

Rapport d'état d'avancement

Compétition ECS'2024

Projet	Projet 12 : Développement d'une plateforme Web immersive via WebXR
Equipe	Oussama Fayz , Youssef Moustaid
Encadrants	M. Mohmed Tabaa , Mme. Houda Mouttalib
Date du rapport	08/08/2024

I. Contexte

Le projet vise à créer une plateforme immersive utilisant des technologies avancées telles que WebXR. Cette plateforme permettra une interaction enrichie dans un environnement 3D pour des fins éducatives.

II. Travaux accomplis durant la semaine

- Mise en place de multer pour gérer le stockage des images sur le serveur.
- Développement de l'API de front-end.
- Mise en place des endpoints nécessaires pour permettre au front-end de communiquer avec l'API.
- Utilisation de JSON Web Tokens (JWT) pour gérer les sessions utilisateurs.
- Développement des pages de création de cours et de création de groupe.
- Développement d'un context menu.
- Début du développement des pages de création de quiz.

III. Perspectives de travail pour la semaine prochaine

- Poursuivre le développement des fonctionnalités de l'API front-end.
- Poursuivre les tests pour assurer la stabilité du système.
- Développement de la fonctionnalité de session.
- Développement du CMS en React pour l'administrateur du système.
- Développement des pages : forgot password, edit course, view course.

IV. Formations suivies

- Formation en React.js :
Nom : **Create Web Applications with React.js.**
Lien : <https://openclassrooms.com/en/courses/7132446-create-a-web-application-with-react-js/7206291-understand-the-purpose-of-react-js>
- Formation en A-frame :
Nom : **Developing AR/VR/MR/XR Apps with WebXR , Unity and Unreal.**
Lien : <https://www.coursera.org/learn/develop-augmented-virtual-mixed-extended-reality-applications-webxr-unity-unreal/home/module/2>
- Formation en Node.js:
Nom : **Go Full-Stack With Node.js, Express, and MongoDB.**
Lien : <https://openclassrooms.com/en/courses/5614116-go-full-stack-with-node-js-express-and-mongodb/5614123-set-up-your-coding-environment>

V. Quelques images de notre travail :

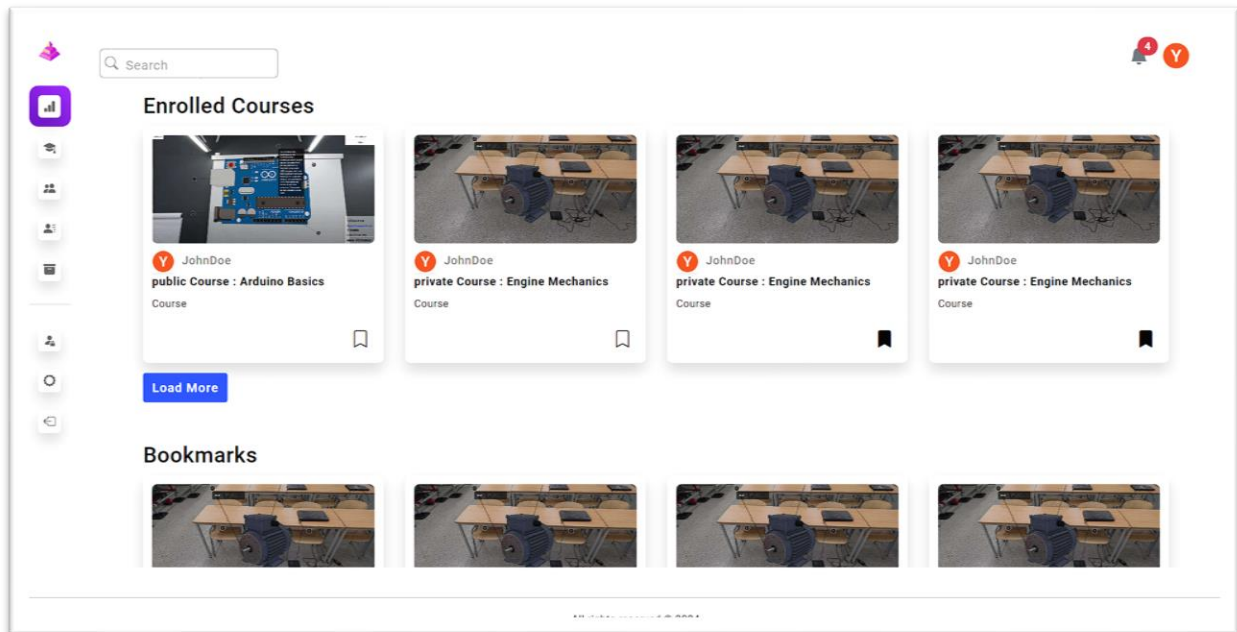


Fig 1 : Dashboard

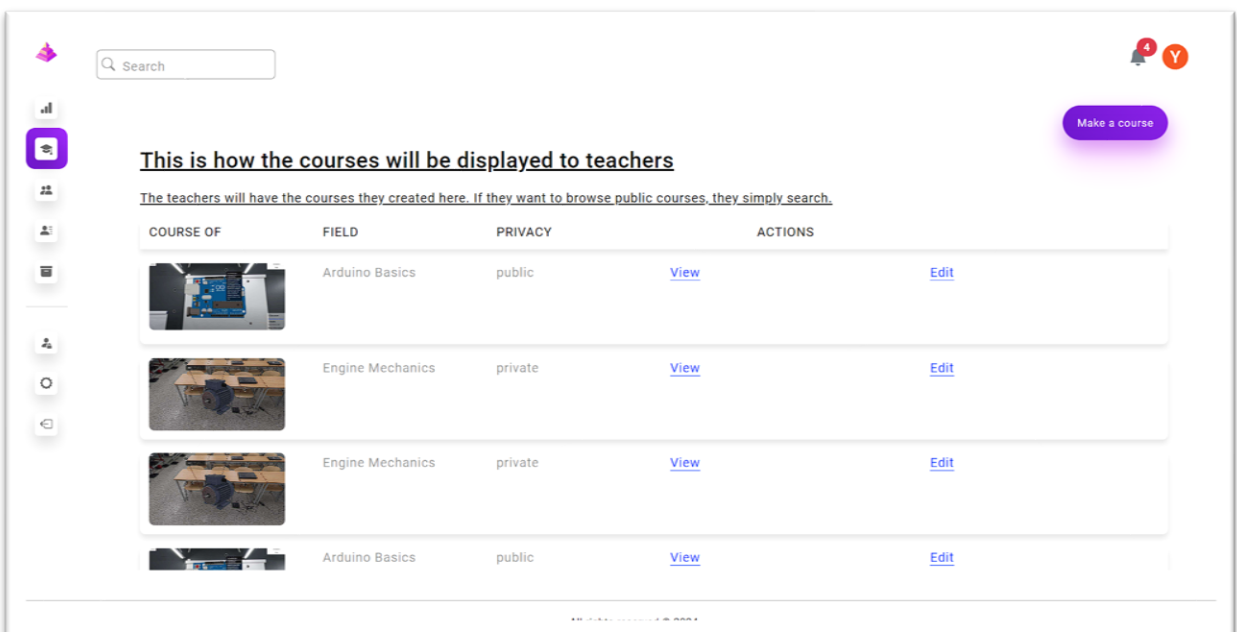


Fig 2 : Courses



Fig 3 : Group chat

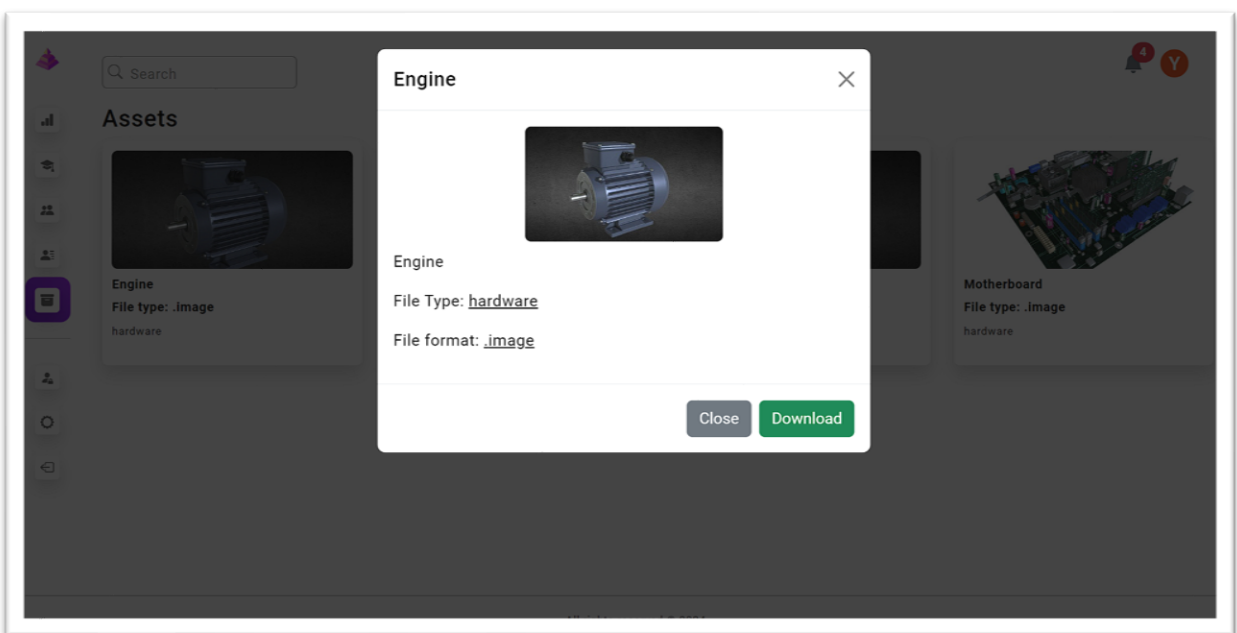
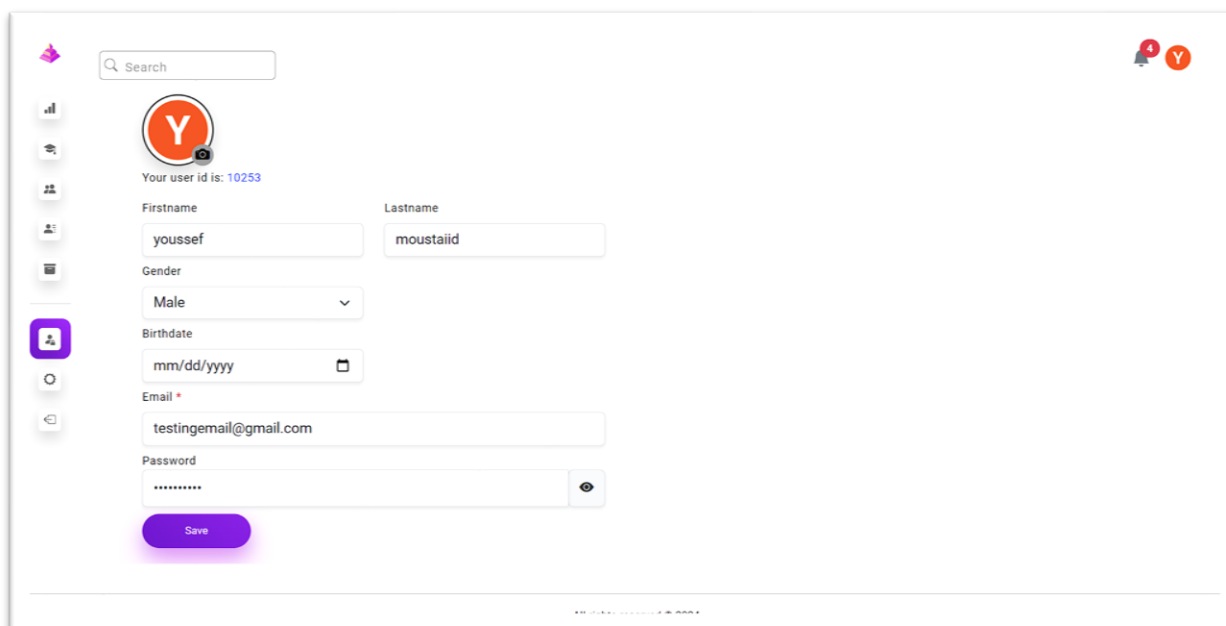


Fig 4 : Uploaded assets



Search

Your user id is: 10253

Firstname:
 Lastname:

Gender:

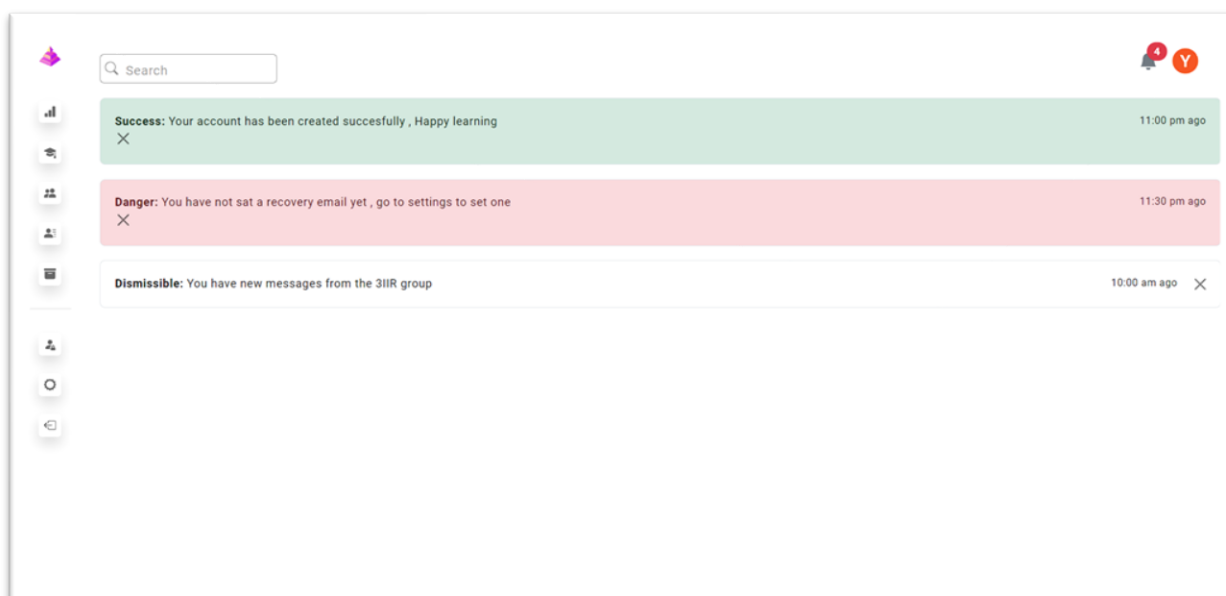
Birthdate:

Email:

Password:

Save

Fig 5 : Profile



Search

Success: Your account has been created succesfully , Happy learning
 11:00 pm ago

Danger: You have not sat a recovery email yet , go to settings to set one
 11:30 pm ago

Dismissible: You have new messages from the 3IIR group
 10:00 am ago

Fig 6 : Notifications

Course Details

Course Title *

Description *

Category *

Software ▾

Environment *

School ▾

Privacy *

Private ▾

[Next](#)

Fig 7 : Create course page 1 – Course Meta data

←
[Go back](#)

File Upload

Course PDF *

Choose File
Black and White Minimalist Simple Artist CV Resume (1).pdf


Video * (please note that the video should be a YouTube embed code; any other format is not accepted)

clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" referrerpolicy="strict-origin-when-cross-origin" allowfullscreen></frame>

Video Preview:

100 SECONDS OF

Watch later
Share



Previous
Next

Fig 8 : Create course page 2 – Required course assets

Additional Assets

Asset name *

Asset Type *

3D model (.obj, .gltf)

▼

Asset File *

Choose File
uploads_files_3834193_classroom.obj

+ Add Asset

Previous

Next

Fig 9 : Create course page 3 – Additional assets

Course Preview

Course Details

Course Title: Arduino Course

Description: Arduino Course

Type:

Category: software

Environment: school

Privacy:

Video:


File Uploads

Type: 3d-model

Name: Arduino Card

File: uploads_files_3834193_classroom.obj

Size: 17444 K bytes



Previous

Confirm

Fig 10 : Create course page 4 – Review and confirm

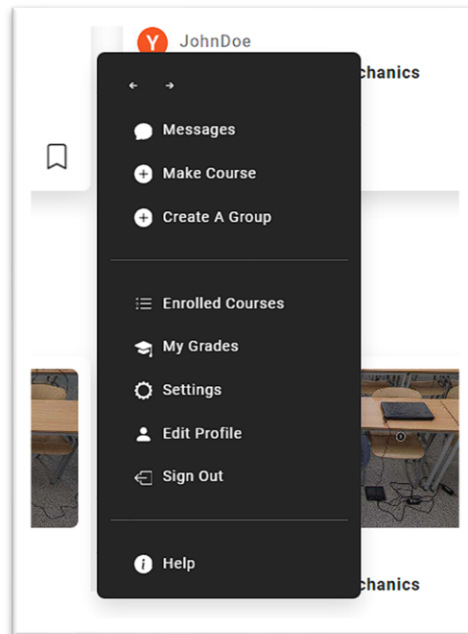


Fig 11:Context menu

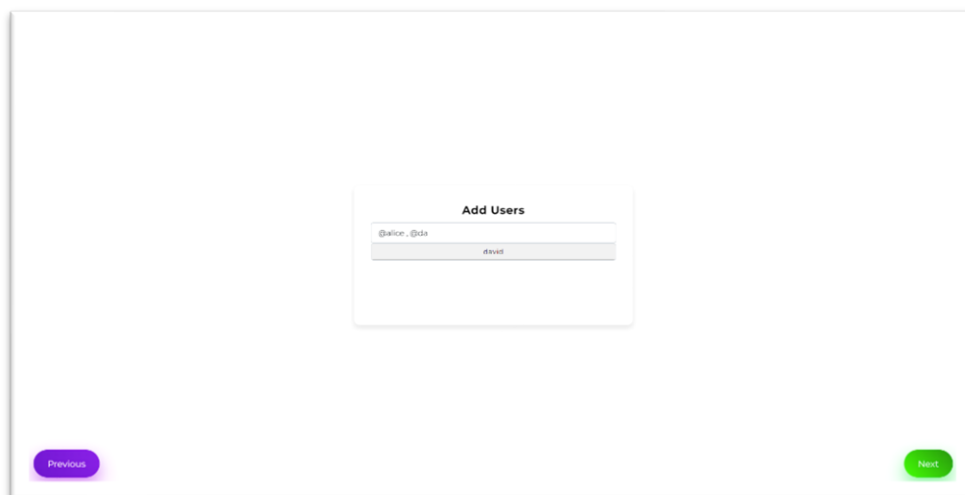


Fig : 12 : Add users to make a group


```

55: // token)); // token hook useEffect has a missing dependency: 'getStudents'. Either include it or remove the dependency array.
56: return (
57:   <div className="students mt-5">
58:     <table className="table shadow">
59:       <thead>
60:         <tr>
61:           <th scope="col">#</th>
62:           <th scope="col">Id</th>
63:           <th scope="col">Email</th>
64:           <th scope="col">Courses Attended</th>
65:         </tr>
66:       </thead>
67:       <tbody>
68:         {students.map((student, index) => (
69:           <tr scope="row" className="small-scale-on-hover" key={index}>
70:             <td>
71:               <img className="profile-img" src={defaultUserProfile} alt="profile" />
72:               <span className="ms-2">{student.firstname} {student.lastname}</span>
73:             </td>
74:             <td>{student.idstudent}</td>
75:             <td>{student.email}</td>
76:             <td>10</td>
77:           </tr>
78:         ))}
79:       </tbody>
80:     </table>
81:   </div>

```

PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL PORTS

fayzo@DESKTOP-3NN3VOA MINGW64 ~/OneDrive/Bureau/frontend+api (main)

powershell
bash
powershell

Fig 13 : List of students

```

7: const Students = () => {
12:
13:   useEffect(() => {
14:     const storedToken = localStorage.getItem('token');
15:     if (!storedToken) {
16:       navigate('/sign-in');
17:     } else {
18:       setToken(storedToken);
19:     }
20:   }, [navigate]);
21:
22:   useEffect(() => {
23:     const storedIdTeacher = localStorage.getItem('idteacher');
24:     setIdTeacher(storedIdTeacher);
25:   }, []);
26:   console.log(idTeacher);
27:
28:   const getStudents = async () => {
29:     try {
30:       const response = await fetch('http://localhost:4200/students', {
31:         method: 'GET',
32:         headers: {
33:           'Content-Type': 'application/json',
34:           'Authorization': 'Bearer ${token}'
35:         },
36:       });
37:
38:       if (!response.ok) {
39:         throw new Error('Failed to fetch students');
40:       }

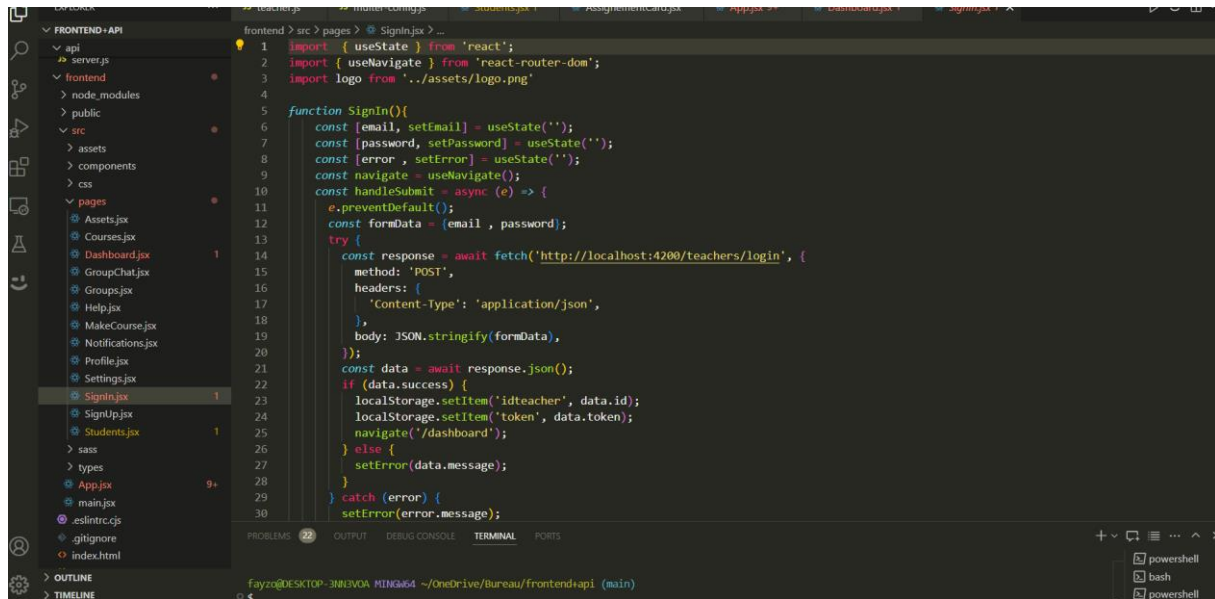
```

PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL PORTS

fayzo@DESKTOP-3NN3VOA MINGW64 ~/OneDrive/Bureau/frontend+api (main)

powershell
bash

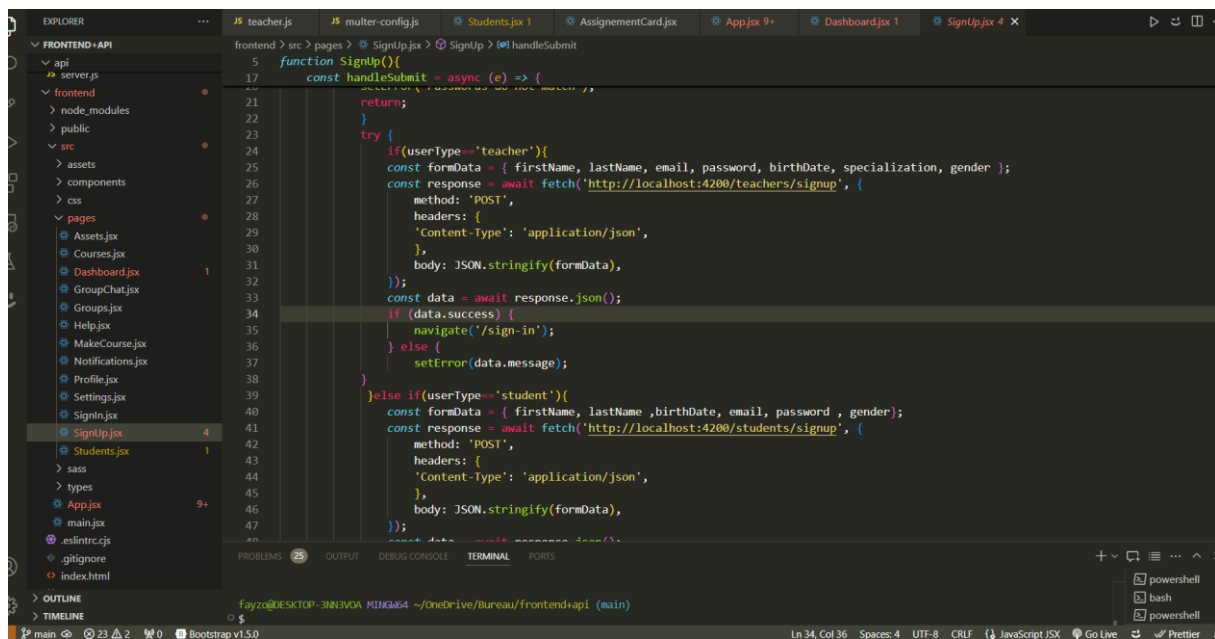
Fig 14 : Request to get list of students and token verification



```

1  import { useState } from 'react';
2  import { useNavigate } from 'react-router-dom';
3  import logo from '../assets/logo.png';
4
5  function SignIn() {
6    const [email, setEmail] = useState('');
7    const [password, setPassword] = useState('');
8    const [error, setError] = useState('');
9    const navigate = useNavigate();
10   const handleSubmit = async (e) => {
11     e.preventDefault();
12     const formData = { email, password };
13     try {
14       const response = await fetch('http://localhost:4200/teachers/login', {
15         method: 'POST',
16         headers: {
17           'Content-Type': 'application/json',
18         },
19         body: JSON.stringify(formData),
20       });
21       const data = await response.json();
22       if (data.success) {
23         localStorage.setItem('idteacher', data.id);
24         localStorage.setItem('token', data.token);
25         navigate('/dashboard');
26       } else {
27         setError(data.message);
28       }
29     } catch (error) {
30       setError(error.message);
31     }
32   }
33 }
  
```

Fig 15 : Login



```

5  function SignUp() {
6    const [firstName, setFirstName] = useState('');
7    const [lastName, setLastName] = useState('');
8    const [birthDate, setBirthDate] = useState('');
9    const [email, setEmail] = useState('');
10   const [password, setPassword] = useState('');
11   const [gender, setGender] = useState('');
12   const handleSubmit = async (e) => {
13     e.preventDefault();
14     const formData = {
15       firstName,
16       lastName,
17       birthDate,
18       email,
19       password,
20       gender,
21     };
22     try {
23       const response = await fetch('http://localhost:4200/teachers/signup', {
24         method: 'POST',
25         headers: {
26           'Content-Type': 'application/json',
27         },
28         body: JSON.stringify(formData),
29       });
30       const data = await response.json();
31       if (data.success) {
32         navigate('/sign-in');
33       } else {
34         setError(data.message);
35       }
36     } else if (userType == 'student') {
37       const formData = { firstName, lastName, birthDate, email, password, gender };
38       const response = await fetch('http://localhost:4200/students/signup', {
39         method: 'POST',
40         headers: {
41           'Content-Type': 'application/json',
42         },
43         body: JSON.stringify(formData),
44       });
45       const data = await response.json();
46       if (data.success) {
47         navigate('/sign-in');
48       } else {
49         setError(data.message);
50       }
51     }
52   }
53 }
  
```

Fig 16 : Signup

```
api > src > routes > JS teacher.js > ...
1  const express = require('express');
2  const router = express.Router();
3  const teacherCtrl = require('../controllers/teacher');
4  const auth = require('../middleware/auth');
5  const multer = require('../middleware/multer-config');
6
7  router.post('/signup', teacherCtrl.signup);
8  router.post('/login', teacherCtrl.login);
9  router.get('/:id', teacherCtrl.getTeacher);
10 router.put('/:id', auth, multer, teacherCtrl.updateTeacher);
11
12 module.exports = router;
```

Fig 17 : Multer image upload route integration

```
i > src > middleware > JS multer-config.js > ...
1  const multer = require('multer');
2  const path = require('path');
3
4  const MIME_TYPES = {
5    'image/jpg': 'jpg',
6    'image/jpeg': 'jpg',
7    'image/png': 'png'
8  };
9
10 const storage = multer.diskStorage({
11   destination: (req, file, callback) => {
12     callback(null, '../api/src/images'); // Ensure this folder exists or create it programmatically
13   },
14   filename: (req, file, callback) => {
15     // Extract file extension
16     const extension = MIME_TYPES[file.mimetype];
17     // Generate a sanitized filename with timestamp
18     const name = file.originalname
19       .replace(/\s+/g, '_') // Replace spaces with underscores
20       .replace(/\.[^./]+$/, ''); // Remove existing file extension
21     // Construct the new filename
22     callback(null, `${name}_${Date.now()}.${extension}`);
23   }
24 });
25
26 module.exports = multer({ storage: storage }).single('image');
```

Fig :18 : Multer configuration to handle image upload