



CSE471: System Analysis and Design Project Report

Project Title: Thesis Portal

Group No: 01, CSE471 Lab Section: 05, Fall 2024	
ID	Name
21201532	Tasmin Ahmed Oni
22101881	Afsana Akter Mim
21201757	Riba Rahman
21201241	Nirvik Shaha Rudra

Submission Date:02/09/2025

Table of Contents

1. System Request	3
Business need:	3
Business requirements:	3
Business value:	3
Special issues or constraints:	4
2. Functional Requirements	5
3. Technology (Framework, Languages)	6
4. Backend Development	6
5. User Interface Design	12
6. Frontend Development	15
7. User Manual	37
8. Performance and Network Analysis	48
9. Github Repo [Public] Link	51
10. Link of Deployed Project	51
11. Individual Contribution	52
12. References	53

1. System Request

Business need

The Thesis Portal is designed to streamline and digitize the entire thesis management process within academic institutions. This platform provides a centralized, interactive, and secure platform where students, supervisors, and administrators can collaborate seamlessly. It ensures smooth submission tracking, supervisor-student coordination, structured evaluation, and transparent progress monitoring throughout the thesis lifecycle.

Business requirements

Using the Thesis Portal, students, supervisors, and administrators will be able to manage the entire thesis process in one centralized platform. The specific functionality that the system should have includes the following:

- Develop a secure and role-based platform for student, supervisor, and administrator access with profile management.
- Enable thesis submission tracking for Registration, P1, P2, and P3, including supervisor approval, feedback, and grading.
- Provide group formation and supervisor assignment features to simplify collaboration and workload distribution.
- Support real-time notifications, messaging, and progress tracking to ensure smooth communication and timely updates.
- Maintain a repository of previous research documents with search, filtering, and bookmarking tools for academic reference.
- Equip supervisors and administrators with dashboards for monitoring submissions, evaluating work, and managing users.
- Offer additional tools such as one-click thesis templates and AI-powered report summarization to enhance efficiency.

Business value

We expect that the Thesis Portal will significantly improve the efficiency and transparency of the thesis process by providing a centralized platform for students, supervisors, and administrators. The portal will streamline submissions, supervisor assignments, feedback collection, and evaluation, reducing delays and miscommunication common in manual processes.

The system's user-friendly design, real-time notifications, and progress tracking features will ensure students stay on schedule while supervisors can manage and evaluate multiple groups effectively. By reducing delays and miscommunication, the portal enhances academic quality and collaboration. Its scalable design ensures adaptability to institutional growth and evolving academic needs, supporting long-term usability and value.

Special issues or constraints

1. Data Privacy and Security

Since the portal handles sensitive student and faculty data (like grades, personal info, and research papers), it must have strong security to prevent hacking, data leaks, or unauthorized access.

2. Device Compatibility

Although the system is responsive, it may not look or work the same on all devices (especially older phones or browsers). Testing on different screen sizes is needed.

3. AI Features Limitations

The AI summarizer may not always generate accurate summaries or suggestions, especially for complex or poorly formatted papers. Users should not rely on it 100%.

4. Scalability Issues

If too many users (students, teachers, admins) use the system at the same time, performance might slow down unless the system is optimized for high traffic.

5. Maintenance and Updates

The portal will need regular maintenance, bug fixes, and updates. If not updated, it might face compatibility issues or security risks in the future.

6. Group Management Complexity

Group formation, supervisor requests, and coordination between students can get complex, especially if users don't follow proper steps. It may lead to confusion without clear instructions.

1. Functional Requirements

→ Role-based Authentication

The system supports different user roles (Student, Teacher, and Admin) with appropriate permissions.

→ User Registration

Students, teachers, and administrators can create accounts by providing details such as name, email, ID, and password.

→ User Login/Logout

Users can securely log in and log out using their registered credentials.

→ Change/Reset Password

Users can reset or update their passwords securely.

→ Profile Page

Students and teachers each have a profile page displaying academic information such as CGPA, department, supervisor expertise, and profile photo.

→ Edit Profile

Students and teachers can update their profile information when necessary.

→ Dashboard

Dashboards allow students and teachers to view their personalized information, thesis submissions, and activities.

→ Admin Control

Admins can manage users, assign or update marks for P1, P2, and P3 submissions, upload and update pre-formatted thesis templates (P1, P2, P3) for student download, adjust supervisor group limits to balance workload, and delete inappropriate posts from the platform.

→ Report Submission

Students can upload Registration, P1, P2, and P3 reports in PDF format along with metadata (title, submission date, and group details).

→ Supervisor Review

Supervisors can approve or reject reports and provide feedback.

→ Grading System

Supervisors and admins can assign marks for P1, P2, and P3 submissions.

→ Progress Tracking

Students can view the timeline of their thesis progress, submission status, and marks.

→ Notifications

Students receive notifications about submission status, supervisor feedback, and marks.

→ Previous Research Repository

A searchable database of past thesis documents is available for students. Faculty can upload new research papers to the repository.

→ Bookmarking

Students can bookmark and remove bookmarks for important resources from the repository.

→ AI Summarizer

The system provides an AI-based summarizer for submitted reports, highlighting strengths, weaknesses, and suggestions.

→ Group Formation

Students can form groups of 1–4 members by sending or accepting group requests.

→ Supervisor Selection

Students can request supervisors or co-supervisors, while supervisors can accept or reject based on availability.

→ Supervisor Assignment

Once confirmed, all group members are automatically linked to the same supervisor.

→ Group Posts

Students can create posts to find group members, with other users able to comment or like.

→ Feedback & Rating System

Students can rate and leave feedback for supervisors after thesis completion.

→ Search & Filter

Students can search for supervisors by name, department, or email, and filter submissions by status, group, or academic year.

→ Real-Time Chat

Students and supervisors can exchange messages in real time.

→ Chat Search

Users can search for specific contacts or messages in the chat system.

→ Chat Alerts

Users are notified when a new message is received.

→ Notification System

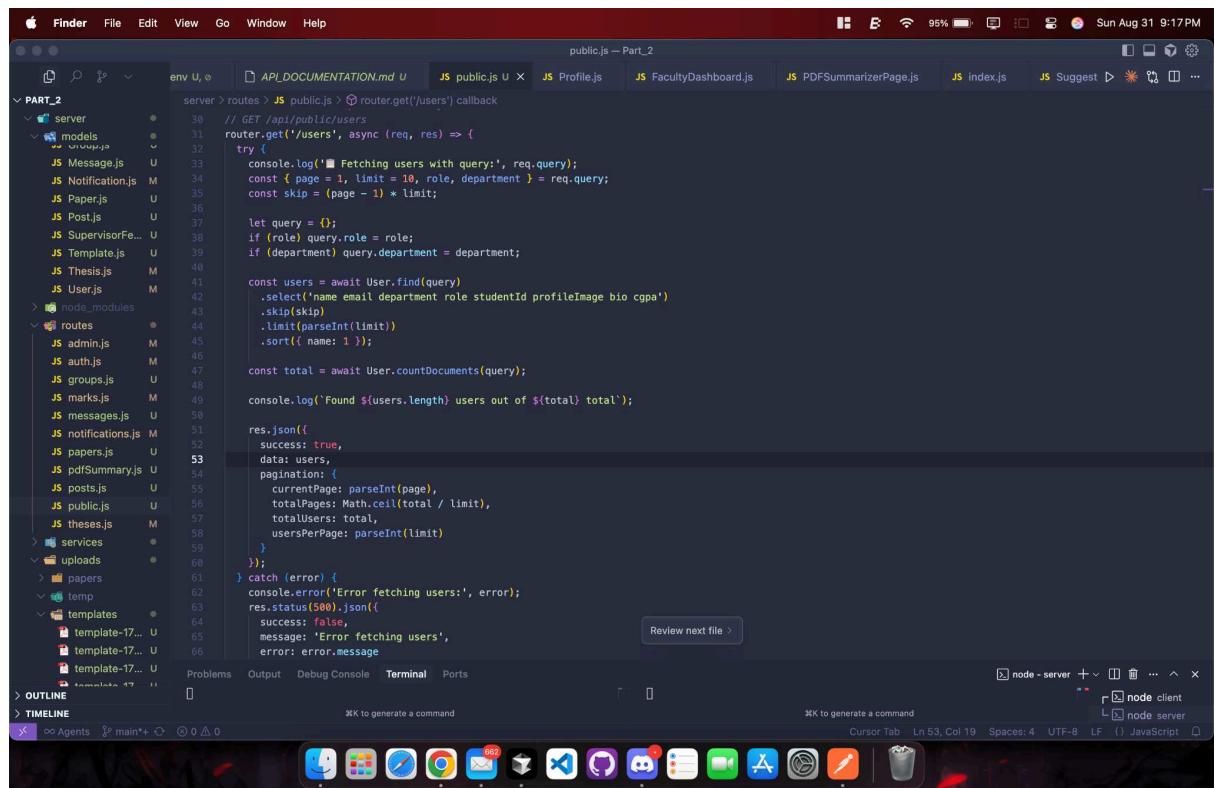
Students are notified about submission status, feedback, marks, and supervisor assignments, while supervisors receive notifications for new requests or submissions.

2. Technology (Framework, Languages)

- Language - JavaScript
- Frontend- React
- Backend- NodeJS and ExpressJS
- Database - MongoDB

3. Backend Development

1. User Info:



```
server > routes > JS public.js -> router.get('/users') callback
      // GET /api/public/users
      router.get('/users', async (req, res) => {
        try {
          console.log(`Fetching users with query: ${req.query}`);
          const { page = 1, limit = 10, role, department } = req.query;
          const skip = (page - 1) * limit;

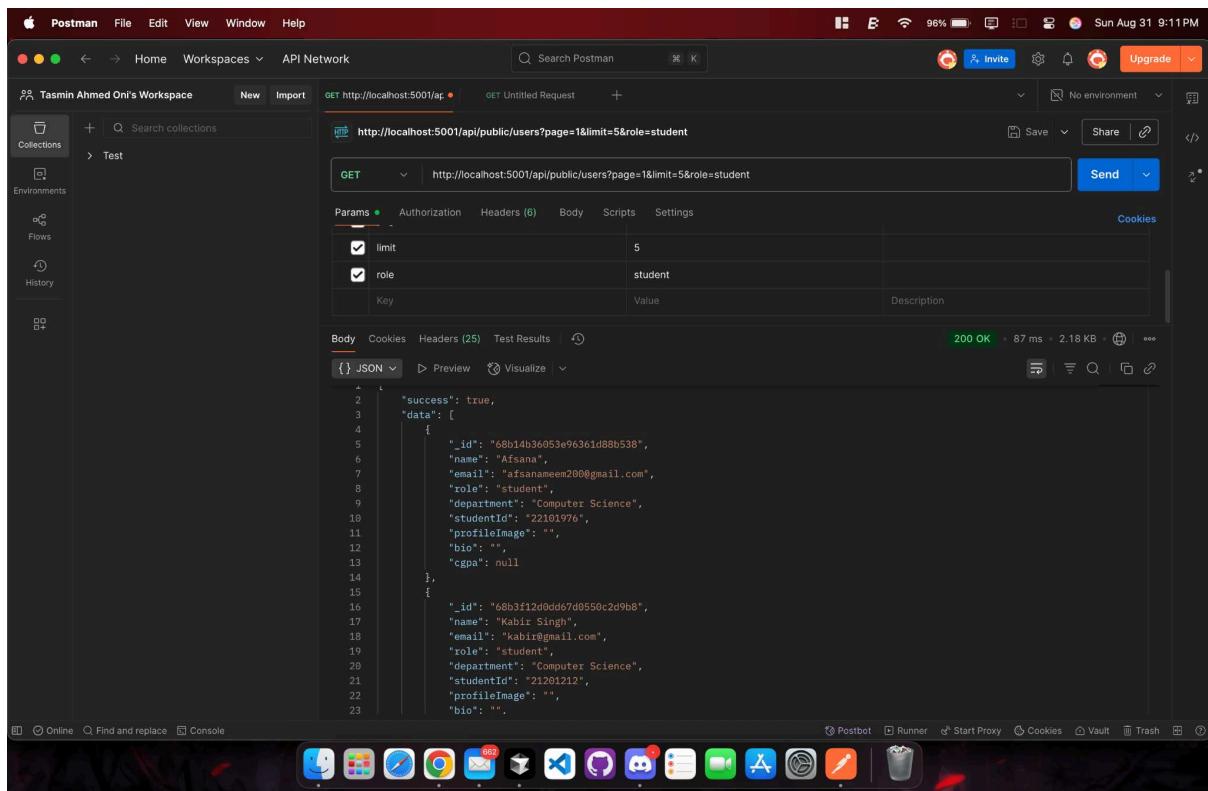
          let query = {};
          if (role) query.role = role;
          if (department) query.department = department;

          const users = await User.find(query)
            .select('name email department role studentId profileImage bio cgs')
            .skip(skip)
            .limit(parseInt(limit))
            .sort({ name: 1 });

          const total = await User.countDocuments(query);

          console.log(`Found ${users.length} users out of ${total}`);
          res.json({
            success: true,
            data: users,
            pagination: {
              currentPage: parseInt(page),
              totalPages: Math.ceil(total / limit),
              totalUsers: total,
              usersPerPage: parseInt(limit)
            }
          });
        } catch (error) {
          console.error('Error fetching users:', error);
          res.status(500).json({
            success: false,
            message: 'Error fetching users',
            error: error.message
          });
        }
      });
    }

  
```

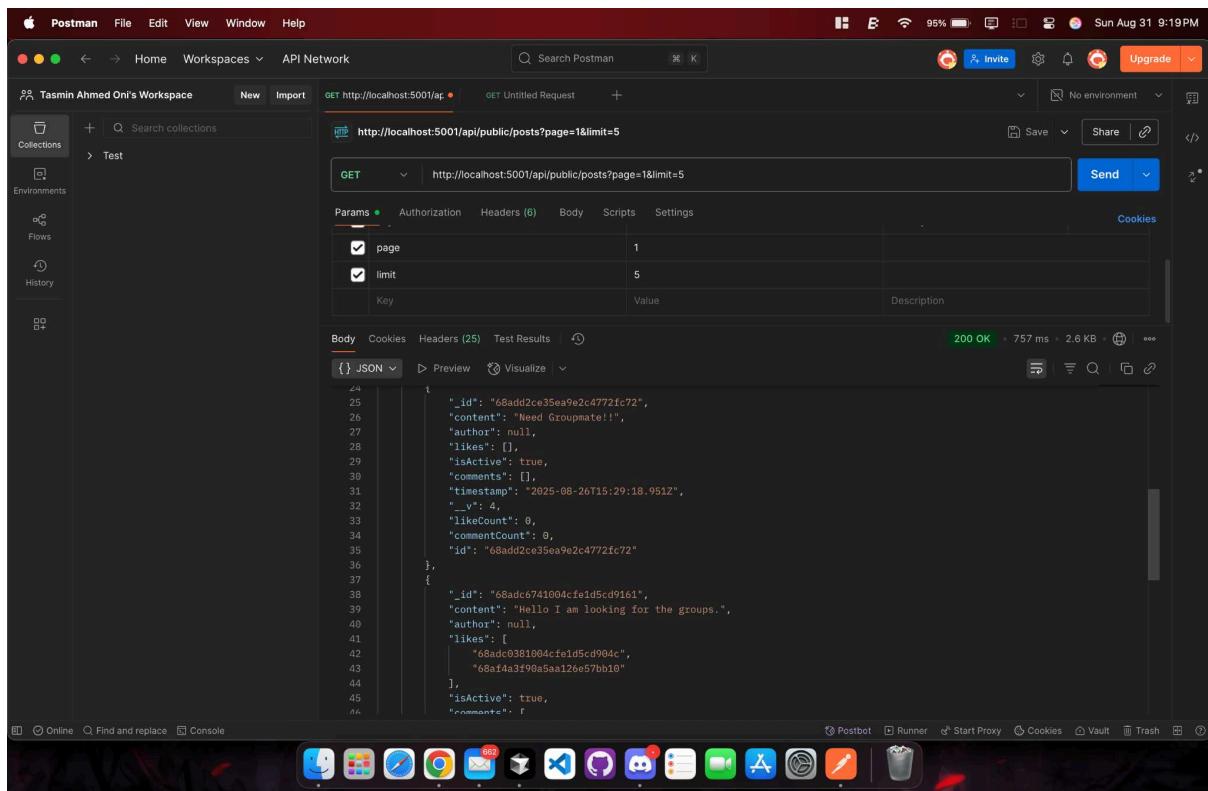


2. Users Posts:

The screenshot shows the Visual Studio Code (VS Code) interface. The top bar includes "Cursor File Edit Selection View Go Run Terminal Window Help" and "Sun Aug 31 9:20PM". The left sidebar shows a file tree with "PART_2" and various files like "server", "models", "routes", "services", "uploads", "temp", and "templates". The main editor window displays the "public.js" file under "API_DOCUMENTATION.md U". The code is as follows:

```
server > routes > JS public.js > router.get('/posts') callback
72 // GET /api/public/posts
73 router.get('/posts', async (req, res) => {
74   try {
75     console.log('Fetching posts with query:', req.query);
76     const { page = 1, limit = 10, author } = req.query;
77     const skip = (page - 1) * limit;
78
79     let query = { isActive: true };
80     if (author) query.author = author;
81
82     const posts = await Post.find(query)
83       .populate('author', 'name email profileImage department role')
84       .populate('comments.author', 'name profileImage')
85       .skip(skip)
86       .limit(parseInt(limit))
87       .sort({ timestamp: -1 });
88
89     const total = await Post.countDocuments(query);
90
91     // Add virtual fields
92     const postsWithCounts = posts.map(post => (
93       ...post.toObject(),
94       likeCount: post.likes.length,
95       commentCount: post.comments.length
96     ));
97
98     console.log(`Found ${posts.length} posts out of ${total} total`); | TAB to jump here
99
100   res.json({
101     success: true,
102     data: postsWithCounts,
103     pagination: {
104       currentPage: parseInt(page),
105       totalPages: Math.ceil(total / limit),
106       totalPosts: total,
107       postsPerPage: parseInt(limit)
108     }
109   })
110 })
```

The bottom status bar shows "node - server" and "node client" with "node server". The bottom navigation bar includes "Problems", "Output", "Debug Console", "Terminal", and "Ports".



3. All Papers:

The screenshot shows the Visual Studio Code (VS Code) interface. The top bar includes "Cursor", "File", "Edit", "Selection", "View", "Go", "Run", "Terminal", "Window", and "Help". The status bar shows "public.js — Part_2" and "Sun Aug 31 9:23PM".

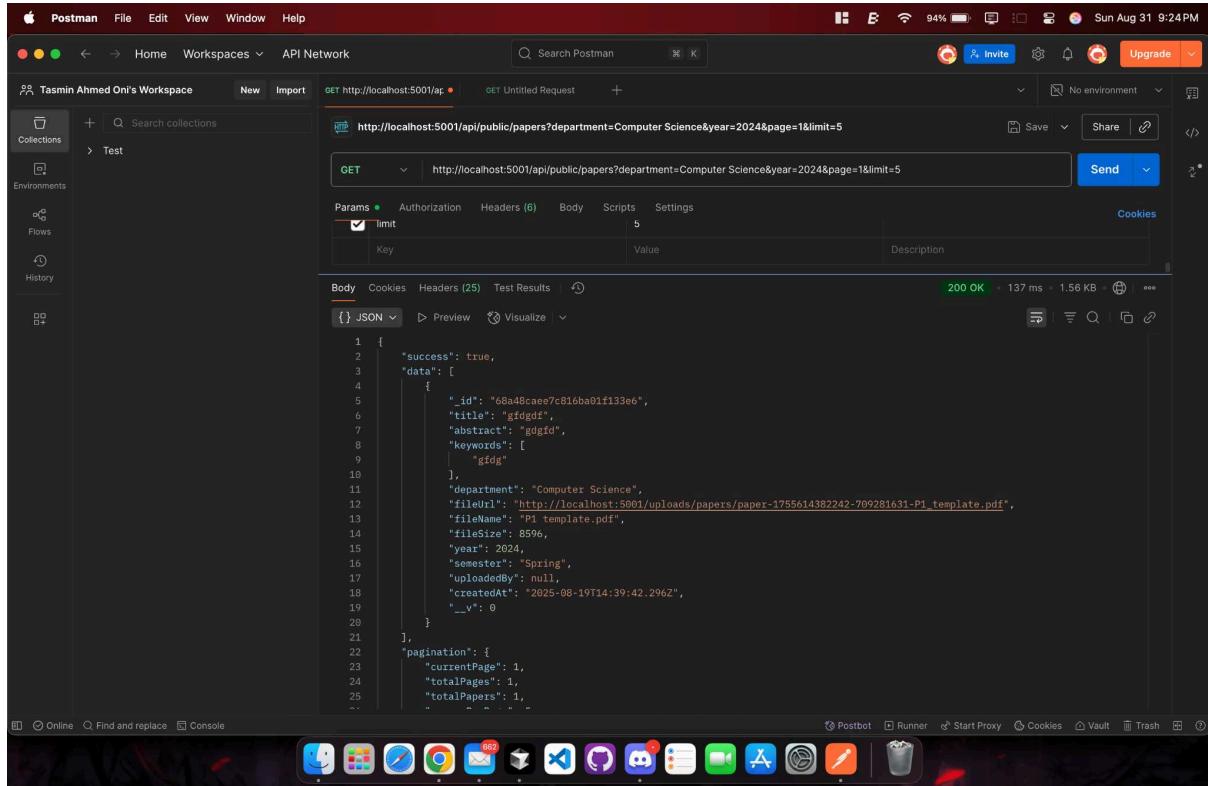
The left sidebar shows a file tree with a project structure under "PART_2". The "public.js" file is open in the center editor. The code is as follows:

```

server > routes > JS public.js U ↗ router.get('/papers') callback: ↗ pagination
120 // 3. Get all papers (public)
121 // GET /api/public/papers
122 router.get('/papers', async (req, res) => {
123   try {
124     console.log('Fetching papers with query:', req.query);
125     const { page = 1, limit = 10, department, year, semester, search } = req.query;
126     const skip = (page - 1) * limit;
127
128     let query = {};
129     if (department) query.department = department;
130     if (year) query.year = parseInt(year);
131     if (semester) query.semester = semester;
132     if (search) {
133       query.$text = { $search: search };
134     }
135
136     const papers = await Paper.find(query)
137       .populate('uploadedBy', 'name email department')
138       .skip(skip)
139       .limit(parseInt(limit))
140       .sort({ createdAt: -1 });
141
142     const total = await Paper.countDocuments(query);
143
144     console.log(`Found ${papers.length} papers out of ${total}`);
145
146     res.json({
147       success: true,
148       data: papers,
149       pagination: {
150         currentPage: parseInt(page),
151         totalPages: Math.ceil(total / limit),
152         totalPapers: total,
153         papersPerPage: parseInt(limit)
154       }
155     });
156   } catch (error) {
157     console.error(`Error fetching papers! ${error}`);
158   }
159 }

```

The bottom of the screen shows the "node - server" terminal with "node client" and "node server" listed, along with other VS Code status indicators like "Cursor Tab", "Ln 149, Col 20", "Spaces: 4", "UTF-8", and "JavaScript".



4. Templates:

```

server > routes > JS public.js > router.get('/templates') callback
16/ // GET /api/public/templates
169 router.get('/templates', async (req, res) => {
try {
  console.log('Fetching templates');
  const templates = await Template.find()
    .populate('uploadedBy', 'name email department role')
    .sort({ uploadDate: -1 });

  console.log(`Found ${templates.length} templates`);

  res.json({
    success: true,
    data: templates,
    totalTemplates: templates.length
  });
} catch (error) {
  console.error('Error fetching templates:', error);
  res.status(500).json({
    success: false,
    message: 'Error fetching templates',
    error: error.message
  });
}
}

```

Postman Screenshot:

```

GET http://localhost:5001/api/public/templates
200 OK
{
  "data": [
    {
      "_id": "68adc7f0be314295378fb63",
      "type": "P1",
      "filename": "template-1756219390960-682090343-cst.pdf",
      "originalName": "cst.pdf",
      "uploadedBy": {
        "_id": "68700b5063663d0d098b5097",
        "name": "Admin",
        "email": "admin@gmail.com",
        "role": "admin",
        "department": "Computer Science"
      },
      "uploadDate": "2025-08-26T14:43:11.074Z",
      "__v": 0
    },
    {
      "_id": "68625a9ae5a2b0184541aa64",
      "type": "P3",
      "filename": "template-1753373338756-675028843-P3 template.pdf",
      "originalName": "P3 template.pdf",
      "uploadedBy": {
        "_id": "68700b5063663d0d098b5097",
        "name": "Admin",
        "email": "admin@gmail.com",
      }
    }
  ]
}

```

5.Notification:

VS Code Screenshot:

```

server > routes > JS public.js > router.get('/notifications') callback
router.get('/notifications', async (req, res) => {
  console.log(`Fetching notifications with query: ${req.query}`);
  const { page = 1, limit = 10, type, isRead } = req.query;
  const skip = (page - 1) * limit;

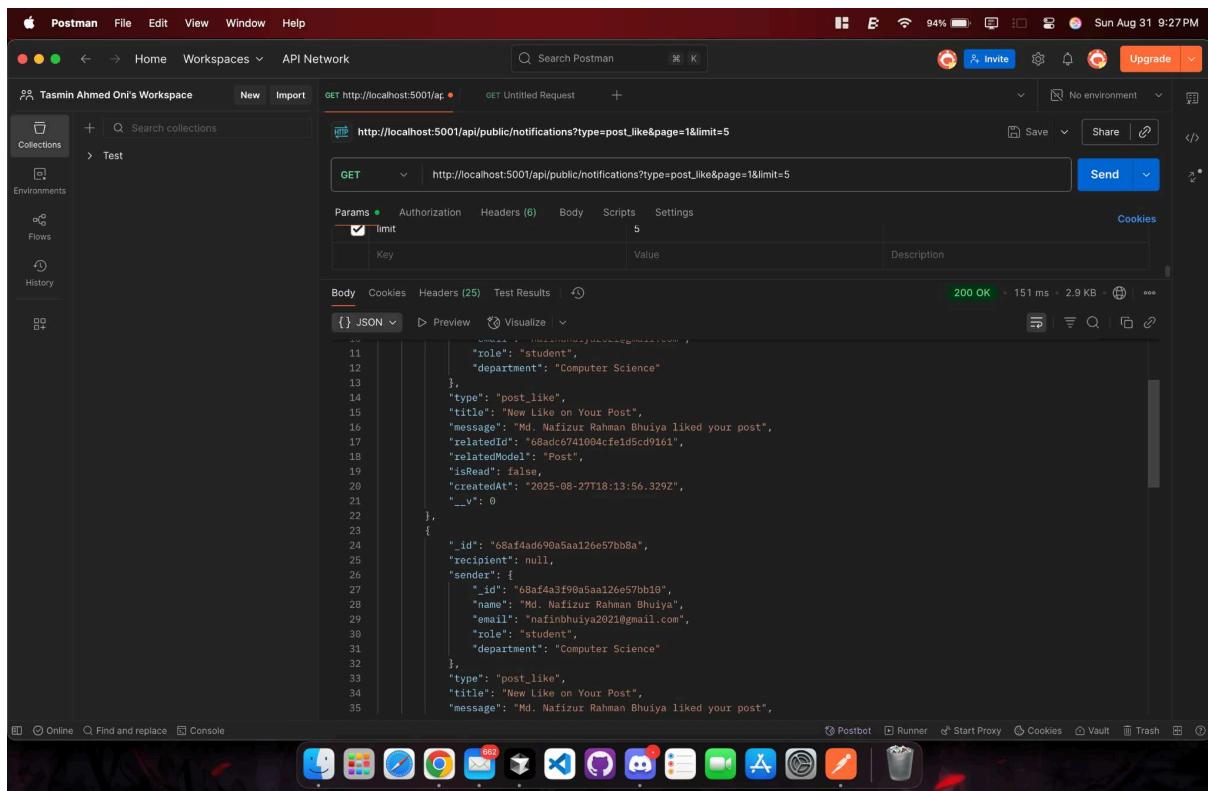
  let query = {};
  if (type) query.type = type;
  if (isRead !== undefined) query.isRead = isRead === 'true';

  const notifications = await Notification.find(query)
    .populate('recipient', 'name email department role')
    .populate('sender', 'name email department role')
    .skip(skip)
    .limit(parseInt(limit))
    .sort({ createdAt: -1 });

  const total = await Notification.countDocuments(query);

  console.log(`Found ${notifications.length} notifications out of ${total}`);
  res.json({
    success: true,
    data: notifications,
    pagination: {
      currentPage: parseInt(page),
      totalPages: Math.ceil(total / limit),
      totalNotifications: total,
      notificationsPerPage: parseInt(limit)
    }
  });
} catch (error) {
  console.error('Error fetching notifications:', error);
  res.status(500).json([
    success: false,
    message: 'Error fetching notifications',
    error: error.message
  ]);
}


```



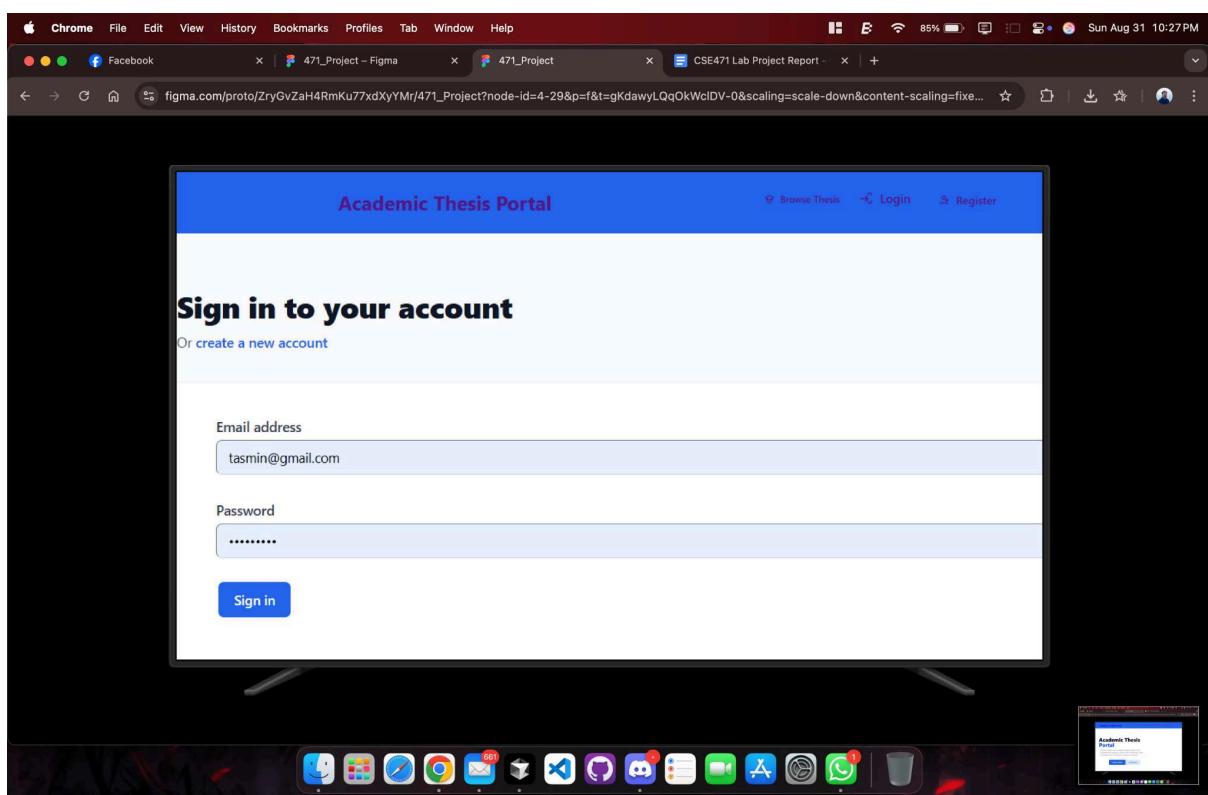
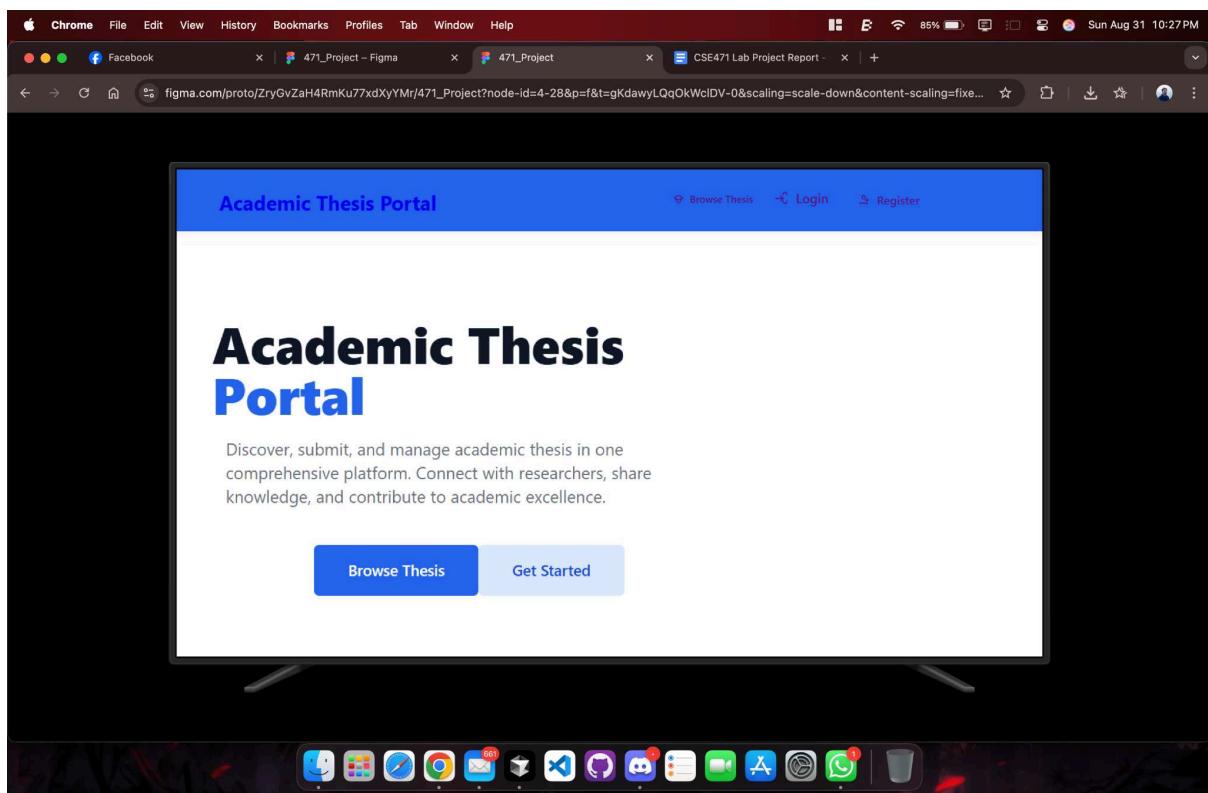
4. User Interface Design

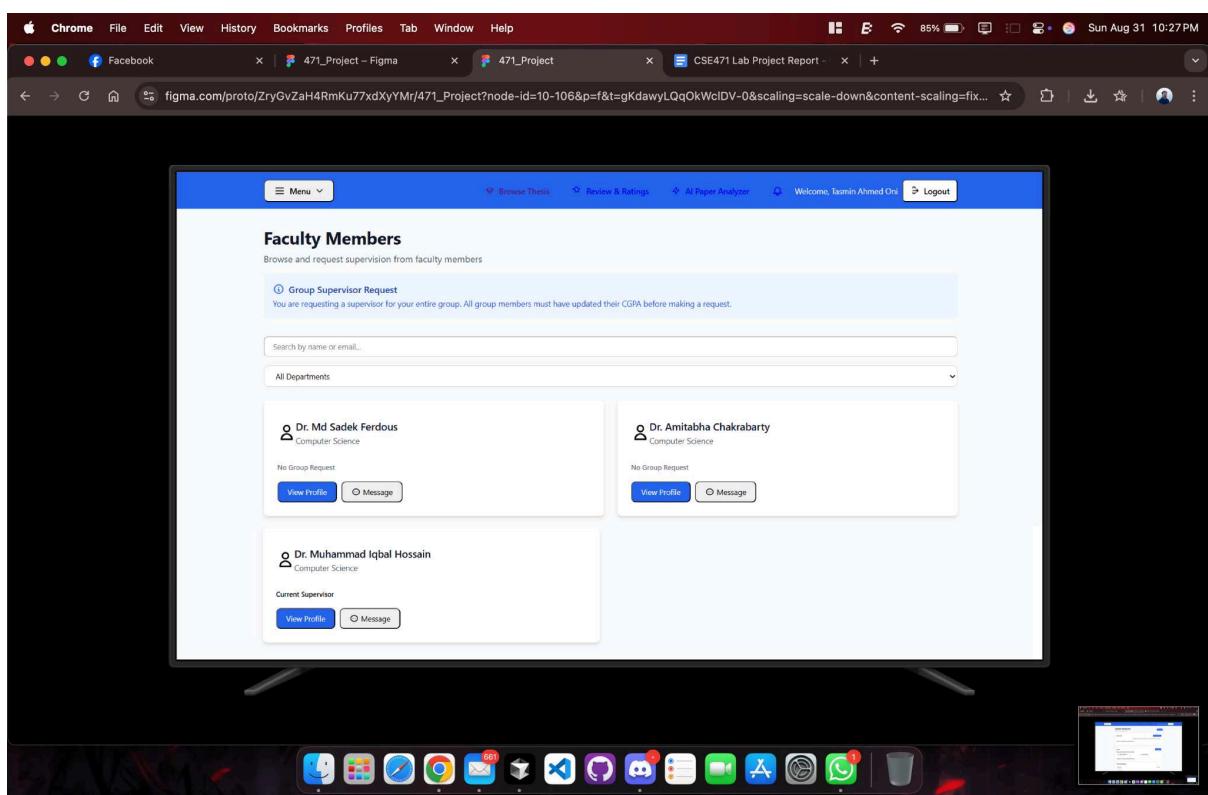
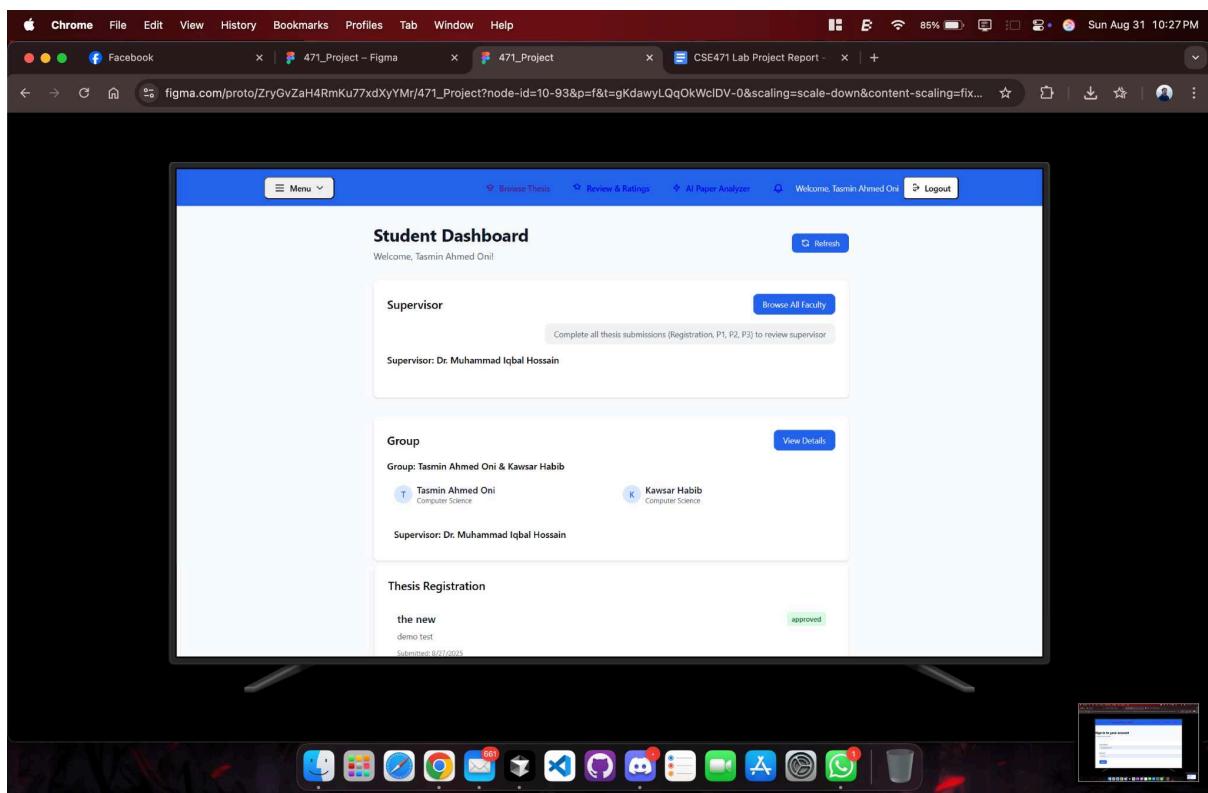
Figma Link:

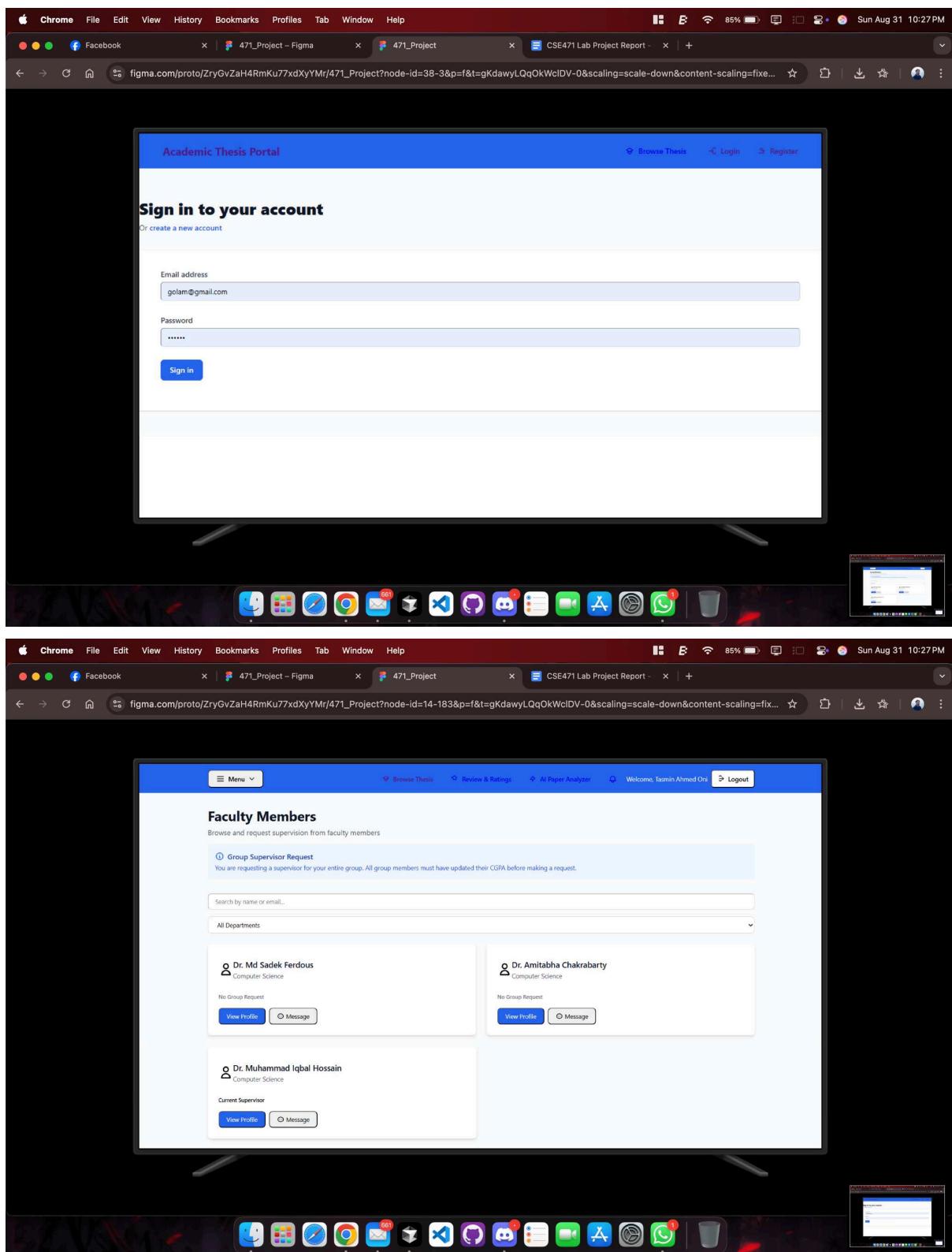
https://www.figma.com/design/ZryGvZaH4RmKu77xdXyYMr/471_Project?node-id=0-1&p=f

Prototype Link:

https://www.figma.com/proto/ZryGvZaH4RmKu77xdXyYMr/471_Project?node-id=4-28&t=ecd1g8ZD6oz0kEE5-1&starting-point-node-id=4%3A28







5. Frontend Development

1. Landing Page: The landing page is the main entry point and introduction to the Academic Thesis Portal, designed to welcome visitors and provide an overview of the platform's features. When users first visit the site, they see a visually appealing hero section with

compelling headlines, descriptive text about the portal's purpose, and call-to-action buttons that guide them to either register for a new account or log in to an existing one.

```

1 import React from 'react';
2 import { Link } from 'react-router-dom';
3 import SectionCarousel from './SectionCarousel';
4
5 const LandingPage = () => {
6   return (
7     <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100">
8       </> Hero Section </div>
9     <div className="relative overflow-hidden">
10      <div className="relative z-10 pb-8 sm:pb-16 md:pb-20 lg:max-w-2xl lg:w-full lg:pb-28 xl:pb-32">
11        <main className="mt-10 mx-auto max-w-7xl px-4 sm:mt-12 sm:px-6 md:mt-20 lg:px-8 xl:mt-28">
12          <div className="text-center lg:text-left">
13            <h1 className="text-4xl tracking-tight font-extrabold text-gray-900 sm:text-5xl md:text-6xl">
14              Academic Thesis</h1>
15              <span className="block text-blue-600 xl:inline">Portal</span>
16            </h1>
17            <p className="mt-3 text-base text-gray-500 sm:mt-5 sm:text-lg sm:max-w-xxl sm:mx-auto md:mt-5 md:text-xl lg:mt-8">
18              Discover, submit, and manage academic thesis in one comprehensive platform.
19              Connect with researchers, share knowledge, and contribute to academic excellence.
20            </p>
21            <div className="mt-5 sm:mt-8 sm:flex sm:justify-center lg:justify-start">
22              <div className="mt-3 sm:mt-0 sm:ml-3">
23                <Link
24                  to="/theses"
25                  className="w-full flex items-center justify-center px-8 py-3 border border-transparent text-base font-medium rounded-md text-white bg-blue-600 sm:px-12 sm:py-4 sm:font-semibold sm:outline-none sm:transition-all hover:bg-blue-700 focus-visible:outline-blue-500">
26                  Browse Thesis
27                </Link>
28              </div>
29            <div className="mt-3 sm:mt-0 sm:ml-3">
30              <Link
31                to="/register"
32                className="w-full flex items-center justify-center px-8 py-3 border border-transparent text-base font-medium rounded-md text-blue-700 sm:px-12 sm:py-4 sm:font-semibold sm:outline-none sm:transition-all hover:bg-blue-700 focus-visible:outline-blue-500">
33                Get Started
34              </Link>
35            </div>
36          </div>
37        </main>
38      </div>
39    </div>
40  );
41
42  export default LandingPage;
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

```

```

1 import React from 'react';
2 import { Link } from 'react-router-dom';
3 import SectionCarousel from './SectionCarousel';
4
5 const LandingPage = () => {
6   return (
7     <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100">
8       </> Hero Section </div>
9     <div className="relative overflow-hidden">
10      <div className="relative z-10 pb-8 sm:pb-16 md:pb-20 lg:max-w-2xl lg:w-full lg:pb-28 xl:pb-32">
11        <main className="mt-10 mx-auto max-w-7xl px-4 sm:mt-12 sm:px-6 md:mt-20 lg:px-8 xl:mt-28">
12          <div className="text-center lg:text-left">
13            <h1 className="text-4xl tracking-tight font-extrabold text-gray-900 sm:text-5xl md:text-6xl">
14              Academic Thesis</h1>
15              <span className="block text-blue-600 xl:inline">Portal</span>
16            </h1>
17            <p className="mt-3 text-base text-gray-500 sm:mt-5 sm:text-lg sm:max-w-xxl sm:mx-auto md:mt-5 md:text-xl lg:mt-8">
18              Discover, submit, and manage academic thesis in one comprehensive platform.
19              Connect with researchers, share knowledge, and contribute to academic excellence.
20            </p>
21            <div className="mt-5 sm:mt-8 sm:flex sm:justify-center lg:justify-start">
22              <div className="mt-3 sm:mt-0 sm:ml-3">
23                <Link
24                  to="/theses"
25                  className="w-full flex items-center justify-center px-8 py-3 border border-transparent text-base font-medium rounded-md text-white bg-blue-600 sm:px-12 sm:py-4 sm:font-semibold sm:outline-none sm:transition-all hover:bg-blue-700 focus-visible:outline-blue-500">
26                  Browse Thesis
27                </Link>
28              </div>
29            <div className="mt-3 sm:mt-0 sm:ml-3">
30              <Link
31                to="/register"
32                className="w-full flex items-center justify-center px-8 py-3 border border-transparent text-base font-medium rounded-md text-blue-700 sm:px-12 sm:py-4 sm:font-semibold sm:outline-none sm:transition-all hover:bg-blue-700 focus-visible:outline-blue-500">
33                Get Started
34              </Link>
35            </div>
36          </div>
37        </main>
38      </div>
39    </div>
40  );
41
42  export default LandingPage;
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

```

Finder File Edit View Go Window Help

LandingPage.js — Part_2

```

client > src > components > Index > JS LandingPage.js > ...
  5  const LandingPage = () => {
  6    > <div>
  7      > <Get Started>
  8      </div>
  9    </div>
 10   </div>
 11   </div>
 12   </div>
 13   <div className="main">
 14     <SectionCarousel />
 15   </div>
 16   <div className="bg-blue-600">
 17     <div className="max-w-2xl mx-auto text-center py-16 px-4 sm:py-20 sm:px-6 lg:px-8">
 18       <h2>Ready to get started?</h2>
 19       <span>Join our academic community today.</span>
 20     </div>
 21     <p>Start exploring thesis, submit your research, and connect with the academic community.</p>
 22   </div>
 23 </div>
 24 </div>
 25 </div>
 26 </div>
 27 </div>
 28 </div>
 29 </div>
 30 </div>
 31 </div>
 32 </div>
 33 </div>
 34 </div>
 35 </div>
 36 </div>
 37 </div>
 38 </div>
 39 </div>
 40 </div>
 41 </div>
 42 </div>
 43 </div>
 44 </div>
 45 <div>
 46   <SectionCarousel />
 47 </div>
 48 <div className="bg-blue-600">
 49   <div className="max-w-2xl mx-auto text-center py-16 px-4 sm:py-20 sm:px-6 lg:px-8">
 50     <h2>Ready to get started?</h2>
 51     <span>Join our academic community today.</span>
 52   </div>
 53   <p>Start exploring thesis, submit your research, and connect with the academic community.</p>
 54 </div>
 55 </div>
 56 </div>
 57 </div>
 58 </div>
 59 </div>
 60 </div>
 61 </div>
 62 </div>
 63 </div>
 64 </div>
 65 <export default LandingPage>;

```

Review next file >

Problems Output Debug Console Terminal Ports

node - server + node client node server

Cursor Tab Ln 1, Col 1 Spaces: 2 UTF-8 CRLF () JavaScript

Sun Aug 31 10:02 PM

Cursor File Edit Selection View Go Run Terminal Window Help

SectionCarousel.js — Part_2

```

client > src > components > Index > JS SectionCarousel.js > ...
  1  import React, { useState, useEffect } from "react";
  2  import ST3 from "../assets/img/ST3.jpg";
  3  import ST from "../assets/img/ST.png";
  4  import ST2 from "../assets/img/ST2.jpg";
  5
  6  const items = [
  7    {
  8      src: ST3,
  9      alt: "Academic Research",
 10      title: "Academic Research",
 11      description: "Discover groundbreaking research and scholarly work"
 12    },
 13    {
 14      src: ST,
 15      alt: "Thesis Management",
 16      title: "Thesis Management",
 17      description: "Efficiently manage and organize your academic thesis"
 18    },
 19    {
 20      src: ST2,
 21      alt: "Knowledge Sharing",
 22      title: "Knowledge Sharing",
 23      description: "Connect with researchers and share academic insights"
 24    }
 25  ];
 26
 27  function SectionCarousel() {
 28    const [activeIndex, setActiveIndex] = useState(0);
 29    const [isHovered, setIsHovered] = useState(false);
 30
 31    const nextSlide = () => {
 32      setActiveIndex(prevIndex) =>
 33        prevIndex === items.length - 1 ? 0 : prevIndex + 1
 34    };
 35  }
 36
 37 </div>

```

Review next file >

Problems Output Debug Console Terminal Ports

node - server + node client node server

Cursor Tab Ln 1, Col 1 Spaces: 2 UTF-8 CRLF () JavaScript

Sun Aug 31 10:02 PM

Cursor File Edit Selection View Go Run Terminal Window Help

SectionCarousel.js — Part_2

```
client > src > components > Index > JS SectionCarousel.js ...  
function SectionCarousel() {  
    return (  
        <div className="py-8 bg-white">  
            <div className="max-w-3xl mx-auto px-4 sm:px-6 lg:px-8">  
                /* PowerPoint-style Presentation Container */  
                <div  
                    className="relative bg-gray-100 rounded-lg shadow-lg overflow-hidden border"  
                    onMouseEnter={() => setIsHovered(true)}  
                    onMouseLeave={() => setIsHovered(false)}  
                    style={{ aspectRatio: '16/9', maxHeight: '400px' }}  
                >  
                /* Slide Counter */  
                <div className="absolute top-4 right-4 z-20 bg-black/70 text-white px-3 py-1 rounded-full text-sm font-medium">  
                    {(activeIndex + 1) / items.length}  
                </div>  
  
                /* Main Slide Area */  
                <div className="relative w-full h-full overflow-hidden bg-white">  
                    /* Only show the current active slide */  
                    <div className="w-full h-full">  
                        <img  
                            src={items[activeIndex].src}  
                            alt={items[activeIndex].alt}  
                            className="w-full h-full object-cover transition-opacity duration-500"  
                        />  
                    </div>  
  
                    /* Clean Title Overlay */  
                    <div className="absolute bottom-0 left-0 right-0 bg-gradient-to-t from-black/80 to-transparent p-6">  
                        <h3 className="text-white text-2xl md:text-3xl font-bold mb-2">  
                            {items[activeIndex].title}  
                        </h3>  
                        <p className="text-gray-200 text-lg">  
                            {items[activeIndex].description}  
                        </p>  
                    </div>  
                </div>  
            </div>  
        </div>  
    )  
}
```

Review next file >

node - server node client

```

client > src > components > Index > JS SectionCarousel.js > ...
e.js JS FacultyDashboard.js JS PDFSummarizerPage.js JS index.js JS SuggestionsDisplay.js JS Navbar.js JS SectionCarousel.js U ...
function SectionCarousel() {
    ...
    <button onClick={nextSlide}>
        ...
    </button>

    /* Progress Bar - PowerPoint Style */
    <div className="absolute right-2 top-1/2 transform -translate-y-1/2 bg-black/50 hover:bg-black/70 text-white p-2 rounded-md transition-all duration-300 ease-out">
        ...
    </div>

    /* Slide Indicators - PowerPoint Style */
    <div className="flex justify-center mt-6 space-x-2">
        items.map((_, index) => (
            <button key={index} onClick={() => goToSlide(index)}>
                ...
            </button>
        )
    )</div>
}

```

```

client > src > components > Index > JS SectionCarousel.js > ...
e.js JS FacultyDashboard.js JS PDFSummarizerPage.js JS index.js JS SuggestionsDisplay.js JS Navbar.js JS SectionCarousel.js U ...
function SectionCarousel() {
    ...
    /* Slide Information Panel */
    <div className="mt-8 bg-gray-50 rounded-lg p-6">
        <div className="text-center">
            <h2 className="text-2xl font-bold text-gray-900 mb-4">
                Academic Thesis Portal
            </h2>
            <div className="grid grid-cols-1 md:grid-cols-3 gap-6">
                items.map((item, index) => (
                    <div key={index} className="p-4 rounded-lg cursor-pointer transition-all duration-300 ${index === activeIndex ? 'bg-blue-100 border-2 border-blue-500 shadow-md' : 'bg-white hover:bg-gray-100 border border-gray-200'}">
                        ...
                        <div onClick={() => goToSlide(index)}>
                            ...
                        </div>
                        <div className="flex items-center mb-3">
                            <div className="w-8 h-8 rounded-full flex items-center justify-center text-white font-bold mr-3 ${index === activeIndex ? 'bg-blue-500' : 'bg-gray-400'}">
                                ...
                            </div>
                            <div>
                                <h3 className="font-semibold text-gray-900">{item.title}</h3>
                                <p className="text-sm text-gray-600">{item.description}</p>
                            </div>
                        </div>
                    </div>
                )</div>
            </div>
        </div>
    </div>
}

```

2. User Profile: The user profile display and editing interface, where users can view and modify their personal information. When a user navigates to their profile page, the frontend fetches their current profile data (including name, email, department, role, profile image, bio, and other details) from the backend API and displays it in a structured format. Users can edit their information by clicking edit buttons, which transforms the display into editable form

fields where they can update their bio, upload new profile pictures, change contact information, or modify other profile attributes. The frontend handles image uploads with preview functionality, validates form inputs, and sends updated data to the backend when users save changes.

The screenshot shows a Mac OS X desktop environment. In the top bar, there are icons for Finder, File, Edit, View, Go, Window, Help, and system status like battery level (90%) and date (Sun Aug 31 9:54PM). The main window is a code editor with tabs for JS files: public.js, Messages.js, Profile.js (the active tab), FacultyDashboard.js, PDFSummarizerPage.js, index.js, and Suggestion. The left sidebar shows a project outline with sections like PART_2, client, build, node_modules, public, src, components, admin, assets, auth, faculty, Index, message, FloatingMessageNotification.js, Messages.js, notifications, posts, CreatePost.js, PostCard.js, PostFeed.js, student, summarizer, thesis, PrivateRoute.js, Profile.js, SessionWarningModal.js, Toaster.js, context, hooks, App.js, index.js, and package-lock.json. The code editor displays a portion of Profile.js with logic for handling profile updates and image changes. At the bottom, there are tabs for OUTLINE, TIMELINE, Agents, Problems, Output, Debug Console, Terminal, and Ports. A status bar at the bottom right shows node - server, node-client, and other system information.

```
Profile.js — Part_2

client > src > components > JS Profile.js > [e] Profile
  const Profile = () => {
    const handleSubmit = async (e) => {
      // Validate CGPA if provided
      if (user.role === 'student' && formData.cgpa !== '') {
        const cgpa = parseFloat(formData.cgpa);
        if (isNaN(cgpa)) || cgpa < 0.0 || cgpa > 4.0) {
          setMessage('CGPA must be a number between 0.00 and 4.00');
          return;
        }
      }

      setLoading(true);
      setMessage('');
      const result = await updateProfile(formData);
      setLoading(false);
      if (result.success) {
        showSuccess('Profile updated successfully!');
        setEditMode(false);
      } else {
        showError(result.message);
        setMessage(result.message);
      }
    };

    const handleImageChange = (e) => {
      const file = e.target.files[0];
      if (file) {
        // Preview the image
        const reader = new FileReader();
        reader.onloadend = () => {
          setImagePreview(reader.result);
        };
        reader.readAsDataURL(file);
      }
    };
  };
}

const handleImageChange = (e) => {
  const file = e.target.files[0];
  if (file) {
    // Preview the image
    const reader = new FileReader();
    reader.onloadend = () => {
      setImagePreview(reader.result);
    };
    reader.readAsDataURL(file);
  }
};


```

```
Profile.js - Part_2
client > src > components > JS Profile.js > [e] Profile
  6  const Profile = () => {
  7    const handleImageUpload = async () => {
  8      setMessage('Please select an image file');
  9      return;
 10    }
 11
 12    setLoading(true);
 13    setMessage('');
 14    const result = await uploadProfileImage(file);
 15    setLoading(false);
 16    if (result.success) {
 17      showSuccess('Profile image updated successfully!');
 18      setImagePreview(null);
 19      inputFileRef.current.value = '';
 20    } else {
 21      showError(result.message);
 22      setMessage(result.message);
 23    }
 24  };
 25
 26  const handlePasswordChange = (e) => {
 27    setPasswordForm({ ...passwordForm, [e.target.name]: e.target.value });
 28  };
 29
 30  const handlePasswordSubmit = async (e) => {
 31    e.preventDefault();
 32
 33    // Validate passwords
 34    if (passwordForm.newPassword !== passwordForm.confirmPassword) {
 35      setMessage('New passwords do not match');
 36      return;
 37    }
 38
 39    if (passwordForm.newPassword.length < 6) {
 40      setMessage('New password must be at least 6 characters long');
 41      return;
 42    }
 43
 44    if (passwordForm.newPassword === passwordForm.oldPassword) {
 45      setMessage('New password cannot be the same as the old password');
 46      return;
 47    }
 48
 49    try {
 50      const response = await axios.post('/api/auth/change-password', {
 51        newPassword: passwordForm.newPassword
 52      }, {
 53        headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }
 54      });
 55
 56      showSuccess('Password changed successfully!');
 57      setPasswordForm({
 58        currentPassword: '',
 59        newPassword: '',
 60        confirmPassword: ''
 61      });
 62      setShowPasswordForm(false);
 63    } catch (error) {
 64      showError(error.response?.data?.message || 'Failed to change password');
 65      setMessage(error.response?.data?.message || 'Failed to change password');
 66    } finally {
 67      setPasswordLoading(false);
 68    }
 69  };
 70
 71  if (!user) {
 72    return (
 73      <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100 flex items-center justify-center">
 74        <div className="animate-spin rounded-full h-32 w-32 border-b-4 border-blue-600 border-t-transparent"></div>
 75      </div>
 76    );
 77  }
 78
 79  return (
 80    <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100 py-8 px-4">
 81      <div className="max-w-4xl mx-auto">
 82        /* Header */
 83        <div className="text-center mb-8">
 84          <h1 className="text-4xl font-bold text-gray-900 mb-2">My Profile</h1>
 85        </div>
 86        ...
 87      </div>
 88    </div>
 89  );
 90}
 91
 92
 93
 94
 95
 96
 97
 98
 99
100
101
102
103
104
105
```

```
Profile.js - Part_2
client > src > components > JS Profile.js > [e] Profile
  6  const Profile = () => {
  7    const handlePasswordSubmit = async (e) => {
  8      const response = await axios.post('/api/auth/change-password', {
  9        newPassword: passwordForm.newPassword
 10      }, {
 11        headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }
 12      });
 13
 14      showSuccess('Password changed successfully!');
 15      setPasswordForm({
 16        currentPassword: '',
 17        newPassword: '',
 18        confirmPassword: ''
 19      });
 20      setShowPasswordForm(false);
 21    } catch (error) {
 22      showError(error.response?.data?.message || 'Failed to change password');
 23      setMessage(error.response?.data?.message || 'Failed to change password');
 24    } finally {
 25      setPasswordLoading(false);
 26    }
 27  };
 28
 29  if (!user) {
 30    return (
 31      <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100 flex items-center justify-center">
 32        <div className="animate-spin rounded-full h-32 w-32 border-b-4 border-blue-600 border-t-transparent"></div>
 33      </div>
 34    );
 35  }
 36
 37  return (
 38    <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100 py-8 px-4">
 39      <div className="max-w-4xl mx-auto">
 40        /* Header */
 41        <div className="text-center mb-8">
 42          <h1 className="text-4xl font-bold text-gray-900 mb-2">My Profile</h1>
 43        </div>
 44        ...
 45      </div>
 46    </div>
 47  );
 48}
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
```

Profile.js — Part_2

```
client > src > components > JS Profile.js > [o] Profile
const Profile = () => {
  </div>

  <div className="grid grid-cols-1 lg:grid-cols-3 gap-8">
    /* Profile Image Section */
    <div className="lg:col-span-1">
      <div className="bg-white rounded-2xl shadow-lg p-6">
        <div className="text-center">
          <div className="relative inline-block cursor-pointer group" onClick={() => inputFileRef.current.click()}>
            <img
              src={imagePreview || (user.profileImage ? user.profileImage.startsWith("http") ? user.profileImage : 'http://localhost:5001/uploads/' + user.profileImage : 'https://ui-avatars.com/api/?name=${encodeURIComponent(user.name)}&size=200&background=3B82F6&color=fff&bold=true')}
              alt="Profile Picture"
              className="w-48 h-48 rounded-full object-cover object-center border-4 border-blue-100 shadow-lg mx-auto mb-4 transition-transform duration-300 group-hover:scale-105 group-focus:outline-none" style={{
                objectFit: 'cover',
                objectPosition: 'center center',
                minWidth: '192px',
                minHeight: '192px',
                maxWidth: '192px',
                maxHeight: '192px'
              }}
            >
            <onLoad={() => {
              // console.log('Profile image loaded successfully:', user.profileImage);
            }}>
            <onError={(e) => {
              // console.log('Profile image failed to load:', user.profileImage);
              e.target.src = 'https://ui-avatars.com/api/?name=${encodeURIComponent(user.name)}&size=200&background=3B82F6&color=fff&bold=true';
            }}>
            </div>
            <input
              ref={fileInputRef}
              type="file"
              accept="image/*"
              onChange={handleImageChange}
              className="hidden"
            />
          </div>
        </div>
      </div>
    </div>
  </div>
}

export default Profile;
```

Review next file >

node - server

⌘K to generate a command

Cursor Tab Ln 224, Col 69 (selected) Spaces: 2 UTF-8

The screenshot shows a Mac OS X desktop with a dark-themed interface. The top bar displays the system status with icons for battery (90%), signal strength, and network. The title bar of the active window, "Profile.js - Part_2", includes a tab bar with other files like "public.js", "Messages.js", "FacultyDashboard.js", "PDFSummarizerPage.js", "index.js", and "Suggestions".

The left sidebar shows the project structure under "PART_2". The "src" folder contains components, messages, notifications, posts, student, summarizer, thesis, and various utility files like "SessionWarningModal.js", "Toaster.js", "context", "css", "hooks", "App.js", and "Index.js".

The main editor area displays the "Profile.js" code, which handles profile picture uploads. It includes imports for "useEffect", "useState", and "useRef", along with "useFormik" from "formik". The component uses "React.FC<Props>" as its type. It features a file input field, a preview image, and a button for uploading. A tooltip "Review next file >" is visible near the bottom right of the editor.

At the bottom, there are tabs for "Problems", "Output", "Debug Console", "Terminal", and "Ports". The status bar at the bottom right shows "node - server" and "node client" with their respective ports (3001 and 3002). The bottom dock contains icons for Finder, Mail, Safari, and other applications.

A screenshot of a Mac OS X desktop environment. At the top, the Dock shows various application icons including Finder, Mail, Safari, and Visual Studio Code. The main window is a terminal window titled "node - server" with the identifier "node - 1". It displays a command history with several entries, including "git pull", "cd ..", and "node index.js". Below the terminal is a file browser window titled "Profile.js – Part_2" which is part of a project structure named "PART_2". The file browser lists numerous files and folders such as "client", "src", "node_modules", "public", "env", "JS public.js", "JS Messages.js", "JS Profile.js", "JS FacultyDashboard.js", "JS PDFSummarizerPage.js", "JS index.js", "JS Suggestions.js", "JS FloatingMessageNotification.js", "JS Messages.js", "JS notifications.js", "JS CreatePost.js", "JS PostCard.js", "JS PostFeed.js", "JS student.js", "JS summarizer.js", "JS thesis.js", "JS PrivateRoutes.js", "JS Profile.js", "JS SessionWarningModal.js", "JS Toaster.js", "JS context.js", "JS hooks.js", "JS App.js", "JS index.js", and "package-lock.json". The "Profile.js" file is currently open in the editor, showing its code. The status bar at the bottom right indicates the date and time as "Sun Aug 31 9:54PM".

```

const Profile = () => {
  </div>
  <div className="space-y-4">
    <div>
      <label>Role</label>
      <span>Student</span>
    </div>
    <div>
      <label>Student ID</label>
      <p>user.studentId || 'Not provided'</p>
    </div>
    <div>
      <label>Phone Number</label>
      <p>user.phone || 'Not provided'</p>
    </div>
    {user.role === 'student' && (
      <div>
        <label>CGPA</label>
        <p>user.cgpa ? `${user.cgpa.toFixed(2)}/4.00` : 'Not provided'</p>
      </div>
    )}
    <br>
    <div>
      <label>Member since</label>
      <p>new Date(user.createdAt).toLocaleDateString()</p>
    </div>
    {user.bio && (
      <div>
        <label>Bio</label>
        <p>text</p>
      </div>
    )}
  </div>

```

```

const Profile = () => {
  <input type="text" name="department" value={formData.department} onChange={handleChange} className="w-full px-4 py-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-transparent transition" required>
  <div>
    <label>Student ID</label>
    <input type="text" name="studentId" value={formData.studentId} onChange={handleChange} className="w-full px-4 py-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-transparent transition" required>
  </div>
  <div>
    <label>Phone Number</label>
    <input type="tel" name="phone" value={formData.phone} onChange={handleChange} className="w-full px-4 py-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-transparent transition" required>
  </div>
  {user.role === 'student' && (
    <div>
      <label>CGPA</label>
      <input type="number" name="cgpa" value={formData.cgpa} onChange={handleChange} className="w-full px-4 py-3 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-transparent transition" required>
    </div>
  )}

```

3. Messege: The messaging system works through a real-time chat interface where users can send and receive messages with other users. When a user types a message and sends it, the frontend captures the input, displays it immediately in the chat window, and sends the message data to the backend API. The system uses WebSocket connections (via Socket.io) to provide real-time updates, so when other users send messages, they appear instantly without

needing to refresh the page. The frontend handles message display with timestamps, user avatars, and different styling for sent vs. received messages. Users can view their message conversations, search other users.

```

client > src > components > message > JS Messages.js > [e] Messages > [e] sendMessage > [e] response
  1 import React, { useState, useEffect, useRef } from 'react';
  2 import axios from 'axios';
  3 import { useAuth } from './../../../context/AuthContext';
  4 import { useSocket } from './../../../context/SocketContext';
  5 import { useLocation } from 'react-router-dom';
  6
  7 const Messages = () => {
  8   const { user } = useAuth();
  9   const { socket, sendMessage: sendSocketMessage } = useSocket();
 10   const location = useLocation();
 11   const [conversations, setConversations] = useState([]);
 12   const [selectedConversation, setSelectedConversation] = useState(null);
 13   const [messages, setMessages] = useState([]);
 14   const [newMessage, setNewMessage] = useState('');
 15   const [loading, setLoading] = useState(true);
 16   const [sendingMessage, setSendingMessage] = useState(false);
 17   const [searchQuery, setSearchQuery] = useState('');
 18   const [searchResults, setSearchResults] = useState([]);
 19   const [showSearch, setSearchShow] = useState(false);
 20   const [searchLoading, setSearchLoading] = useState(false);
 21   const messagesEndRef = useRef(null);
 22   const messageInputRef = useRef(null);
 23   const [messageLoading, setMessageLoading] = useState(false);
 24   const [isTyping, setIsTyping] = useState(false);
 25   const [messageNotification, setMessageNotification] = useState(null);
 26   const [showNotificationPopup, setShowNotificationPopup] = useState(false);
 27   const [statusMessage, setStatusMessage] = useState('');
 28   const conversationStartedRef = useRef(false);
 29   const [startingConversation, setStartingConversation] = useState(false);

// Debug: Log when notification popup state changes
useEffect(() => {
  // console.log('Notification popup state changed:', showNotificationPopup);
  // console.log('Notification data:', messageNotification);
}, [showNotificationPopup, messageNotification]);
  
```



```

  31 // Get auth token
 32
 33 // Focus on the input field for better UX
 34 const inputElement = messageInputRef.current?.querySelector('input');
 35 if (inputElement) {
 36   inputElement.focus();
 37 }
 38, 100); // Small delay to ensure UI has updated
};

// Fetch conversations
const fetchConversations = async () => {
  try {
    const response = await axios.get('/api/messages/conversations', {
      headers: getAuthHeaders()
    });
    setConversations(response.data);
  } catch (error) {
    console.error('Error fetching conversations:', error);
  } finally {
    setLoading(false);
  }
};
  
```

A screenshot of a macOS desktop environment. At the top, there's a red dock bar with icons for various apps like Finder, Mail, and Safari. Below it is a dark-themed window for a code editor. The left sidebar shows a file tree with a complex project structure. The main area displays a large file named 'Messages.js — Part_2'. This file contains several functions for handling messages, including fetching conversations, marking messages as read, updating conversation status, and sending messages. It uses promises, async/await, and axios for API calls. The code editor has tabs for other files like 'Profile.js' and 'FacultyDashboard.js'. At the bottom, there are tabs for 'Output', 'Debug Console', 'Terminal', and 'Ports'. A small terminal window in the bottom right shows some command-line output. The system tray at the top right shows battery level (91%), signal strength, and the date/time (Sun Aug 31 9:47PM).

```
client > src > components > message > JS Messages.js > (e) Messages > (e) sendMessage > (e) response
const Messages = () => {
  const fetchMessages = async (conversationId) => {
    if (!conversationId) return;
    setMessageLoading(true);
    try {
      const response = await axios.get(`/api/messages/conversations/${conversationId}/messages`, {
        headers: getAuthHeaders()
      });
      setMessages(response.data.messages);
    } catch (error) {
      console.error('Error fetching messages:', error);
    } finally {
      setMessageLoading(false);
    }
  };
  // Trigger message update event for navbar
  window.dispatchEvent(new CustomEvent('messageUpdate'));
  // Update conversations to reflect read status
  fetchConversations();
} catch (error) {
  console.error('Error fetching messages:', error);
} finally {
  setMessageLoading(false);
}

// Send message
const sendMessage = async (e) => {
  e.preventDefault();
  if (!newMessage.trim() || !selectedConversation || sendingMe)
    let cleanedMessage = newMessage.trim();
    Review next file >
```

The screenshot shows a Mac OS X desktop environment with several open windows:

- Terminal Window:** The title bar says "node - server". It contains the following Node.js code:

```
const response = await axios.post('/api/messages/send', { receiverId, content: messageContent }, { headers: notAuthHeaders })
```

- Code Editor:** The title bar says "Messages.js — Part_2". The code is part of a file named "Messages.js" and includes logic for sending messages, handling user names, and validating message content.
- File Browser:** The sidebar shows the project structure with files like "client", "build", "node_modules", "public", "src", "components", "admin", "assets", "auth", "faculty", "index", "message", "FloatingMessageNotification.js", "Messages.js", "notifications", "posts", "CreatePost.js", "PostCard.js", "PostFeed.js", "student", "summarizer", "thesis", "PrivateRoute.js", "Profile.js", "SessionWarningModal.js", "Toaster.js", "context", "css", "hooks", "App.js", "index.js", and "package-lock.json".
- System Status:** The top right corner shows battery level (91%), signal strength, and the date/time (Sun Aug 31 9:47 PM).

Messages.js – Part_2

```

client > src > components > message > JS Messages.js > [e] Messages > [e] sendMessage > [e] response
  7 const Messages = () => {
  8   const sendMessage = async (e) => {
  9     ...
 10
 11     // Send real-time message via Socket.io
 12     if (socket) {
 13       sendSocketMessage(receiverId, response.data);
 14     }
 15
 16     // Refresh conversations to update last message
 17     fetchConversations();
 18   } catch (error) {
 19     console.error('Error sending message:', error);
 20     alert('Failed to send message. Please try again.');
 21   } finally {
 22     setSendMessage(false);
 23   }
 24 };
 25
 26 // Search users
 27 const searchUsers = async (query) => {
 28   if (!query || query.trim().length < 2) {
 29     setSearchResults([]);
 30     return;
 31   }
 32
 33   setSearchLoading(true);
 34   try {
 35     const response = await axios.get(`api/messages/search-users?q=${encodeURIComponent(query)}`);
 36     ...
 37   } catch (error) {
 38     ...
 39   }
 40   setSearchResults(response.data);
 41 } catch (error) {
 42   ...
 43   if (error.response?.status === 404) {
 44     setSearchResults([]);
 45   }
 46 }
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525
 526
 527
 528
 529
 530
 531
 532
 533
 534
 535
 536
 537
 538
 539
 540
 541
 542
 543
 544
 545
 546
 547
 548
 549
 550
 551
 552
 553
 554
 555
 556
 557
 558
 559
 559
 560
 561
 562
 563
 564
 565
 566
 567
 568
 569
 570
 571
 572
 573
 574
 575
 576
 577
 578
 579
 579
 580
 581
 582
 583
 584
 585
 586
 587
 588
 589
 589
 590
 591
 592
 593
 594
 595
 596
 597
 598
 599
 599
 600
 601
 602
 603
 604
 605
 606
 607
 608
 609
 609
 610
 611
 612
 613
 614
 615
 616
 617
 618
 619
 619
 620
 621
 622
 623
 624
 625
 626
 627
 628
 629
 629
 630
 631
 632
 633
 634
 635
 636
 637
 638
 639
 639
 640
 641
 642
 643
 644
 645
 646
 647
 648
 649
 649
 650
 651
 652
 653
 654
 655
 656
 657
 658
 659
 659
 660
 661
 662
 663
 664
 665
 666
 667
 668
 669
 669
 670
 671
 672
 673
 674
 675
 676
 677
 678
 679
 679
 680
 681
 682
 683
 684
 685
 686
 687
 688
 689
 689
 690
 691
 692
 693
 694
 695
 696
 697
 698
 699
 699
 700
 701
 702
 703
 704
 705
 706
 707
 708
 709
 709
 710
 711
 712
 713
 714
 715
 716
 717
 718
 719
 719
 720
 721
 722
 723
 724
 725
 726
 727
 728
 729
 729
 730
 731
 732
 733
 734
 735
 736
 737
 738
 739
 739
 740
 741
 742
 743
 744
 745
 746
 747
 748
 749
 749
 750
 751
 752
 753
 754
 755
 756
 757
 758
 759
 759
 760
 761
 762
 763
 764
 765
 766
 767
 768
 769
 769
 770
 771
 772
 773
 774
 775
 776
 777
 778
 779
 779
 780
 781
 782
 783
 784
 785
 786
 787
 788
 789
 789
 790
 791
 792
 793
 794
 795
 796
 797
 798
 799
 799
 800
 801
 802
 803
 804
 805
 806
 807
 808
 809
 809
 810
 811
 812
 813
 814
 815
 816
 817
 818
 819
 819
 820
 821
 822
 823
 824
 825
 826
 827
 828
 829
 829
 830
 831
 832
 833
 834
 835
 836
 837
 838
 839
 839
 840
 841
 842
 843
 844
 845
 846
 847
 848
 849
 849
 850
 851
 852
 853
 854
 855
 856
 857
 858
 859
 859
 860
 861
 862
 863
 864
 865
 866
 867
 868
 869
 869
 870
 871
 872
 873
 874
 875
 876
 877
 878
 879
 879
 880
 881
 882
 883
 884
 885
 886
 887
 888
 889
 889
 890
 891
 892
 893
 894
 895
 896
 897
 898
 899
 899
 900
 901
 902
 903
 904
 905
 906
 907
 908
 909
 909
 910
 911
 912
 913
 914
 915
 916
 917
 918
 919
 919
 920
 921
 922
 923
 924
 925
 926
 927
 928
 929
 929
 930
 931
 932
 933
 934
 935
 936
 937
 938
 939
 939
 940
 941
 942
 943
 944
 945
 946
 947
 948
 949
 949
 950
 951
 952
 953
 954
 955
 956
 957
 958
 959
 959
 960
 961
 962
 963
 964
 965
 966
 967
 968
 969
 969
 970
 971
 972
 973
 974
 975
 976
 977
 978
 979
 979
 980
 981
 982
 983
 984
 985
 986
 987
 988
 989
 989
 990
 991
 992
 993
 994
 995
 996
 997
 998
 999
 999
 1000
 1001
 1002
 1003
 1004
 1005
 1006
 1007
 1008
 1009
 1009
 1010
 1011
 1012
 1013
 1014
 1015
 1016
 1017
 1018
 1019
 1019
 1020
 1021
 1022
 1023
 1024
 1025
 1026
 1027
 1028
 1029
 1029
 1030
 1031
 1032
 1033
 1034
 1035
 1036
 1037
 1038
 1039
 1039
 1040
 1041
 1042
 1043
 1044
 1045
 1046
 1047
 1048
 1049
 1049
 1050
 1051
 1052
 1053
 1054
 1055
 1056
 1057
 1058
 1059
 1059
 1060
 1061
 1062
 1063
 1064
 1065
 1066
 1067
 1068
 1069
 1069
 1070
 1071
 1072
 1073
 1074
 1075
 1076
 1077
 1078
 1079
 1079
 1080
 1081
 1082
 1083
 1084
 1085
 1086
 1087
 1088
 1089
 1089
 1090
 1091
 1092
 1093
 1094
 1095
 1096
 1097
 1098
 1099
 1099
 1100
 1101
 1102
 1103
 1104
 1105
 1106
 1107
 1108
 1109
 1109
 1110
 1111
 1112
 1113
 1114
 1115
 1116
 1117
 1118
 1119
 1119
 1120
 1121
 1122
 1123
 1124
 1125
 1126
 1127
 1128
 1129
 1129
 1130
 1131
 1132
 1133
 1134
 1135
 1136
 1137
 1138
 1139
 1139
 1140
 1141
 1142
 1143
 1144
 1145
 1146
 1147
 1148
 1149
 1149
 1150
 1151
 1152
 1153
 1154
 1155
 1156
 1157
 1158
 1159
 1159
 1160
 1161
 1162
 1163
 1164
 1165
 1166
 1167
 1168
 1169
 1169
 1170
 1171
 1172
 1173
 1174
 1175
 1176
 1177
 1178
 1179
 1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1188
 1189
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1198
 1199
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1208
 1209
 1209
 1210
 1211
 1212
 1213
 1214
 1215
 1216
 1217
 1218
 1219
 1219
 1220
 1221
 1222
 1223
 1224
 1225
 1226
 1227
 1228
 1229
 1229
 1230
 1231
 1232
 1233
 1234
 1235
 1236
 1237
 1238
 1239
 1239
 1240
 1241
 1242
 1243
 1244
 1245
 1246
 1247
 1248
 1249
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257
 1258
 1259
 1259
 1260
 1261
 1262
 1263
 1264
 1265
 1266
 1267
 1268
 1269
 1269
 1270
 1271
 1272
 1273
 1274
 1275
 1276
 1277
 1278
 1279
 1279
 1280
 1281
 1282
 1283
 1284
 1285
 1286
 1287
 1288
 1289
 1289
 1290
 1291
 1292
 1293
 1294
 1295
 1296
 1297
 1298
 1299
 1299
 1300
 1301
 1302
 1303
 1304
 1305
 1306
 1307
 1308
 1309
 1309
 1310
 1311
 1312
 1313
 1314
 1315
 1316
 1317
 1318
 1319
 1319
 1320
 1321
 1322
 1323
 1324
 1325
 1326
 1327
 1328
 1329
 1329
 1330
 1331
 1332
 1333
 1334
 1335
 1336
 1337
 1338
 1339
 1339
 1340
 1341
 1342
 1343
 1344
 1345
 1346
 1347
 1348
 1349
 1349
 1350
 1351
 1352
 1353
 1354
 1355
 1356
 1357
 1358
 1359
 1359
 1360
 1361
 1362
 1363
 1364
 1365
 1366
 1367
 1368
 1369
 1369
 1370
 1371
 1372
 1373
 1374
 1375
 1376
 1377
 1378
 1379
 1379
 1380
 1381
 1382
 1383
 1384
 1385
 1386
 1387
 1388
 1388
 1389
 1390
 1391
 1392
 1393
 1394
 1395
 1396
 1397
 1398
 1399
 1399
 1400
 1401
 1402
 1403
 1404
 1405
 1406
 1407
 1408
 1409
 1409
 1410
 1411
 1412
 1413
 1414
 1415
 1416
 1417
 1418
 1419
 1419
 1420
 1421
 1422
 1423
 1424
 1425
 1426
 1427
 1428
 1429
 1429
 1430
 1431
 1432
 1433
 1434
 1435
 1436
 1437
 1438
 1439
 1439
 1440
 1441
 1442
 1443
 1444
 1445
 1446
 1447
 1448
 1449
 1449
 1450
 1451
 1452
 1453
 1454
 1455
 1456
 1457
 1458
 1459
 1459
 1460
 1461
 1462
 1463
 1464
 1465
 1466
 1467
 1468
 1469
 1469
 1470
 1471
 1472
 1473
 1474
 1475
 1476
 1477
 1478
 1479
 1479
 1480
 1481
 1482
 1483
 1484
 1485
 1486
 1487
 1488
 1489
 1489
 1490
 1491
 1492
 1493
 1494
 1495
 1496
 1497
 1498
 1499
 1499
 1500
 1501
 1502
 1503
 1504
 1505
 1506
 1507
 1508
 1509
 1509
 1510
 1511
 1512
 1513
 1514
 1515
 1516
 1517
 1518
 1519
 1519
 1520
 1521
 1522
 1523
 1524
 1525
 1526
 1527
 1528
 1529
 1529
 1530
 1531
 1532
 1533
 1534
 1535
 1536
 1537
 1538
 1539
 1539
 1540
 1541
 1542
 1543
 1544
 1545
 1546
 1547
 1548
 1549
 1549
 1550
 1551
 1552
 1553
 1554
 1555
 1556
 1557
 1558
 1559
 1559
 1560
 1561
 1562
 1563
 1564
 1565
 1566
 1567
 1568
 1569
 1569
 1570
 1571
 1572
 1573
 1574
 1575
 1576
 1577
 1578
 1579
 1579
 1580
 1581
 1582
 1583
 1584
 1585
 1586
 1587
 1588
 1589
 1589
 1590
 1591
 1592
 1593
 1594
 1595
 1596
 1597
 1598
 1599
 1599
 1600
 1601
 1602
 1603
 1604
 1605
 1606
 1607
 1608
 1609
 1609
 1610
 1611
 1612
 1613
 1614
 1615
 1616
 1617
 1618
 1619
 1619
 1620
 1621
 1622
 1623
 1624
 1625
 1626
 1627
 1628
 1629
 1629
 1630
 1631
 1632
 1633
 1634
 1635
 1636
 1637
 1638
 1639
 1639
 1640
 1641
 1642
 1643
 1644
 1645
 1646
 1647
 1648
 1649
 1649
 1650
 1651
 1652
 1653
 1654
 1655
 1656
 1657
 1658
 1659
 1659
 1660
 1661
 1662
 1663
 1664
 1665
 1666
 1667
 1668
 1669
 1669
 1670
 1671
 1672
 1673
 1674
 1675
 1676
 1677
 1678
 1679
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688
 1689
 1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1707
 1708
 1709
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1718
 1719
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 1727
 1728
 1729
 1729
 1730
 1731
 1732
 1733
 1734
 1735
 1736
 1737
 1738
 1739
 1739
 1740
 1741
 1742
 1743
 1744
 1745
 1746
 1747
 1748
 1749
 1749
 1750
 1751
 1752
 1753
 1754
 1755
 1756
 1757
 1758
 1759
 1759
 1760
 1761
 1762
 1763
 1764
 1765
 1766
 1767
 1768
 1769
 1769
 1770
 1771
 1772
 1773
 1774
 1775
 1776
 1777
 1778
 1779
 1779
 1780
 1781
 1782
 1783
 1784
 1785
 1786
 1787
 1788
 1789
 1789
 1790
 1791
 1792
 1793
 1794
 1795
 1796
 1797
 1798
 1799
 1799
 1800
 1801
 1802
 1803
 1804
 1805
 1806
 1807
 1808
 1809
 1809
 1810
 1811
 1812
 1813
 1814
 1815
 1816
 1817
 1818
 1819
 1819
 1820
 1821
 1822
 1823
 1824
 1825
 1826
 1827
 1828
 1829
 1829
 1830
 1831
 1832
 1833
 1834
 1835
 1836
 1837
 1838
 1839
 1839
 1840
 1841
 1842
 1843
 1844
 1845
 1846
 1847
 1848
 1849
 1849
 1850
 1851
 1852
 1853
 1854
 1855
 1856
 1857
 1858
 1859
 1859
 1860
 1861
 1862
 1863
 1864
 1865
 1866
 1867
 1868
 1869
 1869
 1870
 1871
 1872
 1873
 1874
 1875
 1876
 1877
 1878
 1879
 1879
 1880
 1881
 1882
```

Messages.js – Part_2

```

client > src > components > message > JS Messages.js > [e] Messages > [e] sendMessage > [e] response
env U, e JS public.js U JS Messages.js X JS Profile.js JS FacultyDashboard.js JS PDFSummarizerPage.js JS index.js JS Suggestions.js ...
const Messages = () => {
  const startConversation = async (userId) => {
    // Add new conversation to the top of the list
    setConversations(prev => [newConversation, ...prev]);
    setSelectedConversation(newConversation);
  } else {
    // Update existing conversation but mark as new chat
    const updatedConv = { ...existingConv, isNewChat: true };
    setConversations(prev => prev.map(conv =>
      conv.conversationId === conversationId ? updatedConv : conv
    ));
    setSelectedConversation(updatedConv);
  }

  // Clear search interface
  setShowSearch(false);
  setSearchQuery('');
  setSearchResults([]);
}

// Start with empty messages for new chat experience
setMessages([]);
setMessageLoading(false);

// Don't fetch previous messages to maintain "new chat" experience
// User can see previous messages if they select the conversation from sidebar later
// Scroll to message input box for new conversation
scrollToMessageInput();

} catch (error) {
  console.error('Error starting conversation:', error);
  setStatusMessage('Failed to start conversation. Please try again.');
  setTimeout(() => setStatusMessage('', 3000));
} finally {
  setStartingConversation(false);
}
};


```

Review next file >

Problems Output Debug Console Terminal Ports

node - server + - node - allack

Messages.js – Part_2

```

client > src > components > message > JS Messages.js > [e] Messages > [e] sendMessage > [e] response
env U, e JS public.js U JS Messages.js X JS Profile.js JS FacultyDashboard.js JS PDFSummarizerPage.js JS index.js JS Suggestions.js ...
const Messages = () => {
  const formatTime = (timestamp) => {
    return date.toLocaleDateString([], { month: 'short', day: 'numeric' });
  };

  // Load conversations on component mount
  useEffect(() => {
    fetchConversations();
  });

  // Request notification permission
  if (Notification.permission === 'default') {
    Notification.requestPermission().then(permission => {
      // console.log('Notification permission:', permission);
    });
  }

  // Cleanup function to reset ref when component unmounts
  return () => {
    conversationStartedRef.current = false;
  }, [ ]);

  // Socket.io event listeners
  useEffect(() => {
    if (socket) {
      // Listen for new messages
      socket.on('receive_message', (message) => {
        // Show popup notification if message is not from current user
        if (message.sender_id !== user.id) {
          setMessageNotification({
            id: Date.now(),
            senderName: message.sender.name,
            senderImage: message.sender.profileImage,
          });
        }
      });
    }
  }, [ ]);


```

Review next file >

Problems Output Debug Console Terminal Ports

node - server + - node - allack

Messages.js – Part_2

```

client > src > components > message > JS Messages.js > [e] Messages > [e] sendMessage > [e] response
    7  const Messages = () => {
    8    387    useEffect(() => {
    9      388      socket.on('user_typer', ({ isTyping: userIsTyping }) => {
   10        389        // Cleanup listeners
   11        390        return () => {
   12          391          socket.off('receive_message');
   13          392          socket.off('user_typer');
   14        };
   15      }, [socket, selectedConversation]);
   16    );
   17    393    // Handle navigation from faculty list
   18    394    useEffect(() => {
   19      395      if (!location.state?.startConversation) {
   20        396        const { conversationId, otherUser } = location.state.startConversation;
   21
   22        // Check if conversation already exists in our list
   23        let existingConv = conversations.find(conv => conv.conversationId === conversationId);
   24
   25        if (!existingConv) {
   26          // Create a new conversation entry
   27          existingConv = {
   28            conversationId,
   29            otherUser,
   30            lastMessage: null,
   31            unreadCount: 0
   32          };
   33          setConversations(prev => [existingConv, ...prev]);
   34        }
   35
   36        // Select this conversation
   37        setSelectedConversation(existingConv);
   38        fetchMessages(conversationId);
   39
   40        // Scroll to message input box
   41        scrollToMessageInputBox();
   42      }
   43    );
   44    397    // Select this conversation
   45    398    setSelectedConversation(existingConv);
   46    399    fetchMessages(conversationId);
   47
   48    // Scroll to message input box
   49    scrollToMessageInputBox();
  
```

Review next file >

node - server

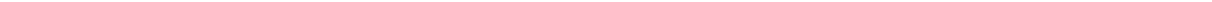
Messages.js – Part_2

```

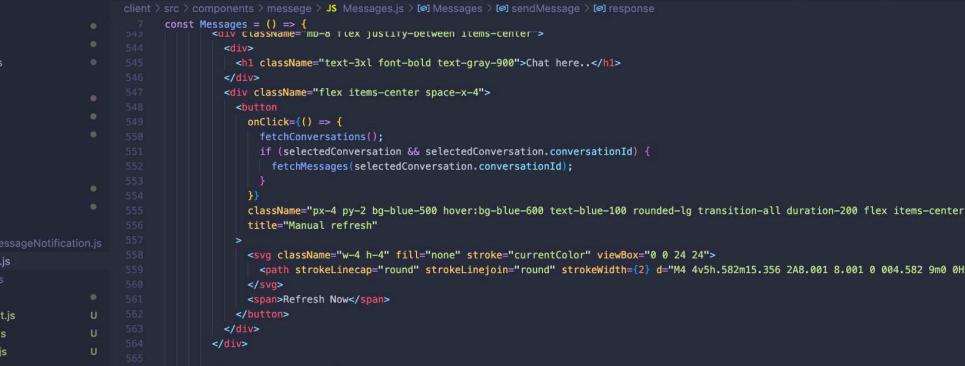
client > src > components > message > JS Messages.js > [e] Messages > [e] sendMessage > [e] response
    7  const Messages = () => {
    8    455    if (loading) {
    9      456      return (
   10        457          <div className="flex justify-center items-center h-screen bg-gradient-to-br from-blue-50 to-indigo-100">
   11          458              <div className="text-center">
   12                  <div className="w-24 mx-auto mb-6 bg-white rounded-full flex items-center justify-center shadow-lg">
   13                      <div className="animate-spin rounded-full h-12 w-12 border-4 border-blue-200 border-t-blue-600"></div>
   14                  <h3 className="text-xxl font-semibold text-gray-700 mb-2">Loading Messages</h3>
   15                  <p className="text-gray-500">Please wait while we set up your messaging experience...</p>
   16              </div>
   17          </div>
   18      );
   19    }
   20
   21    return (
   22        <div className="min-h-screen bg-gray-50 py-8">
   23            /* Message Notification Popup */
   24            &gt; showNotificationPopup && messageNotification && (
   25                <div className="fixed top-4 right-4 z-[9999] animate-bounce" style={{ zIndex: 9999 }}>
   26                    <div className="w-5 h-5 bg-white border border-gray-200 rounded-full shadow-2xl max-w-sm w-full overflow-hidden">
   27                        /* Header */
   28                        <div className="bg-gradient-to-r from-blue-500 to-indigo-600 px-4 py-2 text-white rounded-t-full">
   29                            <div className="flex items-center justify-between">
   30                                <div className="flex items-center space-x-2">
   31                                    <svg className="w-5 h-5 fill:none stroke=currentColor viewBox='0 0 24 24'">
   32                                        <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M8 12h.01M12 12h.01M16 12h.01M21 12c 4.418-4.03 8-9 8s-9-3.582-9" />
   33                                    </svg>
   34                                    <span className="font-semibold text-sm">New Message</span>
   35                                </div>
   36                                <button onClick={() => setShowNotificationPopup(false)} className="text-white/80 hover:text-white transition-colors">
   37                                    <svg className="w-4 h-4 fill:none stroke=currentColor" viewBox='0 0 24 24'>
   38                                        <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M6 18L18 6M 6L12 12" />
   39                                    </svg>
   40                            </div>
   41                        </div>
   42                    </div>
   43                </div>
   44            )
   45            &gt; showNotificationPopup && messageNotification && (
   46                <div className="fixed top-4 right-4 z-[9999] animate-bounce" style={{ zIndex: 9999 }}>
   47                    <div className="w-5 h-5 bg-white border border-gray-200 rounded-full shadow-2xl max-w-sm w-full overflow-hidden">
   48                        /* Header */
   49                        <div className="bg-gradient-to-r from-blue-500 to-indigo-600 px-4 py-2 text-white rounded-t-full">
   50                            <div className="flex items-center justify-between">
   51                                <div className="flex items-center space-x-2">
   52                                    <svg className="w-5 h-5 fill:none stroke=currentColor viewBox='0 0 24 24'">
   53                                        <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M8 12h.01M12 12h.01M16 12h.01M21 12c 4.418-4.03 8-9 8s-9-3.582-9" />
   54                                    </svg>
   55                                    <span className="font-semibold text-sm">New Message</span>
   56                                </div>
   57                                <button onClick={() => setShowNotificationPopup(false)} className="text-white/80 hover:text-white transition-colors">
   58                                    <svg className="w-4 h-4 fill:none stroke=currentColor" viewBox='0 0 24 24'>
   59                                        <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M6 18L18 6M 6L12 12" />
   60                                    </svg>
   61                            </div>
   62                        </div>
   63                    </div>
   64                </div>
   65            )
   66        )
   67    );
   68
   69    // Select this conversation
   70    setSelectedConversation(existingConv);
   71
   72    // Scroll to message input box
   73    scrollToMessageInputBox();
  
```

Review next file >

node - server



A screenshot of a Mac OS X desktop environment. The main focus is a code editor window titled "Messages.js – Part_2". The file contains JSX code for rendering a message notification component. The code includes imports from "messageNotification" and "sendMessage", and uses semantic HTML classes like "text-white", "text-black", and "font-semibold". A "Review next file >" button is visible at the bottom right of the code editor. Below the code editor is a toolbar with icons for Problems, Output, Debug Console, Terminal, and Ports. To the right of the code editor is a terminal window titled "node - server" showing command-line history. The desktop dock at the bottom has icons for various applications including Finder, Mail, Safari, and others. The system status bar at the top right shows the date as "Sun Aug 31 9:48 PM" and battery level as "91%".



A screenshot of a Mac OS X desktop environment. At the top, there's a red dock bar with icons for various applications like Finder, Mail, and Safari. The main screen shows a terminal window with a Node.js application running, displaying logs such as 'node - server' and 'node client'. A floating message notification from 'Messages' is visible, prompting the user to 'Review next file'. In the background, a file browser window is open, showing a project structure with files like 'Messages.js' and 'Profile.js'. The status bar at the bottom shows network connectivity, battery level (91%), and the date and time (Sun Aug 31 9:48PM).

```
const Messages = () =>
  <div className="flex justify-between items-center">
    <div>
      <h1>Chat here...</h1>
    </div>
    <button onClick={() => {
      fetchConversations();
      if (selectedConversation && selectedConversation.conversationId) {
        fetchMessages(selectedConversation.conversationId);
      }
    }}>
      Manual refresh
    </button>
  </div>
  <span>Refresh Now</span>
</div>
</div>

/* Status Message */
(statusMessage && (
  <div className="mb-4 bg-blue-50 border border-blue-200 rounded-lg p-4">
    <div className="flex items-center">
      <div className="flex-shrink-0">
        <svg className="animate-spin h-5 w-5 text-blue-600" fill="none" viewBox="0 0 24 24">
          <circle className="opacity-25" cx="12" cy="12" r="10" stroke="currentColor" strokeWidth="4"/>
          <path className="opacity-75" fill="currentColor" d="M4 12a8 8 0 1 0 16 0v1a8 8 0 1 0 -16 0V12z M4 12a5 5 0 1 1 0 10 5 5 0 1 1 0 -10z M9.5 3.5a1 1 0 1 1 2 0v.5a1 1 0 1 1 -2 0V3.5z" />
        </svg>
      </div>
      <div className="ml-3">
        <p>Review next file >></p>
      </div>
    </div>
  </div>
))
```

A screenshot of a macOS desktop environment. The top bar shows the date and time as "Sun Aug 31 9:48PM". The main window is a Visual Studio Code editor displaying a file named "Messages.js – Part_2". The code is a component definition for a messaging application, containing JSX and some JavaScript logic for handling user search queries and rendering a search input field. A floating message notification from "FloatingMessageNotification.js" is visible in the center of the screen, prompting the user to "Review next file". The left sidebar shows a project structure with various files like "client", "build", "node_modules", "public", "src", "components", "admin", "assets", "auth", "faculty", "index", "message", "notifications", "posts", "CreatePost.js", "PostCard.js", "PostFeed.js", "student", "summarizer", "thesis", "PrivateRoute.js", "Profile.js", "SessionWarningModal.js", "Toaster.js", "context", "css", "hooks", "App.js", "index.js", and "package-lock.json". The bottom navigation bar includes tabs for "Problems", "Output", "Debug Console", "Terminal", and "Ports". A status bar at the bottom right shows "node - server" and other system information.

4.Post: The post works through a form-based interface where users can input text content and optionally attach images. When a user types in the post content area and clicks the submit button, the frontend captures the form data, validates the input, and then sends a POST request to the backend API endpoint (typically '/api/posts') with the post data. The frontend handles the user experience by showing loading states during submission, displaying

success/error messages via the toaster system, and automatically refreshing the post feed to show the newly created post. The component also manages image previews, character limits, and form validation to ensure users can only submit valid posts with proper content.

```

 1 import React, { useState } from 'react';
 2 import { useNavigate } from 'react-router-dom';
 3 import { useAuth } from '../Context/AuthContext';
 4 import { useToaster } from '../Toaster';
 5 import axios from 'axios';
 6
 7 const CreatePost = () => {
 8   const { user } = useAuth();
 9   const navigate = useNavigate();
10   const [showSuccess, showError] = useState('');
11   const [content, setContent] = useState('');
12   const [loading, setLoading] = useState(false);
13   const [error, setError] = useState('');
14
15   const handleSubmit = async (e) => {
16     e.preventDefault();
17
18     if (!content.trim()) {
19       showError('Please write something to post');
20       return;
21     }
22
23     if (content.length > 2000) {
24       showError('Post content cannot exceed 2000 characters');
25       return;
26     }
27
28     // Check if the post contains the user's name
29     const userName = user?.name;
30     if (userName && content.toLowerCase().includes(userName.toLowerCase())) {
31       showError('Please don\'t include your name in the post. Your name will be displayed automatically.');
32       return;
33     }
34
35     setLoading(true);
36     setError('');
37
38   }
39
40   return (
41     <div>
42       <form>
43         <input type="text" value={content} onChange={(e) => setContent(e.target.value)} />
44         <button type="submit" onClick={handleSubmit}>Post</button>
45       </form>
46     </div>
47   );
48 }
49
50
51 const CreatePost = () => {
52   const handleSubmit = async (e) => {
53     e.preventDefault();
54     showSuccess('Post created successfully!');
55     navigate('/posts');
56   } catch (error) {
57     console.error('Error creating post:', error);
58     showError(error.response?.data?.message || 'Failed to create post');
59   } finally {
60     setLoading(false);
61   }
62
63   if (user?.role !== 'student') {
64     return (
65       <div>
66         <div>
67           <h3>Access Denied</h3>
68           <p>Only students can create posts.</p>
69         </div>
70       </div>
71     );
72   }
73
74   return (
75     <div>
76       <h3>Post created successfully!</h3>
77       <p>Your post has been published. You can now see it on the main feed.</p>
78     </div>
79   );
80 }
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
5540
5541
5542
5543
5544
5545
5546
5547
5548
5549
55410
55411
55412
55413
55414
55415
55416
55417
55418
55419
55420
55421
55422
55423
55424
55425
55426
55427
55428
55429
55430
55431
55432
55433
55434
55435
55436
55437
55438
55439
55440
55441
55442
55443
55444
55445
55446
55447
55448
55449
55450
55451
55452
55453
55454
55455
55456
55457
55458
55459
55460
55461
55462
55463
55464
55465
55466
55467
55468
55469
55470
55471
55472
55473
55474
55475
55476
55477
55478
55479
55480
55481
55482
55483
55484
55485
55486
55487
55488
55489
55490
55491
55492
55493
55494
55495
55496
55497
55498
55499
554100
554101
554102
554103
554104
554105
554106
554107
554108
554109
554110
554111
554112
554113
554114
554115
554116
554117
554118
554119
554120
554121
554122
554123
554124
554125
554126
554127
554128
554129
554130
554131
554132
554133
554134
554135
554136
554137
554138
554139
554140
554141
554142
554143
554144
554145
554146
554147
554148
554149
554150
554151
554152
554153
554154
554155
554156
554157
554158
554159
554160
554161
554162
554163
554164
554165
554166
554167
554168
554169
554170
554171
554172
554173
554174
554175
554176
554177
554178
554179
554180
554181
554182
554183
554184
554185
554186
554187
554188
554189
554190
554191
554192
554193
554194
554195
554196
554197
554198
554199
554200
554201
554202
554203
554204
554205
554206
554207
554208
554209
554210
554211
554212
554213
554214
554215
554216
554217
554218
554219
554220
554221
554222
554223
554224
554225
554226
554227
554228
554229
554230
554231
554232
554233
554234
554235
554236
554237
554238
554239
554240
554241
554242
554243
554244
554245
554246
554247
554248
554249
554250
554251
554252
554253
554254
554255
554256
554257
554258
554259
554260
554261
554262
554263
554264
554265
554266
554267
554268
554269
554270
554271
554272
554273
554274
554275
554276
554277
554278
554279
554280
554281
554282
554283
554284
554285
554286
554287
554288
554289
554290
554291
554292
554293
554294
554295
554296
554297
554298
554299
554300
554301
554302
554303
554304
554305
554306
554307
554308
554309
554310
554311
554312
554313
554314
554315
554316
554317
554318
554319
554320
554321
554322
554323
554324
554325
554326
554327
554328
554329
554330
554331
554332
554333
554334
554335
554336
554337
554338
554339
554340
554341
554342
554343
554344
554345
554346
554347
554348
554349
554350
554351
554352
554353
554354
554355
554356
554357
554358
554359
554360
554361
554362
554363
554364
554365
554366
554367
554368
554369
554370
554371
554372
554373
554374
554375
554376
554377
554378
554379
554380
554381
554382
554383
554384
554385
554386
554387
554388
554389
554390
554391
554392
554393
554394
554395
554396
554397
554398
554399
554400
554401
554402
554403
554404
554405
554406
554407
554408
554409
554410
554411
554412
554413
554414
554415
554416
554417
554418
554419
554420
554421
554422
554423
554424
554425
554426
554427
554428
554429
554430
554431
554432
554433
554434
554435
554436
554437
554438
554439
554440
554441
554442
554443
554444
554445
554446
554447
554448
554449
554450
554451
554452
554453
554454
554455
554456
554457
554458
554459
554460
554461
554462
554463
554464
554465
554466
554467
554468
554469
554470
554471
554472
554473
554474
554475
554476
554477
554478
554479
554480
554481
554482
554483
554484
554485
554486
554487
554488
554489
554490
554491
554492
554493
554494
554495
554496
554497
554498
554499
554500
554501
554502
554503
554504
554505
554506
554507
554508
554509
5545010
5545011
5545012
5545013
5545014
5545015
5545016
5545017
5545018
5545019
55450100
55450101
55450102
55450103
55450104
55450105
55450106
55450107
55450108
55450109
55450110
55450111
55450112
55450113
55450114
55450115
55450116
55450117
55450118
55450119
554501100
554501101
554501102
554501103
554501104
554501105
554501106
554501107
554501108
554501109
554501110
554501111
554501112
554501113
554501114
554501115
554501116
554501117
554501118
554501119
5545011100
5545011101
5545011102
5545011103
5545011104
5545011105
5545011106
5545011107
5545011108
5545011109
5545011110
5545011111
5545011112
5545011113
5545011114
5545011115
5545011116
5545011117
5545011118
5545011119
55450111100
55450111101
55450111102
55450111103
55450111104
55450111105
55450111106
55450111107
55450111108
55450111109
55450111110
55450111111
55450111112
55450111113
55450111114
55450111115
55450111116
55450111117
55450111118
55450111119
554501111100
554501111101
554501111102
554501111103
554501111104
554501111105
554501111106
554501111107
554501111108
554501111109
554501111110
554501111111
554501111112
554501111113
554501111114
554501111115
554501111116
554501111117
554501111118
554501111119
5545011111100
5545011111101
5545011111102
5545011111103
5545011111104
5545011111105
5545011111106
5545011111107
5545011111108
5545011111109
5545011111110
5545011111111
5545011111112
5545011111113
5545011111114
5545011111115
5545011111116
5545011111117
5545011111118
5545011111119
55450111111100
55450111111101
55450111111102
55450111111103
55450111111104
55450111111105
55450111111106
55450111111107
55450111111108
55450111111109
55450111111110
55450111111111
55450111111112
55450111111113
55450111111114
55450111111115
55450111111116
55450111111117
55450111111118
55450111111119
554501111111100
554501111111101
554501111111102
554501111111103
554501111111104
554501111111105
554501111111106
554501111111107
554501111111108
554501111111109
554501111111110
554501111111111
554501111111112
554501111111113
554501111111114
554501111111115
554501111111116
554501111111117
554501111111118
554501111111119
5545011111111100
5545011111111101
5545011111111102
5545011111111103
5545011111111104
5545011111111105
5545011111111106
5545011111111107
5545011111111108
5545011111111109
5545011111111110
5545011111111111
5545011111111112
5545011111111113
5545011111111114
5545011111111115
5545011111111116
5545011111111117
5545011111111118
5545011111111119
55450111111111100
55450111111111101
55450111111111102
55450111111111103
55450111111111104
55450111111111105
55450111111111106
55450111111111107
55450111111111108
55450111111111109
55450111111111110
55450111111111111
55450111111111112
55450111111111113
55450111111111114
55450111111111115
55450111111111116
55450111111111117
55450111111111118
55450111111111119
554501111111111100
554501111111111101
554501111111111102
554501111111111103
554501111111111104
554501111111111105
554501111111111106
554501111111111107
554501111111111108
554501111111111109
554501111111111110
554501111111111111
554501111111111112
554501111111111113
554501111111111114
554501111111111115
554501111111111116
554501111111111117
554501111111111118
554501111111111119
5545011111111111100
5545011111111111101
5545011111111111102
5545011111111111103
5545011111111111104
5545011111111111105
5545011111111111106
5545011111111111107
5545011111111111108
5545011111111111109
5545011111111111110
5545011111111111111
5545011111111111112
5545011111111111113
5545011111111111114
5545011111111111115
5545011111111111116
5545011111111111117
5545011111111111118
5545011111111111119
55450111111111111100
55450111111111111101
55450111111111111102
55450111111111111103
55450111111111111104
55450111111111111105
55450111111111111106
55450111111111111107
55450111111111111108
55450111111111111109
55450111111111111110
55450111111111111111
55450111111111111112
55450111111111111113
55450111111111111114
55450111111111111115
55450111111111111116
55450111111111111117
55450111111111111118
55450111111111111119
554501111111111111100
554501111111111111101
554501111111111111102
554501111111111111103
554501111111111111104
554501111111111111105
554501111111111111106
554501111111111111107
554501111111111111108
554501111111111111109
554501111111111111110
554501111111111111111
554501111111111111112
554501111111111111113
554501111111111111114
554501111111111111115
554501111111111111116
554501111111111111117
554501111111111111118
554501111111111111119
5545011111111111111100
5545011111111111111101
5545011111111111111102
5545011111111111111103
5545011111111111111104
5545011111111111111105
5545011111111111111106
5545011111111111111107
5545011111111111111108
5545011111111111111109
5545011111111111111110
55450111111
```

A screenshot of a Mac OS X desktop environment. At the top is a Dock with various application icons. The main window is a terminal application showing a Node.js script named `CreatePost.js`. The script contains code for creating a new post, including a form with fields for title, content, and user info, and logic for handling file uploads. Below the terminal is a file browser showing a project structure with files like `client`, `public`, `src`, and `posts`. The `posts` folder contains files such as `CreatePost.js`, `PostCard.js`, and `PostFeed.js`. The bottom of the screen shows a dock of application icons.

```
const CreatePost = () => {
  <div className="bg-white rounded-xl shadow-lg border border-gray-200 overflow-hidden">
    </> Header
    <div className="bg-gradient-to-r from-blue-600 to-indigo-600 px-6 py-4">
      <h2 className="text-2xl font-bold text-white flex items-center">
        <svg className="w-6 h-6 mr-3" fill="none" stroke="currentColor" viewBox="0 24 24">
          <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M12 4v16m-8H4" />
        </svg>
        Create a New Post
      </h2>
      <p className="text-blue-100 mt-1">Share your thoughts with fellow students</p>
    </div>
    </> Form
    <form onSubmit={handleSubmit} className="p-6">
      <error && (
        <div className="mb-4 bg-red-50 border border-red-200 rounded-lg p-4">
          <div className="flex">
            <div className="flex-shrink-0">
              <svg className="h-5 w-5 text-red-400" viewBox="0 20 20 20" fill="currentColor">
                <path fillRule="evenodd" d="M10 18a8 8 0 0 1 16ZM7.293 1.001c-1.414 0 -2.857 0.723-3.571 1.414l-1.414 1.414L8.586 10l-1.293 1.293a1 1 0 1 1 1.414 1.414" />
            </div>
            <div className="ml-3">
              <p className="text-sm text-red-800">{error}</p>
            </div>
          </div>
        </div>
      )>
      </> User Info
      <div className="flex items-center mb-12 pb-1 border-b border-gray-100">
        <div className="flex-shrink-0">
          {user?.profileImage ? (
            <img
              className="h-8 w-8 rounded-full object-cover border border-gray-200"
              src={user?.profileImage.startsWith('http') ? `http://localhost:5001/uploads/${user?.profileImage}` : user?.profileImage}
            >
            <span style={{ position: 'absolute', left: 0, top: 0, width: '100%', height: '100%' }>
              Review next file
            </span>
          ) : null}
        </div>
      </div>
    </form>
  </div>
}
```



A screenshot of a Mac OS X desktop environment. The top bar shows the system status with battery level (91%), signal strength, and the date (Sun Aug 31 9:44PM). The main window is a code editor titled "CreatePost.js - Part_2". The code is a React component named "CreatePost" that handles user profile image uploads and content input. It includes logic for handling file inputs, setting state, and rendering user profile information. The code editor has syntax highlighting for JavaScript and CSS. Below the code editor are several tabs for other files like "Profile.js", "FacultyDashboard.js", etc. At the bottom of the screen is a dock with various application icons.

```
const CreatePost = () => {
  const [user, setUser] = useState(null);
  const [content, setContent] = useState("");
  const [error, setError] = useState("");

  const handleFileInput = (e) => {
    const file = e.target.files[0];
    if (!file) return;
    const reader = new FileReader();
    reader.onload = (e) => {
      const dataURL = e.target.result;
      const userImage = {
        name: user?.name,
        profileImage: dataURL,
      };
      setUser(userImage);
    };
    reader.readAsDataURL(file);
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    if (!content) {
      setError("Content is required");
      return;
    }
    // Logic to post content to the server
  };

  return (
    <div>
      <div>
        <img alt={user?.profileImage}>
        <span>{user?.name}</span>
      </div>
      <div>
        <h2>What's on your mind?</h2>
        <input type="text" value={content} onChange={setContent} />
        <button type="button" onClick={handleSubmit}>Post</button>
      </div>
    </div>
  );
}
```

The screenshot shows a Mac OS X desktop environment with several open windows:

- Terminal:** A window titled "node - server" containing Node.js code related to a "FacultyDashboard" application.
- Code Editor:** An IDE window titled "CreatePost.js – Part_2" showing the source code for a "CreatePost" component. The code includes JSX for a form with an "input" field, "button" elements for "Cancel" and "Post", and an "svg" element for a loading spinner. The code editor has syntax highlighting and a status bar indicating "File 118, Col 21".
- File Browser:** A sidebar titled "PART_2" listing project files and folders, including "client", "build", "node_modules", "public", "src", "components", "admin", "assets", "auth", "faculty", "Index", "messege", "notifications", and "posts".
- System Status:** A top bar showing system icons for battery, signal, and time ("Sun Aug 31 9:44 PM").
- Dock:** A row of application icons including Finder, Mail, Safari, and others.

5. Notification: The toaster component is used here to provide user feedback and notifications throughout the application. It serves as a centralized notification system that displays temporary messages (like success confirmations, error alerts, or informational updates) to users in a non-intrusive way. When users perform actions such as logging in/out, submitting forms, or when system events occur, the toaster automatically shows relevant messages at the

top or corner of the screen, ensuring users are always informed about the status of their actions without disrupting their workflow or requiring them to navigate to separate notification pages.

Cursor File Edit Selection View Go Run Terminal Window Help

Toaster.js – Part_2

```

client > src > components > JS Toaster.js > ...
1 import React, { createContext, useContext, useState, useCallback } from 'react';
2 import './Toaster.css';
3
4 const ToasterContext = createContext();
5
6 export const useToaster = () => {
7   const context = useContext(ToasterContext);
8   if (!context) {
9     throw new Error('useToaster must be used within a ToasterProvider');
10  }
11  return context;
12};
13
14 export const ToasterProvider = ({ children }) => {
15   const [toasts, setToasts] = useState([]);
16
17   const showToast = useCallback(({ message, type = 'info', duration = 3000 }) => {
18     const id = Date.now() + Math.random();
19     const newToast = {
20       id,
21       message,
22       type,
23       duration
24     };
25
26     setToasts((prev => [...prev, newToast]));
27
28     // Auto remove toast after duration
29     setTimeout(() => {
30       setToasts(prev => prev.filter(toast => toast.id !== id));
31     }, duration);
32
33     return id;
34   });
35
36   const removeToast = useCallback((id) => {
37     setToasts(prev => prev.filter(toast => toast.id !== id));
38   });
39
40   const showSuccess = useCallback(({ message, 'success', duration }) => {
41     return showToast(message, 'success', duration);
42   });
43
44   const showError = useCallback(({ message, duration }) => {
45     return showToast(message, 'error', duration);
46   });
47
48   const showInfo = useCallback(({ message, duration }) => {
49     return showToast(message, 'info', duration);
50   });
51
52   const showWarning = useCallback(({ message, duration }) => {
53     return showToast(message, 'warning', duration);
54   });
55
56   return (
57     <ToasterContext.Provider value={{ showToast, showSuccess, showError, showInfo, showWarning, removeToast }}>
58       {children}
59       <ToasterContainer toasts={toasts} removeToast={removeToast} />
60     </ToasterContext.Provider>
61   );
62 };
63

```

Review next file >

node - server node client node server

Cursor Tab Ln 1, Col 1 Spaces: 2 UTF-8 LF JavaScript

Cursor File Edit Selection View Go Run Terminal Window Help

Toaster.js – Part_2

```

client > src > components > JS Toaster.js > ...
14 export const ToasterProvider = ({ children }) => {
15   const showToast = useCallback(({ message, type = 'info', duration = 3000 }) => {
16     setTimeout(() => {
17       setToasts(prev => prev.filter(toast => toast.id !== id));
18     }, duration);
19
20     return id;
21   });
22
23   const removeToast = useCallback((id) => {
24     setToasts(prev => prev.filter(toast => toast.id !== id));
25   });
26
27   const showSuccess = useCallback(({ message, 'success', duration }) => {
28     return showToast(message, 'success', duration);
29   });
30
31   const showError = useCallback(({ message, duration }) => {
32     return showToast(message, 'error', duration);
33   });
34
35   const showInfo = useCallback(({ message, duration }) => {
36     return showToast(message, 'info', duration);
37   });
38
39   const showWarning = useCallback(({ message, duration }) => {
40     return showToast(message, 'warning', duration);
41   });
42
43   return (
44     <ToasterContext.Provider value={{ showToast, showSuccess, showError, showInfo, showWarning, removeToast }}>
45       {children}
46       <ToasterContainer toasts={toasts} removeToast={removeToast} />
47     </ToasterContext.Provider>
48   );
49 };
50
51
52
53
54
55
56
57
58
59
60
61
62
63

```

Review next file >

node - server node client node server

Cursor Tab Ln 1, Col 1 Spaces: 2 UTF-8 LF JavaScript

```

client > src > components > JS Toaster.js > e| Toast > e| getIcon
  64  const ToasterContainer = ({ toasts, removeToast }) => {
  65    </div>
  66  };
  67  </div>
  68  );
  69  </div>
  70  > e| Toast = ({ toast, onRemove }) => {
  71    const { id, message, type } = toast;
  72
  73    const getIcon = () => {
  74      switch (type) {
  75        case 'success':
  76          return 's';
  77        case 'error':
  78          return 'x';
  79        case 'warning':
  80          return '!';
  81        case 'info':
  82          return 'i';
  83        default:
  84          return 't';
  85      }
  86    };
  87
  88    return (
  89      <div className="toast toast-$<type>">
  90        <div className="toast-icon">{getIcon()}</div>
  91        <div className="toast-message">{message}</div>
  92        <button
  93          className="toast-close"
  94          onClick={() => onRemove(id)}
  95          aria-label="Close notification"
  96        >
  97          <span>x</span>
  98        </button>
  99      </div>
 100    );
 101  );
 102  </div>
 103  );
 104  </div>
 105  );

```

6. User Manual

For new users, this user manual offers a thorough overview of how to use and navigate the Thesis Portal system. To guarantee clarity, it is separated into sections according to user roles (Administrator, Supervisor/Teacher, and Student). Each section contains important features, detailed instructions, and placeholders for screenshots (which can be taken from prototypes or the live system). The portal is designed to be intuitive, with a responsive UI that works on PCs, tablets, and mobile devices.

1. Getting Started: Registration and Login

1.1 User Registration

New users (students, teachers, or admins) must create an account to access the portal. Go to the homepage of the Thesis Portal (for example, by using the URL for the login page). The "Register" button in the upper-right corner should be clicked. Enter the necessary information: Name, Email (preferably an institutional email), Department *, student ID (optional). The password needs to have a minimum of six characters. Choose the role of the user: Student, Instructor, or Administrator. "Submit" will create the account.

Academic Thesis Portal

Browse Thesis Login Register

Create your account

Or [sign in to your existing account](#)

Full Name *

Email address *

Department *

Role *

Activate Windows

Go to Settings to activate Windows

1.2 User Login

Go to the homepage and click "Login." Enter your registered email or User ID and password. Click "Login" to access your dashboard.

Academic Thesis Portal

Browse Thesis Login Register

Create your account

Or [sign in to your existing account](#)

Full Name *

Email address *

Department *

Role *

Activate Windows

Go to Settings to activate Windows

Full Name *

Email address *

Department *

Role *

Activate Windows

Go to Settings to activate Windows

2. Student User Guide

2.1 Student Dashboard

Upon logging in as a student, you will be directed to your personalized Student Dashboard. This is the central hub for managing your thesis process. The dashboard provides quick access to key features and displays relevant information tailored to your current status.

The screenshot shows the Student Dashboard with a blue header bar containing navigation links: 'Menu', 'Browse Thesis', 'Review & Ratings', 'AI Paper Analyzer', 'Welcome, meem', and 'Logout'. Below the header, the title 'Student Dashboard' is displayed, followed by a welcome message 'Welcome, meem!'. A 'Refresh' button is also present. The main content area includes sections for 'Supervisor' (with a note about not requesting one yet), 'My Thesis' (showing no submissions), and 'Download Templates' (with three download buttons for P1, P2, and P3). A sidebar on the right side of the dashboard displays the message 'Activate Windows Go to Settings to activate Windows.'

Navigating the Dashboard Features

Menu: Click the "Menu" button (three horizontal lines) to open a sidebar with additional navigation links (e.g., Dashboard, Profile, Create post, Supervisor/co-supervisor, Marks, Messages).

The screenshot shows the Student Dashboard. At the top, there is a blue header bar with a 'Menu' button on the left and a 'Browse Thesis' link on the right. Below the header, the main content area has a light gray background. On the left side, there is a vertical navigation menu with the following items:

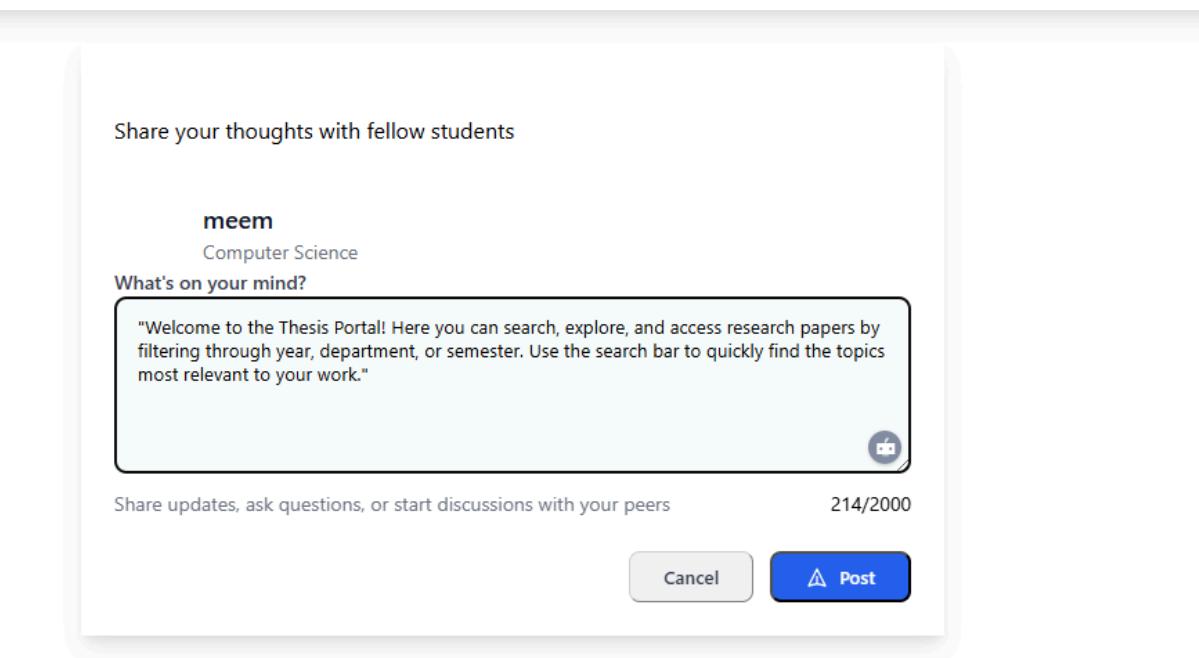
- Dashboard
- +Create Post
- Posts Feed
- Supervisor/Co-supervisor
- Marks
- Messages
- Profile

The central part of the dashboard features two main sections:

- Supervisor:** A box containing the message "You have not requested a supervisor yet." and a link "Browse faculty members and request supervision for your thesis".
- My Thesis:** A box containing the message "No thesis submissions yet."

New Post Create:

Upon selecting the "Create Post" option from the menu bar, the Chat interface appears, enabling students to post messages and engage in real-time communication. The page features a clean layout with a "Refresh Now" button to update the conversation, ensuring the latest messages are visible. A sample exchange displays a student's message in the chat window, with a text input field below for typing new posts, allowing for direct and immediate interaction.



Activate Win
Go to Settings tc

The Student Posts Feed page allows students to connect and share experiences, featuring a feed with posts shared just now, welcoming users, and explaining research access features. Options such as "My Posts," "Create Post," and "Refresh Feed" enable users to manage and update their content, with interactive elements like "Like" and "Comment" buttons enhancing engagement.

A screenshot of the "Student Posts Feed" page. At the top, it says "Connect with fellow students and share your experiences". Below this, it shows "Showing 3 of 3 posts". There is a post by "meem" (Student) from "Computer Science" posted "Just now". The post content is the same as in the previous screenshot: "Welcome to the Thesis Portal! Here you can search, explore, and access research papers by filtering through year, department, or semester. Use the search bar to quickly find the topics most relevant to your work.". Below the post are "0 likes" and "0 comments" buttons. At the top right of the feed area, there are three buttons: "My Posts", "+ Create Post", and "Refresh Feed".

Marks Management:

The Marks Management page provides a clear interface for viewing thesis marks and progress, displaying student information and total scores. It includes a section for the supervisor and CGPA details, with the total score highlighted for easy reference.

Marks Management

View your thesis marks and progress

Student Information

Name

meem

Department

Computer Science

Supervisor

Dr. Muhammad Iqbal Hossain

CGPA

3.55/4.00

Total Score

0/100

Browse Thesis: Click "Browse Thesis" in the navbar to go to the Browse Papers page after logging in. You can look through a searchable database of previous thesis documents that faculty members have posted for academic reference in this section. You can easily find relevant research papers using the search and filter options. Simply enter a keyword, title, or abstract in the search field and press Enter to locate papers of interest. To refine your results, you can filter by year by selecting a specific option from the dropdown menu, or choose a department to narrow down the papers by faculty. Additionally, you may select a semester from the dropdown to filter by academic term. By adjusting these filters and searching again, you can quickly locate the most relevant research papers. Click "My Bookmarks" to see a list of saved papers. Remove bookmarks by unselecting the star icon next to a paper in the bookmarks list.

The screenshot shows the 'Browse Papers' page with a blue header bar containing 'Menu', 'Browse Thesis', 'Review & Ratings', 'AI Paper Analyzer', 'Welcome, meem', and 'Logout'. Below the header is a search bar with placeholder text 'Search by title, abstract, or keywords'. To the right of the search bar are dropdown menus for 'Department' (set to 'All Departments'), 'Year' (set to 'All Years'), and 'Semester' (set to 'All Semesters'). A 'My Bookmarks' button is located in the top right corner of the main content area. The main content area displays a heading 'Results (4 papers)' followed by a list of four items: 'title p' and 'ai'. A watermark in the bottom right corner reads 'Activate Windows' and 'View PDF'.

Review & Ratings: Navigate to view supervisor feedback and ratings (visible only after thesis completion).

The screenshot shows a user interface for managing faculty reviews. At the top, there's a blue header bar with a 'Menu' dropdown, 'Browse Thesis', 'Review & Ratings', 'AI Paper Analyzer', and a 'Welcome, meem' message. Below the header, the title 'Faculty Review & Ratings' is displayed. A large box labeled 'All Faculty' contains a review for 'Dr. Muhammad Iqbal Hossain' from 'Computer Science'. The average rating is 5.00 / 5. Two reviews are shown:

- Faculty:** Dr. Muhammad Iqbal Hossain
Student: riba333
Review: sir was good
Date: 8/27/2025
- Faculty:** Dr. Muhammad Iqbal Hossain
Student: riba22
Review: good
Date: 8/26/2025

At the bottom right of the main box, there's a link 'Activate Wir'.

AI Paper Analyzer: Access the AI-powered summarizer to analyze submitted reports (available after submission).

Upload a paper then click summarizer it will generate a brief summary and if clicked on suggestions it will generate suggestions for the uploaded paper.

The screenshot shows a web-based AI paper analyzer. The browser toolbar at the top includes 'Chrome', 'File', 'Edit', 'View', 'History', 'Bookmarks', 'Profiles', 'Tab', 'Window', and 'Help'. The address bar shows 'localhost:3000/pdf-summarizer'. The main content area has a title 'Paper Analyzer' and a sub-instruction 'Upload a PDF and generate a beautifully formatted academic summary and suggestions for improvement.' Below this, there's a section titled 'AI-Powered Paper Analyzer' with a 'Select PDF File' input field. The file 'Pre_1_Thesis-2.pdf (0.44 MB)' is selected. There are two buttons: 'Generate Summary...' and 'Generate Suggestions...'. To the right of the summary button is a 'Copy Summary' link. The summary itself is a detailed academic text about network security and intrusion detection systems. At the bottom, there's a dark footer bar with various application icons.

💡 Paper Improvement Suggestions

Generated using comprehensive text analysis for reliable academic writing suggestions

🌟 **Strengths Found:**

- > ✓ Employs quantitative research methods
- > ✓ Includes literature review and background research
- > ✓ Good use of transition words for logical flow

📝 **Content Analysis for machine learning, deep learning, artificial intelligence, neural networks, optimization, algorithms, computer science, engineering, technology:**

- > ⓘ Consider adding performance metrics and evaluation criteria
- > ⓘ Include mathematical formulations for key algorithms
- > ⓘ Add comparative analysis with existing methods
- > ⓘ Include technical specifications and requirements
- > ⓘ Add implementation details and constraints

-> ⓘ Consider practical applications and limitations

⚠️ **Issues to Address:**

- > ⓘ Research questions or hypotheses should be explicitly stated
- > ⓘ Consider using more active voice for clarity and engagement
- > ⓘ Fix multiple consecutive spaces for consistent formatting
- > ⓘ Define acronym "IDS" on first use
- > ⓘ Define acronym "NIDS" on first use
- > ⓘ Define acronym "FAR" on first use
- > ⓘ Define acronym "SNIDS" on first use
- > ⓘ Define acronym "ADNIDS" on first use

Copy All Suggestions

New Supervisor Request

Upon sending a supervisor request from the student side, a notification alert appears on the faculty member's dashboard, indicated by the red "5" badge on the **Notifications** bell icon. The provided image shows the **Faculty Dashboard** for Dr. Muhammad Iqbal Hossain, where the dashboard displays a welcoming message, the total seats available (4 out of 9, with 3 individual and 1 group seat), and a "Supervisor Requests" section. The faculty member can review the request, click "Details" for more information, "Accept" to approve, or "Decline" to reject, ensuring an efficient process for managing supervision assignments.

The screenshot shows the Faculty Dashboard for Dr. Muhammad Iqbal Hossain. At the top, there are navigation links for 'Menu', 'Browse Thesis', 'AI Paper Analyzer', and a notifications badge with the number '5'. The main header says 'Faculty Dashboard' and 'Welcome, Dr. Muhammad Iqbal Hossain'. Below this, it displays 'Total Seats: 4/9 (Individual: 3, Groups: 1)' and '5 seats available'. A 'Supervisor Requests' section shows a single entry for 'meem' with details: Department: Computer Science, Email: afsanameem12@gmail.com, Student ID: 111111, CGPA: 3.55/4.00, Requested: 8/29/2025. There are three buttons: 'Details', 'Accept', and 'Decline'. To the right of the dashboard, a small 'Activate Windows' watermark is visible.

The supervisor has accepted the request.

The screenshot shows the Notifications page for Dr. meem. At the top, there are navigation links for 'Menu', 'Browse Thesis', 'Review & Ratings', 'AI Paper Analyzer', and a notifications badge with the number '1'. The main header says 'Notifications' and 'Welcome, meem'. Below this, it says 'View and manage all your notifications'. A notification card for 'Supervisor Request Accepted' states: 'Dr. Muhammad Iqbal Hossain has accepted your supervisor request' and includes a timestamp '0 minutes ago'. There are two buttons: 'Mark Read' and 'Delete'.

2.2 Browse All Faculty

You can browse and request supervision from available faculty members on the Faculty Members tab after selecting "Browse All Faculty" on the Student Dashboard. This portal offers an easy-to-use interface for departmental filtering, faculty search by name or email, and seeing comprehensive biographies of possible supervisors, including Dr. Amitabha Chakrabarty and Dr. Md. Sadek Ferdous from the Computer Science department. With a clear indication of the current request status (e.g., "No Request"), each faculty listing offers the ability to check their profile, send a message, or request supervision. To ensure a seamless procedure for individual supervision assignments, a helpful remark advises you to update your Student ID and CGPA in your profile prior to submitting a supervision request.

Faculty Members

Browse and request supervision from faculty members

① Individual Supervisor Request
You are requesting a supervisor for yourself. Make sure you have updated your Student ID and CGPA in your profile.

Search by name or email...

All Departments

Dr. Md Sadek Ferdous Computer Science No Request View Profile Message Request Supervision	Dr. Amitabha Chakrabarty Computer Science No Request View Profile Message Request Supervision
--------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------

Activate Windows
Go to Settings to activate Windows.

2.3 New Message for Faculty

Upon clicking the "Message" option next to a faculty member's profile on the "Browse All Faculty" page, the Chat interface appears, allowing students to communicate directly with faculty members such as Dr. Muhammad Iqbal Hossain from the Computer Science department.

Chat here..

Messages [+](#) **Dr. Muhammad Iqbal Hossain**
 faculty • Computer Science

Dr. Muhammad Iqbal Hossain (faculty) 12:58 PM You: As salamu alaikum sir , I have submitted the thesis registration form.	meem As salamu alaikum sir , I have submitted the thesis registration form. 12:58 PM
-------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------

Type your message... [Send](#)

Activate Windows
Go to Settings to activate Windows.

2.4 Group Formation Update

Upon clicking "Group Formation" on the Student Dashboard, the Group Formation page appears, offering a platform to find and collaborate with other students for a thesis project. The page displays a list of available students under the "Available Students" section, showcasing profiles with names, IDs, and departments, such as Computer Science students. Each profile includes a "Send Group Request" button, allowing the user to initiate group formation by sending requests to potential members.

2.5 Thesis Registration

Users are given a simple interface to register their thesis subject with their supervisor before submitting the following stages when they visit the Thesis Registration page. The page includes a form where users can enter a 200-character Thesis Title and a 1000-character Thesis Description field that can hold in-depth explanations (for example, a description of cloud computing as a technology that provides services like servers, storage, and networking over the internet). A "Submit Registration" button allows users to complete their registration.

The screenshot shows the 'Thesis Registration' page. It features two main input fields:

- Thesis Title ***: A text input field containing 'Cloud Computing' with a character count of '15/200 characters' below it.
- Thesis Description ***: A larger text input field containing a descriptive paragraph about cloud computing, with a character count of '244/1000 characters' below it. A small camera icon is located in the top right corner of this field.

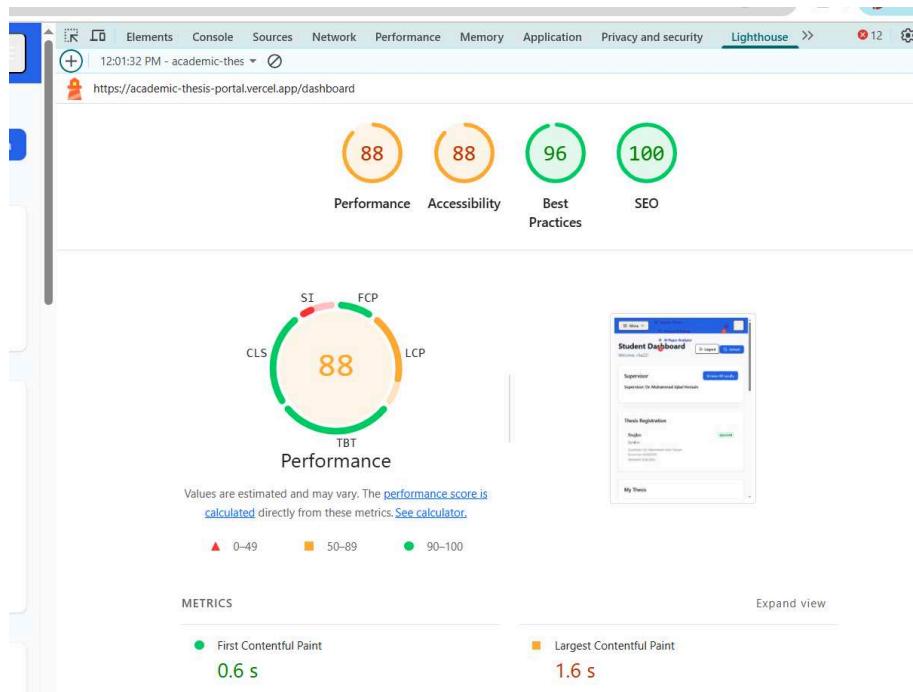
At the bottom right of the form, there is a blue 'Submit Registration' button.

Logout: Click to securely log out of the system.

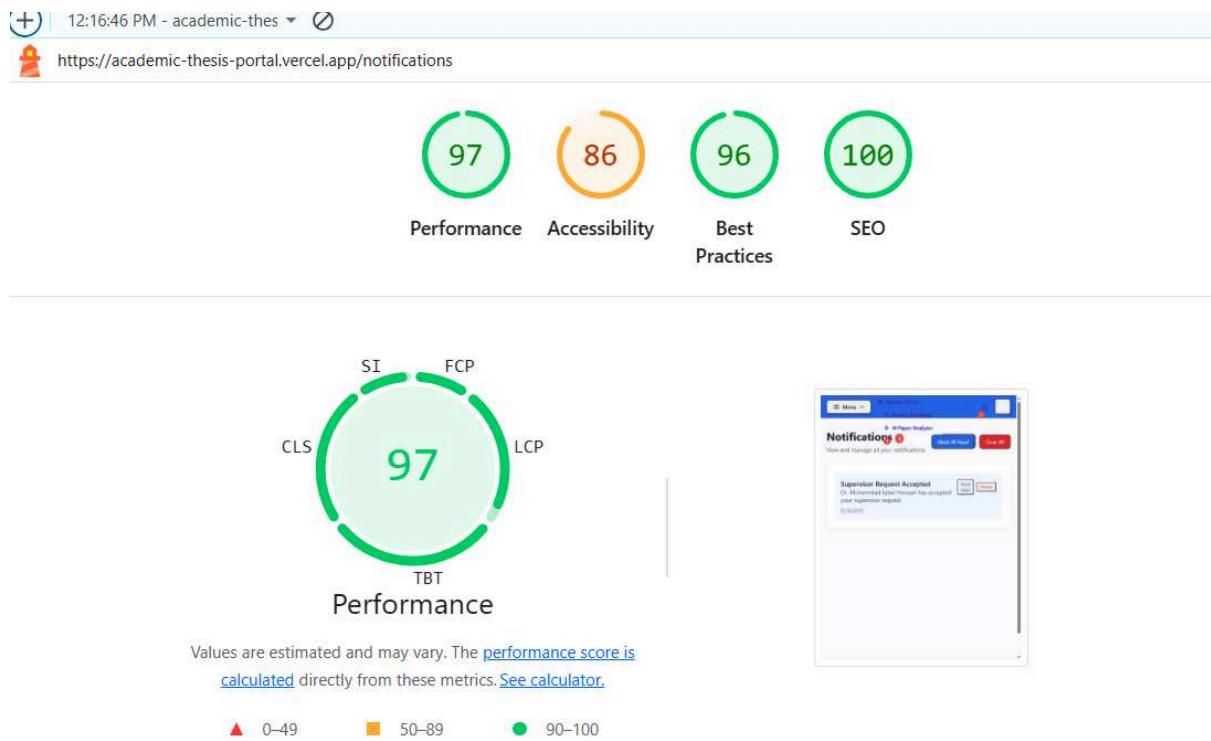
7. Performance and Network Analysis

Desktop Analysis

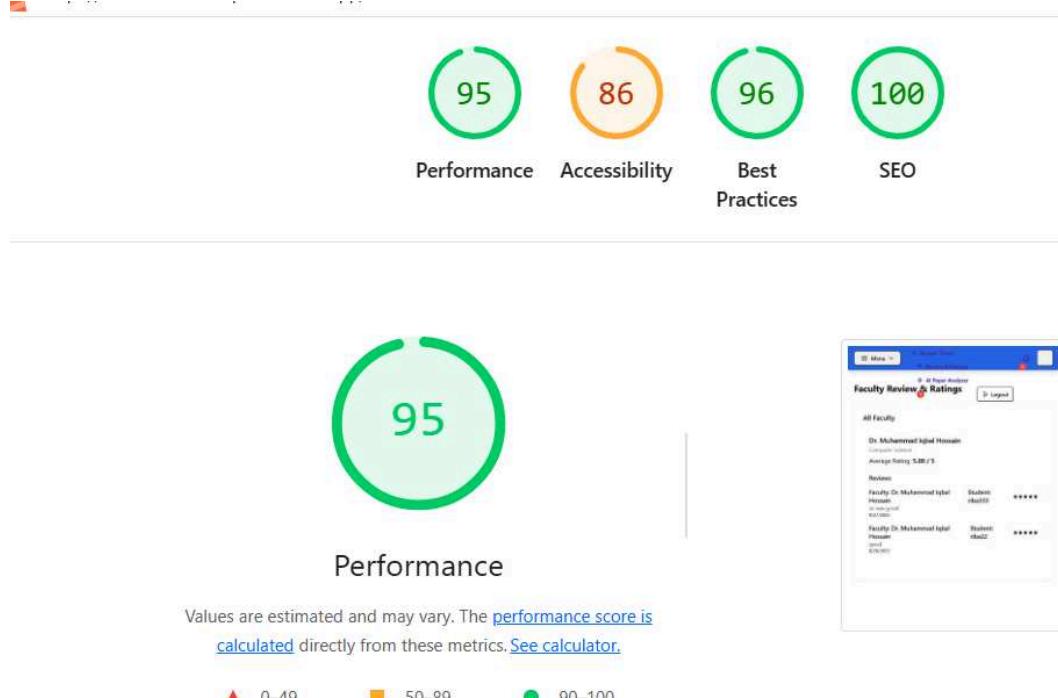
Student Dashboard



Notification Page

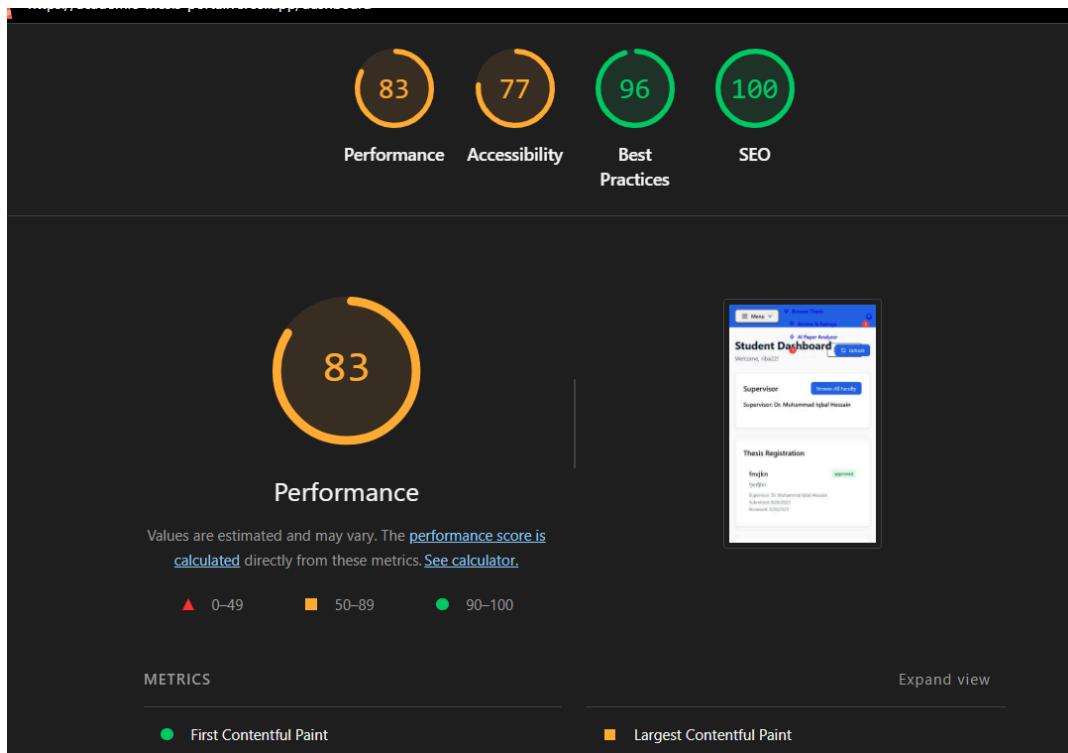


Faculty Review & Rating Page

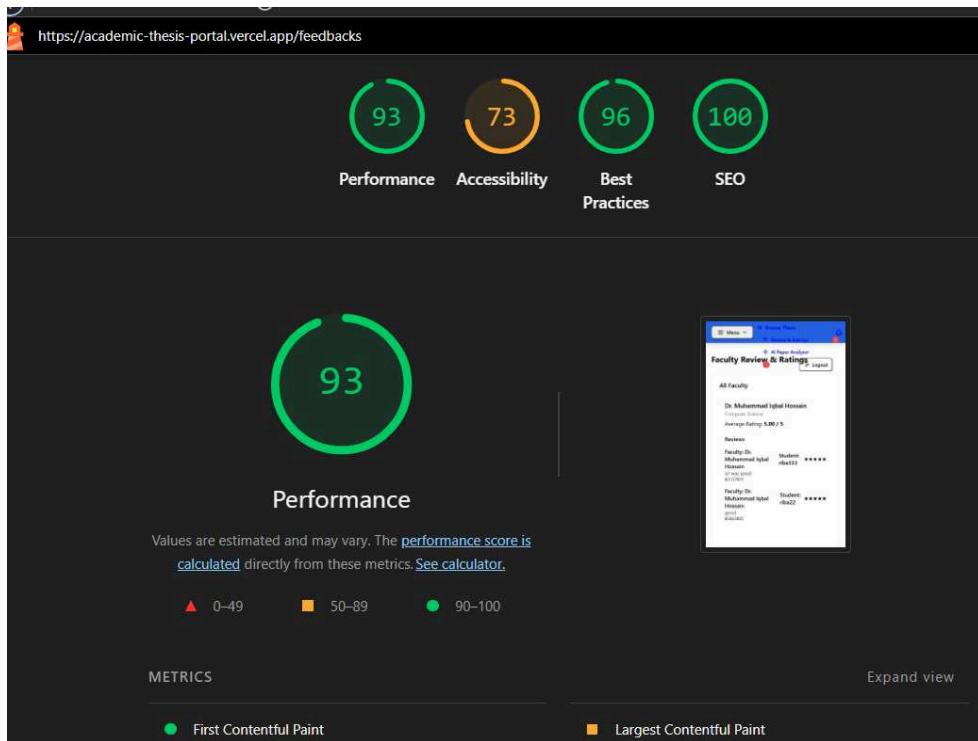


Mobile Analysis

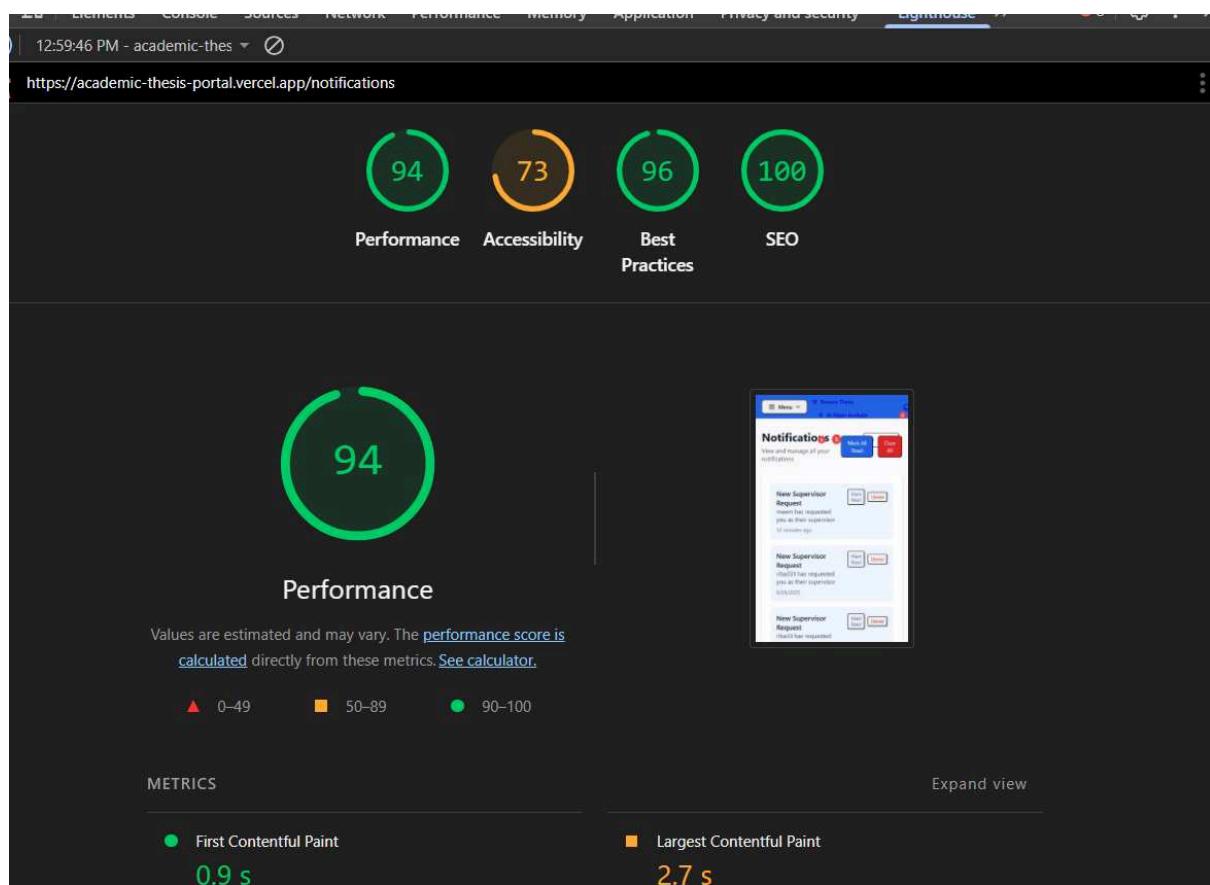
Student Dashboard



Faculty Review & Rating



Notification



8. Github Repo [Public] Link-

<https://github.com/tasminoni/Thesis-Portal>

9. Link of Deployed Project -

<https://academic-thesis-portal.vercel.app/>

10. Individual Contribution

Group member - 01	
Name:Tasmin Ahmed Oni	Student ID:21201532
Functional Requirements that are developed by this member:	
1. Chat/Messaging System	
2. Progress Tracking	
3. Group Formation	
4. Summarize using AI and provide detailed suggestions	

Group member - 02	
Name: Afsana Akter Mim	Student ID:22101881
Functional Requirements that are developed by this member:	
1. Notification System	
2. Thesis Submission	
3. Search, Filter, and Categorization	
4. One-Click Report Template Download	

Group member - 03	
Name: Riba Rahman	Student ID:21201757
Functional Requirements that are developed by this member:	
1. User Authentication and Authorization	
2. Teacher Dashboard/ Student Dashboard	
3. Supervisor Selection	
4. Feedback and Rating System	

Group member - 04

Name: Nirvik Shaha Rudra	Student ID:21201241
Functional Requirements that are developed by this member:	
1. Student Profile and Teacher Profile	
2. Documents on Previous Research	
3. Bookmarking Tool	
4. Admin Dashboard	

11. References

N/A