**CrediChain– Blockchain-Based Academic Record Management**

**PHASE 3 – Final Report and Submission**

**College Name:** VIT Bhopal University

**Name:** Aakansh Yadav

# 1. Objective of the Project

The primary objective of **CrediChain** is to develop a blockchain-based academic record management system that ensures secure, tamper-proof, and decentralized storage of student records. The platform empowers both students and faculty with secure access to academic data while maintaining data integrity, transparency, and auditability. By leveraging blockchain technology, **CrediChain** creates an immutable ledger of academic transactions, preventing unauthorized modifications and ensuring trust in the academic evaluation process.

# 2. Tools & Technologies Used

| **Technology** | **Purpose** |
| --- | --- |
| HTML & CSS | Frontend layout, design, and responsive styling |
| JavaScript | Frontend interactivity, blockchain logic, and API calls |
| Node.js | Backend server, handling API requests and logic |
| SQLite | Lightweight local database for storing academic records |
| Docker | Containerized deployment for consistent environments |
| Flask (Python) | Serves the static frontend over HTTP for client-side interaction |
| crypto (Node.js) | Blockchain hashing (SHA-256) for data integrity in academic records |

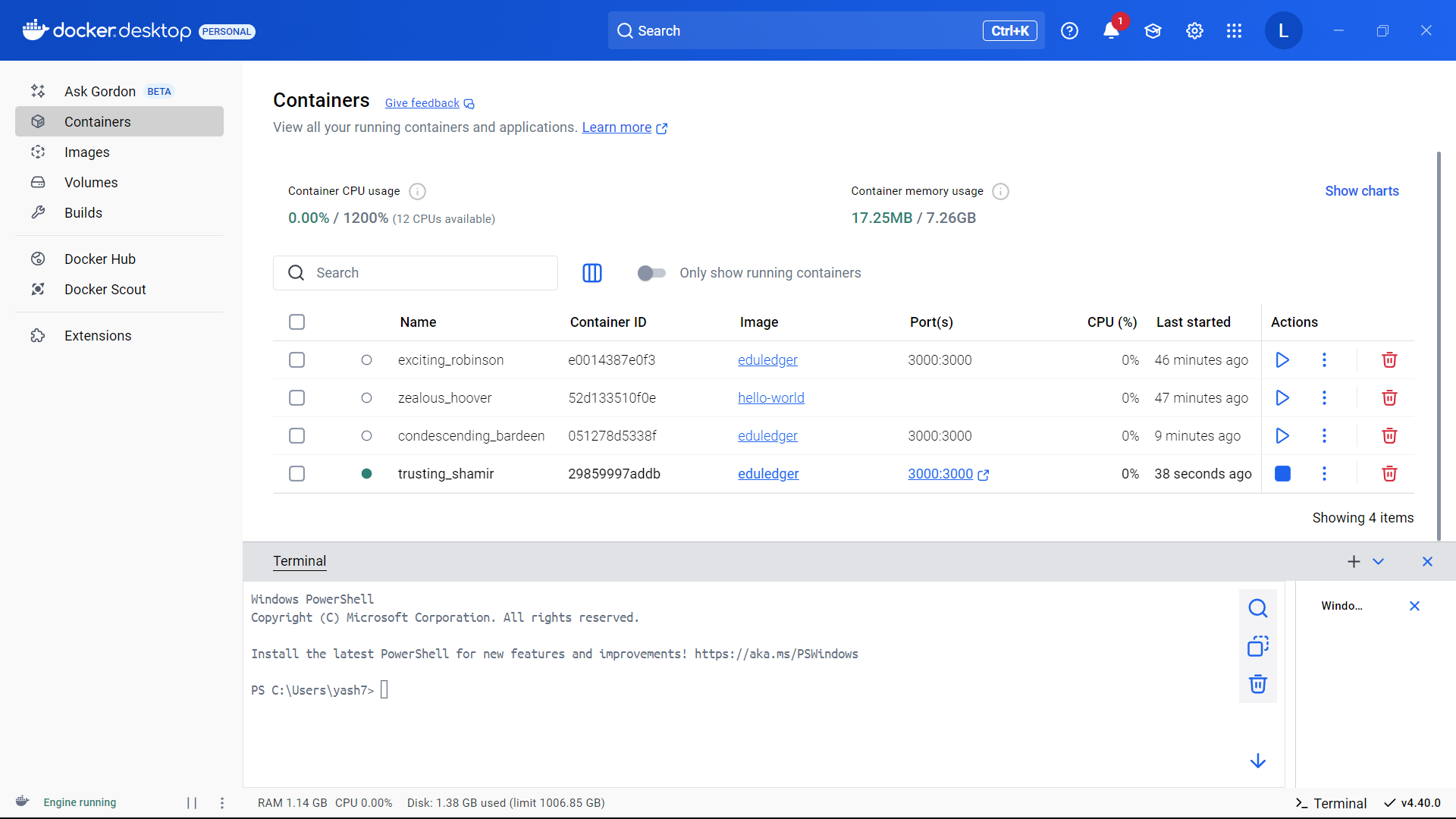
# 3. Summary of Work Done

* **Phase 1**: The concept of **CrediChain** was proposed to address the need for secure, transparent, and tamper-proof academic record management using blockchain technology.
* **Phase 2**: The full-stack solution was implemented, including:
* Custom blockchain logic in JavaScript to ensure data integrity and immutability of academic records.
* Backend API built with Node.js and Express for handling requests related to student records, grades, and faculty management.
* SQLite database for structured and efficient storage of academic data.
* A responsive web-based frontend developed with HTML, CSS, and JavaScript for intuitive user interaction.
* Dockerized deployment environment with Flask serving the static frontend and handling routing.
* **Phase 3**: The blockchain functionality was validated, application logic was tested, and the final report was prepared with comprehensive documentation and visual aids to support the project’s outcomes.

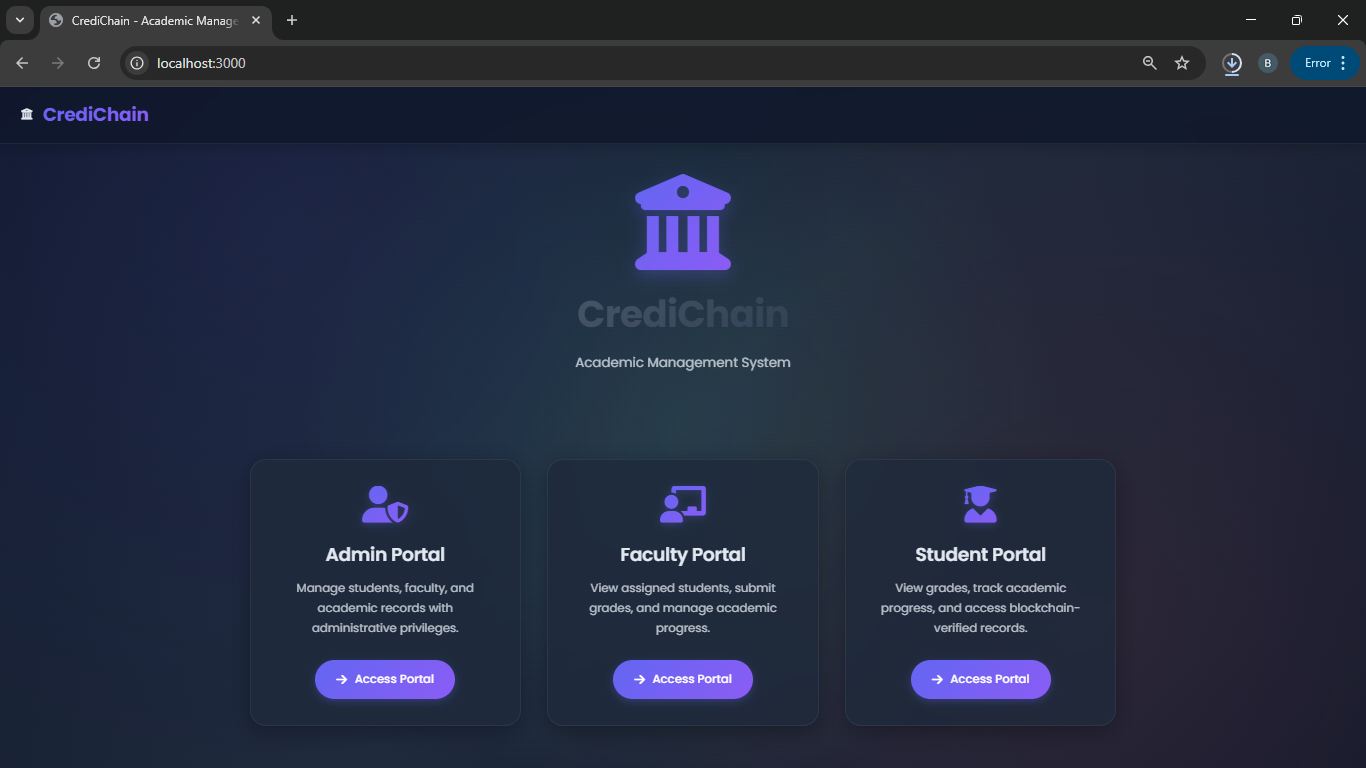
# 4. Features Implemented

* **Proof-of-Work logic** to secure the blockchain and prevent unauthorized tampering.
* **API endpoints** for user registration, login, and management of academic records (student grades, faculty details, etc.).
* **SQLite database integration** for structured storage of academic data.
* **Frontend UI** designed for intuitive user interaction, allowing students, faculty, and administrators to interact with the system.
* **Blockchain chain validation** logic to ensure the integrity of stored records and prevent any tampering.
* **Dockerized deployment** with Flask serving the static frontend and a Node.js backend for a consistent and portable application environment.
* **Data filtering by student ID** for easy retrieval of records specific to users or faculty.
* **Fully functional CRUD operations** for creating, reading, updating, and deleting academic records and user profiles.

**5. Screenshots:**



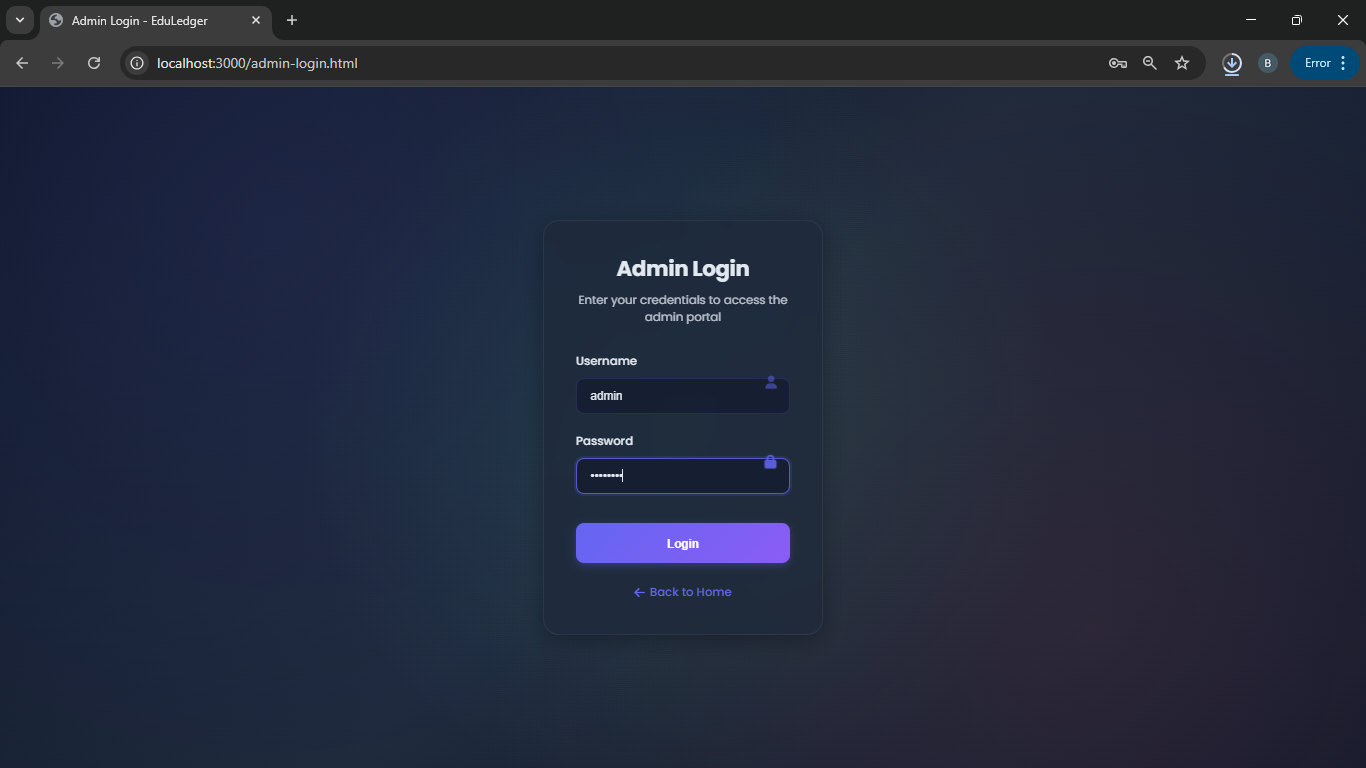
**Figure 1:** Docker Container Running



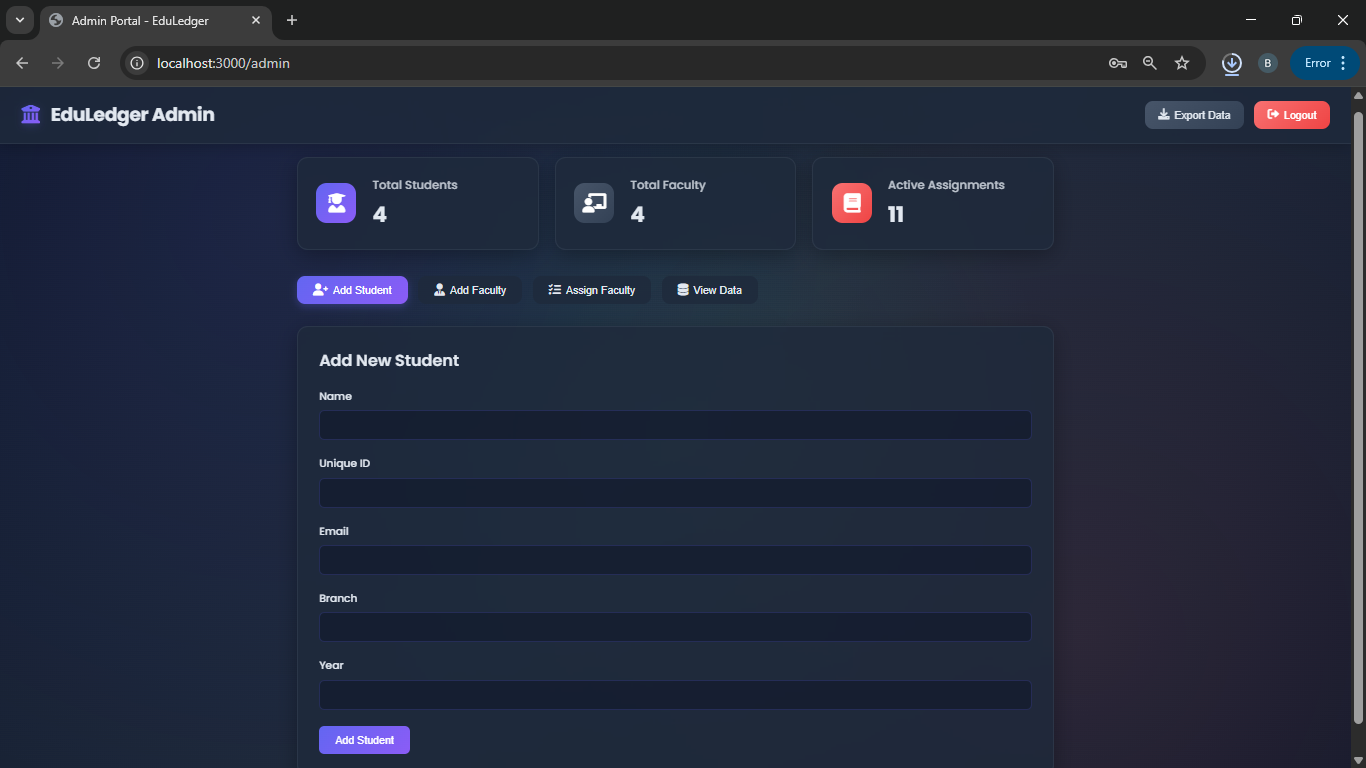
**Figure 2:** Home page of the website



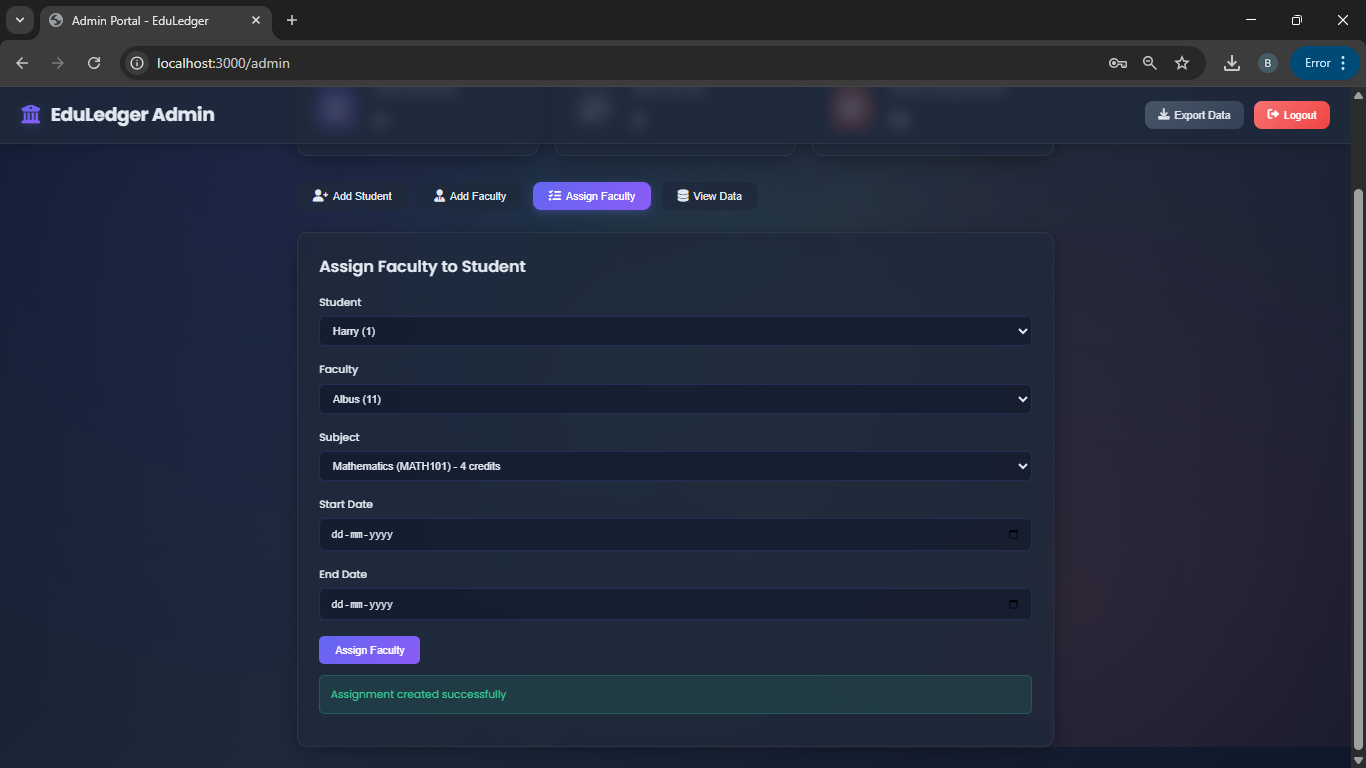
**Figure 3:** Entries made into the database shown in docker terminal



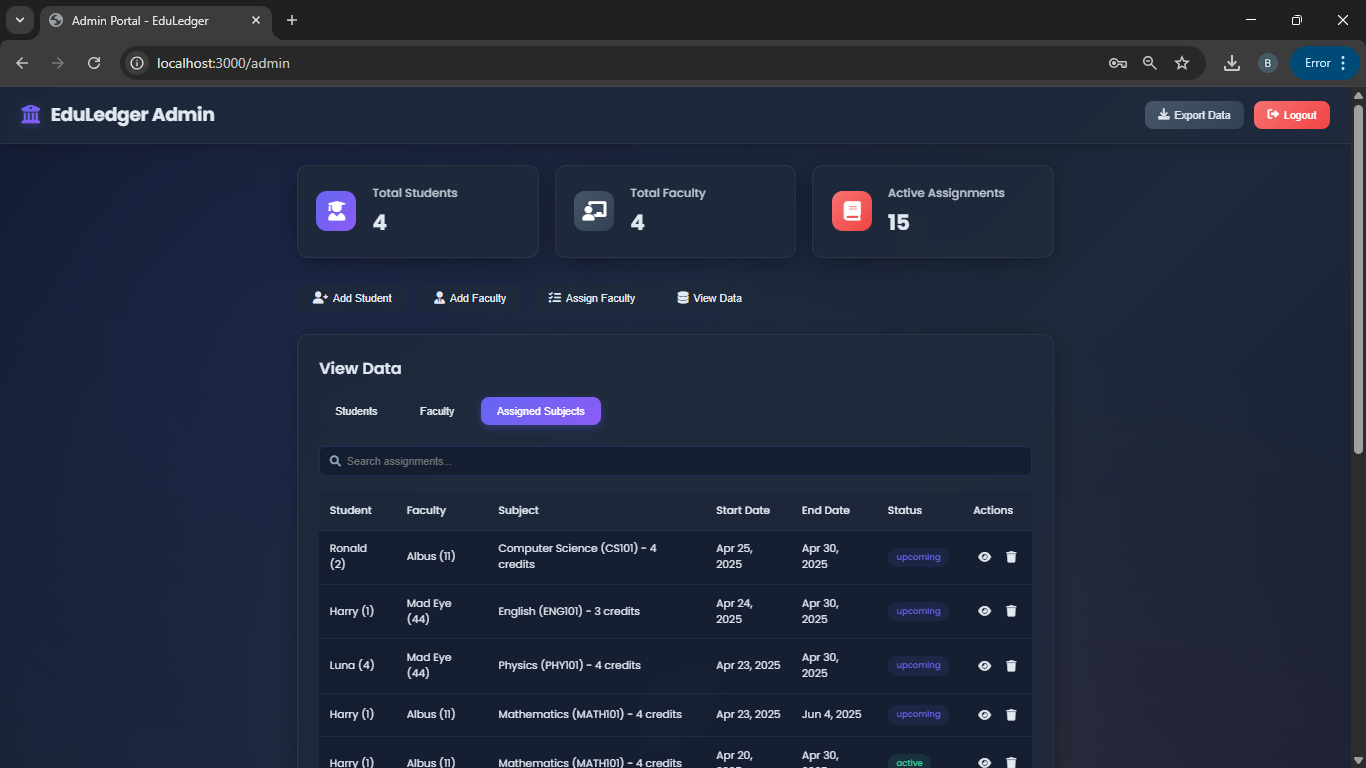
**Figure 4:** Logging in to the website



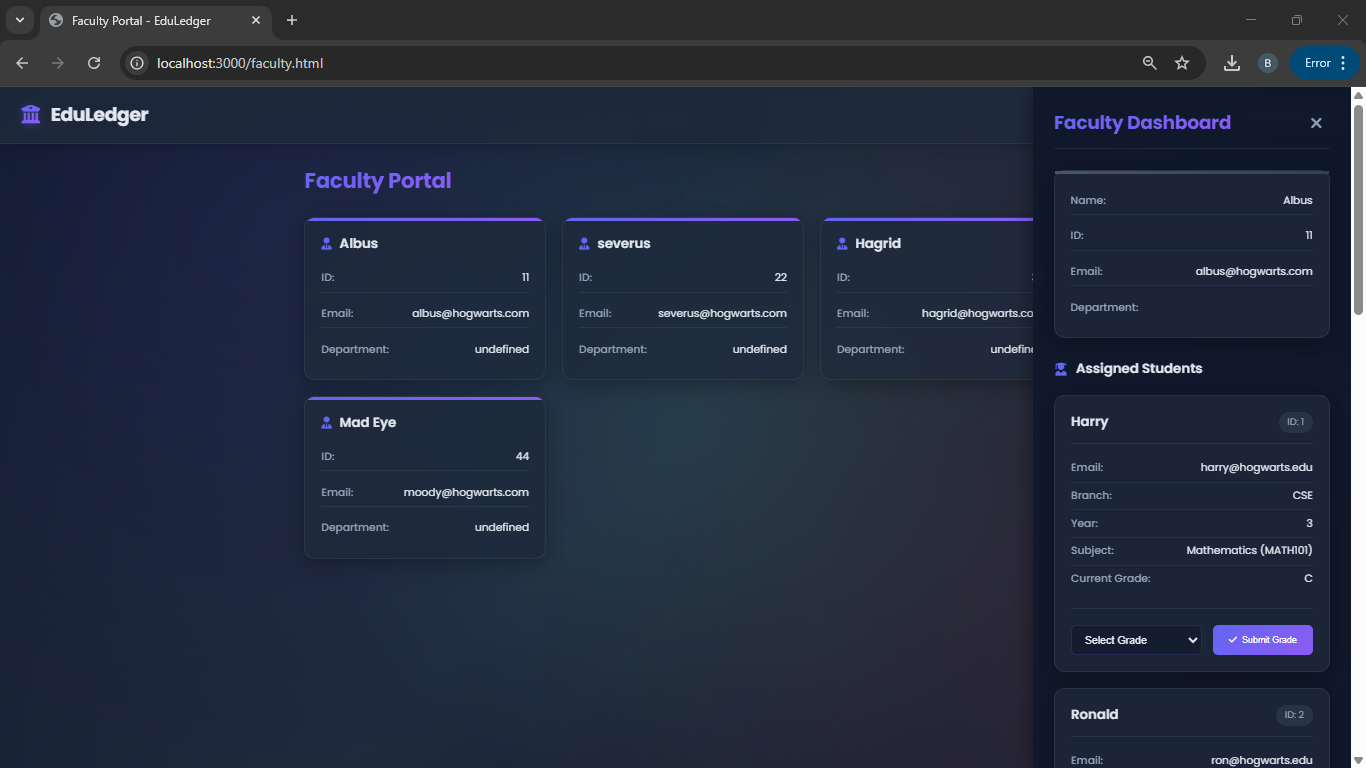
**Figure 5:** Data entry for student



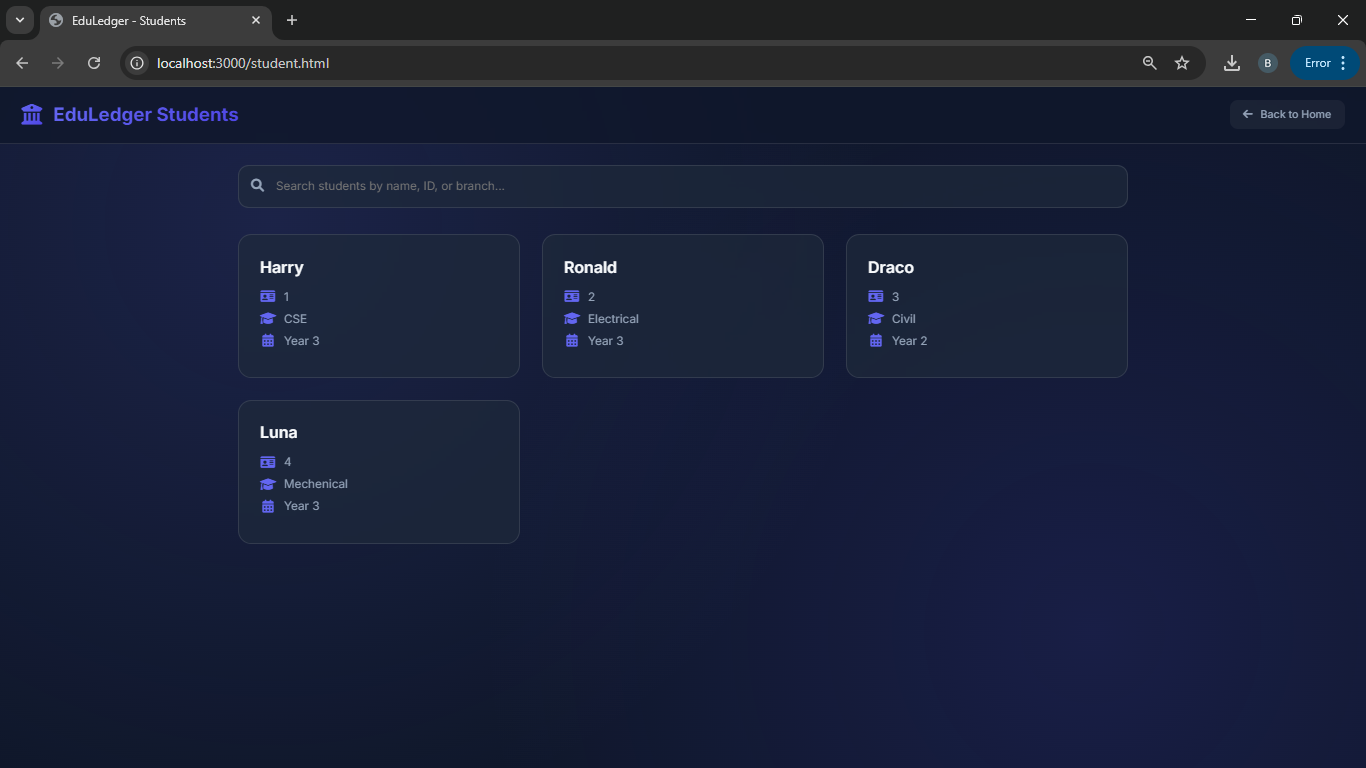
**Figure 6:** Data saving in the database



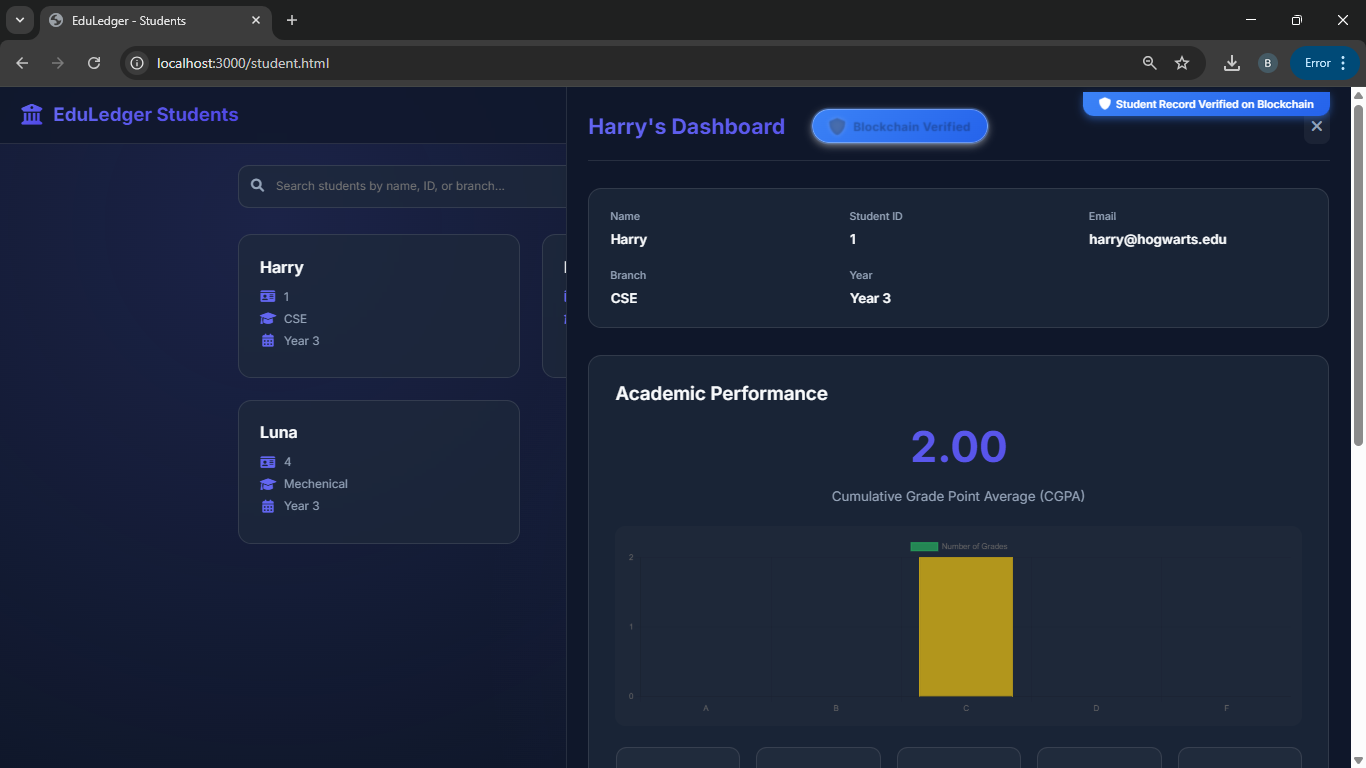
**Figure 7:** Assigned Subjects



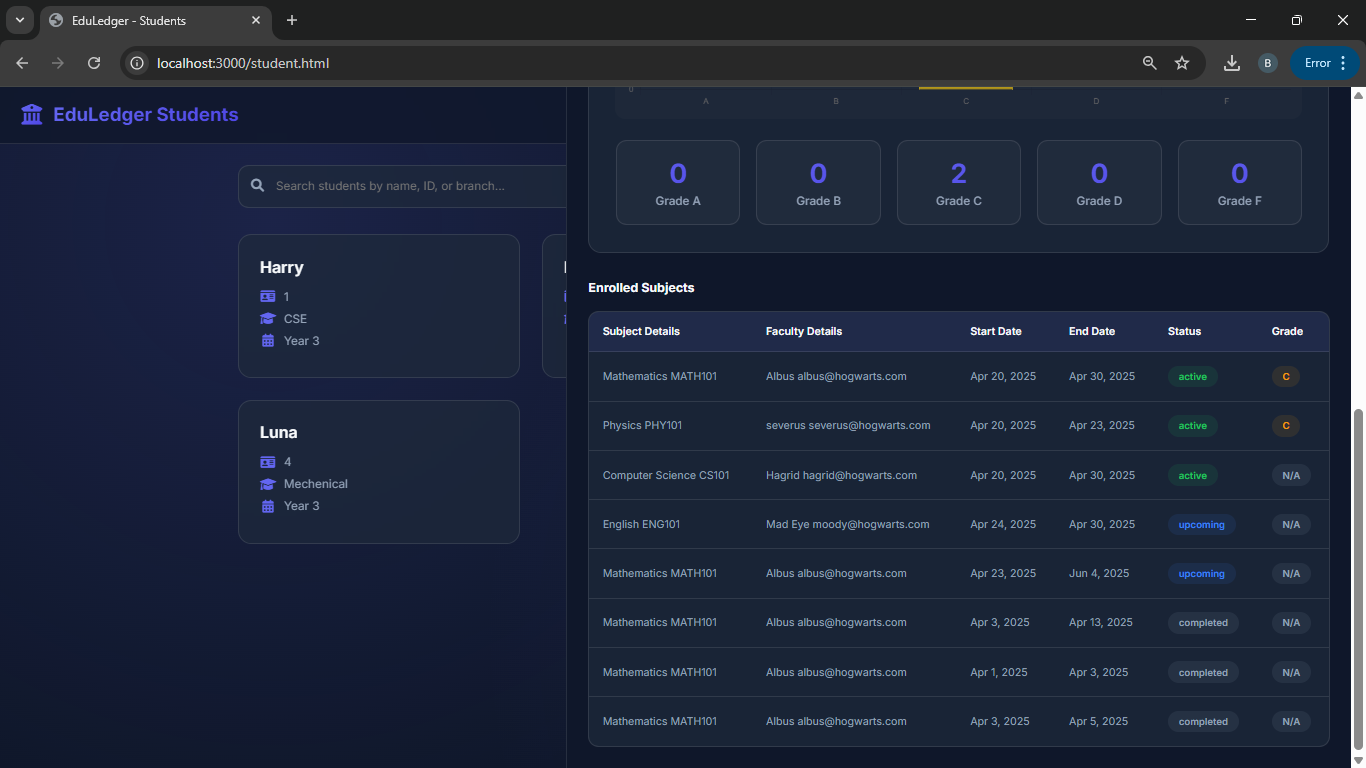
**Figure 8:** Faculty Portal



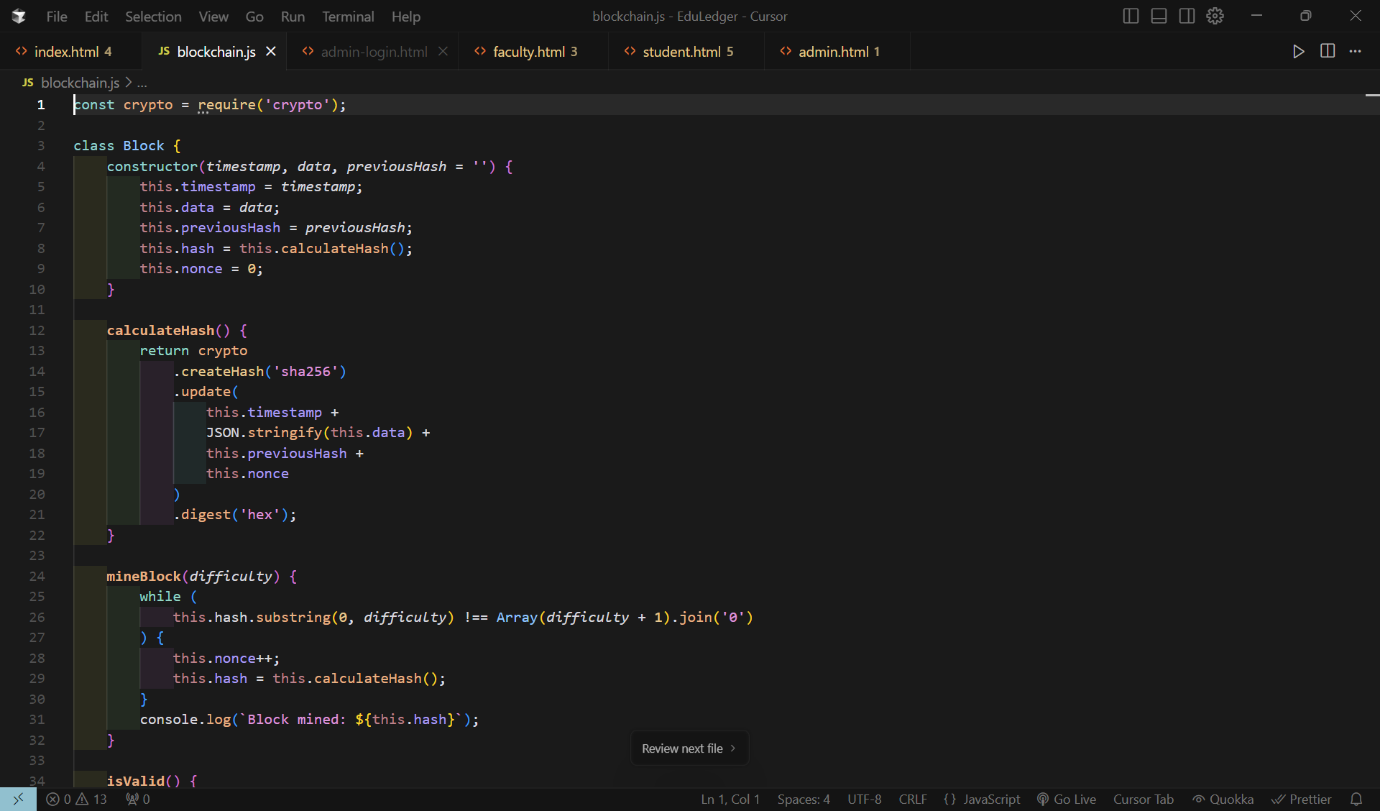
**Figure 9:** Student Dashboard

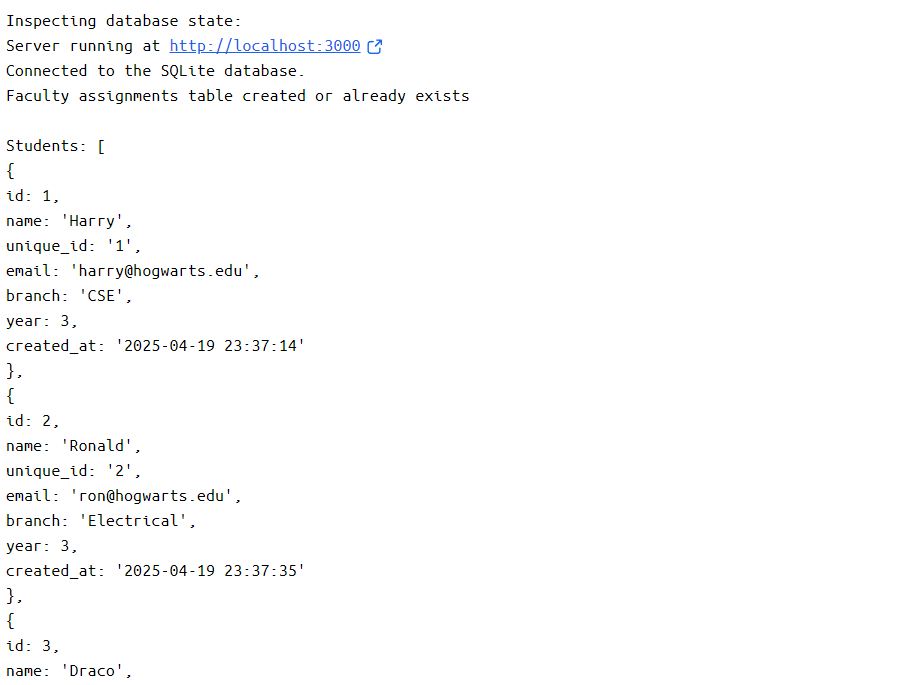


**Figure 10:** Verification

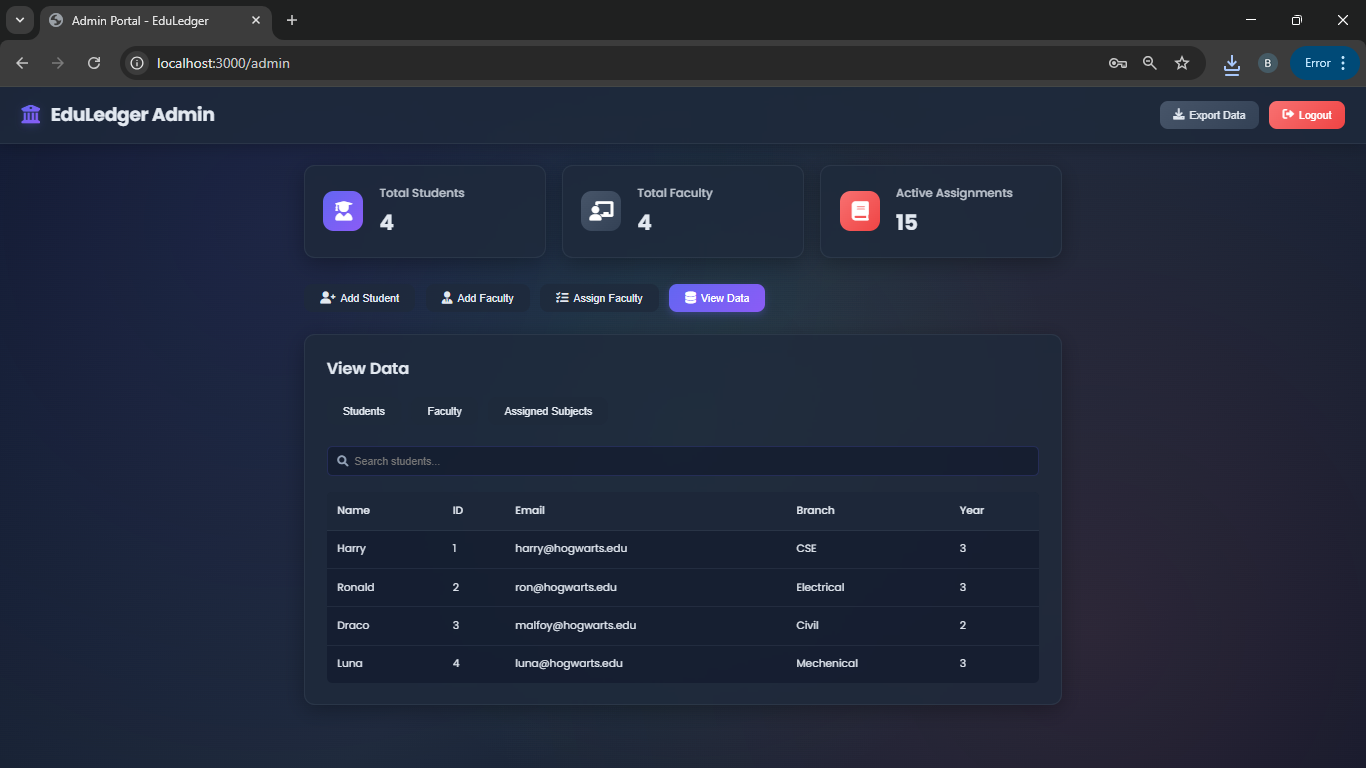


**Figure 11:** Enrolled Subjects

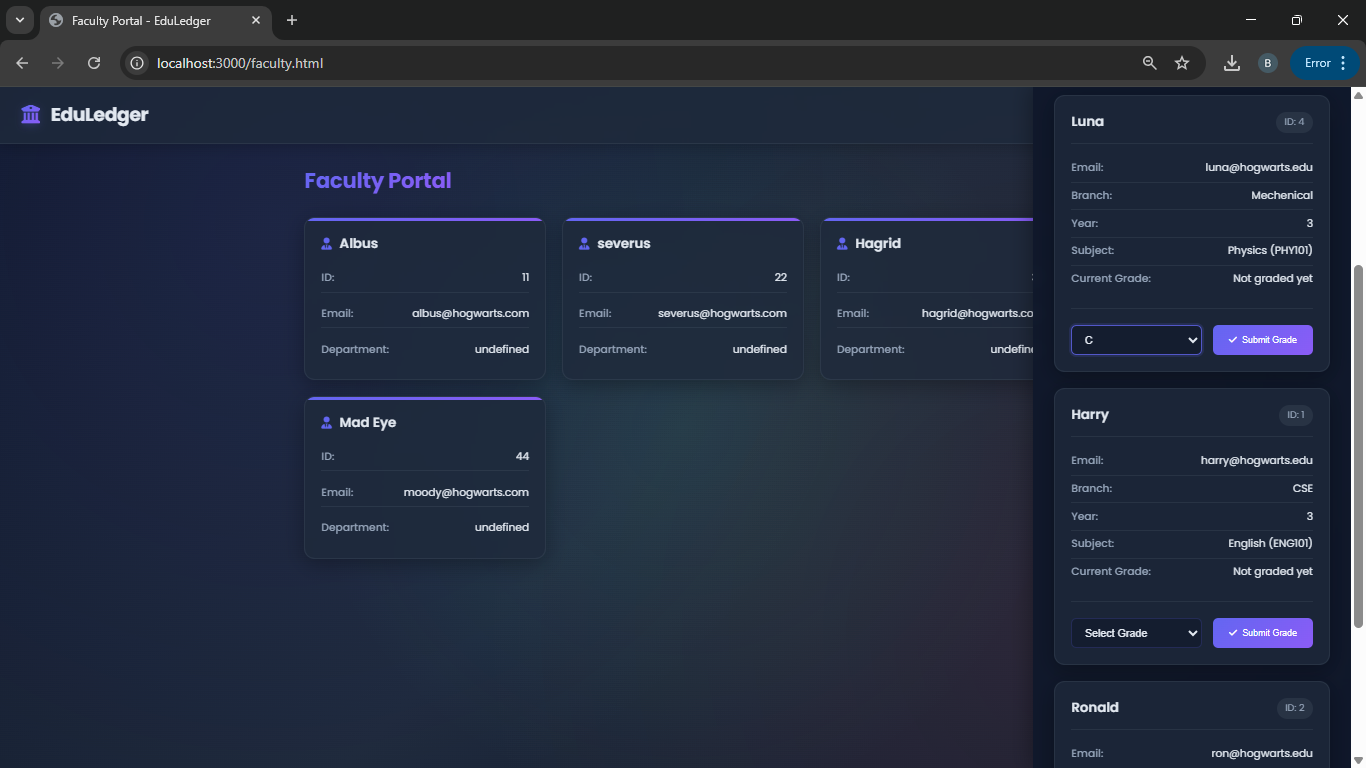
**Figure 12:** Blockchain.js



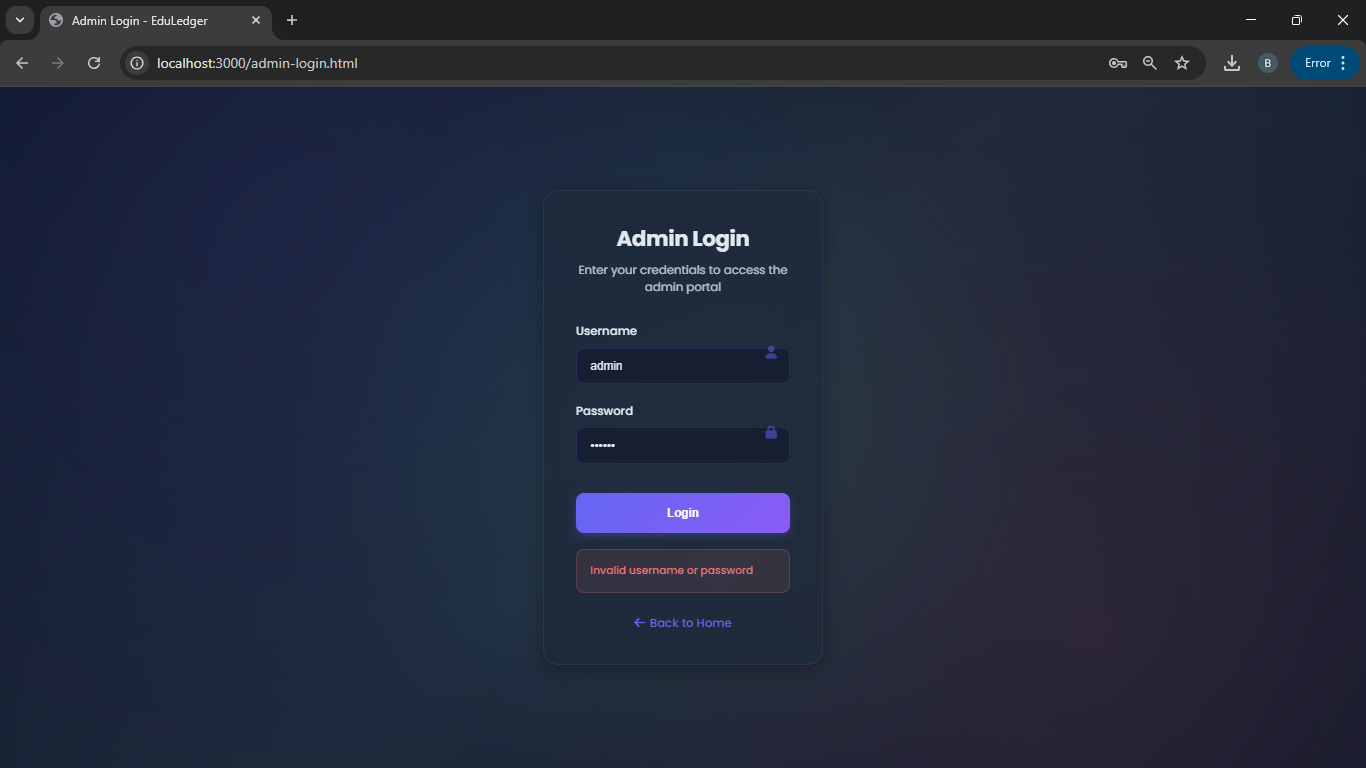
**Figure 13:** Hosting logs



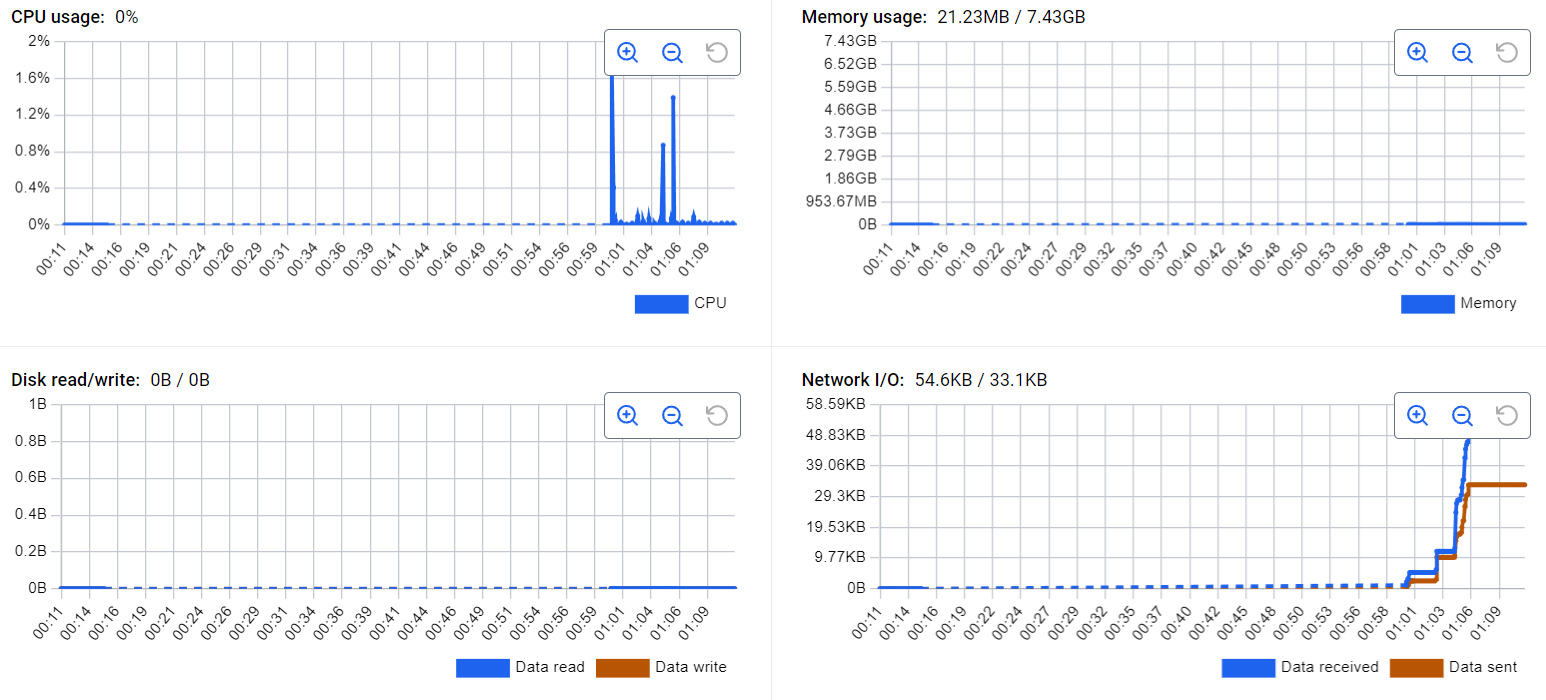
**Figure 14:** Export data



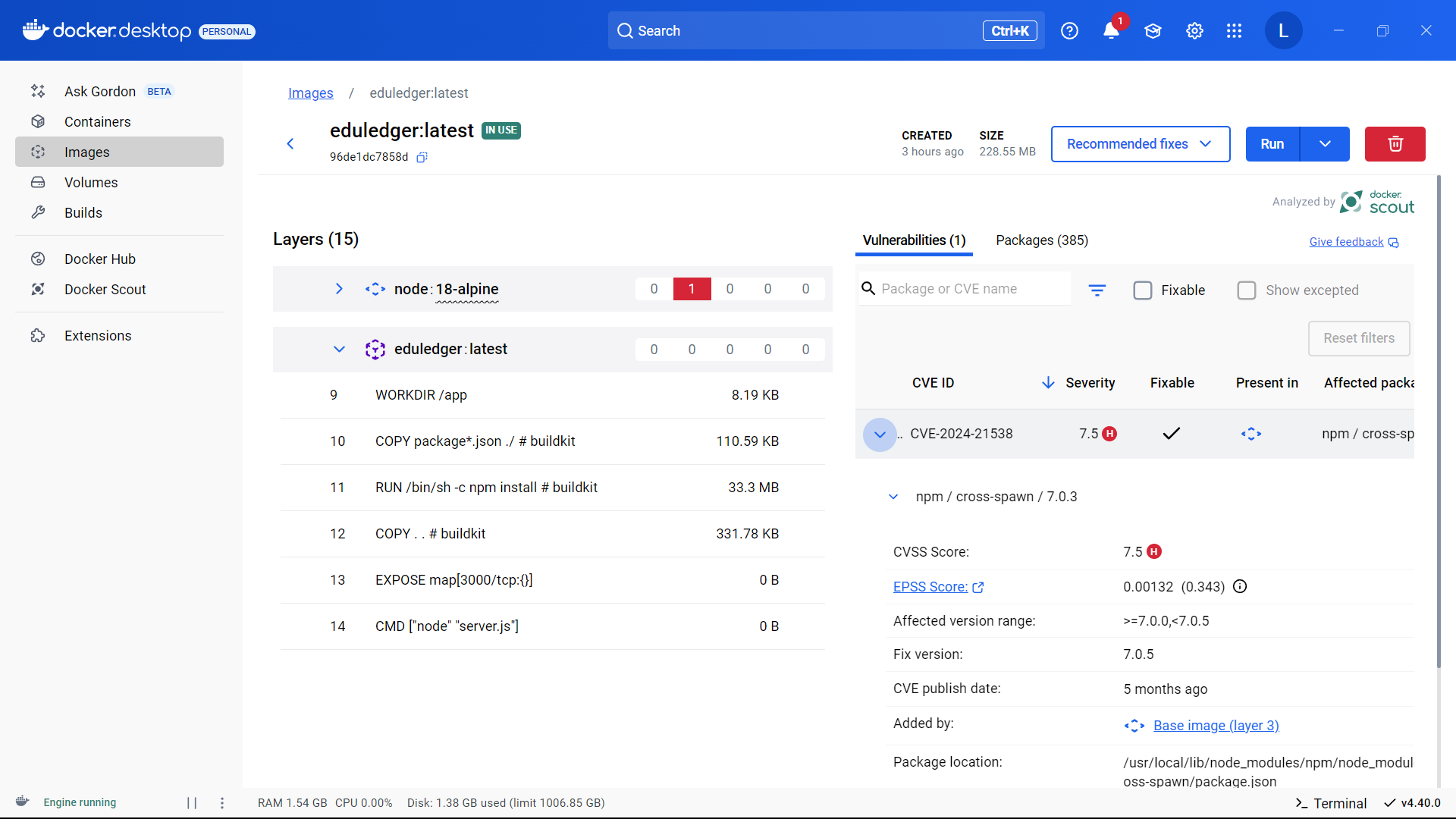
**Figure 15:** Faculty assigning Grades



**Figure 16:** Invalid Credentials while logging in



**Figure 17:** Docker Graphs of resource utilization



**Figure 18:** Layers of Docker File

# 6. Sample Block Structure

Faculty: [

{

id: 1,

name: 'Albus',

unique\_id: '11',

email: 'albus@hogwarts.com',

research\_area: 'All',

post: 'Headmaster',

created\_at: '2025-04-19 23:38:14'

},

{

id: 2,

name: 'severus',

unique\_id: '22',

email: 'severus@hogwarts.com',

research\_area: 'Potions',

post: 'HOD',

created\_at: '2025-04-20 07:42:36'

}

]

# 7. Results and Testing

| **Test Case** | **Input** | **Expected Result** | **Status** |
| --- | --- | --- | --- |
| Register User | Valid username/password | User created with hashed password | ✅Passed |
| Login User | Correct credentials | Login successful, user session initiated | ✅Passed |
| Add Academic Record | Valid student/faculty | Record saved in DB; block mined and added to blockchain | ✅Passed |
| Fetch Records by Student ID | Valid student ID | All matching academic records returned | ✅Passed |
| Blockchain Tampering Detection | Manually altered block data | Blockchain marked as invalid due to hash mismatch | ✅Passed |

# 8. Key Learnings

# Gained in-depth understanding of how blockchain ensures academic data integrity and tamper-resistance.

# Learned to implement custom Proof-of-Work for securing records in a decentralized ledger.

# Developed a full-stack academic management system using Node.js, Express, SQLite, and JavaScript.

# Integrated Docker and Flask for multi-container orchestration and smooth deployment.

# Applied cryptographic hashing and password encryption techniques for user and data security.

# Practiced building modular, scalable backend APIs and frontend components for smooth user interaction.

# 9. Highlights of the Project

* Each academic record is securely hashed and appended to a blockchain, ensuring traceability.
* User-friendly web interface for students and academic administrators.
* Fully Dockerized setup with seamless interaction between frontend, backend, and blockchain engine.
* Complete integration of database, blockchain, REST APIs, and UI in a unified platform.
* Focused on transparency, auditability, and secure record handling in academic institutions.

# 10. Future Scope

* Replace SQLite with MongoDB to enable scalable document-based storage.
* Implement access key-based encryption for role-based visibility of academic records.
* Enable cross-institution record sharing through peer-to-peer blockchain syncing.
* Build a mobile app or progressive web app (PWA) version for easier accessibility.
* Leverage IPFS to store digital certificates or documents with blockchain-verified hashes.
* Transition to cloud-native deployment using platforms like AWS or IBM Cloud for real-world usage.

# 11. Conclusion

**CrediChain** addresses a pressing challenge in academic record management by providing a secure, tamper-proof, and transparent solution using custom-built blockchain technology. The platform ensures that student records—such as grades, certificates, and credentials are permanently stored with integrity and immutability. Through its full-stack implementation, **CrediChain** allows seamless user registration, record addition, and retrieval via a clean, responsive interface.

The integration of a lightweight database (SQLite), a modular backend (Node.js), and a custom blockchain engine demonstrates the practical potential of decentralizing sensitive academic data. By deploying the system in Docker containers and managing frontend routing via Flask, the project achieves a scalable and maintainable architecture.

This working prototype lays a strong foundation for future upgrades, including cloud-based hosting, record encryption, cross-institution sharing, and mobile compatibility. **CrediChain** proves how blockchain can revolutionize academic systems by ensuring trust, accountability, and accessibility in record-keeping.