

REPORT: CSN291 GROUP PROJECT

GROUP 22

PROJECT NAME: STUDENT PORTAL

MEMBER DETAILS:

1. Shreshtha Misra
2. Shabnam Khan
3. Dighiya Sanidhya Ramjiprasad
4. Anand Chourasia
5. Jyoti Chauhan
6. Someshvari Vraj

APPLICATION:

- ❖ Helpful for managing of schedules of students, teachers, etc.
- ❖ Get reminders about upcoming activities.
- ❖ Professors can see availabilities of slots so that no multiple deadlines are planned on the same slots.

INNOVATION:

The year 2020 had a huge impact on the world of academics, besides everything else. The sudden shift to online methodologies of teaching has led to many drawbacks, one of them being the inconvenience students must deal with when keeping track of their daily schedules, assignment submission deadlines, tests and examinations. Different professors use different platforms to announce and collect their announcements, and several deadlines piling up on the same day makes it quite easy to miss out some of them. In addition to this, regular class tests and quizzes make students' schedules even more hard to follow.

Thus, our Student Portal aims to help solve this make schedules less cluttered. It can be used by Professors to check students' schedules before putting out the notice of a test/assignment, so that clashing of several deadlines at the same time is avoided. This application will also give students a list of their upcoming assignments/exam, with the relevant topics and lectures slides/videos for each. It will notify students about all important academic events at one place. The customizable to-do list will enable students to keep a list of all that they want to complete in the upcoming days. Checking email, teams, Moodle etc. could be condensed to checking just one application.

IMPLEMENTATIONS

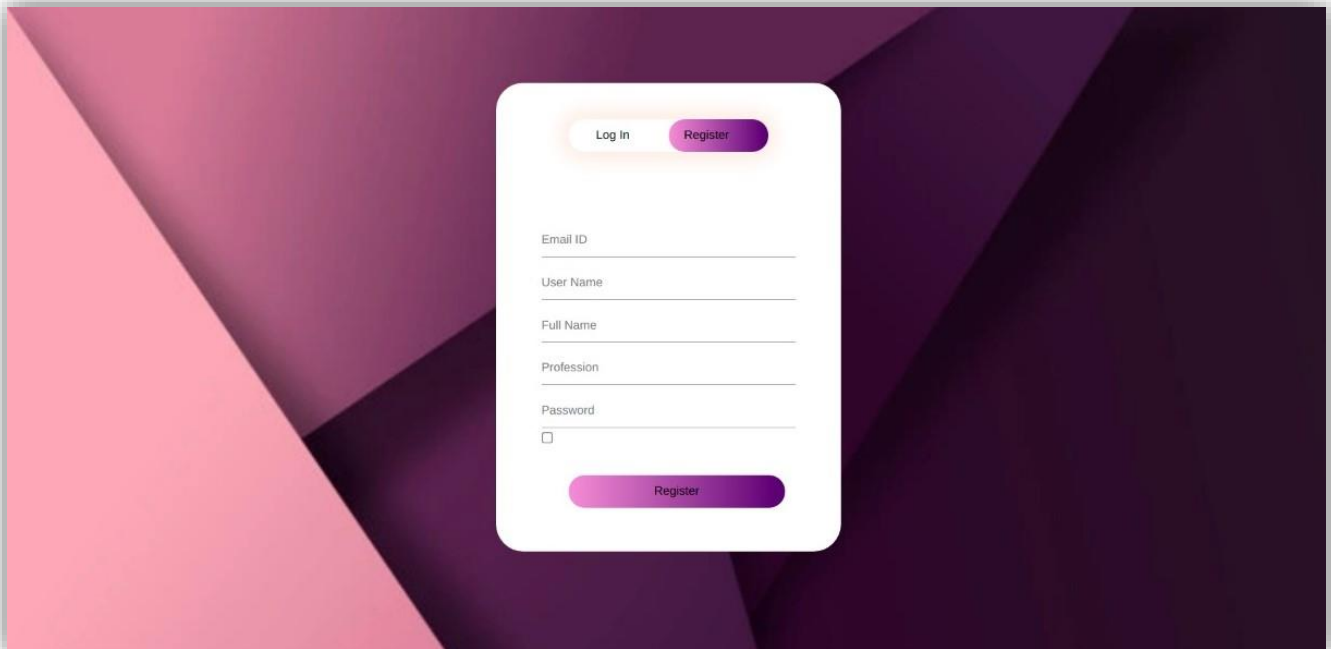
Frontend:

Frontend is built using react.js.

To communicate with backend axios router is used.

1.REGISTER PAGE:

User can create account by entering required information. Once register button clicked, user information will stored be in a noSQL database provided by mongoDB.



The image shows a registration form on a dark purple background with a pink geometric shape on the left. The form is a white rounded rectangle containing a 'Log In' button and a 'Register' button at the top. Below these are input fields for 'Email ID', 'User Name', 'Full Name', 'Profession', and 'Password'. There is a checkbox below the password field and a 'Register' button at the bottom.

[Log In](#) [Register](#)

Email ID

User Name

Full Name

Profession

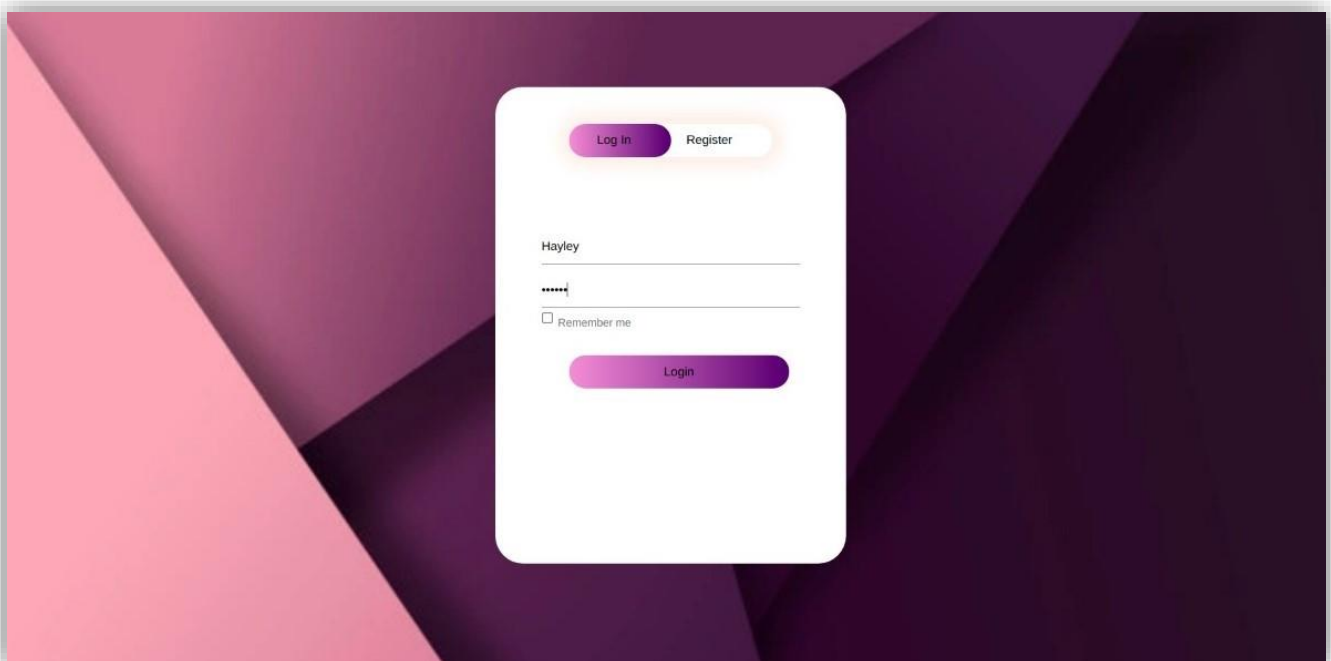
Password

☐

[Register](#)

2.LOGIN PAGE:

In this page we are asked about user name and password. Once login button is clicked, the website will authenticate the user with information stored in mongoDB. Then user is directed to the home page.



The image shows a login form on a dark purple background with a pink geometric shape on the left. The form is a white rounded rectangle containing a 'Log In' button and a 'Register' button at the top. Below these are input fields for 'Hayley' and a password field with masked characters. There is a 'Remember me' checkbox and a 'Login' button at the bottom.

[Log In](#) [Register](#)

Hayley

☐ Remember me

[Login](#)

3.HOME PAGE:

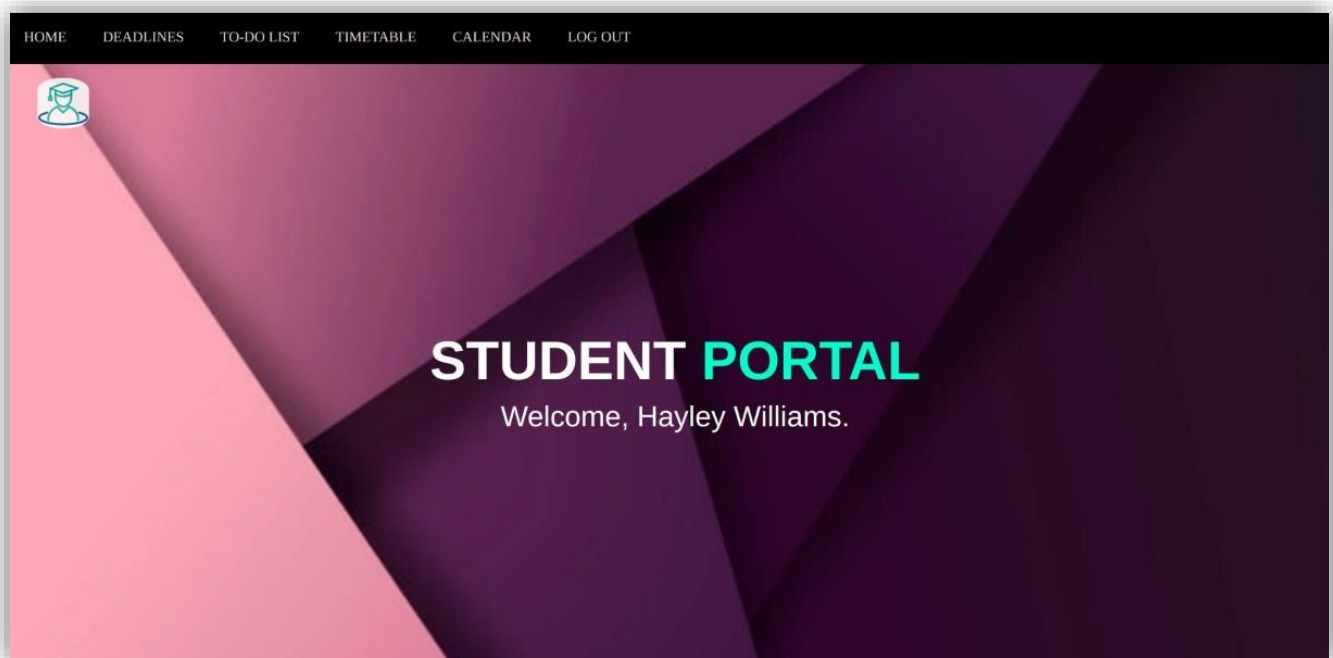
There are many features in navigation bar of home page:

1.DEADLINE

2.TO-DO-LIST

3.TIME TABLE

4.CALENDER



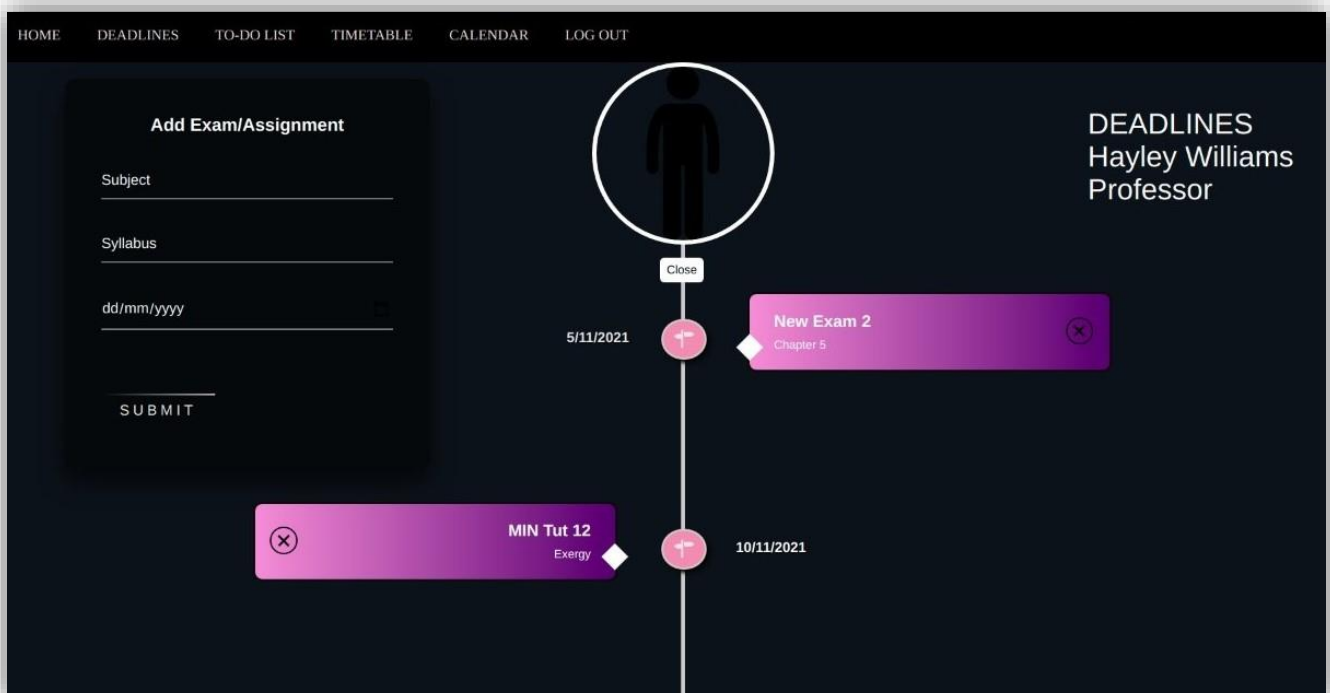
4.DEADLINE:

In this page user can see upcoming deadline. And professor can add any task to the list of deadlines.



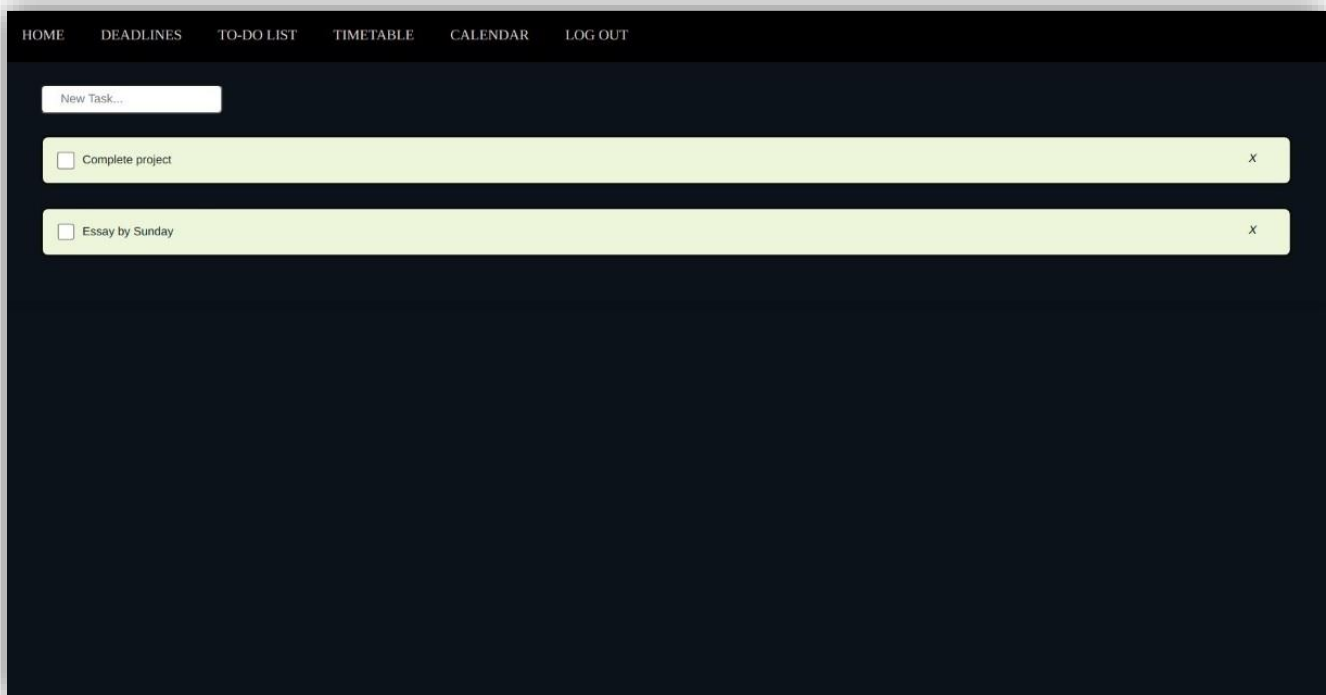
The 'add' button displays a form that 'on submitting' sends the entered information as an object to the backend via axios router (making a post request) and the backend after verifying the request creates a new deadline.

This feature is not available to any student.



5.TO-DO-LIST:

A user (student or professor) can add any task to their to do list. The list contains an input section which sends a post request to the server, and if the request is successfully fulfilled, the page refreshes and the user sees the new task in their to-do list. The user can also delete a particular task by clicking on the 'x' mark that removes the object from the database by sending a 'delete' request.



6.TIME TABLE: In this page user can see their updated time table. And professor also can see time table and can arrange extra classes as per need.

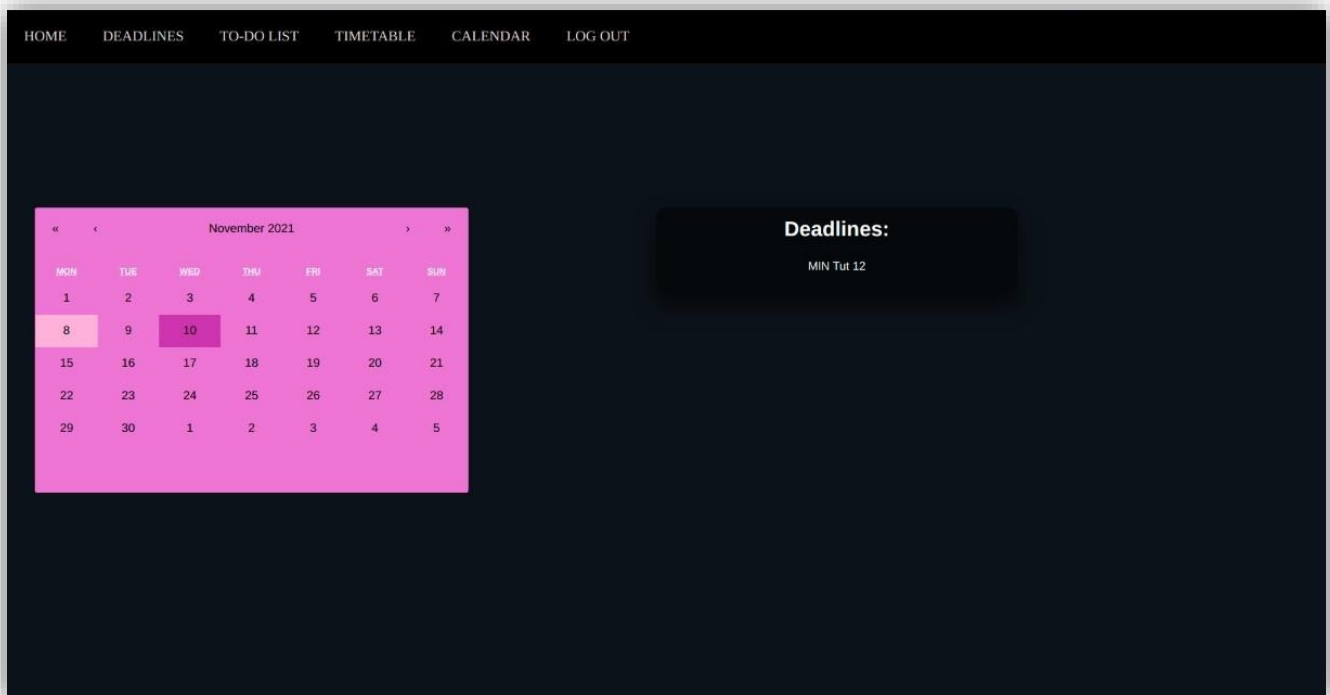
HOME DEADLINES TO-DO LIST TIMETABLE CALENDAR LOG OUT

My timetable

	Monday	Tuesday	Wednesday	thursday	Friday
8:00 AM - 8:55 AM	-	L-HSN-01 Batch-3	-	L-HSN-01 Batch-3	-
9:00 AM - 9:55 AM	L-MIN-106 SS	L-MIN-106 SS	-	L-MIN-106 SS	-
10:00 AM - 10:55 AM	L-CSN-221 PSK	-	L-CSN-221 PSK	-	-
11:05 AM - 12:00 PM	L-ECN-203 SK	-	L-ECN-203 SK	L-CSN-221 PSK	L-ECN-203 SK
12:05 PM - 1:00 PM	L-CSN-291 SK	L-CSN-291 SK	-	L-CSN-291 SK	-
1:00 PM - 2:00 PM	Lunch				
2:00 PM - 2:55 PM	T-CSN-221 PSK	P-CSN-291 SK	-	-	T-CSN-221 PSK
3:00 PM - 3:55 PM	T-MIN-106 O1,O2	P-CSN-291 SK	P-CSN-261 RT	T-MIN-106 O4	T-MIN-106 O3
4:05 PM - 5:00 PM	T-HSN-01 -	P-MIN-106 O1,O2	-	P-MIN-106 O3,O4	T-HSN-01 -
5:05 PM - 6:00 PM	T-HSN-01 -	P-MIN-106 O1	-	P-MIN-106 O3,O4	T-HSN-01 -

7. CALENDAR:

The calendar page displays an interactive calendar that shows the list of activities to be complete on that day. It is an interactive calendar built using 'react-calendar' that on the particular date chose, shows a list of the deadlines on that day. This is particularly useful for professors to see how many deadlines have already been declared on a particular day, which will help them keep students' convenience in mind while deciding a schedule for their assignments.



BACKEND:

The portal_backend folder contains 6 folders in it:

- 1) Controllers
- 2) Models
- 3) Public
- 4) Requests
- 5) Tests
- 6) Utils

- The Controllers folder contains Routers like the examRouter, loginRouter etc. which can make requests (GET, PUT, POST, DELETE) to the Database to interact with the data stored- to get that data, to change that data, to delete that data or to add new data to it.
- When we make a POST request that request bears an authentication bearer token which will verify that the user who is sending the request is a valid user or not.
- The Models folder contains all the schema for defining the collections.

Example:

```
const examSchema = new mongoose.Schema({
  subject: {
    type: String,
    required: true,
    minlength: 5
  },
  date: {
    type: String,
    required: true,
  },
  syllabus : {
```

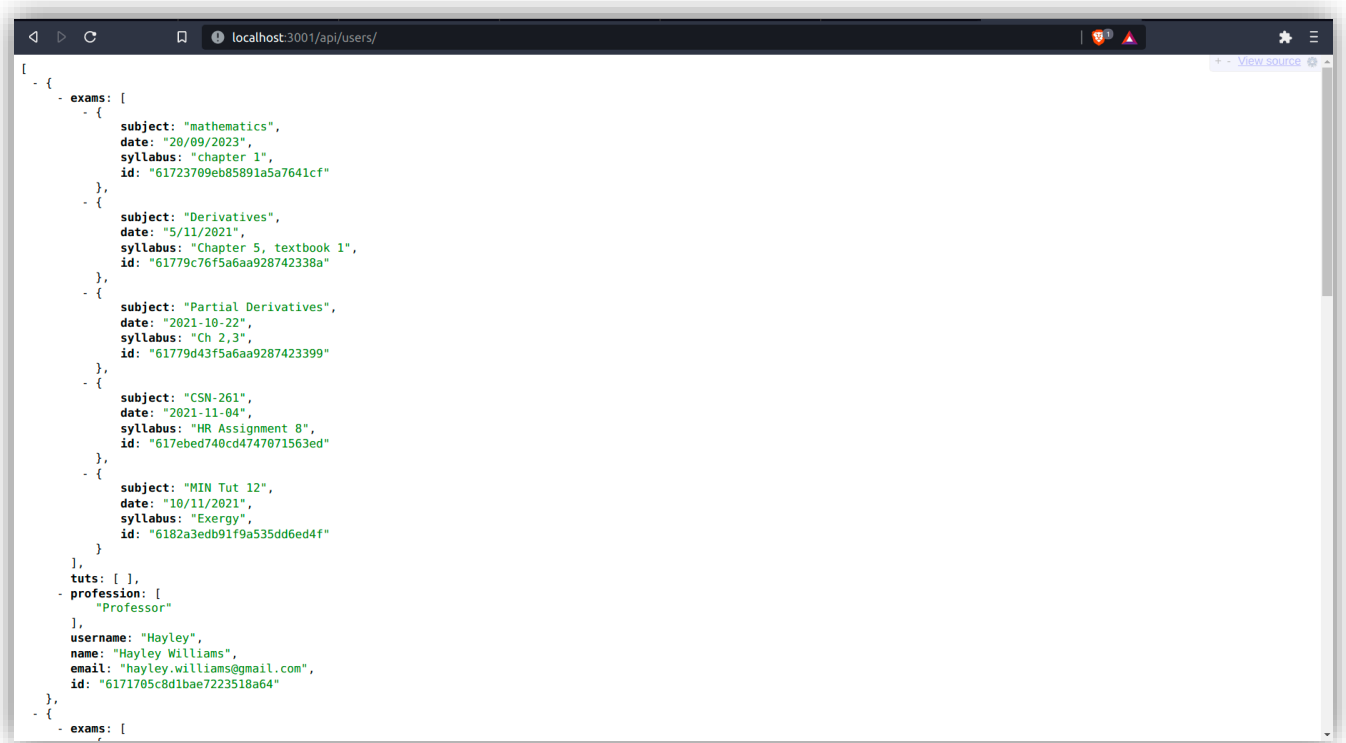


```
    type : String,  
    required:true,  
    minlength:5  
  },  
  user: {  
    type: mongoose.Schema.Types.ObjectId,  
    ref: 'User'  
  }  
})
```

- The portal_backend/requests folder uses vscode rest client to make requests to the database to test if it responds correctly. Through vscode rest client we can send request to the Database and test our code.
- Login: On logging in to the application with the right credentials, the server responds with '200 OK' and a 'token'
 - The server generates a token to identify the user.
 - The browser saves the token until asked to clear localStorage.
 - Now for any further post request to create new tasks/assignments, the token is sent in the header of the request. The backend identifies the user from the token and if successful responds with a '200 OK' message.

THE DATABASE:

The application uses a noSQL database because of the convenience that comes with using mongoDB and the reduction in program complexity that noSQL databases provide. The unstructured nature of data makes it easier to create users that could be associated with multiple exams/tuts/tasks.

A screenshot of a web browser window displaying a JSON document. The browser's address bar shows 'localhost:3001/api/users/'. The JSON document is a collection of user data, including a list of exams, a list of tutorials, a profession, and user details like username, name, email, and id. The JSON is formatted with syntax highlighting.

```
[
  {
    exams: [
      {
        subject: "mathematics",
        date: "20/09/2023",
        syllabus: "chapter 1",
        id: "61723709eb85891a5a7641cf"
      },
      {
        subject: "Derivatives",
        date: "5/11/2021",
        syllabus: "Chapter 5, textbook 1",
        id: "61779c76f5a6aa928742338a"
      },
      {
        subject: "Partial Derivatives",
        date: "2021-10-22",
        syllabus: "Ch 2,3",
        id: "61779d43f5a6aa9287423399"
      },
      {
        subject: "CSN-261",
        date: "2021-11-04",
        syllabus: "HR Assignment 8",
        id: "617ebed740cd4747071563ed"
      },
      {
        subject: "MIN Tut 12",
        date: "10/11/2021",
        syllabus: "Exergy",
        id: "6182a3edb91f9a535dd6ed4f"
      }
    ],
    tuts: [ ],
    profession: [
      "Professor"
    ],
    username: "Hayley",
    name: "Hayley Williams",
    email: "hayley.williams@gmail.com",
    id: "6171705c8d1bae7223518a64"
  },
  {
    exams: [
```

The noSQL Database contains all information about:

- Users' personal details (Name, Password, Profession, etc.)
- User's To-Do List
- Upcoming deadlines for exams and assignments.
- Timetable of that student
- Calendar

NPM PROJECT:

- The Backend is a npm project which is built using node.js and express which provides a REST interface to operate on the data collections. NPM (Node Package Manager) is basically used for managing dependencies of various server-side dependencies.
- We can manage our server-side dependencies manually as well but once our project's dependencies grow it becomes difficult to install and manage. By using NPM it becomes easy, we just need to install NPM once for all dependencies.

WEB TOKENS:

- The application provides user authentication technology which uses web tokens to verify that the user is a valid user or not and further these tokens help them to keep them logged in for a definite period of time such that after that time he will get logged out and he has to login again.
- JSON Web Token is a JSON-based open standard (**RFC 7519**) for creating access tokens that assert some number of claims. For example, a server could generate a token that has the claim "logged in as admin" and provide that to a client.
- This is the most common scenario for using JWT. Once the user is logged in, each subsequent request will include the JWT, allowing the user to access routes, services, and resources that are permitted with that token. Single Sign On is a feature that widely uses JWT nowadays, because of its small overhead and its ability to be easily used across different domains.

CONCLUSION:

Student portal is an easy-to-use website for both the student and professor. The interface is simple and managing one's assignment deadlines and keeping track of quizzes can be easily done with this portal. Also, additional features like timetable and calendar help the user stay up-to date on his academic work.