SEETHI SAHIB MEMORIAL POLYTECHNIC COLLEGE TIRUR - KERALA

DEPARTMENT OF COMPUTER ENGINEERING

PROJECT REPORT

2021-2022

MENTOR SYSTEM

Submitted By ,

SAINATH A & TEAM

SEETHI SAHIB MEMORIAL POLYTECHNIC
COLLEGE TIRUR-KERALA

DEPARTMENT OF COMPUTER ENGINEERING

BONAFIDE CERTIFICATE

Certified that this Project titled "**MENTOR SYSTEM"** is the bonafide work of **Mr. SAINATH A(REG.NO 19130415)** who carried out the project my supervision. Certified further that to the best of my knowledge, the work reported here in does not form part of any other project report or dissertation on the basics of which Diploma or award was conferred on earlier occasion on this or any other scholar.

STAFFINCHARGE **HEAD OFSECTION**

Tirur

……

ABSTRACT

The Student mentoring system is introduced in the College. All the Teachers are involved in the process of mentoring. Every mentor is allotted with about 10 students to take care of them depending upon the department and semester. Every mentor have a list of all the students allotted to him / her with details of Name, register number, roll number , semester. The mentor has a chalked-out responsibilities to take care of all the mentees such as to provide them career counseling, to provide them personal counseling, to support them for any kind of difficulty in their curriculum and in their personal life, to make provision of remedial coaching for them and to always support them as and when required. Mentor system also provide a listener for mentee to speak up their problems.

The mentor also works for finding out hidden talent of the students in various aspects of academic, co – curricular, extra – curricular and extra mural activities so that they can be promoted to do various activities in the concerned area for their holistic development. The mentor also contacts and meets the parents of his / her mentees to discuss their progress and / or any other matter, as and when required.

ACKNOWLEDGEMENT

At the very outset, I would like to give the first honors to god, who gave the wisdom and knowledge to complete this project.

I express my deep gratitude and heartfelt thanks to my project guide **Mr. Saleem KN Head of Computer Department** who guided me for the successful completion of this project. I also thank him for his valuable suggestions, guidance, constant encouragement, boundless cooperation, constructive comments and motivation extended to me for the completion of this project.

I express my sincere gratitude to **Saleem KN Head of Computer Department** for their constant support, valuable suggestions and continuous monitoring without which the successful completion of this project work would not have been possible.

I express my immense pleasure and thankfulness to all other **Teaching and Non- teaching staff of the Department of Computer Engineering, Seethi Sahib Memorial Polytechnic College Tirur** for their cooperation and support.

It will be incomplete if I fail to quote the friends who helped with completing the project. Further, I extend my thanks to my parents for making this endeavour a great success.

SAINATH A

JAYAKRISHNAN KP

MUHAMMED MUNSHID V

MENTOR SYSTEM

SYNOPSIS

Mentor system is a web application that can be simplify the management of college mentoring system. The main objective of this application is to make mentee allocation for a mentor is easy an fast. You can allocate mentee randomly or allocate one by one. It provides a variety of services that include mentee allocation , search a mentee, add mentor, add mentee through form or through load file from excel, mentor list, allocated list.

HOD login

This application is for HOD usage, so only a hod have access to this application. This Login page will give the access to hod through email and password to do all the above mentioned services like mentee allocation , search mentee, add mentor, add mentee through form or through load file from excel, mentor list, allocated list.

INTRODUCTION

1.1 Project Overview

Mentor system is a web application that can be simplify the management of college mentoring system. The main objective of this application is to make mentee allocation for a mentor is easy an fast. You can allocate mentee randomly or allocate one by one. It provides a variety of services that include mentee allocation , search a mentee, add mentor, add mentee through form or through load file from excel, mentor list, allocated list.

HOD login

This application is for HOD usage, so only a hod have access to this application. This Login page will give the access to hod through email and password to do all the above mentioned services like mentee allocation , search mentee, add mentor, add mentee through form or through load file from excel, mentor list, allocated list.

.

1. Can see mentee allocated list of every semester
2. Can add wanted mentor.
3. Add mentee through form.
4. Add mentee through excel file
5. Allocate mentee randomly.
6. Can allocate mentee one by one.
7. Can see mentor list including mentor’ mobile number, email, id.
8. Can search mentee.
9. Can generate allocated list as in pdf, excel format.

Frontend - Libraries

1. React.js
2. Tailwindcss
3. Daisyui

Backend (Using Firebase Features)

1. Authentication. Firebase(database).
2. Netify (To Store Media).
3. Test Labs (Test Application).
4. Firebase

SYSTEM ANALYSIS

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the facts to improve the system. System analysis deals with a detailed study of the various operations performed by the system and their relationship within and outside of the system. System analysis is the heart of the process. Analysis helps us to understand the present system. System analysis specifies what the system should do. A system is a set of components that interact to accomplish some purpose. This chapter explains the analysis process done for **MENTOR SYSTEM.**

Identifying the drawback of the existing system. Perform feasibility study. Identify hardware, software and database requirements. Create system definition that forms the foundation for subsequent work. System analysis helped me to study the existing system and to get the needs of proposed system.

2.1 The Existing System

We don’t have web based mentor system in here. We only have the paper system to manage mentoring system in colleges and schools. It is more complicated and in case of missing it is difficult to recover the data.

Need of the application

The Student mentoring system is introduced in the College. All the Teachers are involved in the process of mentoring. Every mentor is allotted with about 60 to 70 students to take care of them depending upon the program and division. Every mentor prepares a list of all the students allotted to him / her with details of Name, Class, Division, Roll Number, Contact Number and E Mail Id. The mentor has a chalked-out responsibilities to take care of all the mentees such as to provide them career counseling, to provide them personal counseling, to support them for any kind of difficulty in their curriculum, to make provision of remedial coaching for them and to always support them as and when required.

2.2 Requirements of New System

- The motive of this Mentor System web Application is to allow the HOD to make mentee allocation for a mentor is easy an fast.
- This app uses firebase to power up the backend. It is a non relational database and has some fast data fetching capabilities. So the app is fast and reliable.
- Provide Interactive interface through which a HOD can interact with different areas of the web application easily.
- This web application is available in any browser.

2.3 Proposed System

This system proposes solutions to all the above mentioned problems. In this COVID-19 situation we need to keep social distance.In this case the online mentoring method really helps . The main objective of this application is to make mentee allocation for a mentor is easy an fast.

Advantages of Proposed System:

Current mentoring system requires more time and budget to manage this system that solve through online. Best results with short time. Software's required for the system are easily available because of which it becomes a very less expensive system. User friendly GUI can guide the new user to operate the system.

2.4 Feasibility Study

A feasibility analysis involves a detailed assessment of the need, value and practically of a systems development. Feasibility analysis n forms the transparent decision at crucial points during the developmental process as we determine whether it is operationally, economically and technically realistic to proceed with a particular course of action.

Types of Feasibility:

A feasibility analysis usually involves a thorough assessment of the financial (value), technical (practically), and operational (need) aspects of a proposal. From the initial studies it is clear that mentor system is operationally, technically, behaviorally, economically feasible and socially feasible.

Operational Feasibility:

A systems development project is likely to be operationally feasible if it meets the 'needs' and expectations of the organization. User acceptance is an important determinant of operational feasibility. Mentor system has the following functionalities include mentee allocation , search a mentee, add mentor, add mentee through form or through load file from excel, mentor list, allocated list. All functionalities work well as per the requirements of the scheme and deliver the required result in a fast and efficient manner. So mentor system is operationally feasible.

Technical Feasibility:

A systems development project may be regarded as technically feasible or practical if the organization has the necessary expertise and infrastructure to develop, install, operate and maintain the proposed system. Organization will need to make this assessment based on:

- Availability of technically qualified staffin-house for the duration of the project and subsequent maintenance phase.

- Availability of infrastructure in-house to support the development and maintenance of the proposed system.

- The capacity of the proposed system to meet initial performance expectations and accommodate new functionality over the medium term.

Mentor system provides flexibility to manage all mentoring system functions in online.

Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified and mentor system requires only computer/laptop with an

Internet Connection.

Thus the developed system as well within the budget and the cost of the entire project will depend simply on the expenditure incurred for the hardware requirements. The software requirements can be easily fulfilled without any cost.

Social Feasibility:

The acceptance of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make constructive criticism, which is welcomed, as he is the final user of the system.

Software Requirement Specification(SRS)

There are many students out there who have different problems and difficulties. Many of them are afraid or shy to share their problems to teachers. Our goal is to solve their these kind of problems. The real task is to bring out their problems. It is one of the complicated tasks to complete. Our web app gives a mentor for all mentees to become their listener. It will fix the complicated problems of mentees.

Purpose

This project is aimed for chalked-out responsibilities to take care of all the mentees such as to provide them career counseling, to provide them personal counseling, to support them for any kind of difficulty in their curriculum and in their personal life, to make provision of remedial coaching for them and to always support them as and when required. Mentor system also provide a listener for mentee to speak up their problems.

Scope

This system is aimed at a digitized way to manage mentoring system. It

will save a lot of money and also time. It's way more efficient than the current paper process.

Definitions

- SRS- Software Requirement Specification
- Visual Studio Code – Code Editor
- React.js – Frontend framework.
- Firebase -Backend

Overview

This system attempts to provide an online mentoring system for department HODs.

General Description

The system provides only access to a hod of the department. Hod need to go through a login process to use the system with a email and password.

He is the one who has all rights to do mentee allocation , search a mentee. He can add new mentor and mentee through form or through load file from excel. and he can view mentor list and allocated list.

Module Description

This section provides a requirement overview of the system. Various functionalities implemented by the system will be,

HOD :

- Login
- Add Mentor
- Add Mentee
- Mentor – Mentee Allocation
- Mentor – Mentee Allocated List
- Edit Allocation List
- Search a Mentee
- Logout

SYSTEM DESIGN

HOD

Only hod has the rights to access the system. The hod can log in to the system by using his specific email and password. He is the one who has all rights to do mentee allocation , search a mentee. He can add new mentor and mentee through form or through load file from excel. and he can view mentor list and allocated list. Also he can delete mentee from allocated list and he can reallocate a allocated mentee. Hod can also can generate allocated list in pdf format or excel format if wanted to edit the list later.

3.1 Input Design

Input design is the process of determining inputs to a particular project. Input design determines whether the user interacts with the computer in an efficient manner. Mentor system uses the following different User Interfaces for inputting values or data to the system.

3.2 Output Design

The output design has been done so that the results of processing should be communicated to the user. Effective output design will improve the clarity and performance of outputs. Following are some of the Output User Interfaces designed for mentor system:

3.3 Database Design

In this design process ,the information domain model created during analysis is transformed to data structures that will implement the software for data and information storage. After we collect the data required for the application we'll create each model for the application. We're using an non-relational database called Firebase. It stores data as collections and each collection has its own sub collection containing the related data.

3.4 Process Design

In process design, the overall structure of the process is checked out. The design is carried out using top-down design strategy. First the major modules are identified then they are divided into sub modules at the lowest level and they are addressed as a single function of a whole system.

SYSTEM IMPLEMENTATION

System Implementation

Coding Standards Used

It is a set of standard guide lines which are / should be used when writing the source code for a program

- Naming convention: We use camel cases to name variables and functions.
- Component based: Were splitted up the entire app into different components.
- DRY principle is used to avoid replication of code.
- Used less global variables.
- A better folder structure is used for maintain ability.
- Application logic and business logic is separated.

Coding environment used

Mentor System is an web application so to simulate and build an web app Visual studio code is needed. It uses React.js framework, it was based on the node.js environment, so Node.js stable 18.4.0 is needed.

Visual studio code is used to write the entire application. Also, it needs some important plugins to code smoothly. It includes (Bracket pair colorizer, ES6/ES7 React redux snippets, JavaScript)

Hardware and software implementation

Hardware specification

- **Processor**: Intel core i5 or above.
- **Ram**: 8GB or above.
- **Hard disk**: 256 or above.

- **Input Devices**: Keyboard, mouse.
- **Output devices**: Monitor

Software specification

- **Operating system:** Window 7 or above/ Mac/Linux.
- **Frontend:** React.js
- **Backend:** Firebase.
- **Visual Studio Code:** 1.68(stable)
- **Web Browser :** Google chrome

SYSTEM TESTING

Software testing is a process of running with intent of finding errors in software. Software testing assures the quality of software and represents final review of other phases of software like specification, design, code generation etc.

Unit Testing

Every Single field in the design of the project is entered with different kinds of values to know the acceptance and each time make sure that the values are saving to the server system.

Integration Testing

In integration testing a system consisting of different modules is tested for problems arising from component interaction. Integration testing should be developed from the system specification. Firstly, a minimum configuration must be integrated and tested.

In my project I have done integration testing in a bottom up fashion i.e. in this project I have started construction and testing with atomic modules. After unit testing the modules are integrated one by one and then tested the system for problems arising from component interaction.

Validation Testing

It provides final assurances that software meets all functional, behavioral & performance requirement. Black box testing techniques are used.

There are three main components

- Validation test criteria (no. in place of no. & char in place of char)

- Configuration review (to ensure the completeness of s/w configuration.)

- Alpha & Beta testing-Alpha testing is done at developer's site i.e. at home & Beta testing once it is deployed. Since I have not deployed my application, I could not do the Beta testing.

Test Cases- I have used a number of test cases for testing the product. There were different cases for which different inputs were used to check whether desired output is produced or not.

1. Testing hod login with correct and incorrect emails and passwords

2. Addition of new mentors and mentees.

3. Mentee allocation after clear the mentor list.

4. Reallocation and Deletion of allocated mentees.

5. Searching mentees.

White Box Testing

For this testing technique all possible test cases are generated for testing every statement of subroutines, functions and modules of a class. Using these test cases every statement of functions and subroutines of a class are executed at least once for finding errors. Here all conditional branching statements and loop statements are tested. These errors are corrected after white box testing process.

CONCLUSION

Mentor system is proposed to simplify the mentoring system in colleges. The aim is create a better way to manage mentor system instead of current paper system. This method is secure and easy for use. It encourages to help mentees in a simple way. Through this the mentor also works for finding out hidden talent of the students in various aspects of academic, co – curricular, extra – curricular and extra mural activities so that they can be promoted to do various activities in the concerned area for their holistic development.

SCOPE FOR FUTURE ENHANCEMENT

The project has a very vast scope in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion.

The following are the future scope for the project.

- Creating a fast and slow learner identification based on two entities which is class test mark and number of back papers.

- Can add a student login for mentees to know their academic details.

- Creating a private chatting system for mentees to share their personal or academic problems to their mentor.

- The current system is confined only for one department. It can be extended a whole college.

- Can be create android application of this project for easy usage.

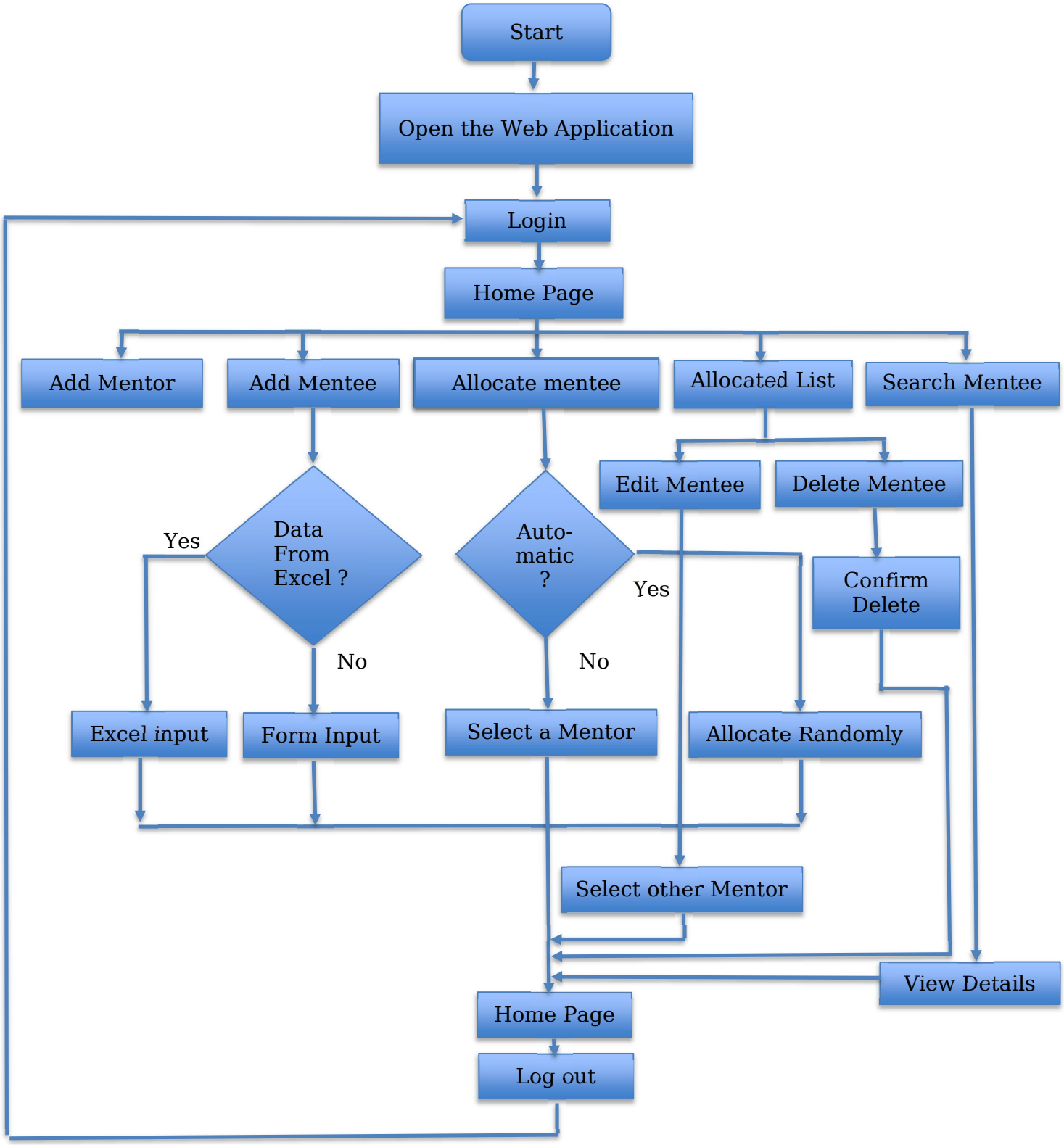
- Can be include more mentee details like health information and personal information.

BIBLIOGRAPHY

References

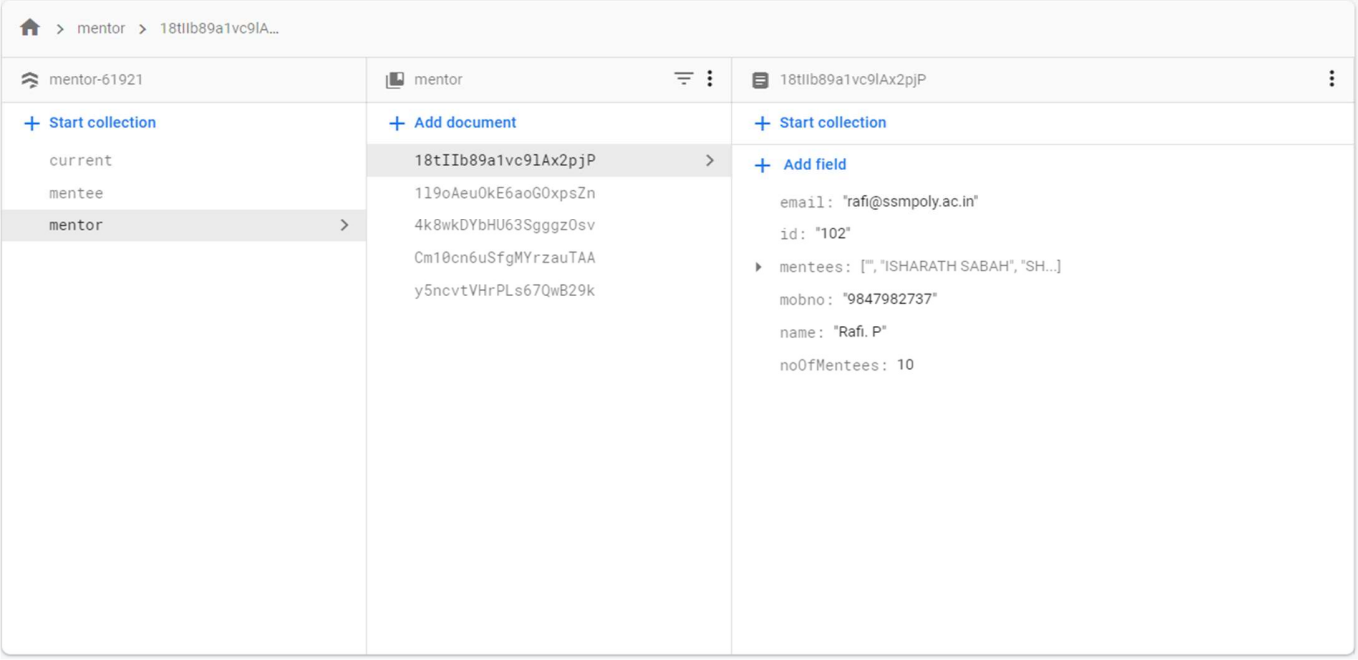
- https://nodejs.org/en
- https://reactjs.org/
- https://www.wikipedia.org/

APPENDIX

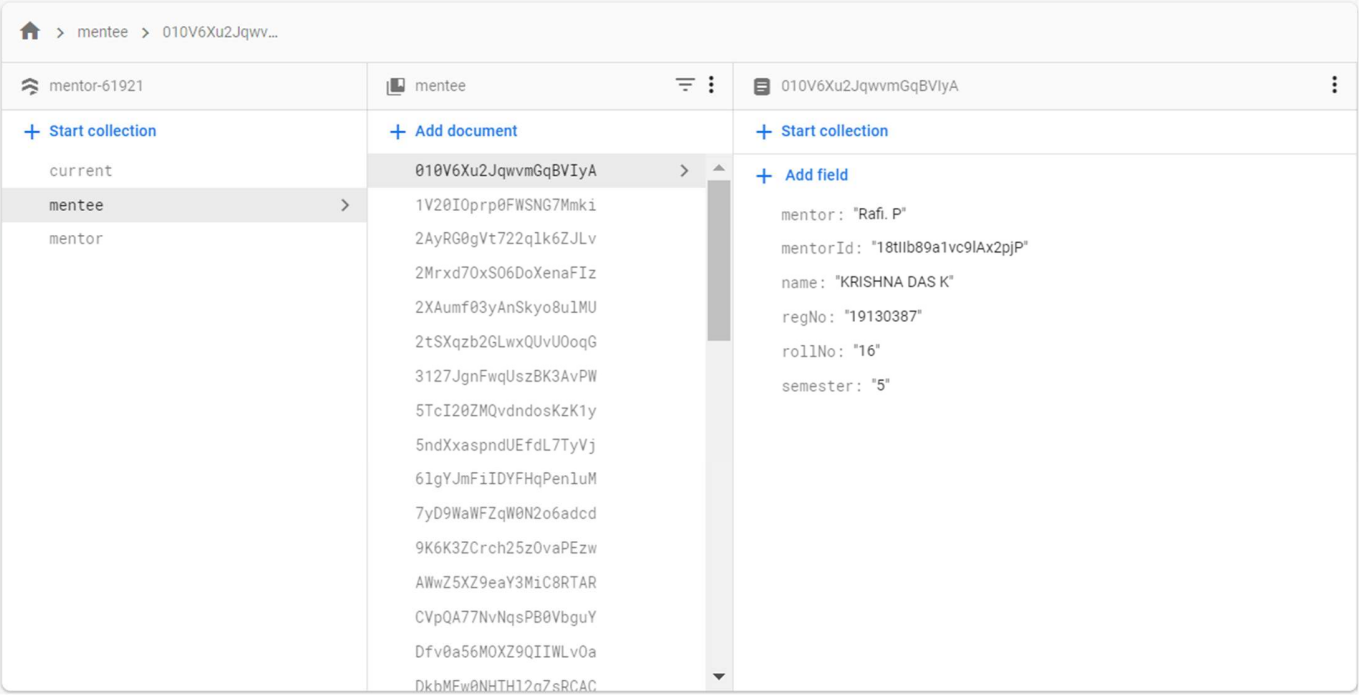

Start
Open the Web Application
Login
Home Page
Add Mentor
Add Mentee
Allocate mentee
Allocated List
Search Mentee
Edit Mentee
Delete Mentee
Yes
Data From Excel ?
Auto-matic ?
Yes
Confirm Delete
No
No
Excel input
Form Input
Select a Mentor
Allocate Randomly
Select other Mentor
View Details
Home Page
Log out

Database Design

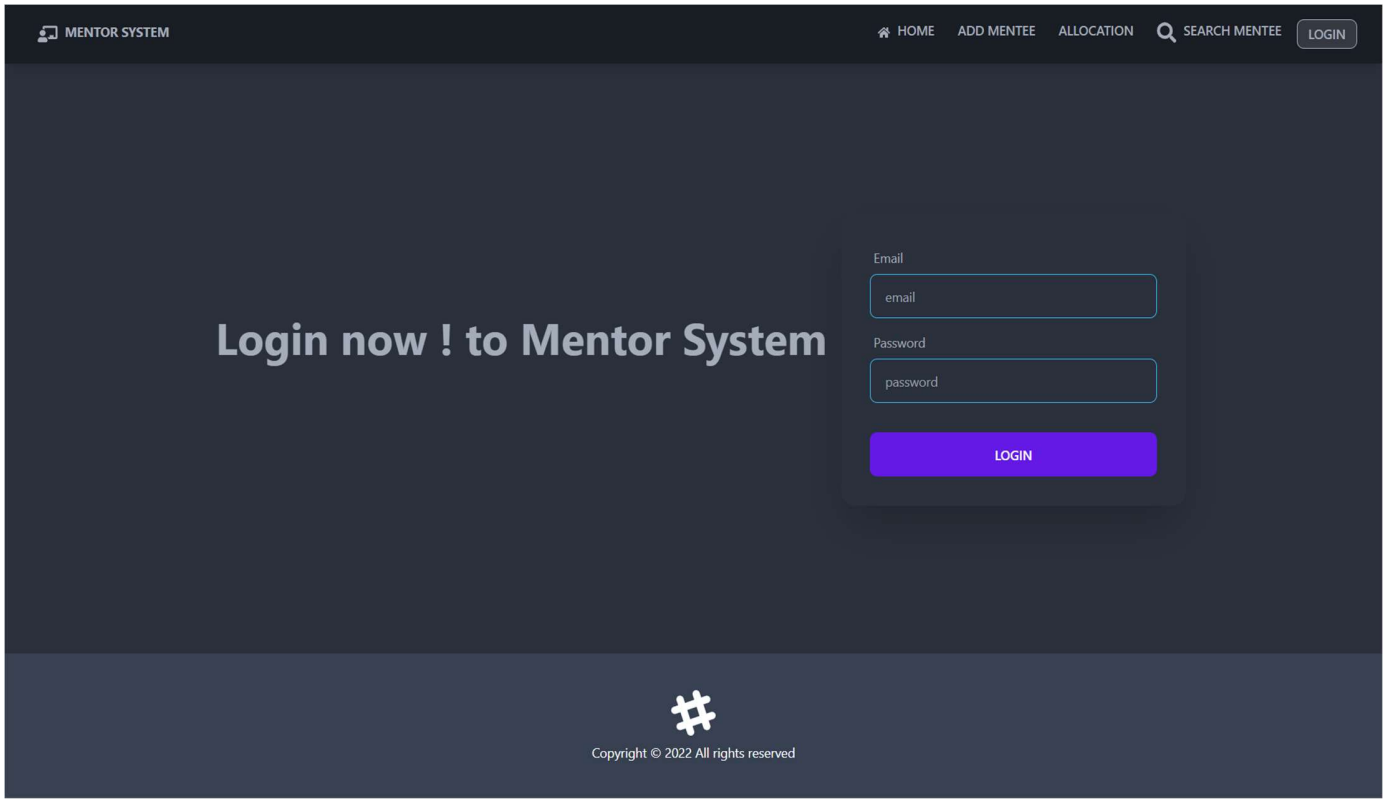
MENTOR

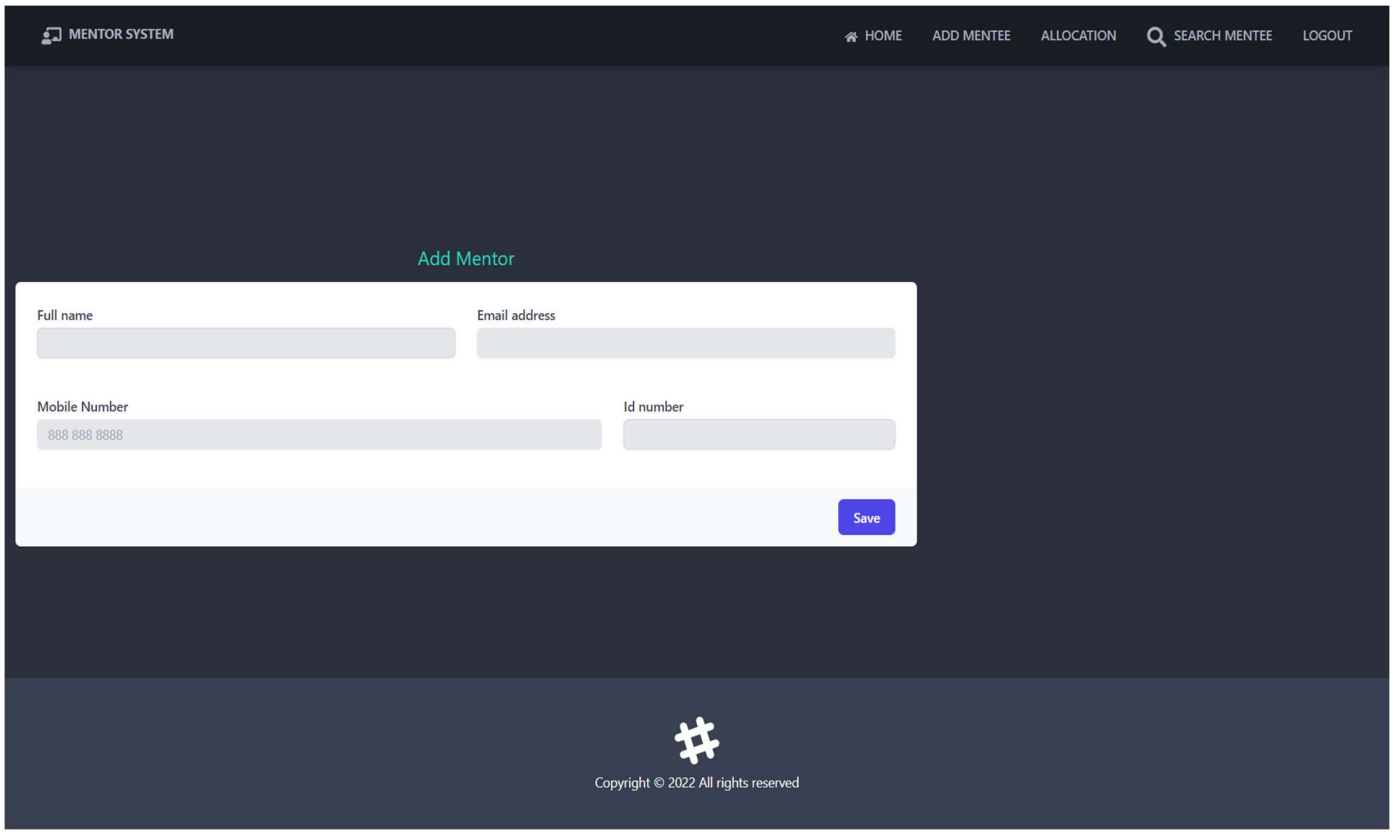


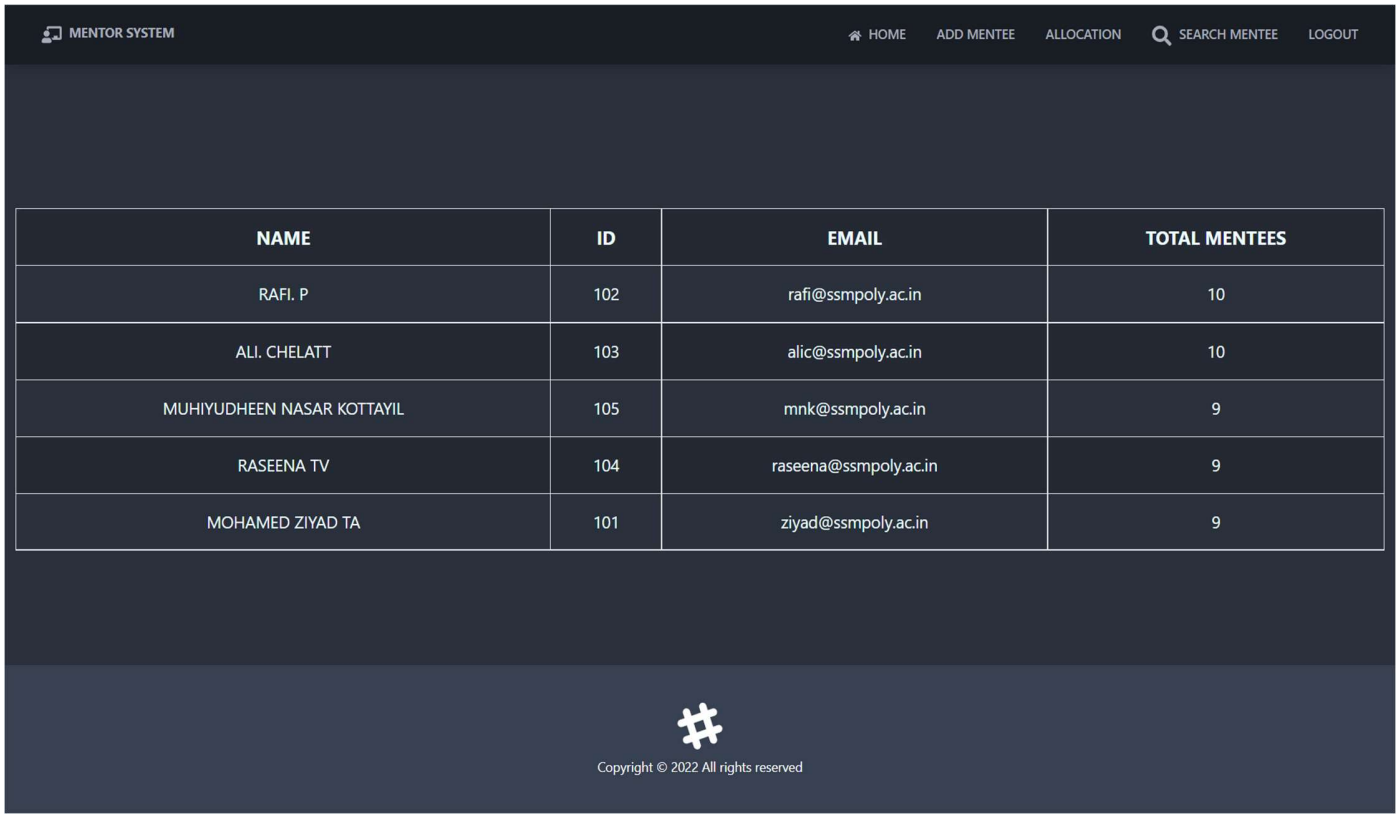
MENTEE



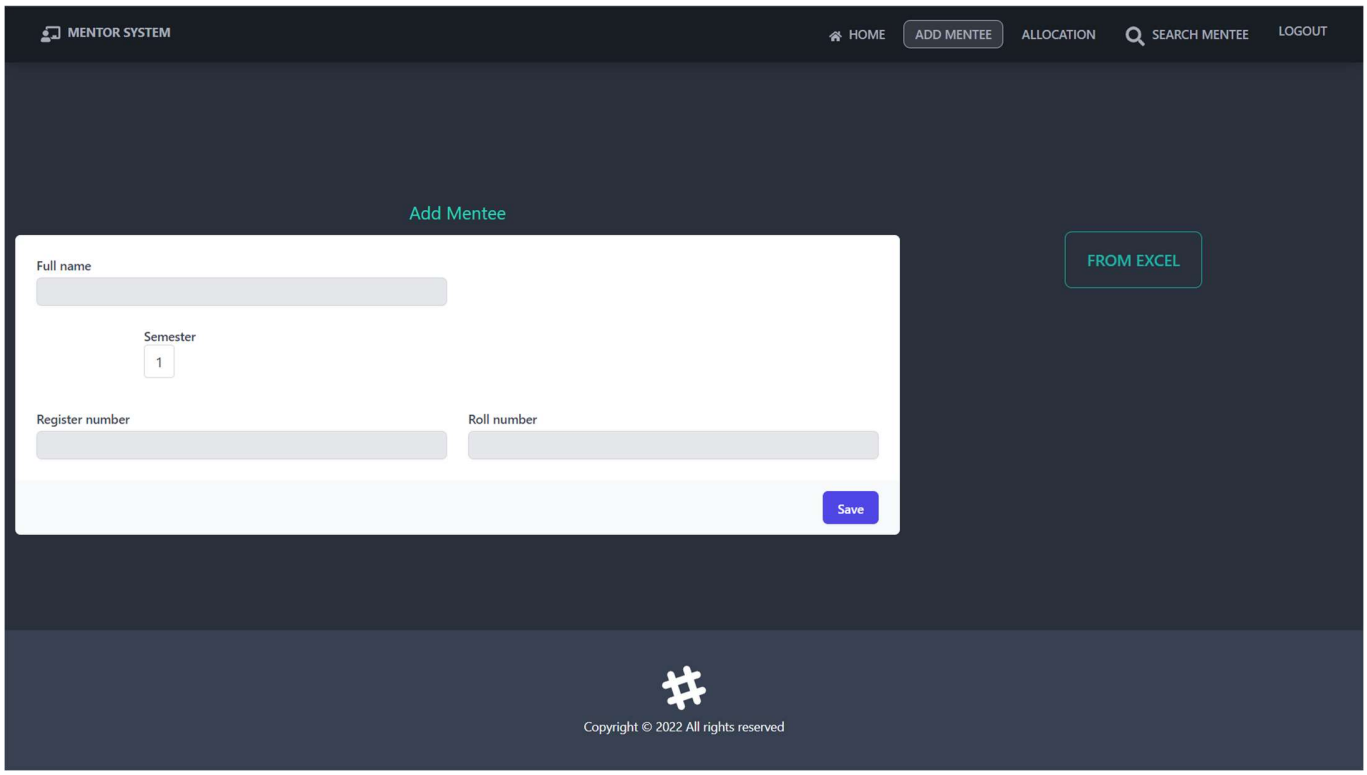
SAMPLE SCREENSHOT

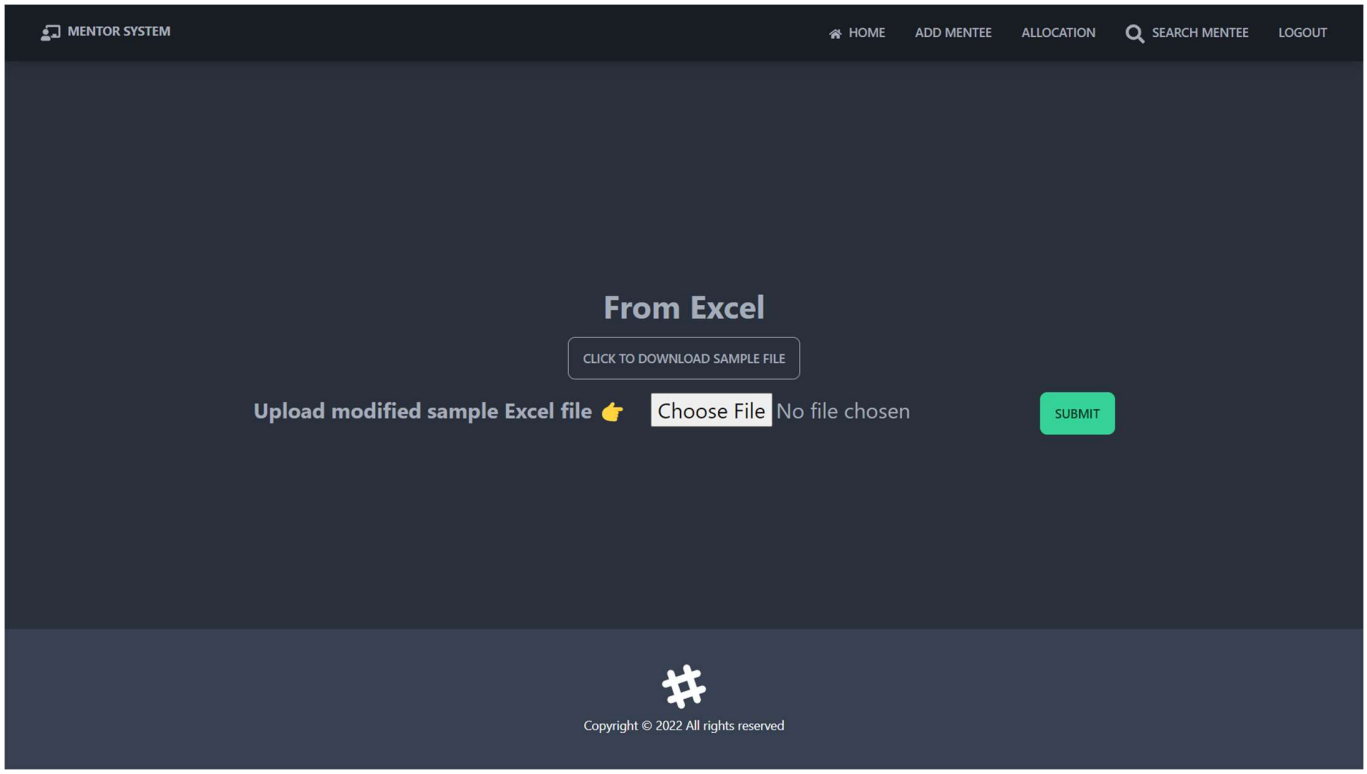






NAME	ID	EMAIL	TOTAL MENTEES
RAFI. P	102	rafi@ssmpoly.ac.in	10
ALI. CHELATT	103	alic@ssmpoly.ac.in	10
MUHIYUDHEEN NASAR KOTTAYIL	105	mnk@ssmpoly.ac.in	9
RASEENA TV	104	raseena@ssmpoly.ac.in	9
MOHAMED ZIYAD TA	101	ziyad@ssmpoly.ac.in	9


MENTOR SYSTEM
HOME
ADD MENTEE
ALLOCATION
SEARCH MENTEE
LOGOUT
Add Mentee
Full name
Semester
1
Register number
Roll number
Save
FROM EXCEL
Copyright © 2022 All rights reserved


MENTOR SYSTEM
HOME
ADD MENTEE
ALLOCATION
SEARCH MENTEE
LOGOUT
From Excel
CLICK TO DOWNLOAD SAMPLE FILE
Upload modified sample Excel file
Choose File
No file chosen
SUBMIT
Copyright © 2022 All rights reserved

MENTOR SYSTEM

HOME ADD MENTEE ALLOCATION SEARCH MENTEE LOGOUT

Allocation page

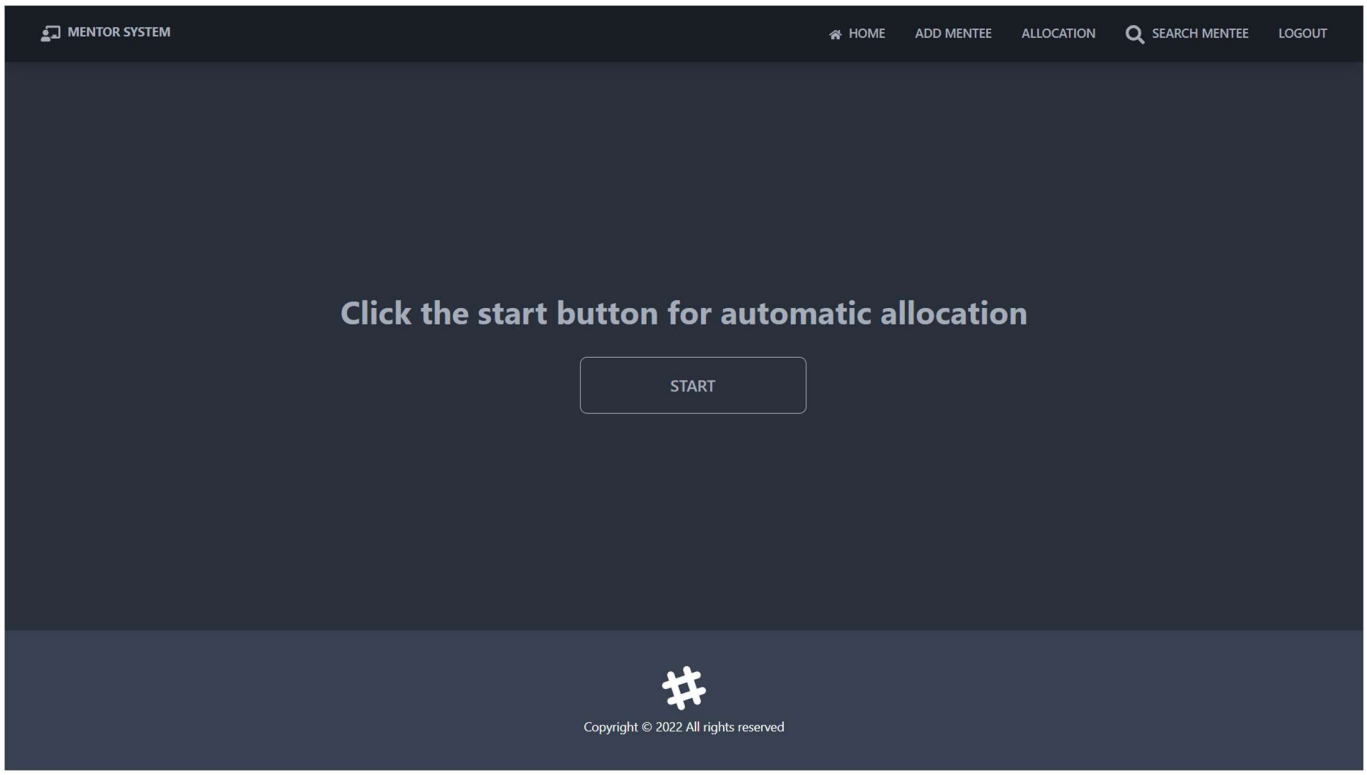
CLICK HERE AUTOMATICALLY ALLOCATE MENTEES

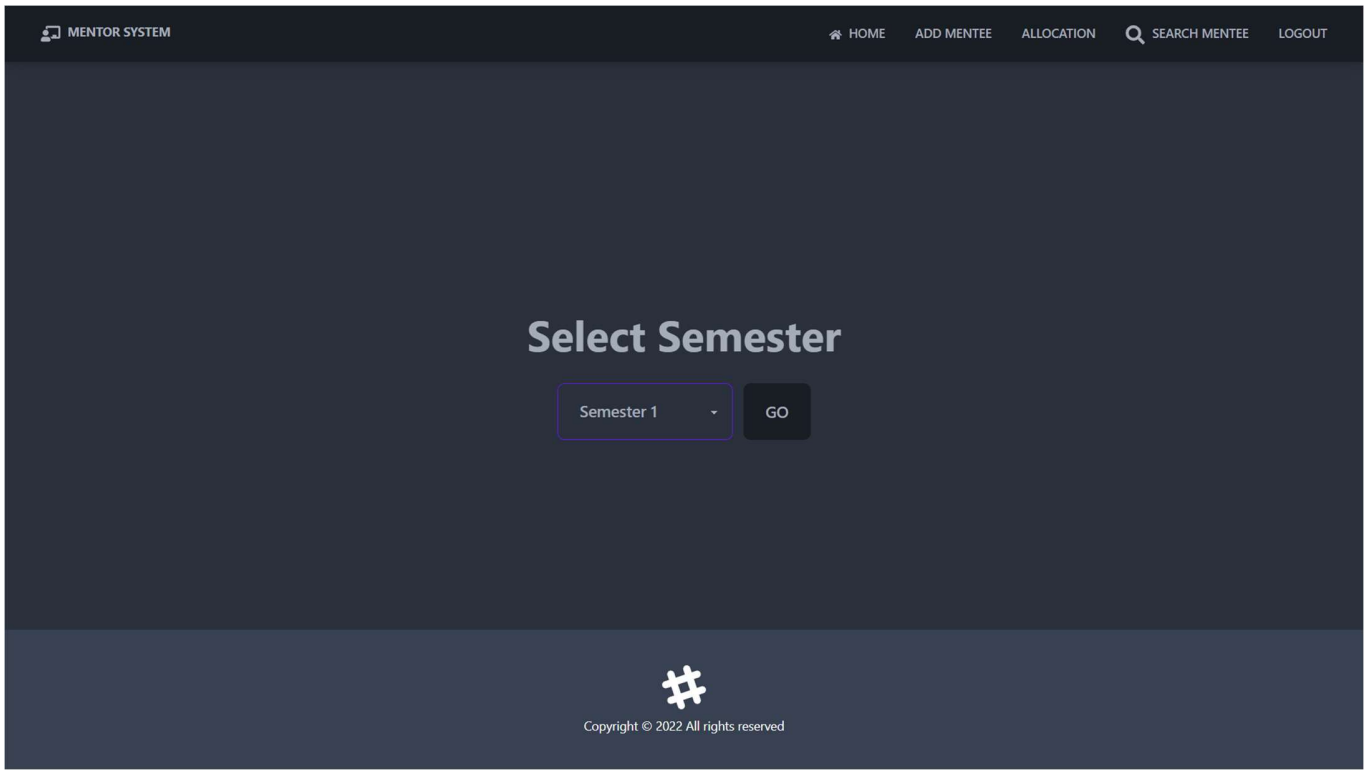
REGNO	SEMESTER	ROLLNO	NAME	ALLOCATE
17130401	5	1	ISHARATH SABAH	ALLOCATE
17131557	5	2	SHAHMA N	ALLOCATE
19130369	5	3	ADITHYA S	ALLOCATE
19130370	5	4	AISWARYA A	ALLOCATE
19130374	5	5	ANZEERA. P.P	ALLOCATE
19130375	5	6	ARSHITHA T	ALLOCATE

MENTOR SYSTEM

HOME ADD MENTEE ALLOCATION SEARCH MENTEE LOGOUT

NAME	ID	EMAIL	TOTAL MENTEES	ALLOCATE
RAFI. P	102	rafi@ssmpoly.ac.in	1	ALLOCATE
ALI. CHELATT	103	alic@ssmpoly.ac.in	0	ALLOCATE
MUHIYUDHEEN NASAR KOTTAYIL	105	mnk@ssmpoly.ac.in	0	ALLOCATE
RASEENA TV	104	raseena@ssmpoly.ac.in	0	ALLOCATE
MOHAMED ZIYAD TA	101	ziyad@ssmpoly.ac.in	0	ALLOCATE


MENTOR SYSTEM
HOME
ADD MENTEE
ALLOCATION
SEARCH MENTEE
LOGOUT
Click the start button for automatic allocation
START
Copyright © 2022 All rights reserved


MENTOR SYSTEM
HOME
ADD MENTEE
ALLOCATION
SEARCH MENTEE
LOGOUT
Select Semester
Semester 1
GO
Copyright © 2022 All rights reserved

MENTOR SYSTEM

HOME ADD MENTEE ALLOCATION SEARCH MENTEE LOGOUT

Allocated List

EXPORT TO EXCEL EXPORT TO PDF EDIT

REGNO	ROLLNO	NAME	MENTOR
17130401	1	ISHARATH SABAH	Rafi. P
17131557	2	SHAHMA N	Ali. Chelatt
19130369	3	ADITHYA S	Ali. Chelatt
19130370	4	AISWARYA A	Muhiyudheen Nasar Kottayil
19130374	5	ANZEERA. P.P	Muhiyudheen Nasar Kottayil
19130375	6	ARSHITHA T	Raseena TV
19130376	7	ASIF SHAHEER. V.P	Raseena TV
19130377	8	ASIYA M V	Mohamed Ziyad TA

MENTOR SYSTEM

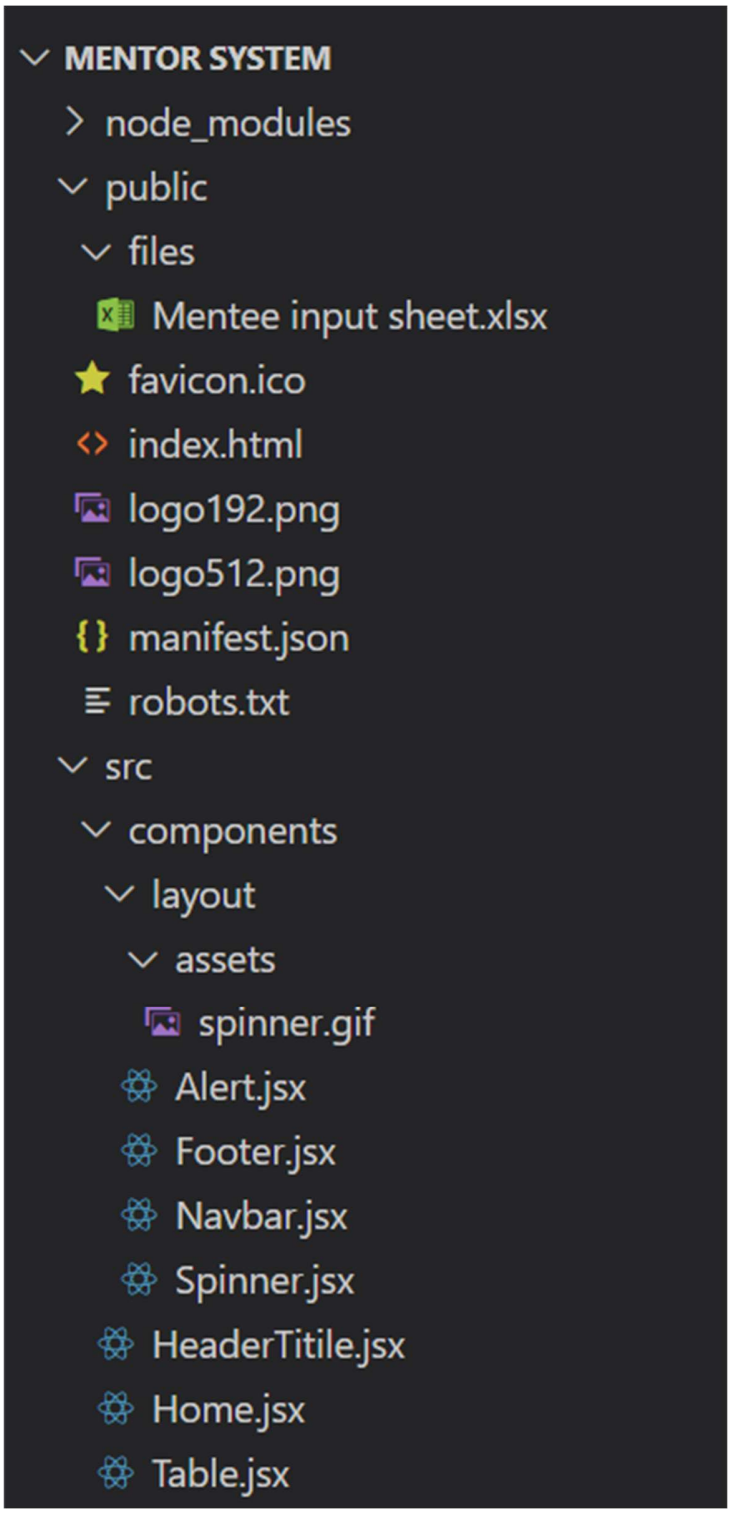
HOME ADD MENTEE ALLOCATION SEARCH MENTEE LOGOUT

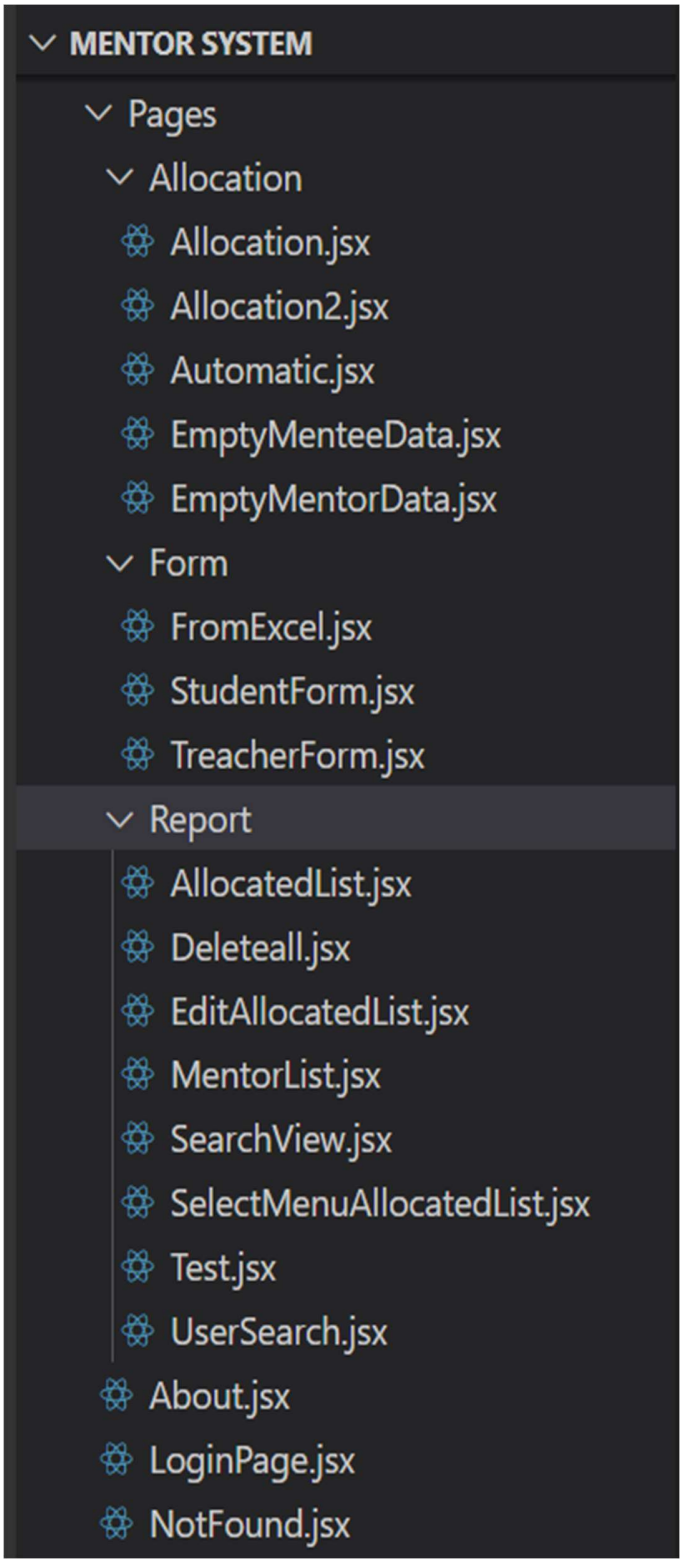
Edit Allocated List

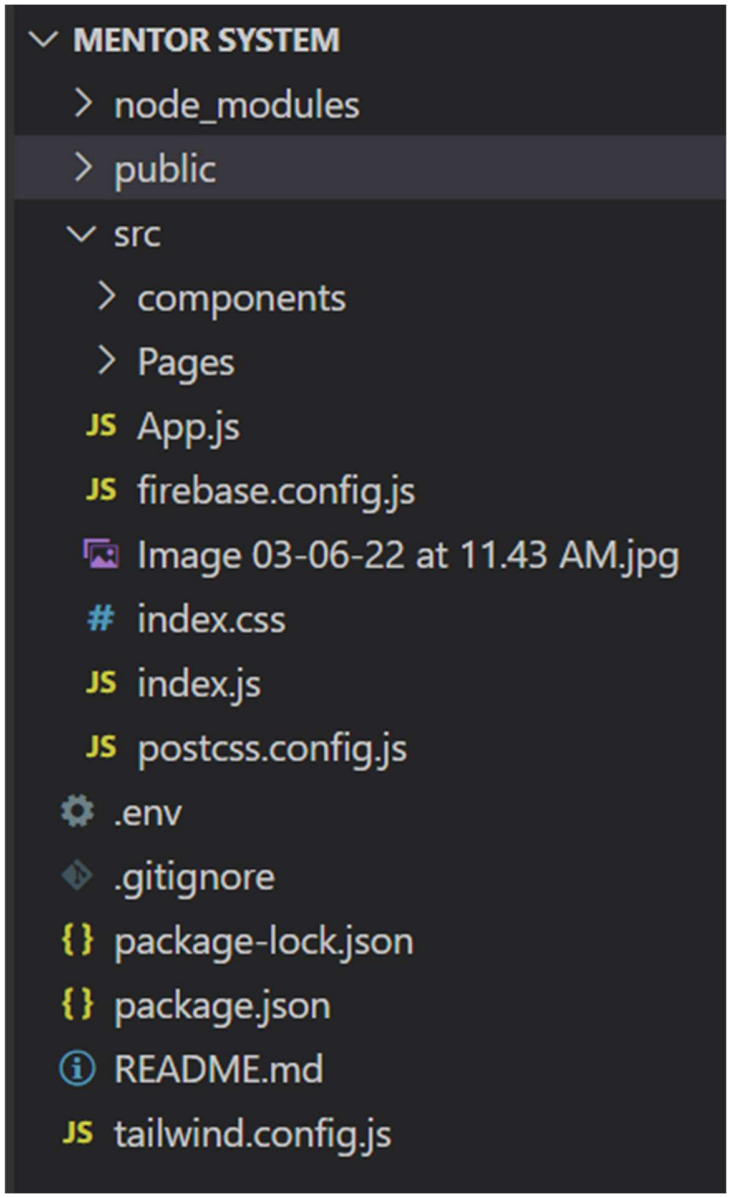
REGNO	ROLLNO	NAME	MENTOR	REALLOCATE	DELETE
17130401	1	ISHARATH SABAH	Rafi. P	REALLOCATE	DELETE
17131557	2	SHAHMA N	Ali. Chelatt	REALLOCATE	DELETE
19130369	3	ADITHYA S	Ali. Chelatt	REALLOCATE	DELETE
19130370	4	AISWARYA A	Muhiyudheen Nasar Kottayil	REALLOCATE	DELETE
19130374	5	ANZEERA. P.P	Muhiyudheen Nasar Kottayil	REALLOCATE	DELETE
19130375	6	ARSHITHA T	Raseena TV	REALLOCATE	DELETE
19130376	7	ASIF SHAHEER. V.P	Raseena TV	REALLOCATE	DELETE

Sample Code

Project structure




MENTOR SYSTEM
Pages
Allocation
Allocation.jsx
Allocation2.jsx
Automatic.jsx
EmptyMenteeData.jsx
EmptyMentorData.jsx
Form
FromExcel.jsx
StudentForm.jsx
TreacherForm.jsx
Report
AllocatedList.jsx
Deleteall.jsx
EditAllocatedList.jsx
MentorList.jsx
SearchView.jsx
SelectMenuAllocatedList.jsx
Test.jsx
UserSearch.jsx
About.jsx
LoginPage.jsx
NotFound.jsx


MENTOR SYSTEM
node_modules
public
src
components
Pages
App.js
firebase.config.js
Image 03-06-22 at 11.43 AM.jpg
index.css
index.js
postcss.config.js
.env
.gitignore
package-lock.json
package.json
README.md
tailwind.config.js

index.html
Mentor System\public\index.html

```
<!DOCTYPE html>
<html lang="en" data-theme="dark">

<head>

  <title>Mentor System</title>
</head>

<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>

</body>

</html>
```

manifest.json
Mentor System\public\manifest.json

```
{
  "short_name": "React App",
  "name": "Create React App Sample",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logo192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "logo512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

App.js

Mentor System\src\App.js

```
import { BrowserRouter as Router, Route, Routes } from "react-router-dom"

import Navbar from "./components/layout/Navbar"
import Footer from "./components/layout/Footer"

//context

//pages
import TreacherForm from "./Pages/Form/TreacherForm"
import NotFound from "./Pages/NotFound"
import StudentForm from "./Pages/Form/StudentForm"
import Home from "./components/Home"
import Allocation from "./Pages/Allocation/Allocation"
import Allocation2 from "./Pages/Allocation/Allocation2"
import AllocatedList from "./Pages/Report/AllocatedList"
import EditAllocatedList from "./Pages/Report/EditAllocatedList"
import SelectMenuAllocatedList from "./Pages/Report/SelectMenuAllocatedList"
import Automatic from "./Pages/Allocation/Automatic"
import FromExcel from "./Pages/Form/FromExcel"
import Deleteall from "./Pages/Report/Deleteall"
import LoginPage from "./Pages/LoginPage"
import MentorList from "./Pages/Report/MentorList"
import UserSearch from "./Pages/Report/UserSearch"
import Test from "./Pages/Report/Test"
import SearchView from "./Pages/Report/SearchView"
function App() {
  return (
    <Router>
      <div className='flex flex-col justify-between h-screen'>
        <Navbar />
        <main className='container mx-auto px-3 pd-12'>
          <Routes>
            <Route path='/AddMentor' element={<TreacherForm />} />
            <Route path='/AddMentee' element={<StudentForm />} />
            <Route path='/AddMentee/FromExcel' element={<FromExcel />} />
            <Route path='/home' element={<Home />} />
            <Route path='/notfound' element={<NotFound />} />
            {/* <Route path='/about' element={<NotFound />} /> */}
            <Route path='/Allocation' element={<Allocation />} />
            <Route path='/Allocation2' element={<Allocation2 />} />
            <Route path='/AllocatedList' element={<AllocatedList />} />
            <Route path='/EditAllocatedList' element={<EditAllocatedList />} />

            <Route
              path='/SelectMenuAllocatedList'
              element={<SelectMenuAllocatedList />}
            />
            <Route path='/Automatic' element={<Automatic />} />
            <Route path='/FromExcel' element={<FromExcel />} />
```

```
            <Route path='/Deleteall' element={<Deleteall />} />


            <Route path='/' element={<LoginPage />} />
            <Route path='/MentorList' element={<MentorList />} />
            <Route path='/Search' element={<UserSearch />} />
            <Route path='/SearchView' element={<SearchView />} />
            <Route path='/test' element={<Test />} />

            <Route path='/*' element={<NotFound />} />
          </Routes>
        </main>
        <Footer />
      </div>
    </Router>
  )
}

export default App
```

firebase.config.js
Mentor System\src\firebase.config.js

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app"
import { getFirestore } from "firebase/firestore" // TODO: Add SDKs for Firebase products that you want
to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyB2Uwq-oUweIrxqltYQ4Q9bWKOGpsU0b_0",
  authDomain: "mentor-61921.firebaseapp.com",
  projectId: "mentor-61921",
  storageBucket: "mentor-61921.appspot.com",
  messagingSenderId: "955747965952",
  appId: "1:955747965952:web:0ba9cc0c7039162ae3cce5",
}

// Initialize Firebase
initializeApp(firebaseConfig)

export const db = getFirestore()
```

index.css
Mentor System\src\index.css

```
@tailwind base;
@tailwind components;
@tailwind utilities;

.custom-card-image .card.image-full:before {
  border-radius: 0.5rem;
  opacity: 0.45;
}
```

index.js
Mentor System\src\index.js

```
import React from "react"
//import "antd/dist/antd.css"
import ReactDOM from "react-dom"
import "./index.css"
import App from "./App"

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById("root")
)
```

HeaderTitile.jsx
Mentor System\src\components\HeaderTitile.jsx

```
import React from "react"

function HeaderTitile({ children }) {
  return (
    <div>
      <div className='hero'>
        <div className='text-center  hero-content'> </div>
        <div className='max-w-full'>
          <h1 className=' text-8xl   font-bold mb-8'>
            <div className='w-full flex justify-center'></div>
            <div>
              <h3 className='text-5xl mt-5 text-center mx-auto'>{children}</h3>
            </div>
          </h1>
        </div>
```

```
      </div>
    </div>
  )
}

export default HeaderTitile
```

Home.jsx
Mentor System\src\components\Home.jsx

```
import { FaChalkboardTeacher } from "react-icons/fa"
import { Link } from "react-router-dom"

function Home() {
  return (
    <div className='hero'>
      <div className='text-center  hero-content'> </div>
      <div className='max-w-full'>
        <h1 className=' text-8xl   font-bold mb-8'>
          <div className='w-full flex justify-center'>
            <FaChalkboardTeacher className='  mr-5 inline-flex' />
          </div>
          <div className='text-center mx-auto'>
            Mentor System
            <h3 className='text-5xl mt-5 text-center mx-auto'>welcome</h3>
          </div>
        </h1>

        <div className='flex-1 px-2 mx-2'>
          <div className='flex justify-end'>
            <Link
              to='/SelectMenuAllocatedList'
              className='mr-2 btn  btn-outline btn-info btn-lg'
            >
              Allocated List
            </Link>

            {/* <Link to='/about' className='btn btn-ghost btn-sm rounded-btn'>
              About
            </Link> */}
            <Link
              to='/AddMentor'
              className=' mr-2 btn  btn-outline btn-info btn-lg'
            >
              Add Mentor
            </Link>
            <Link
              to='/AddMentee'
              className='  mr-2 btn  btn-outline btn-info btn-lg'
            >
              Add Mentee
            </Link>
```

```
            <Link
              to='/Allocation'
              className=' mr-2 btn  btn-outline btn-info btn-lg'
            >
              Allocation
            </Link>

            <Link
              to='/MentorList'
              className=' mr-2 btn  btn-outline btn-info btn-lg'
            >
              Mentor List
            </Link>

            <br />
            <br />
            <br />
          </div>
        </div>
      </div>
    </div>
  )
}

export default Home
```

Table.jsx
Mentor System\src\components\Table.jsx

```
import React from "react"

function Table() {
  return (
    <div className='overflow-x-auto'>
      <table className='table table-zebra w-full'>
        {/* <!-- head --> */}
        <thead>
          <tr>
            <th className='border'></th>
            <th className='border'>Name</th>
            <th className='border'>Job</th>
            <th className='border'>Favorite Color</th>
          </tr>
        </thead>
        <tbody>
          {/* <!-- row 1 --> */}
          <tr>
            <th className='border'>1</th>
            <td className='border'>Cy Ganderton</td>
            <td>Quality Control Specialist</td>
            <td>Blue</td>
          </tr>
```

```
        </tbody>
      </table>
    </div>
  )
}

export default Table
```

Alert.jsx
Mentor System\src\components\layout\Alert.jsx

```
import { useContext } from "react"
import AlertContext from "../../context/alert/AlertContext"

function Alert() {
  const { alert } = useContext(AlertContext)
  return (
    alert !== null && (
      <p className='flex items-start mb-4 space-x-2'>
        {alert.type === "error" && (
          <svg
            className='w-6 h-6 flex-none mt-0.5'
            fill='none'
            viewBox='0 0 24 24'
          >
            <circle cx='12' cy='12' r='12' fill='#FECDD3'></circle>
            <path
              d='M8 8l8 8M16 8l-8 8'
              stroke='#B91C1C'
              strokeWidth='2'
            ></path>
          </svg>
        )}
        <p className='flex-1 text-base font-semibold leading-7 text-white'>
          <strong>{alert.msg}</strong>
        </p>
      </p>
    )
  )
}

export default Alert
```

Footer.jsx
Mentor System\src\components\layout\Footer.jsx

```
function Footer() {
  const footerYear = new Date().getFullYear()

  return (
    <footer className='footer p-10 bg-gray-700 text-primary-content footer-center'>
      <div>
        <svg
          width='50'
          height='50'
          viewBox='0 0 24 24'
          xmlns='http://www.w3.org/2000/svg'
          fillRule='evenodd'
          clipRule='evenodd'
          className='inline-block fill-current'
        >
          <path d='M22.672 15.226l-2.432.811.841 2.515c.33 1.019-.209 2.127-1.23 2.456-1.15.325-2.148-
.321-2.463-1.226l-.84-2.518-5.013 1.677.84 2.517c.391 1.203-.434 2.542-1.831 2.542-.88 0-1.601-.564-1.86-
1.314l-.842-2.516-2.431.809c-1.135.328-2.145-.317-2.463-1.229-.329-1.018.211-2.127 1.231-2.456l2.432-
.809-1.621-4.823-2.432.808c-1.355.384-2.558-.59-2.558-1.839 0-.817.509-1.582 1.327-1.846l2.433-.809-.842-
2.515c-.33-1.02.211-2.129 1.232-2.458 1.02-.329 2.13.209 2.461 1.229l.842 2.515 5.011-1.677-.839-2.517c-
.403-1.238.484-2.553 1.843-2.553.819 0 1.585.509 1.85 1.326l.841 2.517 2.431-.81c1.02-.33 2.131.211 2.461
1.229.332 1.018-.21 2.126-1.23 2.456l-2.433.809 1.622 4.823 2.433-.809c1.242-.401 2.557.484 2.557 1.838 0
.819-.51 1.583-1.328 1.847m-8.992-6.428l-5.01 1.675 1.619 4.828 5.011-1.674-1.62-4.829z'></path>
        </svg>
        <p>Copyright &copy; {footerYear} All rights reserved</p>
      </div>
    </footer>
  )
}

export default Footer
```

Navbar.jsx
Mentor System\src\components\layout\Navbar.jsx

```
import { FaChalkboardTeacher, FaHome, FaSearch } from "react-icons/fa"

import { Link, useMatch, useResolvedPath } from "react-router-dom"
import PropTypes from "prop-types"

function Navbar({ title }) {
  return (
    <nav className='navbar mb-12 shadow-lg bg-neutral text-neutral-content'>
      <div className='container mx-auto'>
```

```
        <div className='flex-none px-2 mx-2'>
          <ul>
            <CustomIcon to='/home' className='font-bold align-middle'>
              <FaChalkboardTeacher className='inline pr-2 text-3xl' />

              {title}
            </CustomIcon>
          </ul>
        </div>

        <div className='flex-1 px-2 mx-2'>
          <div className='flex justify-end'>
            <ul>
              <CustomLink to='/home'>
                {" "}
                <FaHome className='mx-2 inline-flex ' />
                HOME
              </CustomLink>

              <CustomLink to='/AddMentee'>Add Mentee</CustomLink>
              <CustomLink to='/Allocation'>Allocation</CustomLink>
              <CustomLink to='/Search'>
                <FaSearch className='inline pr-2 text-3xl' />
                Search mentee{" "}
              </CustomLink>
            </ul>

            <CustomButton to='/'>
              <FaHome className='mx-2 inline-flex ' />
              Logout
            </CustomButton>

            {/* <Link to='/about' className='btn btn-ghost btn-sm rounded-btn'>
              About
            </Link> */}
          </div>
        </div>
      </div>
    </nav>
  )
}

function CustomLink({ to, children, ...props }) {
  const resolvedPath = useResolvedPath(to)
  const loginpage = useResolvedPath("/")

  const isActive = useMatch({ path: resolvedPath.pathname, end: true })
  const isActiveLoginPage = useMatch({ path: loginpage.pathname, end: true })

  return (
    <li
      className={
        isActive
          ? "btn btn-ghost btn-sm rounded-btn btn-active btn-outline btn-disabled m-1"
          : isActiveLoginPage
```

```
          ? "btn btn-ghost btn-sm rounded-btn btn-disabled"
          : "btn btn-ghost btn-sm rounded-btn mx-1"
      }
    >
      <Link to={to} {...props}>
        {children}
      </Link>
    </li>
  )
}

function CustomButton({ to, children, ...props }) {
  const loginpage = useResolvedPath("/")

  const isActiveLoginPage = useMatch({ path: loginpage.pathname, end: true })

  return (
    <li
      className={
        isActiveLoginPage
          ? "btn btn-ghost btn-sm rounded-btn btn-active btn-outline btn-disabled m-1"
          : "btn btn-ghost btn-sm rounded-btn mx-1"
      }
    >
      <Link to={to} {...props}>
        {isActiveLoginPage ? "Login" : "Logout"}
      </Link>
    </li>
  )
}

function CustomIcon({ to, children, ...props }) {
  const loginpage = useResolvedPath("/")

  const isActiveLoginPage = useMatch({ path: loginpage.pathname, end: true })

  return (
    <li
      className={
        isActiveLoginPage
          ? "btn btn-ghost btn-sm rounded-btn btn-disabled"
          : "btn btn-ghost btn-sm rounded-btn mx-1"
      }
    >
      <Link to={to} {...props}>
        {children}
      </Link>
    </li>
  )
}

Navbar.defaultProps = {
  title: "Mentor System",
}
```

```
Navbar.propTypes = {
  title: PropTypes.string,
}

export default Navbar
```

Spinner.jsx
Mentor System\src\components\layout\Spinner.jsx

```
import React from "react"
import spinner from "./assets/spinner.gif"
function Spinner() {
  return (
    <div className='w-100 mt-20'>
      <img
        className='text-center mx-auto'
        src={spinner}
        width={180}
        alt='Loading ...'
      ></img>
    </div>
  )
}

export default Spinner
```

About.jsx
Mentor System\src\Pages\About.jsx

```
function About() {
  return (
    <>
      <h1 className='text-6xl mb-4'>Github Finder</h1>
      <p className='mb-4 text-2xl font-light'>
        A React app to search GitHub profiles and see profile details. This
        project is part of the
        <a href='https://www.udemy.com/course/modern-react-front-to-back/'>
          {" "}
          React Front To Back
        </a>{" "}
        Udemy course by
        <strong>
          <a href='https://traversymedia.com'> Brad Traversy</a>
        </strong>
        .
      </p>
```

```
      <p className='text-lg text-gray-400'>
        Version <span className='text-white'>1.0.0</span>
      </p>
      <p className='text-lg text-gray-400'>
        Layout By:
        <a className='text-white' href='https://twitter.com/hassibmoddasser'>
          Hassib Moddasser
        </a>
      </p>
    </>
  )
}

export default About
```

LoginPage.jsx
Mentor System\src\Pages\LoginPage.jsx

```
import React from "react"

// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

//navigate to the /HOME page
import { useNavigate } from "react-router-dom"

function LoginPage() {
  const DefaultPassword = "123"
  const DefaultEmail = "test@test.com"

  let confirmemail = false

  let confirmPassword = false

  const navigate = useNavigate()

  function checkInputs() {
    // trim to remove the whitespaces
    console.log("checking input")
    const email = document.getElementById("email")
    const password = document.getElementById("password")

    const emailValue = email.value.trim()
    const passwordValue = password.value.trim()
    console.log("emailValue", emailValue)
    console.log(passwordValue)

    if (emailValue === "") {
      toast.error(email, "Email cannot be blank")
```

```
      setErrorFor(email)
    } else if (!isEmail(emailValue)) {
      toast.error(email, "Not a valid email")
      setErrorFor(email)
    } else if (emailValue !== DefaultEmail) {
      toast.error("please check your email")
    } else {
      setSuccessFor(email)
      confirmemail = true
    }

    // password checking function
    if (passwordValue === "") {
      toast.error(password, "Password cannot be blank")
      setErrorFor(password)
    } else if (passwordValue !== DefaultPassword) {
      toast.error(password, "please check your password")
      setErrorFor(password)
    } else {
      setSuccessFor(password)
      confirmPassword = true
    }
  }

  function setErrorFor(input) {
    const formControl = input
    formControl.className = "input input-bordered input-error"
    formControl.value = ""
  }

  function setSuccessFor(input) {
    const formControl = input
    formControl.className = "input input-bordered input-success"
  }

  function isEmail(email) {
    // eslint-disable-next-line no-useless-escape
    return /^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-
9]{1,3}\.[0-9]{1,3}])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/.test(
      email
    )
  }

  const formInputs = (e) => {
    e.preventDefault()
    checkInputs()
    if (confirmemail && confirmPassword) {
      console.log("success")
      navigate("/Home")
    }
  }

  return (
    <>
      <div className='w-full flex justify-center'>
```

```
        <ToastContainer autoClose={3000} />
        {/* <FaChalkboardTeacher className='  mr-5 inline-flex' />
         */}
        <div className='hero-content flex-col lg:flex-row-reverse'>
          <div className='text-center md:text-left'>
            <h1 className='text-5xl font-bold'>Login now ! to Mentor System</h1>
            <p className='py-6'></p>
          </div>{" "}
        </div>

        <div className='card flex-shrink-0 w-full max-w-sm shadow-2xl bg-base-100'>
          <form id='form' class='form' onSubmit={formInputs}>
            <div className='card-body '>
              <div className='form-control'>
                <label className='label'>
                  <span className='label-text'>Email</span>
                </label>
                <input
                  required
                  id='email'
                  type='email'
                  placeholder='email'
                  className='input input-bordered input-info'
                />
              </div>
              <div className='form-control'>
                <label className='label'>
                  <span className='label-text'>Password</span>
                </label>
                <input
                  required
                  id='password'
                  type='password'
                  placeholder='password'
                  className='input input-bordered input-info'
                />
              </div>
              <div className='form-control mt-6'>
                <button type='submit' className='btn btn-primary'>
                  Login
                </button>
              </div>
            </div>
          </form>
        </div>
      </div>
      <br />
      <br />
    </>
  )
}

export default LoginPage
```

NotFound.jsx
Mentor System\src\Pages\NotFound.jsx

```
import { FaHome } from "react-icons/fa"
import { Link } from "react-router-dom"

function NotFound() {
  return (
    <div className='hero'>
      <div className='text-center hero-content'> </div>
      <div className='max-w-lg'>
        <h1 className=' text-8xl font-bold mb-8'>Oops!</h1>
        <p className='text-5xl mb-8'>404 - page not found!</p>
        <Link to='/' className='btn btn-primary btn-lg'>
          <FaHome className='mr-2' /> Go back to Home
        </Link>
      </div>
    </div>
  )
}

export default NotFound
```

Allocation.jsx
Mentor System\src\Pages\Allocation\Allocation.jsx

```
import { useEffect, useState } from "react"
import { useNavigate, Link } from "react-router-dom"
import {
  collection,
  query,
  where,
  getDocs,
  updateDoc,
  doc,
} from "firebase/firestore"
import { db } from "../../firebase.config"

import Spinner from "../../components/layout/Spinner"
import EmptyMenteeData from "./EmptyMenteeData"

//page
import HeaderTitile from "../../components/HeaderTitile"

// tostify
```

```
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function Allocation() {
  const [loading, setLoading] = useState(true)
  const [menteeData, setmenteeData] = useState("")
  const navigate = useNavigate()
  useEffect(() => {
    // get the mentee data where the mentor is not assigned
    const getData = async () => {
      try {
        const q = query(collection(db, "mentee"), where("mentor", "==", ""))

        const menteeFetchData = []
        const querySnapshot = await getDocs(q)

        querySnapshot.forEach((doc) => {
          return menteeFetchData.push({
            ...doc.data(),
            id: doc.id,
          })
        })
        console.log(menteeFetchData)
        menteeFetchData.sort((a, b) => (a.regNo > b.regNo ? 1 : -1))
        setmenteeData(menteeFetchData)
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }

    getData()
  }, [])

  const onClick = (event) => {
    setLoading(true)
    //seting temp ( selected mentee)id value in firebase
    try {
      const userRef = doc(db, "current", "mentee")
      updateDoc(userRef, {
        id: event.target.id,
      })
      setLoading(false)
      navigate("/Allocation2")
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  return (
    <>
      <div className='overflow-x-auto'>
        {loading ? (
          <Spinner />
        ) : menteeData && menteeData.length > 0 ? (
```

```
<>
  <ToastContainer autoClose={3000} />
  <HeaderTitile>
    Allocation page
    <br />
    <br />
    <p className='text-2xl >'>
      <Link className='btn  btn-outline' to='/Automatic' download>
        click here Automatically Allocate mentees
      </Link>
    </p>
  </HeaderTitile>
  <table className='table table-zebra w-full mb-4'>
    {/* <!-- head --> */}
    <thead>
      <tr>
        <th className='border text-xl text-center text-cyan-50'>
          regNo
        </th>
        <th className='border text-xl text-center text-cyan-50'>
          semester
        </th>
        <th className='border text-xl text-center text-cyan-50'>
          rollNo
        </th>
        <th className='border text-xl text-center text-cyan-50'>
          name
        </th>
        <th className='border text-xl text-center text-cyan-50'>
          Allocate
        </th>
      </tr>
    </thead>
    <tbody>
      {/* <!-- row 1 --> */}
      {menteeData.map((item) => (
        <tr key={item.regNo}>
          <td className='border text-lg text-center text-cyan-50'>
            {item.regNo}
          </td>
          <td className='border text-lg text-center text-cyan-50'>
            {item.semester}
          </td>
          <td className='border text-lg text-center text-cyan-50'>
            {item.rollNo}
          </td>
          <td className='border text-lg text-center text-cyan-50'>
            {item.name.toUpperCase()}
          </td>
          <td className='border text-lg text-center text-cyan-50'>
            {
              <button
                id={item.id}
                onClick={onClick}
                className='btn btn-outline btn-accent'
```

```
                        >
                          Allocate
                        </button>
                      }
                    </td>
                  </tr>
                ))}
              </tbody>
            </table>
          </>
        ) : (
          <div className=''>
            <EmptyMenteeData />
          </div>
        )}
      </div>
    </>
  )
}

export default Allocation
```

Allocation2.jsx
Mentor System\src\Pages\Allocation\Allocation2.jsx

```
import { useEffect, useState } from "react"
import { useNavigate } from "react-router-dom"

//firebase
import {
  collection,
  query,
  getDocs,
  getDoc,
  updateDoc,
  doc,
} from "firebase/firestore"
import { db } from "../../firebase.config"

// Components
import EmptyMentorData from "./EmptyMentorData"

import Spinner from "../../components/layout/Spinner"

// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function Allocation2() {
  let menteeID // to store the mentee id from the current table (where pass the value from the
allocation.jsx)
  const [loading, setLoading] = useState(true) // for  storing state loading
```

```
const [mentorData, setmentorData] = useState("") // for storing all the mentor data
// const [mentorSingleData, setmentorSingleData] = useState({
//   name: "",
//   id: "",
//   mobno: "",
//   email: "",
//   noOfMentees: 0,
//   mentees: [],
// }) // for storing Single the mentor data

const [menteeData, setmenteeData] = useState({
  name: "",
  regNo: "",
  rollNo: "",
  semester: "",
  mentorId: "",
  mentor: "",
}) // for storing  the mentee  data (full data of the mentee which id is menteeFetchId )

function isEmpty(val) {
  return val === undefined || val == null || val.length <= 0 ? true : false
}

const navigate = useNavigate()

//   fetch mentor details single ,
//    copy mentor name into mentee fileds
//     (id, name) ,
//      update with fire base mentee ,
//      copy mentee name in to arry and increment no,of student
//      update with fire base mentor ,
//      clear the current variable

useEffect(() => {
  const getmenteeData = async () => {
    // to fetch data (object ) of the (single)mentee
    try {
      const docRef = doc(db, "mentee", menteeID)
      const docSnap = await getDoc(docRef)

      if (docSnap.exists()) {
        let temp = []
        temp = docSnap.data()
        temp.docId = menteeID
        setmenteeData(temp)
      } else {
        console.log("No such document!")
      }
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  // fetch all mentor data
  const getData = async () => {
```

```
    try {
      const q = query(collection(db, "mentor"))
      // console.log("fetch all mentor data   getData")

      const mentorFetchData = []
      const querySnapshot = await getDocs(q)

      querySnapshot.forEach((doc) => {
        return mentorFetchData.push({
          ...doc.data(),
          docId: doc.id,
          // noOfMentees: doc.data().mentees.length,
        })
      })
      console.log(mentorFetchData)

      setmentorData(mentorFetchData)

      await getmenteeData() // call menteeData
      //setLoading(false)
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  // fetch the temp id (selected mentee)
  const getId = async () => {
    // console.log("fetch the temp id")
    try {
      const q2 = query(collection(db, "current"))

      const temp = []
      const querySnapshot2 = await getDocs(q2)

      querySnapshot2.forEach((doc) => {
        return temp.push({
          ...doc.data(),
        })
      })

      menteeID = temp[0].id // eslint-disable-line
      // console.log(temp[0].id)
      //setmenteeFetchId(menteeID)

      await getData()
      setLoading(false)
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  getId()
}, [])

// Allocation part
```

```
  const onClick = async (e) => {
    setLoading(true)

    let mentorid = e.currentTarget.id //  id of selected mentor
    console.log(`mentorid ${mentorid}`)
    let mentorobj = mentorData[mentorid] //  object of selected mentor
    console.log(mentorobj)
    //  object of selected mentor

    // mentee update

    //update mentee data state with mentor and mentor-id field
    setmenteeData((prevState) => ({
      ...prevState,
      mentor: mentorobj.name,
      mentorId: mentorobj.docId,
    }))
    //update mentee data payload with mentor and mentor-id field  ---- useState an async function only
change the end for function so if want to use the updateted value with in the function you would not
choose the useState here i'm using a payload variable
    const payload = {
      ...menteeData,
      mentor: mentorobj.name,
      mentorId: mentorobj.docId,
    }
    // firebase update -- mentee data update  "mentor" ,"mentorId" field
    try {
      const userRef = doc(db, "mentee", payload.docId)
      delete payload.docId
      updateDoc(userRef, payload)
      setLoading(false)
      // navigate("/")
    } catch (error) {
      toast.error("Could not fetch data")
    }

    // mentor update

    //  update mentor data state with mentees array and noOfMentees field
    let prevStateMentees = mentorobj.mentees //  mentor mentees array to check if the mentee array is
already allocated

    // payload for mentor update    // update mentor data payload with mentees array and noOfMentees
field
    let mentorPayload = {}

    isEmpty(prevStateMentees)
      ? (mentorPayload = {
          ...mentorobj,
          noOfMentees: mentorobj.noOfMentees + 1,
          mentees: [`${menteeData.name}`],
        })
      : (mentorPayload = {
          ...mentorobj,
          noOfMentees: mentorobj.noOfMentees + 1,
```

```
        mentees: [...mentorobj.mentees, `${menteeData.name}`],
      })

  console.log(mentorPayload)

  // firebase update -- mentor data update  "mentees"  (array) ,"noOfMentees" field
  try {
    const userRef = doc(db, "mentor", mentorobj.docId)
    delete mentorPayload.docId
    updateDoc(userRef, mentorPayload)
    setLoading(false)
    navigate("/home")
  } catch (error) {
    toast.error("Could not fetch data")
  }
}
return (
  <>
    <div className='overflow-x-auto '>
      {loading ? (
        <Spinner />
      ) : mentorData && mentorData.length > 0 ? (
        <>
          <ToastContainer />
          <table className='table table-zebra w-full mb-4'>
            {/* <!-- head --> */}

            <thead>
              <tr>
                <th className='border text-xl text-center text-cyan-50'>
                  name
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  id
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  email
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  Total mentees
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  Allocate
                </th>
              </tr>
            </thead>
            <tbody>
              {/* <!-- row 1 --> */}
              {mentorData.map((item, index) => (
                <tr key={item.docId}>
                  <td className='border text-lg text-center text-cyan-50'>
                    {item.name.toUpperCase()}
                  </td>
                  <td className='border text-lg text-center text-cyan-50'>
                    {item.id}
```

```
                    </td>
                    <td className='border text-lg text-center text-cyan-50'>
                      {item.email}
                    </td>
                    <td className='border text-lg text-center text-cyan-50'>
                      {" "}
                      {item.noOfMentees}
                    </td>
                    <td className='border text-lg text-center text-cyan-50'>
                      {
                        <button
                          id={index}
                          onClick={(e) => onClick(e)}
                          className='btn btn-outline btn-accent'
                        >
                          Allocate
                        </button>
                      }
                    </td>
                  </tr>
                ))}
              </tbody>
            </table>
          </>
        ) : (
          <div className=''>
            <EmptyMentorData />
          </div>
        )}
      </div>
    </>
  )
}

export default Allocation2
```

Automatic.js
Mentor System\src\Pages\Allocation\Automatic.jsx

```
import { useEffect, useState } from "react"
import { useNavigate } from "react-router-dom"
import {
  collection,
  query,
  where,
  getDocs,
  updateDoc,
  doc,
  getDoc,
} from "firebase/firestore"
```

```
import { db } from "../../firebase.config"

// Components
import EmptyMentorData from "./EmptyMentorData"

import Spinner from "../../components/layout/Spinner"

// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function Automatic() {
  const [loading, setLoading] = useState(true)
  const [menteeData, setmenteeData] = useState("") // for storing all the mentee data
  const navigate = useNavigate()
  let temp = []

  // 2
  const [mentorData, setmentorData] = useState("") // for storing all the mentor data

  useEffect(() => {
    // get the mentee data where the mentor is not assigned
    const getMenteeData = async () => {
      try {
        const q = query(collection(db, "mentee"), where("mentor", "==", ""))

        const menteeFetchData = []
        const querySnapshot = await getDocs(q)

        querySnapshot.forEach((doc) => {
          return menteeFetchData.push({
            ...doc.data(),
            id: doc.id,
          })
        })

        setmenteeData(menteeFetchData)
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }

    getMenteeData() // calling the mentee data function

    // fetch all mentor data
    const getMentorData = async () => {
      try {
        const q = query(collection(db, "mentor"))
        // console.log("fetch all mentor data   getData")

        const mentorFetchData = []
        const querySnapshot = await getDocs(q)

        querySnapshot.forEach((doc) => {
```

```
        return mentorFetchData.push({
          ...doc.data(),
          docId: doc.id,
          noOfMentees: doc.data().mentees.length,
        })
      })
      console.log(mentorFetchData)
      setmentorData(mentorFetchData)

      //setLoading(false)
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  getMentorData() // calling the mentor data function
}, [])

const Allocation = async () => {
  let Mentee = JSON.parse(JSON.stringify(menteeData))

  let Mentor = JSON.parse(JSON.stringify(mentorData))

  Mentee.sort(function (a, b) {
    return a.regNo - b.regNo
  })
  for (let i = 0; i < Mentee.length; i++) {
    // mentees odrder by regNo

    //Mentor.sort((a, b) => (a.noOfMentees > b.noOfMentees ? 1 : -1))

    Mentor.sort(function (x, y) {
      return x.noOfMentees - y.noOfMentees
    })

    // mentors order by noOfMentees
    console.log(Mentor) // odrder by no of mentees
    console.log(Mentor[0]) // odrder by no of mentees

    //Allocation2.jsx

    // function isEmpty(val) {
    //   return val === undefined || val == null || val.length <= 0
    //     ? true
    //     : false
    // }

    let mentorid = 0 //  id of selected mentor    // where here always 0 selected

    let mentorobj = Mentor[mentorid] //  object of selected mentor

    //  object of selected mentor

    //update mentee data state with mentor and mentor-id field
```

```
      //update mentee data payload with mentor and mentor-id field  ---- useState an async function only
change the end for function so if want to use the updateted value with in the function you would not
choose the useState here i'm using a payload variable
      const payload = {
        ...Mentee[i],
        mentor: mentorobj.name,
        mentorId: mentorobj.docId,
      }

      //console.log(payload.id)
      // firebase update -- mentee data update  "mentor" ,"mentorId" field

      const onLoad2 = async () => {
        try {
          const userRef = doc(db, "mentee", payload.id)
          delete payload.id
          await updateDoc(userRef, payload)
          setLoading(false)
        } catch (error) {
          toast.error("Could not fetch data")
        }
      }

      onLoad2()
      // mentor update

      //update mentor data state with mentee and mentee-id field co- poli

      // payload for mentor update    // update mentor data payload with mentees array and noOfMentees
field

      // to fetch data (object ) of the (single)mentor
      try {
        const docRef = doc(db, "mentor", mentorobj.docId)
        const docSnap = await getDoc(docRef)

        if (docSnap.exists()) {
          temp = docSnap.data()
          temp.docId = mentorobj.docId
          console.log(temp)
        } else {
          console.log("No such document!")
        }
      } catch (error) {
        toast.error("Could not fetch data")
      }

      let mentorPayload = {}
      console.log(temp.mentees.length)
      if (temp.mentees.length === 0) {
        mentorPayload = {
          ...temp,
          noOfMentees: temp.noOfMentees + 1,
          mentees: [`${Mentee[i].name}`],
        }
```

```
      //   console.log("hi here 2")
    } else {
      //  console.log("hi here")
      mentorPayload = {
        ...temp,
        noOfMentees: temp.noOfMentees + 1,
        mentees: [...temp.mentees, `${Mentee[i].name}`],
      }
    }
    Mentor[0].noOfMentees = temp.noOfMentees + 1 // add to Mentor object for sorting

    console.log(mentorPayload)

    // firebase update -- mentor data update  "mentees"  (array) ,"noOfMentees" field
    const onLoad = async () => {
      try {
        const userRef = doc(db, "mentor", mentorobj.docId)
        delete mentorPayload.docId
        await updateDoc(userRef, mentorPayload)
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }
    onLoad()
  } // end of for loop
}

const onClick = (e) => {
  Allocation()
  alert("Allocation Done")
  navigate("/home")
}
return (
  <>
    <div className='overflow-x-auto '>
      {loading ? (
        <Spinner />
      ) : mentorData && mentorData.length > 0 ? (
        <>
          <ToastContainer />

          <div className='hero'>
            <div className='text-center  hero-content'> </div>
            <div className='max-w-full'>
              <h5 className=' text-4xl   font-bold mb-8'>
                Click the start button for automatic allocation
                <div className='w-full flex justify-center mt-8'>
                  <div className='  mr-5 inline-flex mt-8' />
                  <button
                    onClick={onClick}
                    className='btn btn-outline btn-wide btn-lg'
                  >
                    start
                  </button>
```

```
                  </div>
                </h5>
              </div>
            </div>
          </>
        ) : (
          <div className=''>
            <EmptyMentorData />
          </div>
        )}
      </div>
    </>
  )
}

export default Automatic
```

EmptyMenteeData.jsx
Mentor System\src\Pages\Allocation\EmptyMenteeData.jsx

```
import { Link } from "react-router-dom"

function EmptyMenteeData() {
  return (
    <div className='hero'>
      <div className='text-center  hero-content'> </div>
      <div className='max-w-full'>
        <h1 className=' text-8xl   font-bold mb-8'>
          <div className='w-full flex justify-center'></div>
          <div>
            <h3 className='text-5xl mt-5 text-center mx-auto'>
              Allocated everyone
            </h3>
          </div>
        </h1>

        <div className='flex-1 px-2 mx-2'>
          <div className='flex justify-center'>
            <Link to='/home' className='mr-2 btn  btn-outline btn-info btn-lg'>
              HOME
            </Link>

            <Link
              to='/AddMentee'
              className='  mr-2 btn  btn-outline btn-info btn-lg'
            >
              Add Mentee
            </Link>
          </div>
        </div>
      </div>
```

```
    </div>
  )
}

export default EmptyMenteeData
```

EmptyMentorData.jsx
Mentor System\src\Pages\Allocation\EmptyMentorData.jsx

```
import { Link } from "react-router-dom"

function EmptyMentorData() {
  return (
    <div className='hero'>
      <div className='text-center  hero-content'> </div>
      <div className='max-w-full'>
        <h1 className=' text-8xl   font-bold mb-8'>
          <div className='w-full flex justify-center'></div>
          <div>
            <h3 className='text-5xl mt-5 text-center mx-auto'>
              Mentor list is empty
            </h3>
          </div>
        </h1>

        <div className='flex-1 px-2 mx-2'>
          <div className='flex justify-center'>
            <Link to='/home' className='mr-2 btn  btn-outline btn-info btn-lg'>
              HOME
            </Link>

            <Link
              to='/AddMentor'
              className=' mr-2 btn  btn-outline btn-info btn-lg'
            >
              Add Mentor
            </Link>
          </div>
        </div>
      </div>
    </div>
  )
}

export default EmptyMentorData
```

FromExcel.jsx
Mentor System\src\Pages\Form\FromExcel.jsx

```
//XLSX
import * as xlsx from "xlsx"

import { useNavigate, Link } from "react-router-dom"
import { db } from "../../firebase.config"
import { setDoc, doc, collection } from "firebase/firestore"
//page
import HeaderTitile from "../../components/HeaderTitile"

// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function FromExcel() {
  let jsonData
  const navigate = useNavigate()
  const readUploadFile = (e) => {
    e.preventDefault()
    if (e.target.files) {
      const reader = new FileReader()
      reader.onload = (e) => {
        const data = e.target.result
        const workbook = xlsx.read(data, { type: "array" })
        const sheetName = workbook.SheetNames[0]
        const worksheet = workbook.Sheets[sheetName]
        jsonData = xlsx.utils.sheet_to_json(worksheet)
        console.log(jsonData.length)
      }
      reader.readAsArrayBuffer(e.target.files[0])
    }
  }

  const onSubmit = (e) => {
    if (jsonData.length) {
      for (let i = 0; i < jsonData.length; i++) {
        console.log(jsonData[i])
        try {
          const mentor = doc(collection(db, "mentee"))
          const formDataCopy = { ...jsonData[i] }
          formDataCopy.mentorId = ""
          formDataCopy.mentor = ""
          formDataCopy.semester = jsonData[i].semester.toString()
          formDataCopy.name = jsonData[i].name.trim()
          formDataCopy.rollNo = jsonData[i].rollNo.toString()
          formDataCopy.regNo = jsonData[i].regNo.toString()
          setDoc(mentor, formDataCopy)
          console.log(formDataCopy)
```

```
        } catch (error) {
          console.log(error)
        }
      }

      navigate("/home")
      alert("Data Uploaded Successfully")
    } else {
      toast.error("Data is Empty")
    }

    e.preventDefault()
  }

  return (
    <>
      <ToastContainer autoClose={3000} />
      <div>
        <HeaderTitile>
          <div className='text-4xl mt-0'>From Excel</div>
          <p className=' mt-4 text-2xl'>
            {" "}
            <Link
              className='btn btn-outline'
              to='/files/Mentee input sheet.xlsx'
              target='_blank'
              download
            >
              click to download sample file
            </Link>
          </p>
          <form className='mx-auto' onSubmit={onSubmit}>
            <label htmlFor='upload' className='text-2xl'>
              Upload modified sample Excel file 👉{" "}
            </label>
            <input
              className='ml-5 text-2xl'
              type='file'
              name='upload'
              id='upload'
              onChange={readUploadFile}
            />
            <button className='btn btn-success' type='submit'>
              Submit
            </button>
          </form>{" "}
        </HeaderTitile>
      </div>
    </>
  )
}

export default FromExcel
```

StudentForm.jsx
Mentor System\src\Pages\Form\StudentForm.jsx

```
import { useState } from "react"
import { useNavigate } from "react-router-dom"
import { db } from "../../firebase.config"
import { setDoc, doc, collection } from "firebase/firestore"
import { Link } from "react-router-dom"
function StudentForm() {
  const [formData, setformData] = useState({
    name: "",
    regNo: "",
    rollNo: "",
    semester: "1",
  })

  const { name, regNo, rollNo, semester } = formData

  const navigate = useNavigate()

  const onChange = (e) => {
    setformData((prevState) => ({
      ...prevState,
      [e.target.id]: e.target.value.toUpperCase().trim(),
    }))
  }

  // send to firebase
  const onSubmit = async (e) => {
    e.preventDefault()
    try {
      const mentor = doc(collection(db, "mentee"))
      const formDataCopy = { ...formData }
      formDataCopy.mentorId = ""
      formDataCopy.mentor = ""
      await setDoc(mentor, formDataCopy)
      console.log(formDataCopy)

      navigate("/home")
    } catch (error) {
      console.log(error)
    }
  }

  return (
    <div className='mt-10 sm:mt-0 '>
      <div className='md:grid md:grid-cols-3 md:gap-6 '>
        <div className='mt-5 md:mt-0  md:col-span-2'>
          <form onSubmit={onSubmit}>
            <div className='w-full pb-3 text-center text-teal-400 text-xl'>
```

```
              <h1>Add Mentee</h1>
            </div>

            <div className='shadow overflow-hidden sm:rounded-md'>
              <div className='px-4 py-5 bg-white sm:p-6'>
                <div className='grid grid-cols-6 gap-6'>
                  <div className='col-span-6 sm:col-span-3'>
                    <label
                      htmlFor='name'
                      className='block text-sm font-medium text-gray-700'
                    >
                      Full name
                    </label>
                    <input
                      required
                      type='text'
                      name='name'
                      id='name'
                      value={name}
                      onChange={onChange}
                      className='mt-1 focus:ring-indigo-500 bg-gray-200 input focus:border-indigo-500
block w-full  shadow-sm sm:text-sm border-gray-300 rounded-md text-black input-sm'
                    />
                  </div>
                  {
                    //-------------------------------------------------------------
                  }
                  <div className='flex justify-center col-span-6 sm:col-span-4'>
                    <div className='mb-3 xl:w-96 '>
                      <label
                        htmlFor='semester'
                        className='block text-sm font-medium text-gray-700'
                      >
                        Semester
                      </label>
                      <select
                        onChange={onChange}
                        value={semester}
                        id='semester'
                        className=' form-select appearance-none block px-3 py-1.5 text-base font-normal
text-gray-700 bg-white bg-clip-padding bg-no-repeat border border-solid border-gray-300 rounded
transition ease-in-outm-0
                              focus:text-gray-700 focus:bg-white focus:border-blue-600 focus:outline-none
'
                        aria-label='1'
                      >
                        {/* <option selected>Open this select menu</option> */}
                        <option select='selected' value='1'>
                          1
                        </option>
                        <option value='2'>2</option>
                        <option value='3'>3</option>
                        <option value='4'>4</option>
                        <option value='5'>5</option>
                        <option value='6'>6</option>
```

```
                    </select>
                  </div>
                </div>
                {
                  //-------------------------------------------------------------
                }
                <div className='col-span-6 sm:col-span-3'>
                  <label
                    htmlFor='regNo'
                    className='block text-sm font-medium text-gray-700'
                  >
                    Register number
                  </label>

                  <input
                    onChange={onChange}
                    type='number'
                    name='regNo'
                    required
                    id='regNo'
                    value={regNo}
                    min='10000000'
                    max='99999999'
                    className=' bg-gray-200 input mt-1 focus:ring-indigo-500 focus:border-indigo-500
block w-full shadow-sm sm:text-sm border-gray-300 rounded-md text-black input-sm'
                  />
                </div>

                {
                  //-------------------------------------------------------------
                }

                <div className='col-span-6 sm:col-span-3'>
                  <label
                    htmlFor='rollNo'
                    className='block text-sm font-medium text-gray-700'
                  >
                    Roll number
                  </label>

                  <input
                    onChange={onChange}
                    type='number'
                    name='rollNo'
                    required
                    id='rollNo'
                    value={rollNo}
                    min='1'
                    max='99'
                    className=' bg-gray-200 input  mt-1  focus:ring-indigo-500 focus:border-indigo-500
block w-full shadow-sm sm:text-sm border-gray-300 rounded-md text-black input-sm'
                  />
                </div>

                {
```

```
                    //-----------------------------------------------------------
                  }
                </div>
              </div>
              <div className='px-4 py-3 bg-gray-50 text-right sm:px-6'>
                <button
                  type='submit'
                  className='inline-flex justify-center py-2 px-4 border border-transparent shadow-sm
text-sm font-medium rounded-md text-white bg-indigo-600 hover:bg-indigo-700 focus:outline-none
focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500'
                >
                  Save
                </button>
              </div>
            </div>
          </form>
        </div>
        <div className='flex mt-9 justify-center'>
          <Link
            to='/FromExcel'
            className=' mr-2 btn  btn-outline btn-accent btn-lg'
          >
            From Excel
          </Link>
        </div>
      </div>
    </div>
  )
}

export default StudentForm
```

TreacherForm.jsx
Mentor System\src\Pages\Form\TreacherForm.jsx

```
import { useState } from "react"
import { useNavigate } from "react-router-dom"
import { db } from "../../firebase.config"
import { setDoc, doc, collection } from "firebase/firestore"

function TreacherForm() {
  const [formData, setformData] = useState({
    name: "",
    id: "",
    mobno: "",
    email: "",
  })

  const { name, email, mobno, id } = formData

  const navigate = useNavigate()
```

```
  const onChange = (e) => {
    setformData((prevState) => ({
      ...prevState,
      [e.target.id]: e.target.value,
    }))
  }

  const onSubmit = async (e) => {
    e.preventDefault()
    try {
      const mentor = doc(collection(db, "mentor"))
      const formDataCopy = { ...formData }
      formDataCopy.noOfMentees = 0
      formDataCopy.mentees = [""]
      await setDoc(mentor, formDataCopy)
      console.log(formDataCopy)

      navigate("/home")
    } catch (error) {
      console.log(error)
    }
  }

  return (
    <div className='mt-10 sm:mt-0 '>
      <div className='md:grid md:grid-cols-3 md:gap-6 '>
        <div className='mt-5 md:mt-0  md:col-span-2'>
          <form onSubmit={onSubmit}>
            <div className='w-full pb-3 text-center text-teal-400 text-xl'>
              <h1>Add Mentor</h1>
            </div>

            <div className='shadow overflow-hidden sm:rounded-md'>
              <div className='px-4 py-5 bg-white sm:p-6'>
                <div className='grid grid-cols-6 gap-6'>
                  <div className='col-span-6 sm:col-span-3 mb-4 '>
                    <label
                      htmlFor='name'
                      className='block text-sm font-medium text-gray-700'
                    >
                      Full name
                    </label>
                    <input
                      required
                      type='text'
                      name='name'
                      value={name}
                      onChange={onChange}
                      id='name'
                      className='mt-1 focus:ring-indigo-500  bg-gray-200 input focus:border-indigo-500
block w-full  shadow-sm sm:text-sm border-gray-300 rounded-md text-black input-sm'
                    />
                  </div>
                  <div className='col-span-6 sm:col-span-3 mb-4'>
```

```
                    <label
                      htmlFor='email'
                      className='block text-sm font-medium text-gray-700'
                    >
                      Email address
                    </label>
                    <input
                      required
                      type='email'
                      onChange={onChange}
                      name='email'
                      value={email}
                      id='email'
                      className='mt-1 focus:ring-indigo-500  bg-gray-200 inputfocus:border-indigo-500
block w-full shadow-sm sm:text-sm border-gray-300 rounded-md text-black input-sm'
                    />
                  </div>
                  <div className='col-span-6 mb-4 sm:col-span-4'>
                    <label
                      htmlFor='mobno'
                      className='block text-sm font-medium   text-gray-700'
                    >
                      Mobile Number
                    </label>

                    <input
                      type='number'
                      name='mobno'
                      value={mobno}
                      onChange={onChange}
                      id='mobno'
                      min='1000000000'
                      max='999999999999'
                      placeholder='888 888 8888'
                      // pattern='[0-9]{3} [0-9]{3} [0-9]{4}'
                      maxLength='12'
                      title='Ten digits code'
                      required
                      className='mt-1 focus:ring-indigo-500  bg-gray-200 inputfocus:border-indigo-500
block w-full shadow-sm sm:text-sm border-gray-300 rounded-md text-black input-sm'
                    />
                  </div>
                  <div className='col-span-6 sm:col-span-3 lg:col-span-2'>
                    <label
                      htmlFor='id'
                      className='block text-sm font-medium text-gray-700'
                    >
                      Id number
                    </label>

                    <input
                      type='number'
                      name='id'
                      value={id}
                      onChange={onChange}
```

```
                      required
                      id='id'
                      min='1'
                      max='99999'
                      className='mt-1 bg-gray-200 input focus:ring-indigo-500 focus:border-indigo-500
block w-full shadow-sm sm:text-sm border-gray-300 rounded-md text-black input-sm'
                    />
                  </div>
                </div>
              </div>
              <div className='px-4 py-3 bg-gray-50 text-right sm:px-6'>
                <button
                  type='submit'
                  className='inline-flex justify-center py-2 px-4 border border-transparent shadow-sm
text-sm font-medium rounded-md text-white bg-indigo-600 hover:bg-indigo-700 focus:outline-none
focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500'
                >
                  Save
                </button>
              </div>
            </div>
          </form>
        </div>
      </div>
    </div>
  )
}

export default TreacherForm
```

AllocatedList.jsx
Mentor System\src\Pages\Report\AllocatedList.jsx

```
import { useEffect, useState } from "react"
import { Link } from "react-router-dom"
import {
  collection,
  query,
  where,
  getDocs,
  doc,
  getDoc,
} from "firebase/firestore"
import { db } from "../../firebase.config"
// components
import HeaderTitile from "../../components/HeaderTitile"
import Spinner from "../../components/layout/Spinner"
//pdf
import jsPDF from "jspdf"
import autoTable from "jspdf-autotable"
// tostify
```

```
import { toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"
//XLSX
import * as XLSX from "xlsx"

function AllocatedList() {
  const [loading, setLoading] = useState(true)
  const [menteeData, setmenteeData] = useState("")

  let tempSemester = ""

  useEffect(() => {
    // get data from firebase where selected semester
    const getData = async () => {
      try {
        const colRef = collection(db, "mentee")

        const q = query(colRef, where("semester", "==", tempSemester))

        let menteeFetchData = []
        const querySnapshot = await getDocs(q)

        querySnapshot.forEach((doc) => {
          return menteeFetchData.push({
            ...doc.data(),
            id: doc.id,
          })
        })
        menteeFetchData = menteeFetchData.filter((item) => item.mentor !== "")
        menteeFetchData.sort((a, b) => (a.regNo > b.regNo ? 1 : -1))
        setmenteeData(menteeFetchData)
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }

    // fetch the temp semester value from firebase
    const getId = async () => {
      // console.log("fetch the temp id")
      try {
        const docRef = doc(db, "current", "semester")
        const docSnap = await getDoc(docRef)

        if (docSnap.exists()) {
          //console.log("Document data:", docSnap.data().semester)
          tempSemester = docSnap.data().semester // eslint-disable-line
        } else {
          // doc.data() will be undefined in this case
          console.log("No such document!")
        }

        console.log(tempSemester)

        await getData()
```

```
      setLoading(false)
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  getId()
}, [])

let exportMenteeData = JSON.parse(JSON.stringify(menteeData))

//console.log(exportMenteeData)
const handleOnExport = () => {
  // delete the id and menterId field from the exportMenteeData
  exportMenteeData.forEach((item, index) => {
    delete exportMenteeData[index].mentorId
    delete exportMenteeData[index].id
    delete exportMenteeData[index].semester
  })

  var wb = XLSX.utils.book_new()
  var ws = XLSX.utils.json_to_sheet(exportMenteeData, {
    header: ["regNo", "rollNo", "name", "mentor"], // ordered cols
  })
  XLSX.utils.book_append_sheet(wb, ws, "Sheet1")
  XLSX.writeFile(wb, "AllocatedList.xlsx")
}

const handleOnPrint = async () => {
  let info = []
  menteeData.forEach((e) => {
    info.push([e.regNo, e.rollNo, e.name, e.mentor])
  })
  console.log(info)

  const head = [["REGISTER NO", "ROLL NO", "NAME", "MENTOR"]]

  const doc = new jsPDF()

  autoTable(doc, {
    head: head,
    body: [...info],
    theme: "grid",
  })

  doc.save("Mentee allocated List.pdf")
}
return (
  <>
    <div className='overflow-x-auto'>
      {loading ? (
        <Spinner />
      ) : menteeData && menteeData.length > 0 ? (
        <>
          <HeaderTitile>
```

```
            <div className=''>Allocated List</div>
          </HeaderTitile>
          <div className='  px-2 mx-2'>
            <div className='flex   justify-end'>
              <button
                id={tempSemester}
                onClick={handleOnExport}
                className='btn  m-3 btn-lg   btn-outline btn-accent'
              >
                Export to Excel
              </button>

              <button
                className='btn btn-outline  m-3 btn-lg btn-error'
                onClick={handleOnPrint}
              >
                Export to PDF
              </button>

              <Link
                to='/EditAllocatedList'
                className='btn btn-outline btn-warning  m-3 btn-lg'
              >
                Edit
              </Link>
            </div>

            <table className='table table-zebra w-full mb-4'>
              {/* <!-- head --> */}

              <thead>
                <tr>
                  <th className='border text-xl text-center text-cyan-50'>
                    regNo
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    rollNo
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    name
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    Mentor
                  </th>
                </tr>
              </thead>
              <tbody>
                {/* <!-- row 1 --> */}
                {menteeData.map((item) => (
                  <tr key={item.regNo}>
                    <td className='border text-lg text-center text-cyan-50'>
                      {item.regNo}
                    </td>

                    <td className='border text-lg text-center text-cyan-50'>
```

```
                    {item.rollNo}
                  </td>

                  <td className='border text-lg text-center text-cyan-50'>
                    {item.name.toUpperCase()}
                  </td>

                  <td className='border text-lg text-center text-cyan-50'>
                    {item.mentor}
                  </td>
                </tr>
              ))}
            </tbody>
          </table>
        </div>
      </>
    ) : (
      <div className='hero'>
        <div className='text-center  hero-content'> </div>
        <div className='max-w-full'>
          <h1 className=' text-8xl   font-bold mb-8'>
            <div className='w-full flex justify-center'></div>
            <div>
              <h3 className='text-5xl mt-5 text-center mx-auto'>
                No one allocated
              </h3>
            </div>
          </h1>

          <div className='flex-1 px-2 mx-2'>
            <div className='flex justify-center'>
              <Link
                to='/home'
                className='mr-2 btn  btn-outline btn-info btn-lg'
              >
                HOME
              </Link>

              <Link
                to='/AddMentee'
                className='  mr-2 btn  btn-outline btn-info btn-lg'
              >
                Add Mentee
              </Link>
            </div>
          </div>
        </div>
      </div>
    )}
  </div>
 </>
)
}

export default AllocatedList
```

Deleteall.jsx
Mentor System\src\Pages\Report\Deleteall.jsx

```
import { useEffect, useState } from "react"
import { useNavigate, Link } from "react-router-dom"
import HeaderTitile from "../../components/HeaderTitile"
//firebase
import {
  collection,
  query,
  getDocs,
  getDoc,
  updateDoc,
  doc,
  where,
  deleteDoc,
} from "firebase/firestore"
import { db } from "../../firebase.config"

import Spinner from "../../components/layout/Spinner"

// confirm alert

import "react-confirm-alert/src/react-confirm-alert.css" // Import css

// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function Deleteall() {
  const [loading, setLoading] = useState(true)
  let [menteeData, setmenteeData] = useState("")
  let mentordata
  let [mentorData, setmentorData] = useState({
    name: "",
    id: "",
    mobno: "",
    email: "",
    noOfMentees: 0,
    mentees: [],
  })
  const navigate = useNavigate()
  let tempSemester = ""
  let useEffectcounter = ""

  // remove item
  // const removeItem = (arr, item) => {
  //   let newArray = [...arr]
  //   const index = newArray.findIndex((element) => element === item)
  //   if (index !== -1) {
  //     newArray.splice(index, 1)
```

```
//      return newArray
//    }
// }
//loading

const SpinnerLoading = () => {
  setLoading(!loading)
}

useEffect(() => {
  // get data from firebase where selected semester
  const getData = async () => {
    try {
      const colRef = collection(db, "mentee")

      const q = query(colRef, where("semester", "==", tempSemester))

      let menteeFetchData = []
      const querySnapshot = await getDocs(q)

      querySnapshot.forEach((doc) => {
        return menteeFetchData.push({
          ...doc.data(),
          id: doc.id,
        })
      })
      menteeFetchData = menteeFetchData.filter((item) => item.mentor !== "")
      menteeFetchData.sort((a, b) => (a.regNo > b.regNo ? 1 : -1))
      setmenteeData(menteeFetchData)
      setLoading(false)
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  // fetch the temp semester value from firebase
  const getId = async () => {
    // console.log("fetch the temp id")
    try {
      const q2 = query(collection(db, "current"))

      const temp = []
      const querySnapshot2 = await getDocs(q2)

      querySnapshot2.forEach((doc) => {
        return temp.push({
          ...doc.data(),
        })
      })

      tempSemester = temp[1].semester // eslint-disable-line
      console.log(tempSemester)

      await getData()
      setLoading(false)
```

```
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }

    getId()
  }, [useEffectcounter])

  // Relocate mentee event listener
  const onClickReallocate = (e) => {
    const docId = e.target.id // mentee id(document id)

    const selectedMenteeObj = menteeData.find(({ id }) => id === docId) // store  selected  mentee
object  data

    SpinnerLoading()

    Reallocate(selectedMenteeObj) // call Reallocate function for  1. Fetch the mentor doc 2.Remove name
from the array 3. Update mentor doc in firebase
    //seting temp ( selected mentee)id value in firebase

    // const newArray = menteeData.mentee   // error in this line menteeData.mentee not working we want
mentorData.mentee
    // //.filter((item) => item !== "ttt")
    // console.log(newArray)
    try {
      const userRef = doc(db, "current", "mentee")
      updateDoc(userRef, {
        id: e.target.id,
      })

      navigate("/Allocation2")
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  // Reallocate mentee
  const Reallocate = async (MeObj) => {
    // fetch the mentor doc
    console.log(MeObj.mentorId)
    const getmentorData = async () => {
      // to fetch data (object ) of the (single)mentor
      try {
        const docRef = doc(db, "mentor", MeObj.mentorId)
        const docSnap = await getDoc(docRef)
        if (docSnap.exists()) {
          let temp = []
          temp = docSnap.data()
          temp.docId = MeObj.mentorId
          setmentorData(temp)
          mentordata = temp
          console.log(temp)
        } else {
          console.log("No such document!")
```

```
      }
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  await getmentorData() // call getmentorData function

  // remove name from the array

  const removeName = () => {
    console.log(mentordata.mentees)
    console.log(MeObj.name)
    let fil = mentordata.mentees.filter((item) => item !== MeObj.name)
    console.log(`fil`)
    console.log(fil)

    setmentorData((prevState) => ({
      ...mentordata,
      mentees: fil,
    }))
  }
  removeName()

  // call removeName function
  console.log(mentorData)

  setTimeout(() => {
    setLoading(false)
  }, 2000)
}

// Delete mentee event listener
//   const onClickConfirmDelete = (e) => {
//     const docId = e.target.id

//     confirmAlert({
//       title: "Confirm to submit",
//       message: "Are you sure to do this.",
//       buttons: [
//         {
//           label: "Yes",
//           onClick: () => onClickDelete(docId),
//         },
//         {
//           label: "No",
//           onClick: () => {
//             toast.error("Delete cancelled")
//           },
//         },
//       ],
//     })
//   }

const onClickDelete = (e) => {
```

```
    const id = e.target.id
    console.log(id)
    const docRef = doc(db, "mentee", id)
    deleteDoc(docRef)
      .then(() => {
        let tempdata = menteeData.filter((item) => item.id !== id)
        console.log(tempdata)
        setmenteeData(tempdata)
      })
      .catch((err) => {
        toast.error("Could not delete" + err.message)
      })
  }

  //  console.log(menteeData)
  return (
    <>
      <div className='overflow-x-auto'>
        {loading ? (
          <Spinner />
        ) : menteeData && menteeData.length > 0 ? (
          <div className='  px-2 mx-2'>
            <HeaderTitile>Edit Allocated List</HeaderTitile>
            <ToastContainer autoClose={3000} />
            <table className='table table-zebra w-full mb-4'>
              {/* <!-- head --> */}

              <thead>
                <tr>
                  <th className='border text-xl text-center text-cyan-50'>
                    regNo
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    rollNo
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    name
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    Mentor
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    Reallocate
                  </th>
                  <th className='border text-xl text-center text-cyan-50'>
                    Delete
                  </th>
                </tr>
              </thead>
              <tbody>
                {/* <!-- row 1 --> */}
                {menteeData.map((item, index) => (
                  <tr key={item.regNo}>
                    <td className='border text-lg text-center text-cyan-50'>
                      {item.regNo}
```

```
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.rollNo}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.name.toUpperCase()}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.mentor}
                </td>
                <td className='border text-lg text-center text-cyan-50'>
                  {
                    <button
                      id={item.id}
                      name={item.name}
                      onClick={(e) => onClickReallocate(e)}
                      className='btn btn-outline btn-accent'
                    >
                      Reallocate
                    </button>
                  }
                </td>
                <td className='border text-lg text-center text-cyan-50'>
                  {
                    <button
                      id={item.id} // DOC ID
                      // onClick={(e) => onClickConfirmDelete(e)}
                      onClick={(e) => onClickDelete(e)}
                      className='btn btn-outline btn-error btn-accent'
                    >
                      Delete
                    </button>
                  }
                </td>
              </tr>
            ))}
          </tbody>
        </table>
      </div>
    ) : (
      <div className='hero'>
        <div className='text-center  hero-content'> </div>
        <div className='max-w-full'>
          <h1 className=' text-8xl   font-bold mb-8'>
            <div className='w-full flex justify-center'></div>
            <div>
              <h3 className='text-5xl mt-5 text-center mx-auto'>
                No one allocated
              </h3>
            </div>
          </h1>
```

```
              <div className='flex-1 px-2 mx-2'>
                <div className='flex justify-center'>
                  <Link
                    to='/'
                    className='mr-2 btn  btn-outline btn-info btn-lg'
                  >
                    HOME
                  </Link>

                  <Link
                    to='/AddMentee'
                    className='  mr-2 btn  btn-outline btn-info btn-lg'
                  >
                    Add Mentee
                  </Link>
                </div>
              </div>
            </div>
          </div>
        )}
      </div>
    </>
  )
}

export default Deleteall
```

EditAllocatedList.jsx
Mentor System\src\Pages\Report\EditAllocatedList.jsx

```
import { useEffect, useState } from "react"
import { useNavigate, Link } from "react-router-dom"
import HeaderTitile from "../../components/HeaderTitile"
//firebase
import {
  collection,
  query,
  getDocs,
  getDoc,
  updateDoc,
  doc,
  where,
  deleteDoc,
} from "firebase/firestore"
import { db } from "../../firebase.config"

import Spinner from "../../components/layout/Spinner"

// confirm alert
import { confirmAlert } from "react-confirm-alert" // Import
import "react-confirm-alert/src/react-confirm-alert.css" // Import css
```

```
// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function EditAllocatedList() {
  const [loading, setLoading] = useState(true)
  let [menteeData, setmenteeData] = useState("")
  let mentordata
  let [mentorData, setmentorData] = useState({
    name: "",
    id: "",
    mobno: "",
    email: "",
    noOfMentees: 0,
    mentees: [],
  })
  const navigate = useNavigate()
  let tempSemester = ""
  let useEffectcounter = ""

  const SpinnerLoading = () => {
    setLoading(!loading)
  }

  useEffect(() => {
    // get data from firebase where selected semester
    const getData = async () => {
      try {
        const colRef = collection(db, "mentee")

        const q = query(colRef, where("semester", "==", tempSemester))

        let menteeFetchData = []
        const querySnapshot = await getDocs(q)

        querySnapshot.forEach((doc) => {
          return menteeFetchData.push({
            ...doc.data(),
            id: doc.id,
          })
        })
        menteeFetchData = menteeFetchData.filter((item) => item.mentor !== "")
        menteeFetchData.sort((a, b) => (a.regNo > b.regNo ? 1 : -1))
        setmenteeData(menteeFetchData)
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }

    // fetch the temp semester value from firebase
    const getId = async () => {
      // console.log("fetch the temp id")
      try {
        const q2 = query(collection(db, "current"))
```

```
        const docRef = doc(db, "current", "semester")
        const docSnap = await getDoc(docRef)

        if (docSnap.exists()) {
          //console.log("Document data:", docSnap.data().semester)
          tempSemester = docSnap.data().semester // eslint-disable-line
        } else {
          // doc.data() will be undefined in this case
          console.log("No such document!")
        }

        const temp = []
        const querySnapshot2 = await getDocs(q2)

        querySnapshot2.forEach((doc) => {
          return temp.push({
            ...doc.data(),
          })
        })

        console.log(tempSemester)

        await getData()
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }

    getId()
  }, [useEffectcounter])

  // Relocate mentee event listener
  const onClickReallocate = (e) => {
    const docId = e.target.id // mentee id(document id)

    const selectedMenteeObj = menteeData.find(({ id }) => id === docId) // store  selected  mentee
object  data

    SpinnerLoading()

    // call Reallocate function for  1. Fetch the mentor doc 2.Remove name from the array 3. Update
mentor doc in firebase
    //seting temp ( selected mentee)id value in firebase

    // const newArray = menteeData.mentee   // error in this line menteeData.mentee not working we want
mentorData.mentee
    // //.filter((item) => item !== "ttt")
    // console.log(newArray)

    // Reallocate mentee
    const Reallocate = async (MeObj) => {
      await DeleteFromArray(MeObj)
```

```
    try {
      const userRef = doc(db, "current", "mentee")
      await updateDoc(userRef, {
        id: e.target.id,
      })

      navigate("/Allocation2")
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  Reallocate(selectedMenteeObj) // call Reallocate mentee function
}
const DeleteFromArray = async (MeObj) => {
  // fetch the mentor doc
  console.log(MeObj)
  const getmentorData = async () => {
    // to fetch data (object ) of the (single)mentor
    try {
      const docRef = doc(db, "mentor", MeObj.mentorId)
      const docSnap = await getDoc(docRef)
      if (docSnap.exists()) {
        let temp = []
        temp = docSnap.data()
        temp.docId = MeObj.mentorId
        setmentorData(temp)
        mentordata = temp
        console.log(temp)
      } else {
        console.log("No such document!")
      }
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  await getmentorData() // call getmentorData function

  // remove name from the array

  const removeName = () => {
    console.log(mentordata.mentees)
    console.log(MeObj.name)
    let fil = mentordata.mentees.filter((item) => item !== MeObj.name)
    console.log(`fil `)
    console.log(fil)

    const mentorPayload = {
      ...mentordata,
      noOfMentees: mentordata.noOfMentees - 1,
      mentees: [...fil],
    }

    console.log("mentorPayload")
```

```
      console.log(mentorPayload)
      let payloadid = mentorPayload.docId
      delete mentorPayload.docId
      console.log(payloadid)

      try {
        const userRef = doc(db, "mentor", payloadid)
        delete mentorPayload.docId
        updateDoc(userRef, mentorPayload)
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }

      setmentorData((prevState) => ({
        ...mentordata,
        mentees: fil,
      }))
    }
    removeName() // remove the name from the array

    setTimeout(() => {
      // call removeName function
      console.log(mentorData)
    }, 1000)

    setTimeout(() => {
      setLoading(false)
    }, 1000)
  }

  // Delete mentee event listener
  const onClickConfirmDelete = (e) => {
    const docId = e.target.id

    confirmAlert({
      title: "Confirm to submit",
      message: "Are you sure to do this.",
      buttons: [
        {
          label: "Yes",
          onClick: () => onClickDelete(docId),
        },
        {
          label: "No",
          onClick: () => {
            toast.error("Delete cancelled")
          },
        },
      ],
    })
  }

  const onClickDelete = async (docId) => {
    //const docId = e.target.id // mentee id(document id)
```

```
  console.log(docId)
  const selectedMenteeObj = menteeData.find(({ id }) => id === docId)

  await DeleteFromArray(selectedMenteeObj)
  const docRef = doc(db, "mentee", docId)
  deleteDoc(docRef)
    .then(() => {
      let tempdata = menteeData.filter((item) => item.id !== docId)
      console.log(tempdata)
      setmenteeData(tempdata)
    })
    .catch((err) => {
      toast.error("Could not delete" + err.message)
    })
}

//  console.log(menteeData)
return (
  <>
    <div className='overflow-x-auto'>
      {loading ? (
        <Spinner />
      ) : menteeData && menteeData.length > 0 ? (
        <div className='  px-2 mx-2'>
          {" "}
          <HeaderTitile>
            <div className='text-4xl'>Edit Allocated List</div>
          </HeaderTitile>
          <ToastContainer autoClose={3000} />
          <table className='table table-zebra w-full mb-4'>
            {/* <!-- head --> */}

            <thead>
              <tr>
                <th className='border text-xl text-center text-cyan-50'>
                  regNo
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  rollNo
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  name
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  Mentor
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  Reallocate
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  Delete
                </th>
              </tr>
            </thead>
            <tbody>
```

```
            {/* <!-- row 1 --> */}
            {menteeData.map((item, index) => (
              <tr key={item.regNo}>
                <td className='border text-lg text-center text-cyan-50'>
                  {item.regNo}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.rollNo}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.name.toUpperCase()}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.mentor}
                </td>
                <td className='border text-lg text-center text-cyan-50'>
                  {
                    <button
                      id={item.id}
                      name={item.name}
                      onClick={(e) => onClickReallocate(e)}
                      className='btn btn-outline btn-accent'
                    >
                      Reallocate
                    </button>
                  }
                </td>
                <td className='border text-lg text-center text-cyan-50'>
                  {
                    <button
                      id={item.id} // DOC ID
                      onClick={(e) => onClickConfirmDelete(e)}
                      className='btn btn-outline btn-error btn-accent'
                    >
                      Delete
                    </button>
                  }
                </td>
              </tr>
            ))}
          </tbody>
        </table>
      </div>
    ) : (
      <div className='hero'>
        <div className='text-center  hero-content'> </div>
        <div className='max-w-full'>
          <h1 className=' text-8xl   font-bold mb-8'>
            <div className='w-full flex justify-center'></div>
            <div>
              <h3 className='text-5xl mt-5 text-center mx-auto'>
                No one allocated
```

```
                </h3>
              </div>
            </h1>

            <div className='flex-1 px-2 mx-2'>
              <div className='flex justify-center'>
                <Link
                  to='/home'
                  className='mr-2 btn  btn-outline btn-info btn-lg'
                >
                  HOME
                </Link>

                <Link
                  to='/AddMentee'
                  className='  mr-2 btn  btn-outline btn-info btn-lg'
                >
                  Add Mentee
                </Link>
              </div>
            </div>
          </div>
        </div>
      )}
    </div>
  </>
 )
}

export default EditAllocatedList
```

MentorList.jsx
Mentor System\src\Pages\Report\MentorList.jsx

```
import { useEffect, useState } from "react"

//firebase
import { collection, query, getDocs } from "firebase/firestore"
import { db } from "../../firebase.config"

// Components
import EmptyMentorData from "../Allocation/EmptyMentorData"

import Spinner from "../../components/layout/Spinner"

// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function MentorList() {
  const [loading, setLoading] = useState(true) // for  storing state loading
  const [mentorData, setmentorData] = useState("")
```

```
useEffect(() => {
  // fetch all mentor data
  const getData = async () => {
    try {
      const q = query(collection(db, "mentor"))
      // console.log("fetch all mentor data   getData")

      const mentorFetchData = []
      const querySnapshot = await getDocs(q)

      querySnapshot.forEach((doc) => {
        return mentorFetchData.push({
          ...doc.data(),
          docId: doc.id,
          // noOfMentees: doc.data().mentees.length,
        })
      })
      console.log(mentorFetchData)

      setmentorData(mentorFetchData)
      setLoading(false)

      // call menteeData
      //setLoading(false)
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  getData() // calling the mentee data function
}, [])

return (
  <>
    <div className='overflow-x-auto '>
      {loading ? (
        <Spinner />
      ) : mentorData && mentorData.length > 0 ? (
        <>
          <ToastContainer />
          <table className='table table-zebra w-full mb-4'>
            {/* <!-- head --> */}

            <thead>
              <tr>
                <th className='border text-xl text-center text-cyan-50'>
                  name
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  id
                </th>
                <th className='border text-xl text-center text-cyan-50'>
                  email
                </th>
```

```
                <th className='border text-xl text-center text-cyan-50'>
                  Total mentees
                </th>
                {/* <th className='border text-xl text-center text-cyan-50'>
                  Allocate
                </th> */}
              </tr>
            </thead>
            <tbody>
              {/* <!-- row 1 --> */}
              {mentorData.map((item, index) => (
                <tr key={item.docId}>
                  <td className='border text-lg text-center text-cyan-50'>
                    {item.name.toUpperCase()}
                  </td>
                  <td className='border text-lg text-center text-cyan-50'>
                    {item.id}
                  </td>
                  <td className='border text-lg text-center text-cyan-50'>
                    {item.email}
                  </td>
                  <td className='border text-lg text-center text-cyan-50'>
                    {" "}
                    {item.noOfMentees}
                  </td>
                </tr>
              ))}
            </tbody>
          </table>
        </>
      ) : (
        <div className=''>
          <EmptyMentorData />
        </div>
      )}
    </div>
  </>
 )
}

export default MentorList
```

SearchView.jsx
Mentor System\src\Pages\Report\SearchView.jsx

```
// components
import HeaderTitile from "../../components/HeaderTitile"
import Spinner from "../../components/layout/Spinner"
//firebase
import { useEffect, useState } from "react"
import { Link } from "react-router-dom"
import {
```

```
  collection,
  query,
  where,
  getDocs,
  doc,
  getDoc,
  orderBy,
  limit,
} from "firebase/firestore"
import { db } from "../../firebase.config"

// tostify
import { toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function SearchView() {
  const [menteeData, setmenteeData] = useState("")
  const [loading, setLoading] = useState(true)

  let tempSemester = ""

  useEffect(() => {
    // get data from firebase where selected semester
    const getData = async () => {
      console.log(`tempSemester ${tempSemester}`)
      try {
        const colRef = collection(db, "mentee")

        const q = query(
          colRef,
          orderBy("name"),
          limit(1),
          where("name", ">=", tempSemester.toUpperCase())
        )
        let menteeFetchData = []
        const querySnapshot = await getDocs(q)

        querySnapshot.forEach((doc) => {
          return menteeFetchData.push({
            ...doc.data(),
            id: doc.id,
          })
        })
        menteeFetchData = menteeFetchData.filter((item) => item.mentor !== "")

        menteeFetchData.sort((a, b) => (a.regNo > b.regNo ? 1 : -1))
        console.log(menteeFetchData)
        setmenteeData(menteeFetchData)
        setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }

    // fetch the temp name value from firebase
```

```
  const getId = async () => {
    try {
      const docRef = doc(db, "current", "search")
      const docSnap = await getDoc(docRef)

      if (docSnap.exists()) {
        //console.log("Document data:", docSnap.data().semester)
        tempSemester = docSnap.data().name // eslint-disable-line
      } else {
        // doc.data() will be undefined in this case
        console.log("No such document!")
      }

      console.log(tempSemester)

      await getData()
      //setLoading(false)
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  getId()
}, [])

return (
  <>
    <>
      <div className='overflow-x-auto'>
        {loading ? (
          <Spinner />
        ) : menteeData && menteeData.length > 0 ? (
          <>
            <HeaderTitile>
              <div className=''>Mentee details</div>
            </HeaderTitile>
            <div className='  px-2 mx-2'>
              <table className='table table-zebra w-full mb-4'>
                {/* <!-- head --> */}

                <thead>
                  <tr>
                    <th className='border text-xl text-center text-cyan-50'>
                      semester
                    </th>
                    <th className='border text-xl text-center text-cyan-50'>
                      reg No
                    </th>
                    <th className='border text-xl text-center text-cyan-50'>
                      roll No
                    </th>
                    <th className='border text-xl text-center text-cyan-50'>
                      name
                    </th>
                    <th className='border text-xl text-center text-cyan-50'>
```

```
                Mentor
              </th>
            </tr>
          </thead>
          <tbody>
            {/* <!-- row 1 --> */}
            {menteeData.map((item) => (
              <tr key={item.regNo}>
                <td className='border text-lg text-center text-cyan-50'>
                  {item.semester}
                </td>
                <td className='border text-lg text-center text-cyan-50'>
                  {item.regNo}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.rollNo}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.name.toUpperCase()}
                </td>

                <td className='border text-lg text-center text-cyan-50'>
                  {item.mentor}
                </td>
              </tr>
            ))}
          </tbody>
        </table>
      </div>
    </>
  ) : (
    <div className='hero'>
      <div className='text-center  hero-content'> </div>
      <div className='max-w-full'>
        <h1 className=' text-8xl   font-bold mb-8'>
          <div className='w-full flex justify-center'></div>
          <div>
            <h3 className='text-5xl mt-5 text-center mx-auto'>
              No student in this name{console.log(tempSemester)}
            </h3>
          </div>
        </h1>

        <div className='flex-1 px-2 mx-2'>
          <div className='flex justify-center'>
            <Link
              to='/home'
              className='mr-2 btn  btn-outline btn-info btn-lg'
            >
              HOME
            </Link>

            <Link
```

```
                    to='/AddMentee'
                    className='  mr-2 btn  btn-outline btn-info btn-lg'
                  >
                    Add Mentee
                  </Link>
                </div>
              </div>
            </div>
          </div>
        )}
      </div>
    </>
  </>
)
}

export default SearchView
```

SelectMenuAllocatedList.jsx
Mentor System\src\Pages\Report\SelectMenuAllocatedList.jsx

```
import { useState } from "react"
import { useNavigate } from "react-router-dom"
import { toast } from "react-toastify"
//firebase
import { updateDoc, doc } from "firebase/firestore"
import { db } from "../../firebase.config"

// Components
import HeaderTitile from "../../components/HeaderTitile"

function SelectMenuAllocatedList() {
  const navigate = useNavigate()

  const [formData, setformData] = useState({
    semester: "1",
  })

  const { semester } = formData

  const [loading, setLoading] = useState(true)

  const onChange = (e) => {
    // send to firebase
    setformData((prevState) => ({
      ...prevState,
      [e.target.id]: e.target.value,
    }))
  }
  console.log(loading) // unuserd variyable no-unused-vars
```

```
  const handleSubmit = (e) => {
    e.preventDefault()
    setLoading(true)
    //seting temp ( selected mentee)id value in firebase
    try {
      const userRef = doc(db, "current", "semester")
      console.log(semester)
      updateDoc(userRef, {
        semester: semester,
      })
      setLoading(false)
      // setformData({ semester: semester })
      navigate("/AllocatedList")
    } catch (error) {
      toast.error("Could not fetch data")
    }
  }

  return (
    <div className=''>
      <HeaderTitile>Select Semester</HeaderTitile>
      <form onSubmit={handleSubmit}>
        <div className='flex justify-center'>
          <div className=''>
            <select
              className='select mr-3  select-lg select-primary '
              value={semester}
              id='semester'
              onChange={onChange}
            >
              <option disabled select='selected'>
                Select semester ?
              </option>
              <option value='1'>Semester 1</option>
              <option value='2'>Semester 2</option>
              <option value='3'>Semester 3</option>
              <option value='4'>Semester 4</option>
              <option value='5'>Semester 5</option>
              <option value='6'>Semester 6</option>
            </select>
          </div>
          <div className=''>
            <button type='submit' className='btn btn-lg'>
              Go
            </button>
          </div>
        </div>
      </form>
    </div>
  )
}

export default SelectMenuAllocatedList
```

UserSearch.jsx
Mentor System\src\Pages\Report\UserSearch.jsx

```
import { useState, useEffect } from "react"
import { useNavigate } from "react-router-dom"
import {
  collection,
  query,
  limit,
  orderBy,
  getDocs,
  where,
  doc,
  updateDoc,
} from "firebase/firestore"
import { db } from "../../firebase.config"

// components
import HeaderTitile from "../../components/HeaderTitile"

// tostify
import { ToastContainer, toast } from "react-toastify"
import "react-toastify/dist/ReactToastify.css"

function UserSearch() {
  const navigate = useNavigate()

  //firebase

  const [value, setValue] = useState("")

  const [menteeData, setmenteeData] = useState([])
  // const [loading, setLoading] = useState(true) // for  storing state loading

  useEffect(() => {
    const getData = async () => {
      if (value === "") {
        console.log("value is empty")
      } else {
        try {
          const colRef = collection(db, "mentee")

          const q = query(
            colRef,
            orderBy("name"),
            limit(6),
            where("name", ">=", value.toUpperCase())
          )

          let menteeFetchData = []
          const querySnapshot = await getDocs(q)
```

```
        querySnapshot.forEach((doc) => {
          return menteeFetchData.push({
            ...doc.data(),
            id: doc.id,
          })
        })
        menteeFetchData = menteeFetchData.filter((item) => item.mentor !== "")
        //  menteeFetchData.sort((a, b) => (a.name > b.name ? 1 : -1))

        // list()

        menteeFetchData = menteeFetchData.filter((item) => {
          const searchTerm = value.toLowerCase()
          const fullName = item.name.toLowerCase()

          return (
            searchTerm &&
            fullName.startsWith(searchTerm) &&
            fullName !== searchTerm
          )
        })

        setmenteeData(menteeFetchData)
        // setLoading(false)
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }
  }
  getData()
  // calling the mentee data function
}, [value])

// filteredData = JSON.parse(JSON.stringify(menteeData))

// input value change event
const onChange = (event) => {
  setValue(event.target.value)
}

// input value submit eventt
const onSearch = (e) => {
  e.preventDefault()
  const searchTerm = document.getElementById("inputText").value.trim()

  if (searchTerm === "") {
    alert("please enter something", "error")
  } else {
    // searchUsers(text)

    setValue(searchTerm)
    console.log("search ", searchTerm)

    e.preventDefault()
```

```
    // const searchTerm = document.getElementById("inputText").value

    console.log(menteeData)
    if (searchTerm === "") {
      alert("please enter something")
    } else {
      // searchUsers(text)

      setValue(searchTerm)
      console.log("search ", searchTerm)

      //seting temp ( selected mentee)id value in firebase
      try {
        const userRef = doc(db, "current", "search")
        updateDoc(userRef, {
          name: searchTerm,
        })

        navigate("/SearchView")
      } catch (error) {
        toast.error("Could not fetch data")
      }
    }
  }
}

return (
  <>
    <ToastContainer autoClose={3000} />
    <div className='grid grid-cols-1 xl:grid-cols-2 lg:grid-cols-2 md:grid-cols-2 mb-8 gap-8'>
      <HeaderTitile>
        <div className=''>Find Mentee</div>
      </HeaderTitile>
      <div>
        <form onSubmit={(e) => onSearch(e)}>
          <div className='form-control '>
            <div className='dropdown dropdown-open absolute'>
              <input
                id='inputText'
                type='text'
                className=' pr-40   w-50 bg-gray-200 input input-lg inset-x-px text-black'
                placeholder='Search'
                value={value}
                onChange={onChange}
              />

              {menteeData.map((item) => (
                <ul
                  tabIndex='0'
                  class=' menu p-2  mt-0 shadow bg-base-100
                  rounded-box'
                >
                  <li
                    className='dropdown-item'
                    onClick={() => setValue(item.name)}
```

```
                    // className='dropdown-row'
                    key={item.name}
                  >
                    {item.name}
                  </li>
                </ul>
              ))}
              <button
                type='submit'
                className='absolute top-0 right-0 rounded-l-none w-36 btn btn-lg'
                onClick={() => onSearch(value)}
              >
                Go
              </button>
            </div>
          </div>
        </form>
      </div>
      {/* {10 >
        // 10 = users.length
        0 && (
        <div>
          <button
            onClick={handleClearBtn}
            className='btn absolute  btn-ghost btn-lg'
          >
            Clear
          </button>
        </div>
      )} */}
    </div>

    <div class=''>
      {/* <label tabindex='0' class='btn m-1'>
        Click
      </label> */}
    </div>
  </>
 )
}

export default UserSearch
```

package.json
Mentor System\package.json

```
{
  "name": "github-finder",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.16.2",
    "@testing-library/react": "^12.1.3",
    "@testing-library/user-event": "^13.5.0",

    "daisyui": "^2.8.0",
    "firebase": "^9.6.8",
    "jspdf": "^2.5.1",
    "jspdf-autotable": "^3.5.25",
    "react": "^17.0.2",
    "react-confirm-alert": "^2.8.0",
    "react-dom": "^17.0.2",
    "react-icons": "^4.3.1",
    "react-lodash": "^0.1.2",
    "react-router-dom": "^6.2.1",
    "react-scripts": "5.0.0",
    "react-toastify": "^8.2.0",
    "web-vitals": "^2.1.4",
    "xlsx": "^0.18.5"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  },
```

```
  "devDependencies": {
    "autoprefixer": "^10.4.2",
    "postcss": "^8.4.6",
    "tailwindcss": "^3.0.23"
  }
}
```

tailwind.config.js
Mentor System\tailwind.config.js

```
module.exports = {
  content: ["./src/**/*.{js,jsx,ts,tsx}"],
  theme: {
    extend: {},
  },
  plugins: [require("daisyui")],
}
```

Team Members

SAINATH A
Coding

JAYAKRISHNAN KP
Testing

MUHAMMED MUNSHID V
Documentation