**Batch T4**

**Practical No. 4**

**Title of Assignment :**

**MCQ EXAM APP**

**Student Name: Parshwa Herwade**

**Student PRN: 22510064**

**Title**

**Design and Implementation of a Web-Based MCQ Exam Management System**

**Objective / Aim**

* **Develop a scalable web-based application** for managing online multiple-choice examinations.
* **Facilitate distributed question contribution** by teaching assistants, with centralized editing by the course coordinator.
* **Enable dynamic test creation** from a comprehensive question bank.
* **Implement flexible exam administration**: support fixed-time exams as well as on-demand tests with a defined time limit.
* **Provide real-time feedback** to students on their scores upon exam completion.
* **Utilize modern technologies:** Angular 19 for the frontend, NodeJS for the backend, MySQL for data storage, and Tailwind CSS for a responsive, modern UI.

**Introduction**

The modern education landscape increasingly relies on digital platforms for assessment and evaluation. This project focuses on designing and implementing an online multiple-choice exam management system that not only automates exam scheduling and grading but also supports distributed question contribution. The system caters to various user roles such as teachers, teaching assistants, and students, ensuring secure access and role-specific functionalities. By leveraging Angular, NodeJS, MySQL, and Tailwind CSS, the application aims to provide an intuitive and efficient interface for both exam administrators and students.

**Theory / Algorithms**

The application design integrates several key algorithms and theoretical concepts:

* **User Authentication and Authorization:**  
  Implements role-based access control (RBAC) to manage different user types (teachers, teaching assistants, and students). Secure login, session management, and password .
* **Question Bank Management:**  
  A CRUD (Create, Read, Update, Delete) model facilitates the addition, modification, and deletion of questions. This includes support for multimedia (images) and complex mathematical expressions, which are rendered using appropriate libraries
* **Grading System:**  
  Upon test completion, the system compares submitted answers against a predefined answer key.
* **Report Generation:**  
  Generic report algorithms extract and compile data from various tables (users, exams, question banks) to produce interactive dashboards and detailed performance reports.

**Data Flow Diagram (DFD) – Level 1**

1. **User Authentication:**
   * **Input:** Login credentials
   * **Process:** Validate user details, assign role
   * **Output:** User session token, access rights
2. **Question Bank Entry:**
   * **Input:** New question data (text, image, mathematical expressions)
   * **Process:** Validate and store question details
   * **Output:** Updated question bank database
3. **Exam Administration:**
   * **Input:** Test creation request (selection of questions, timing parameters)
   * **Process:** Assemble test, schedule exam (fixed time/on-demand)
   * **Output:** Published exam instance for students
4. **Result Generation:**
   * **Input:** Student responses, exam completion signal
   * **Process:** Automated grading, score computation, report compilation
   * **Output:** Score feedback to student and updated exam statistics on dashboard

**Procedure**

1. **Requirement Analysis:**
   * Gathered detailed requirements and defined use cases for teachers, TAs, and students.
2. **System Design:**
   * Prepared architecture diagrams, flowcharts, and a DFD for data flow.
   * Selected technology stack: Angular 19, NodeJS, MySQL, CSS.
3. **Development Phase:**
   * **Frontend:** Developed UI components using Angular and styled with Tailwind CSS.
   * **Backend:** Created RESTful API endpoints in NodeJS for user management, exam scheduling, and question bank operations.
   * **Database:** Designed MySQL schema to support relational data for users, questions, exams, and results.
4. **Testing:**
   * Unit tests for individual modules and integration tests for overall system flow.
   * User acceptance testing (UAT) was conducted with sample datasets.
5. **Deployment:**
   * Deployed the application on a cloud platform, ensuring scalability and security.
6. **Documentation:**
   * Prepared comprehensive documentation including system architecture, user manuals, and maintenance guides.

**Actual Experiments/Simulation, Results / Observations**

**Experimentation**

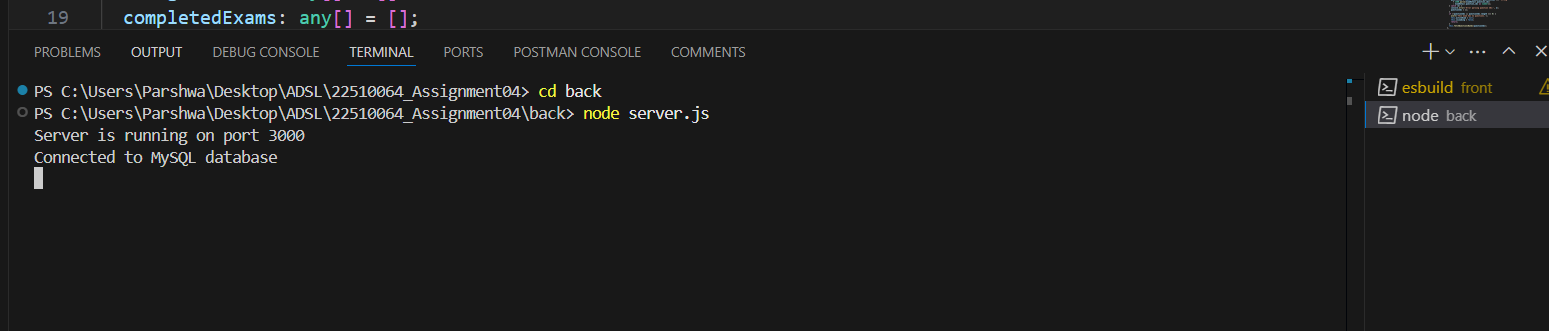
* **Simulation of Exam Creation:**
  + A demo exam was created using the question bank entry form with multimedia support.
  + The system allowed multiple TAs to contribute and the course coordinator to edit questions seamlessly.
* **Live Testing:**
  + Conducted live exam sessions using both fixed scheduling and dynamic time-limited modes.
  + Real-time feedback was provided immediately after exam completion.

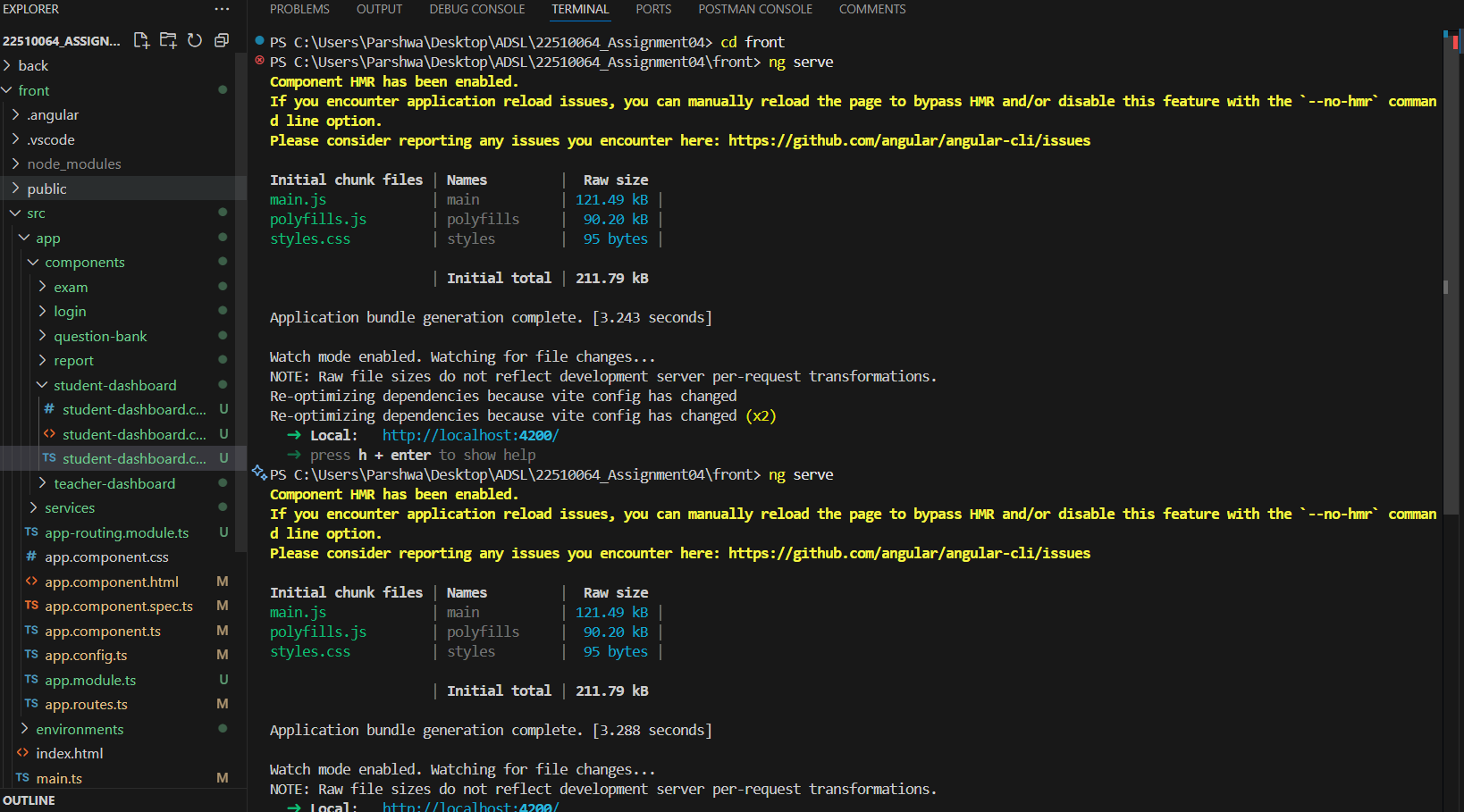
**Results / Observations**

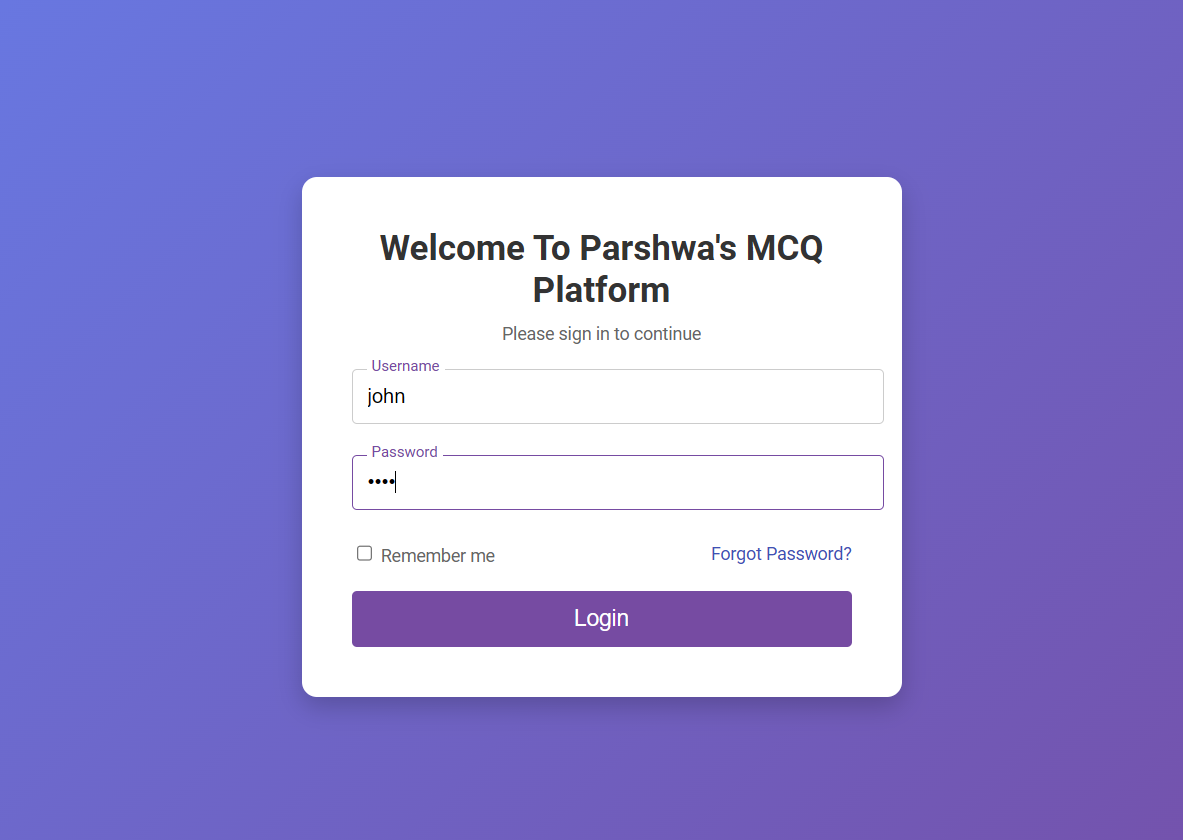
* **User Management:**
  + Successful segregation of roles and secure login for teachers, TAs, and students.
* **Question Bank Entry:**
  + Effective handling of images and mathematical expressions ensured clarity in exam questions.
* **Exam Administration:**
  + The scheduling algorithm performed reliably, enforcing time limits and managing simultaneous sessions.
* **Dashboard and Reports:**
  + Real-time statistics on exam status (ongoing, terminated, completed) were accurately displayed.

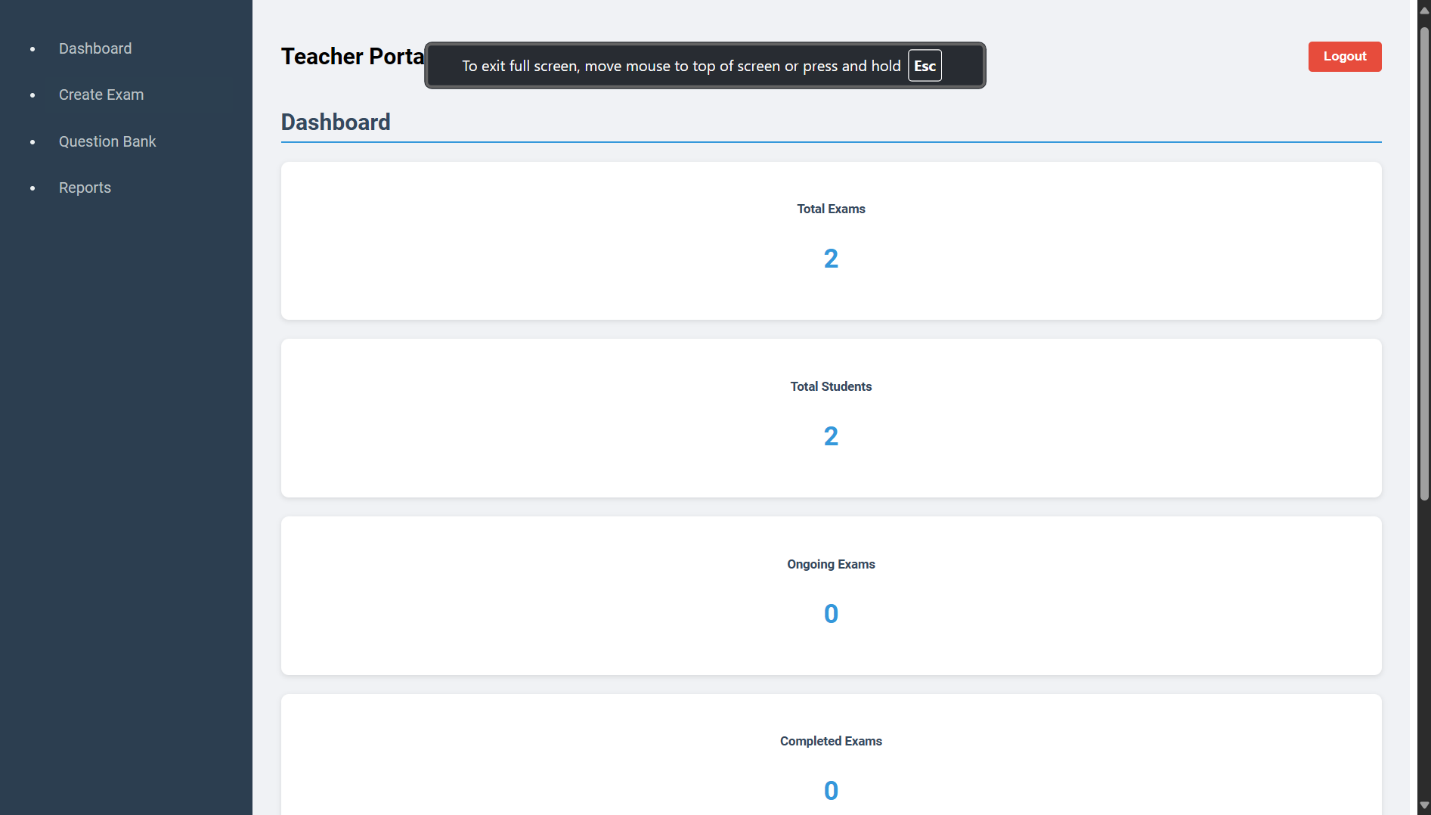
**Conclusion**

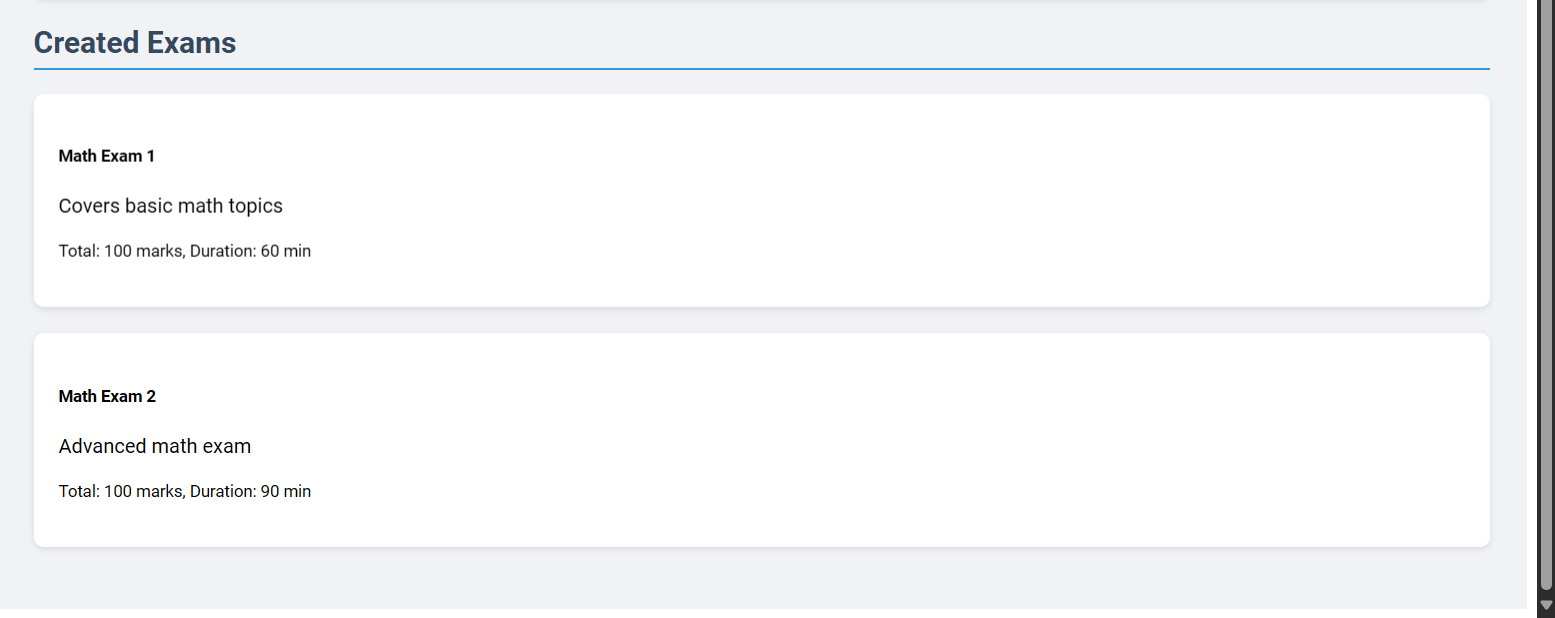
The project successfully demonstrates the design and implementation of a robust web-based MCQ exam management system. The use of Angular, NodeJS, MySQL, and CSS ensured a responsive, secure, and scalable application. Key functionalities such as distributed question contribution, were effectively integrated. The system has the potential to streamline online assessments and enhance the overall educational experience by providing immediate feedback and comprehensive reporting.

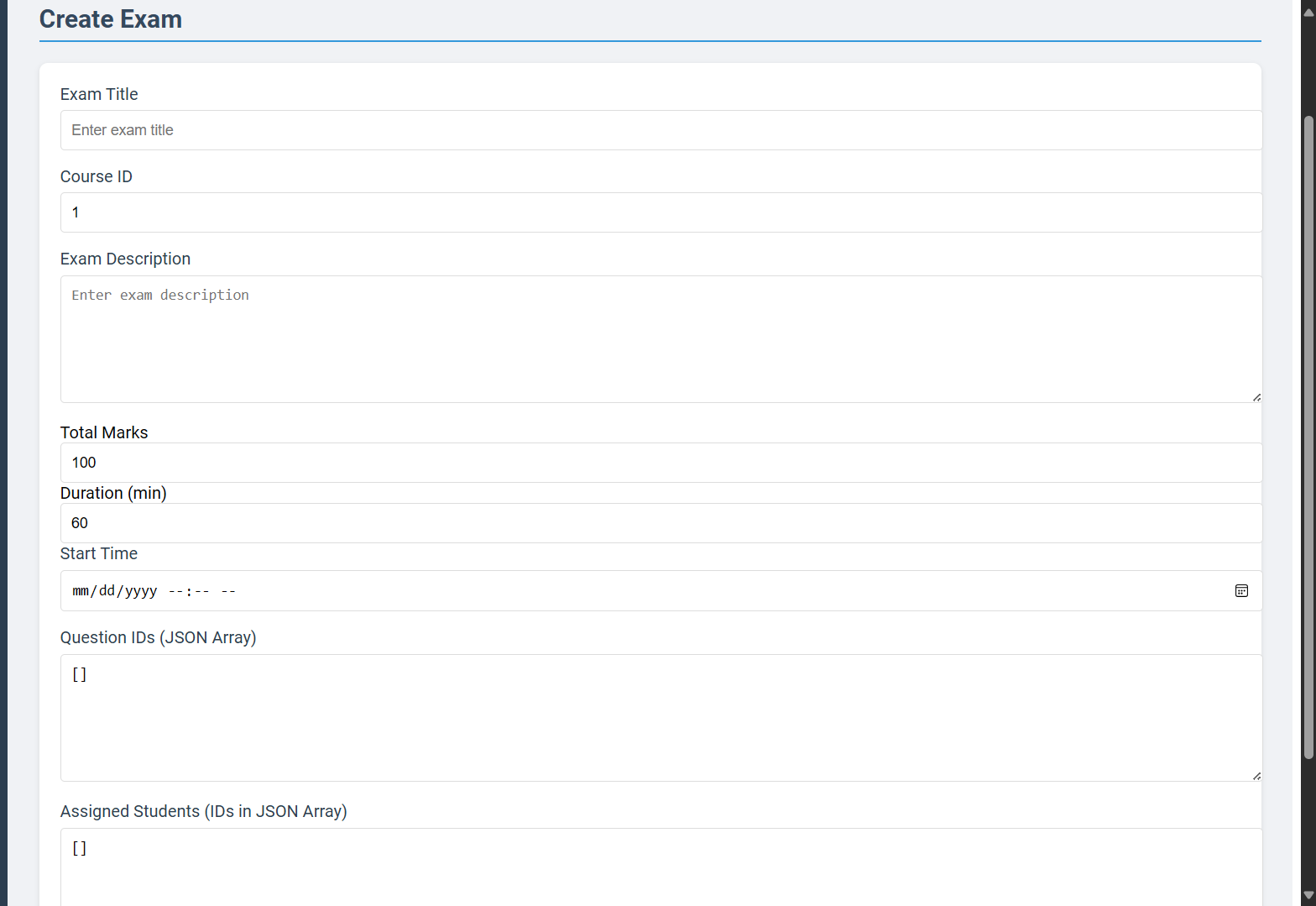


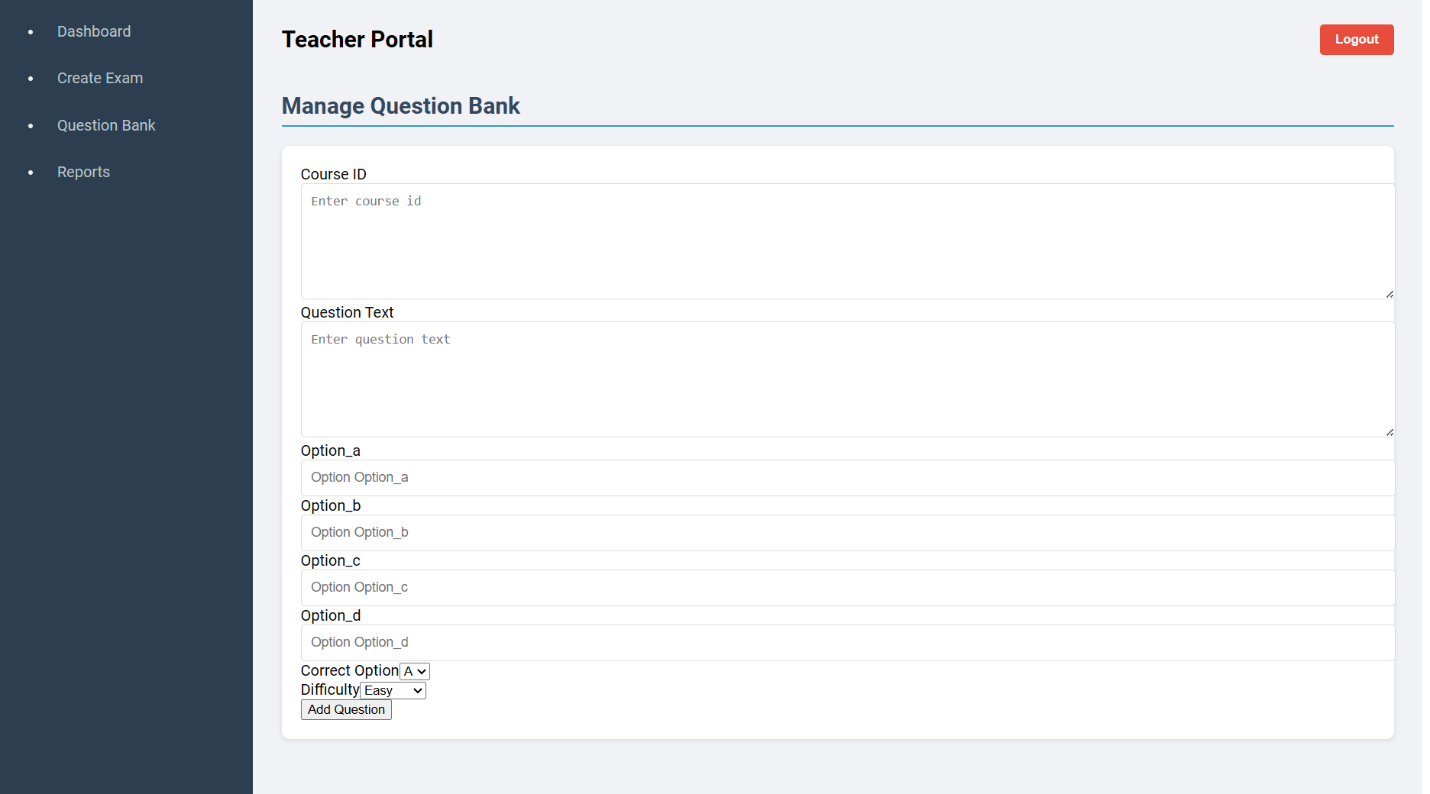


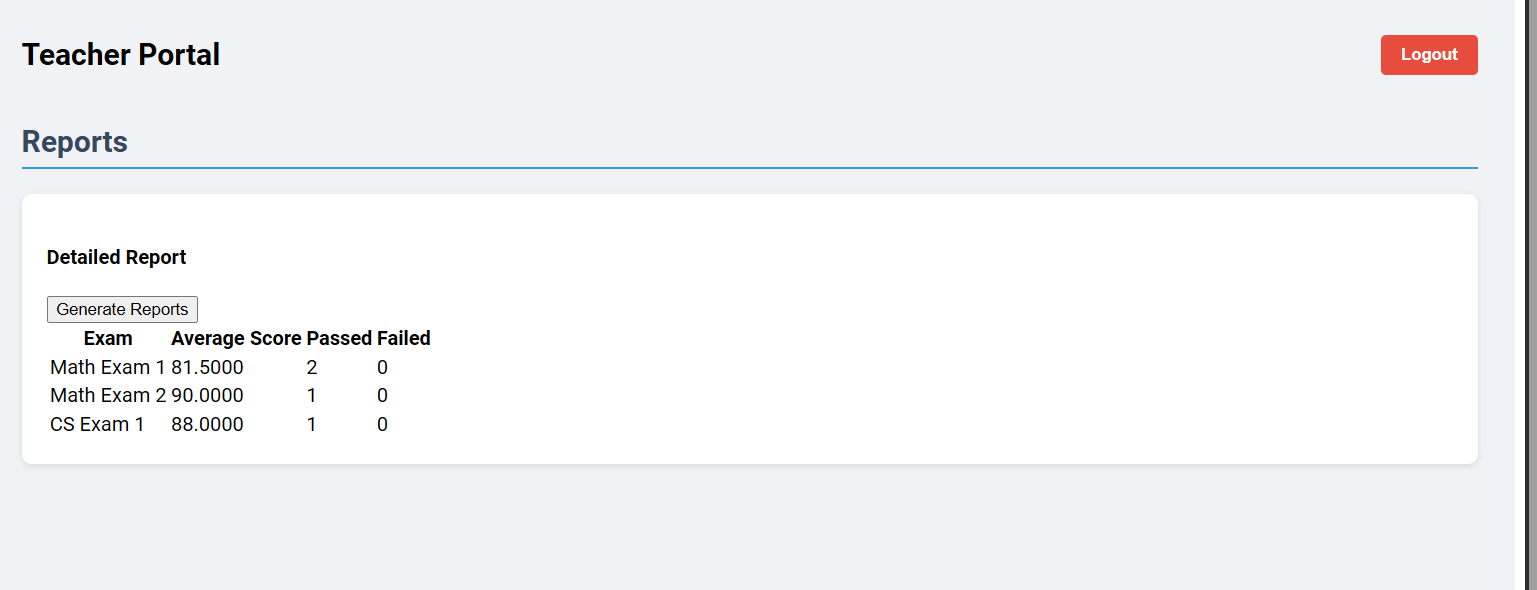
TEACHER PORTAL:  












STUDENT PORTAL:  
