**LearnHub: Full Stack MERN Project Documentation**

**1. Introduction**

**Project Title:** LearnHub: Your Skill Enhancement Center

**Team ID :** LTVIP2025TMID55863  
**Team Members:**

* Enikepalli Naveen Sri Sai
* Deevi Sruthi
* Devarakonda Anusha Rani
* Daliparthi Jerush

**2. Project Overview**

**Purpose:**  
To create a comprehensive online learning platform where learners can enroll in courses, track their progress, and receive certifications, while educators can create and manage courses.

**Features:**

* User registration and login for educators and learners
* Course creation and management by educators
* Course enrollment and tracking by learners
* Certification upon completion
* Admin dashboard for platform management
* Multilingual content support

**3. Architecture**

**Frontend:**  
Developed using React.js with Vite for faster build and performance. Styled using Material UI and Bootstrap for responsive design.

**Backend:**  
Node.js with Express.js to build RESTful APIs for user auth, course management, and content handling.

**Database:**  
MongoDB with Mongoose ODM. Collections include users, courses, and enrollments. Schemas maintain relations between learners and their courses.

**4. Setup Instructions**

**Prerequisites:**

* Node.js and npm
* MongoDB Community Server

**Installation:**

# Clone the repository

git clone https://github.com/NaveenEnikepalli/Learn-Hub-Your-skill-Enhancement-Center.git

# Frontend setup

cd frontend

npm install

# Backend setup

cd ../backend

npm install

# Create .env in backend with:

PORT=5000

MONGODB\_URI=<your\_mongo\_uri>

JWT\_SECRET=<your\_jwt\_key>

**5. Folder Structure**

**Client (frontend):**

* /src/components: Reusable components (e.g., Navbar, CourseCard)
* /src/pages: Registration, Dashboard, CoursePage, etc.
* /src/services: API call files using Axios

**Server (backend):**

* /models: Mongoose schemas (User.js, Course.js)
* /routes: Express routes (auth, courses, admin)
* /middleware: Auth middleware (JWT verification)
* /controllers: Business logic for API requests

**6. Running the Application**

# Frontend

cd frontend

npm run dev

# Backend

cd backend

npm start

**7. API Documentation**

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| POST | /api/auth/register | Register a new user |
| POST | /api/auth/login | Login a user |
| POST | /api/courses | Create a new course |
| GET | /api/courses | Get list of courses |
| POST | /api/enroll | Enroll in a course |
| GET | /api/progress/:id | Get learner progress |

**8. Authentication**

* JWT-based authentication used for securing protected routes.
* Tokens stored in localStorage for session handling.
* Role-based authorization (admin, educator, learner).

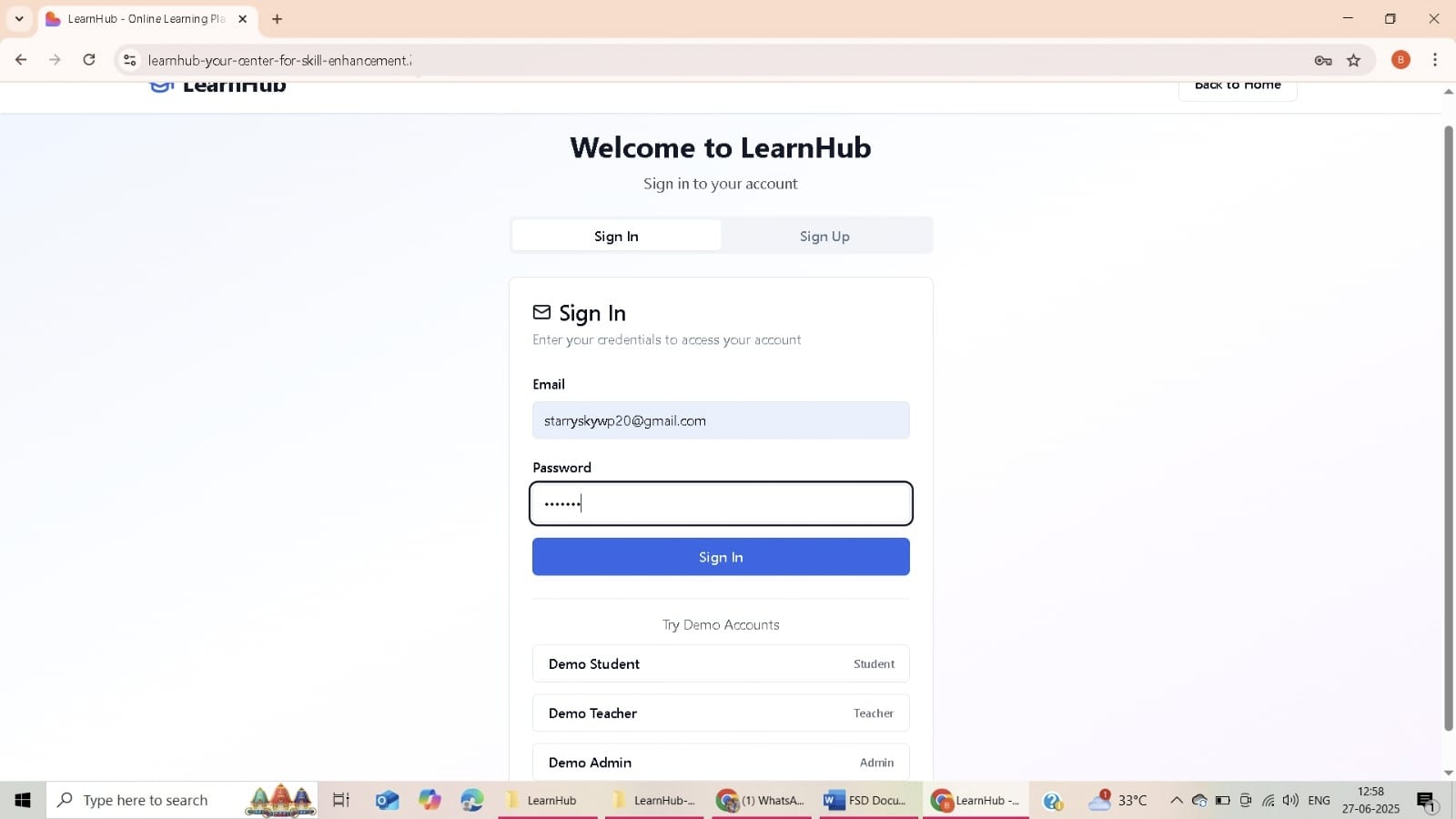
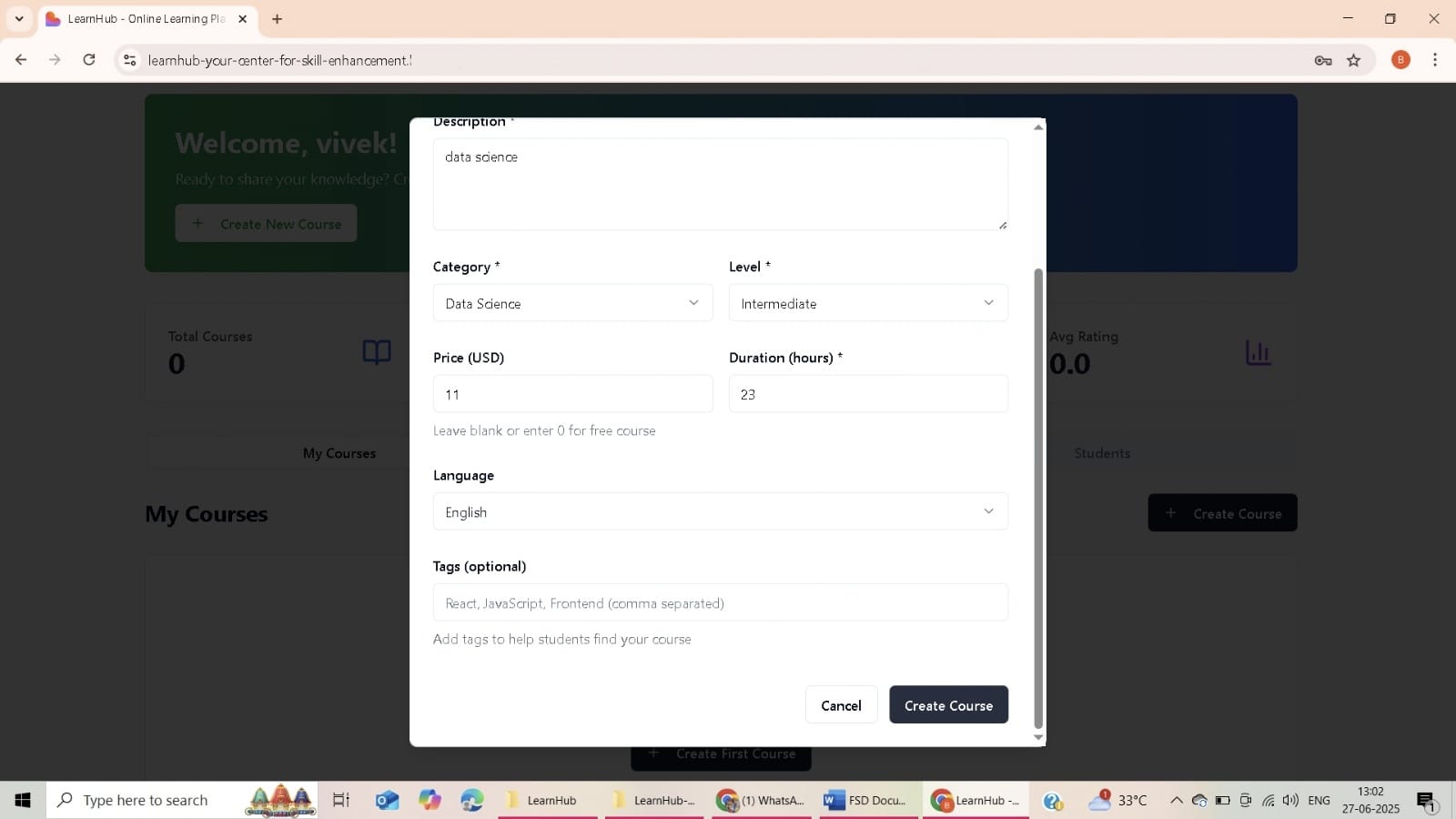
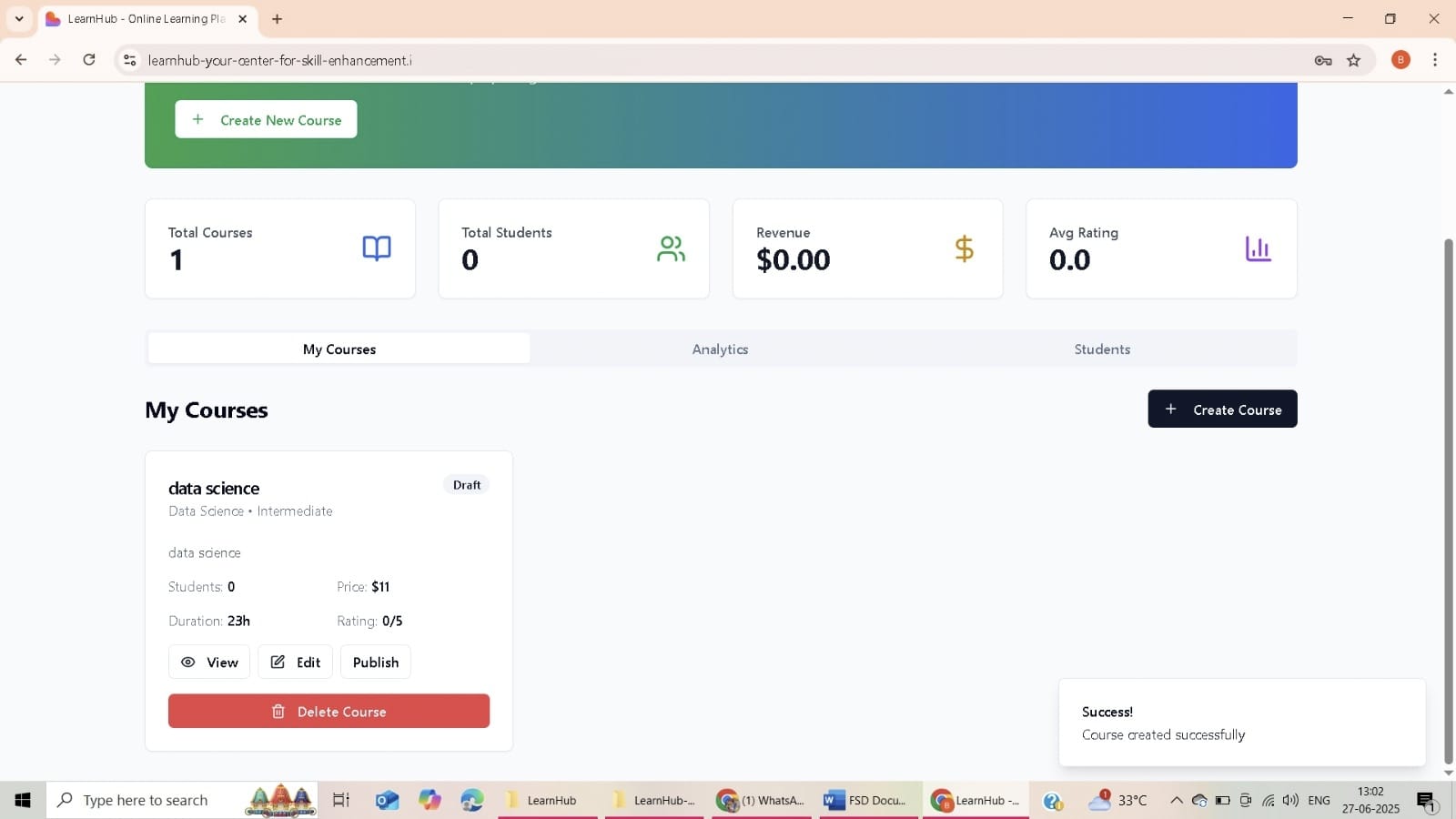
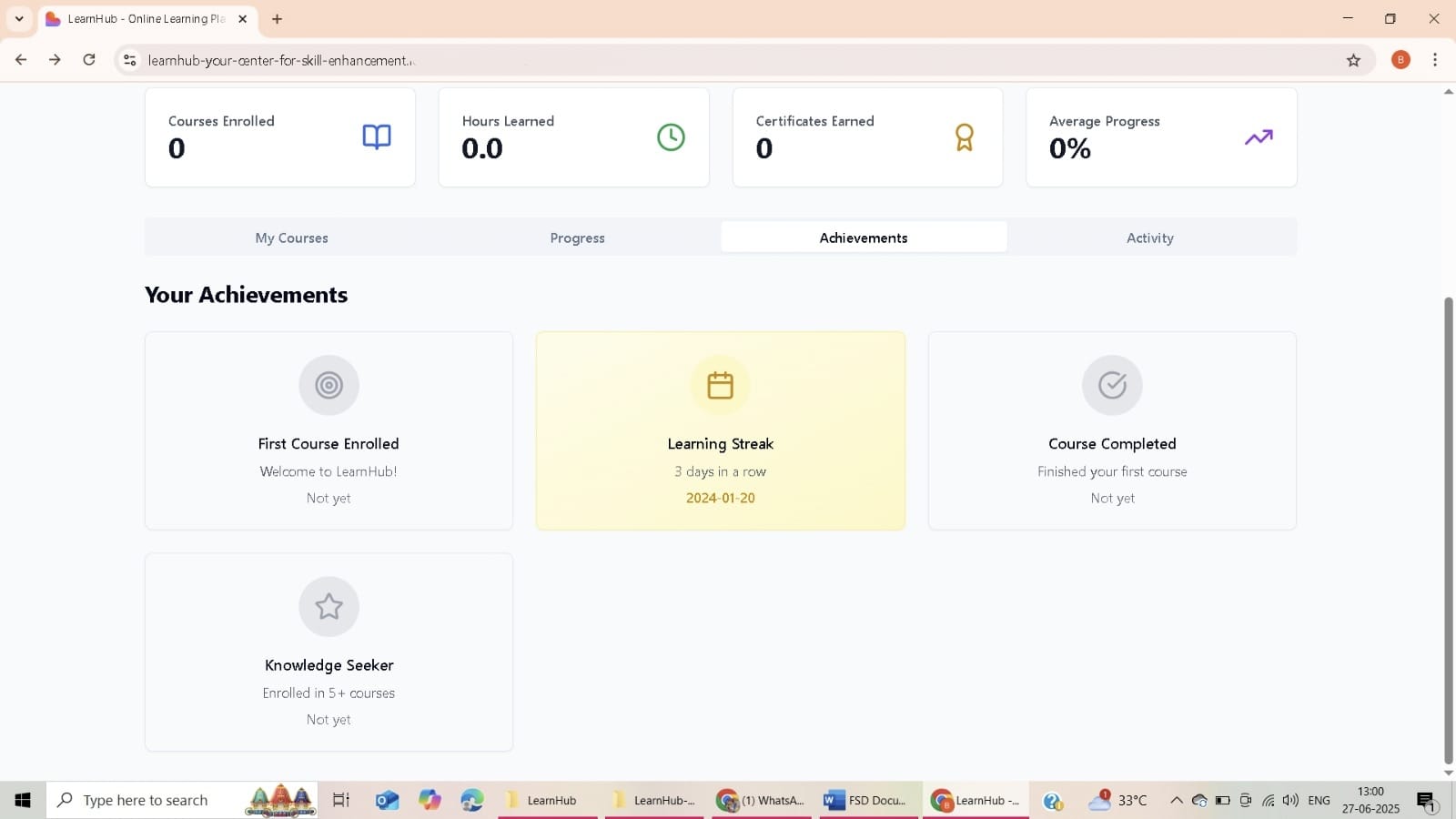
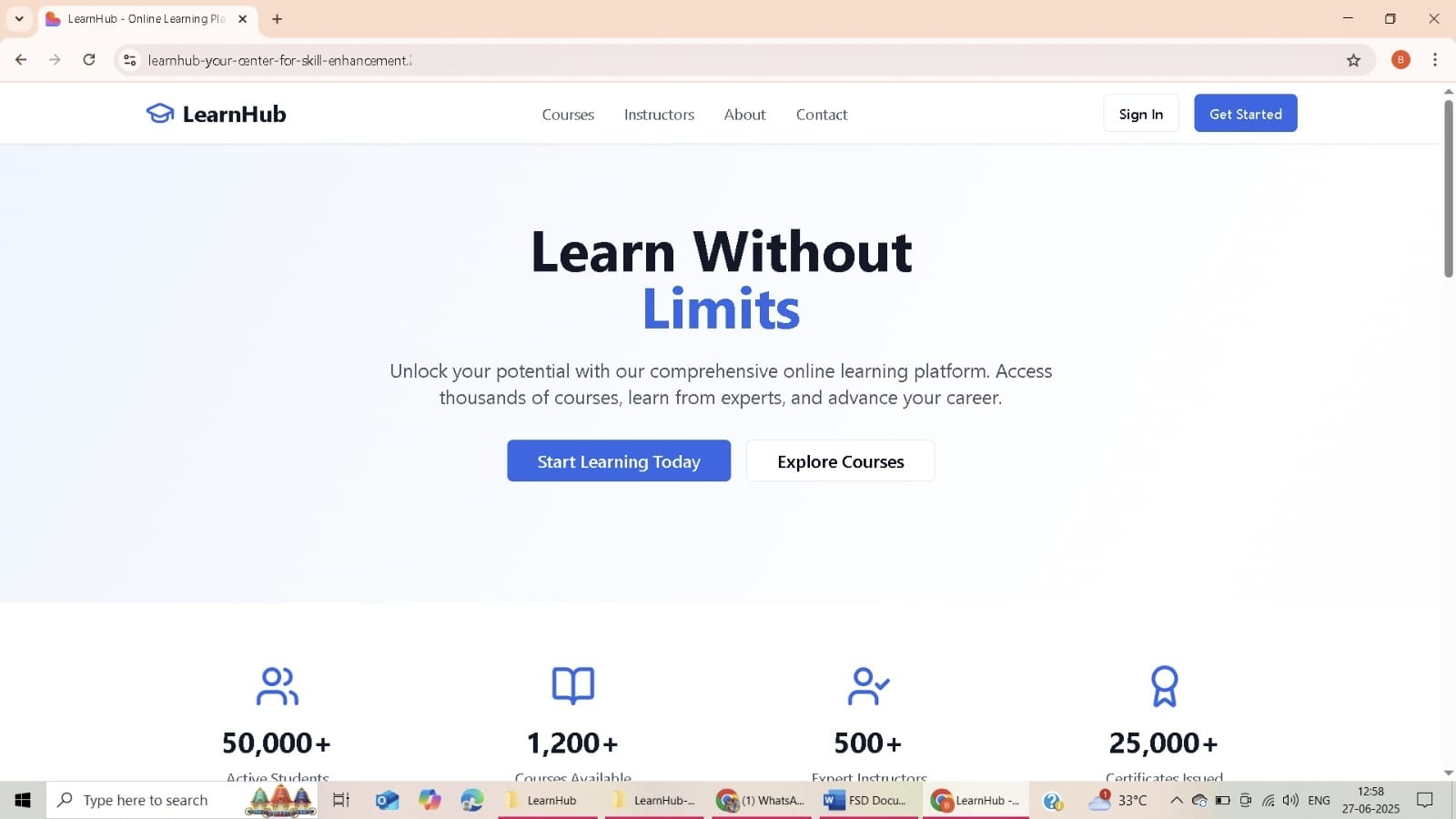
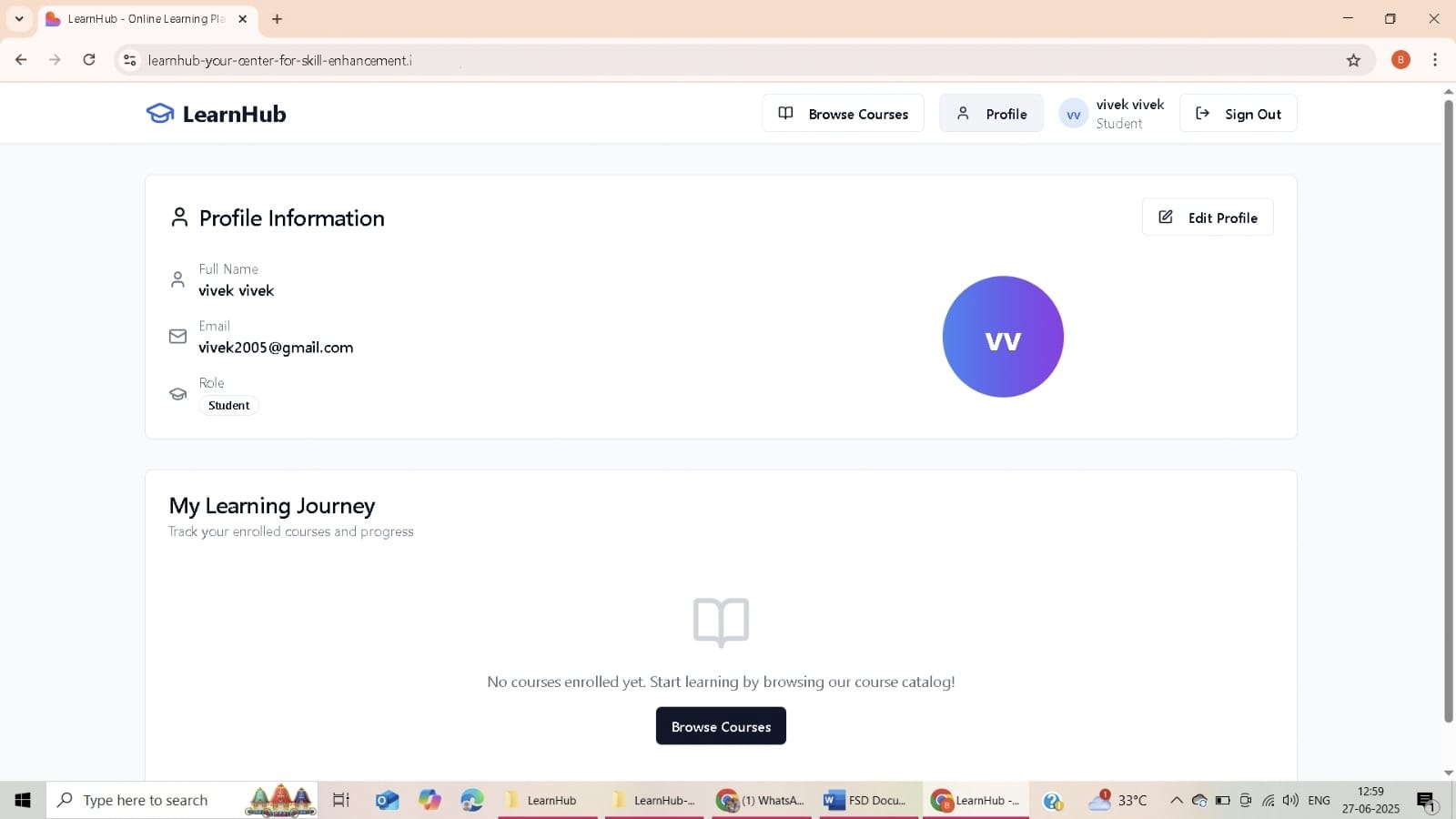
**9. User Interface**

* Simple and intuitive UI using Material UI and Ant Design.
* Dashboards for learners, educators, and admin.
* Certificate and course progress views.

**10. Testing**

* Unit testing using Jest (backend)
* Manual testing using Postman for API validation
* UI testing for all user roles

**11. Screenshots or Demo**

* Registration Page
* Educator Course Creation Page
* 
* 
* Learner Course Dashboard
* 
* Admin Overview Panel
* 
* Certificate Download Page
* 

**12. Known Issues**

* No video compression during upload
* Limited analytics in admin dashboard
* No notification system (yet)

**13. Future Enhancements**

* Real-time classroom/live video sessions
* Mobile application for learners
* Gamification features for learners
* Educator rating and review system