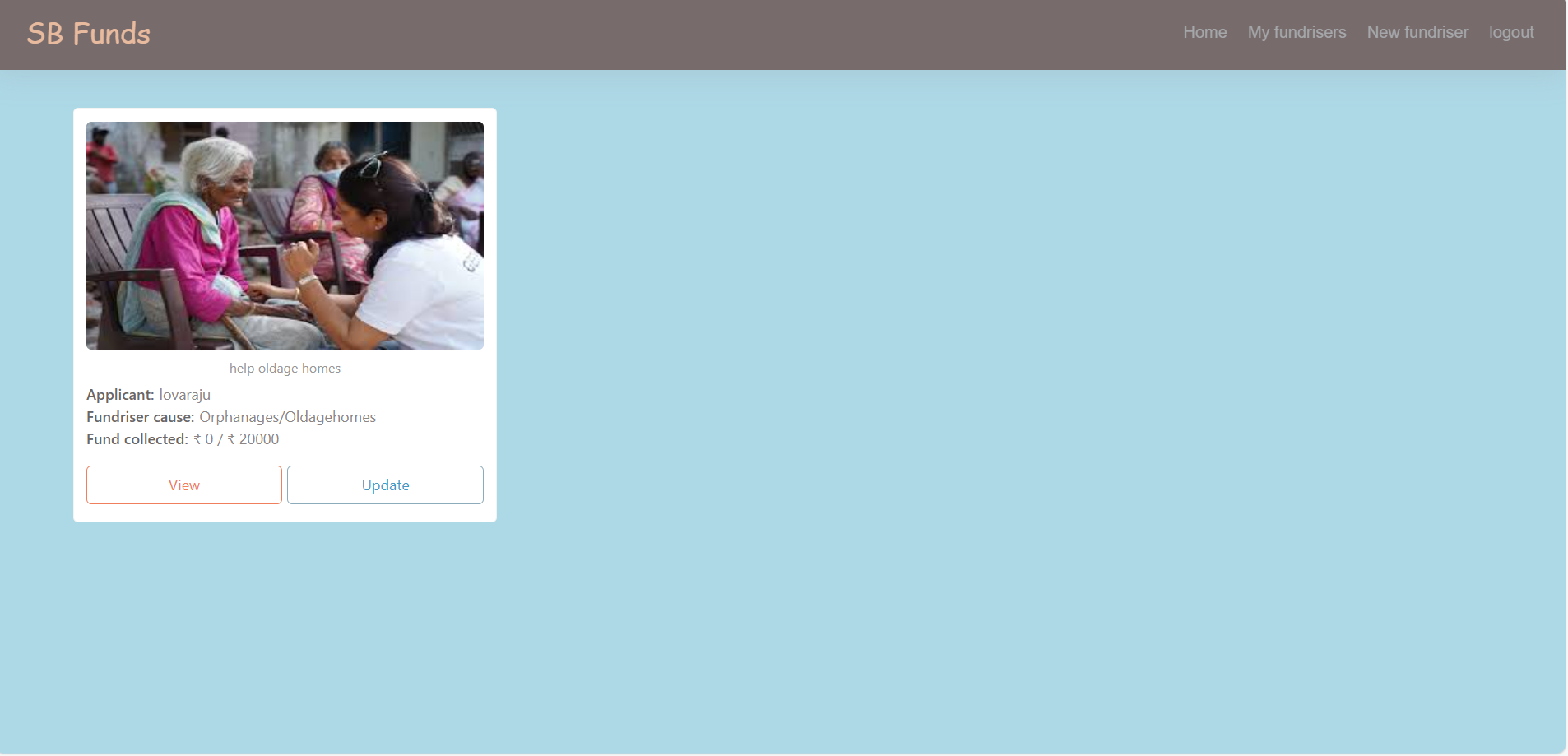


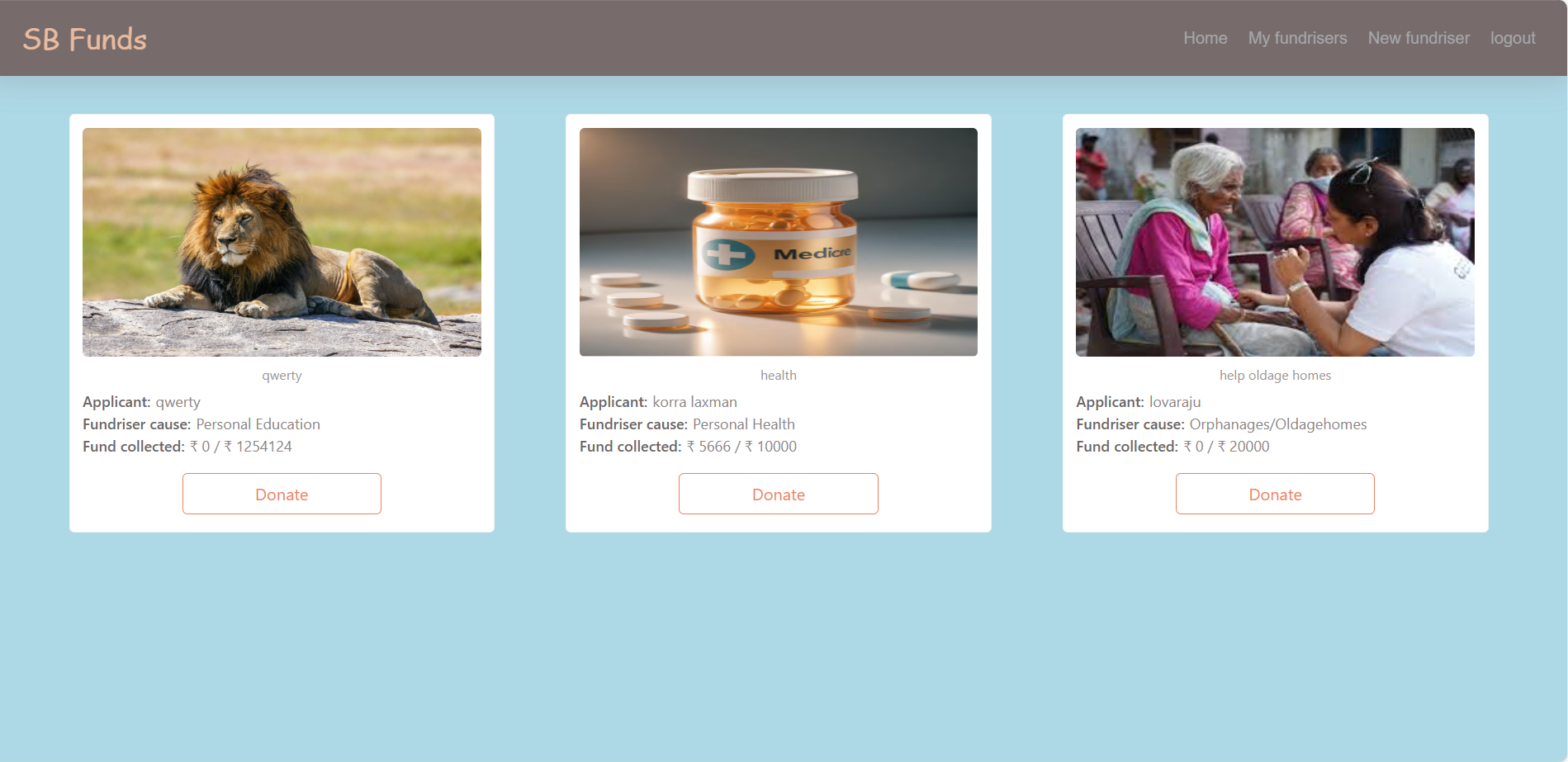












1. **Know issues**

* Payment gateway integration is under development.
* Occasional lag when loading search result with large dataset.

1. **Future enhancement**

* Simplify sign-up with guided tutorials and interactive walkthroughs.
* User reviews and ratings for these application.