|  |  |
| --- | --- |
| A picture of a winding road and trees  Contract monthly claim system  Programming 2B | Abstract  The Contract Monthly Claim System (CMCS) is a web-based prototype designed to simplify the submission and management of academic claims. It enables lecturers to register, log in, and submit claims with supporting details and documents. Programme coordinators and management can review, approve, or reject claims through a structured workflow. Built with C#, MySQL, HTML, CSS, and JavaScript, the system follows an MVC design for security and scalability. The prototype promotes efficiency, transparency, and accountability in the claims approval process.  Amogelang Refilwe Matlhaga  PROG6212 |

**Report on the Contract Monthly Claim System (CMCS)**

**Introduction**

The Contract Monthly Claim System (CMCS) is a web-based prototype developed to streamline the submission and management of academic claims. Traditionally, these claims were managed through manual or partly digital methods, leading to inefficiencies and delays. CMCS addresses these issues by providing a secure and user-friendly digital platform for lecturers, coordinators, and programme management. The system is designed using the Model–View–Controller (MVC) pattern, with C# for the backend, MySQL for database management, and HTML, CSS, and JavaScript for the front end. This architecture ensures scalability, security, and accessibility across devices.

**System Overview**

The CMCS allows lecturers to register, log in, and submit claims that include key details such as the number of sessions, hours worked, hourly rate, module, faculty, and supporting documents. Claims are initially marked as *Pending* and progress through a structured workflow. Coordinators review claims, approving or rejecting them, before management makes the final decision. This workflow promotes accountability and transparency, ensuring each role fulfils its responsibility.

**Objectives and Core Features**

The primary objectives of CMCS are:

* To provide a secure and transparent platform for submitting claims.
* To streamline the approval process across different roles.
* To ensure data integrity through robust database design and validation.

The system incorporates essential features, including user authentication, authorization, secure password storage, file upload restrictions, and claim tracking. User stories emphasize role-based functionality: lecturers manage their own claims, coordinators review pending claims, and programme management finalizes outcomes.

**UML Class Diagram**

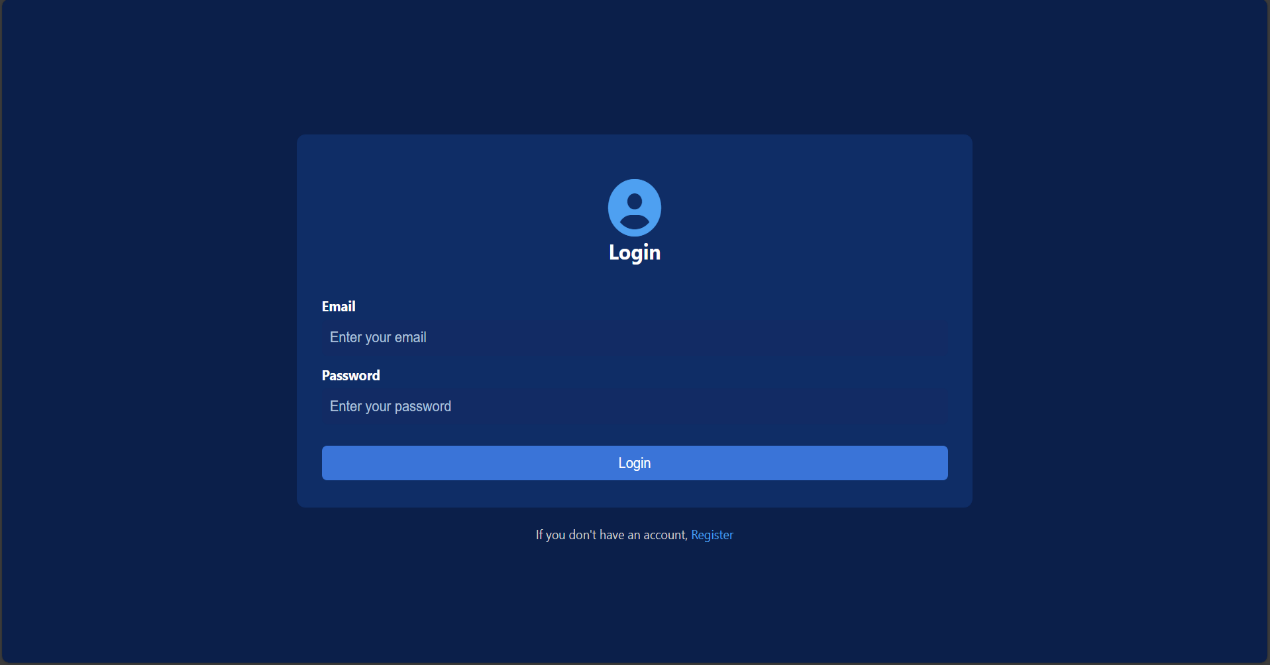
The UML class diagram models two main entities: **Users** and **Claims**.

* The *Users* class includes attributes such as UserID, Full Names, Email, Gender, Role, and Password.
* The *Claims* class records ClaimID, sessions, hours, rate, total amount, module, faculty, documents, claim status, and date.  
  The relationship between Users and Claims is one-to-many, as each lecturer can submit multiple claims. This diagram illustrates the structured database model and highlights the integrity of relationships across the workflow.

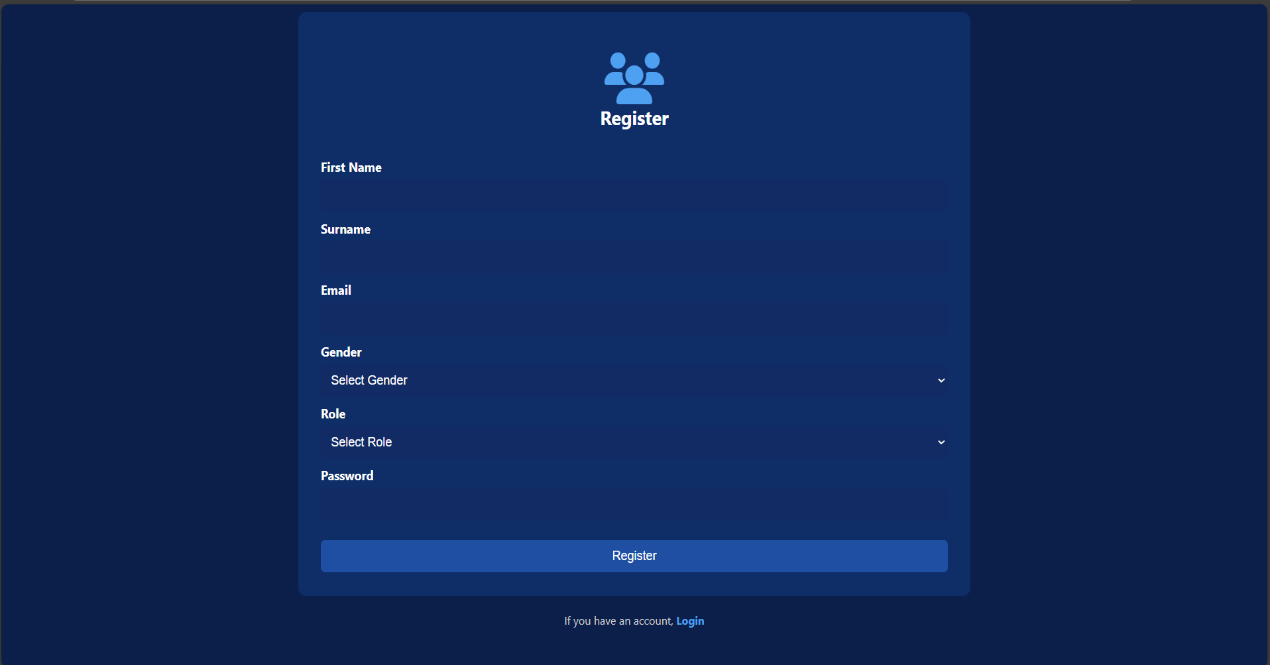
**GUI Design**

The system’s graphical user interface is designed for clarity and ease of navigation. Lecturers access registration, login, claim submission, and claim tracking pages. Coordinators and management have dedicated dashboards for reviewing and finalizing claims. Each page includes relevant input fields and validation mechanisms, ensuring consistency and reducing user errors. For instance, the claim submission page automatically calculates totals and validates document formats, while dashboards display status updates in tabular form for efficiency.

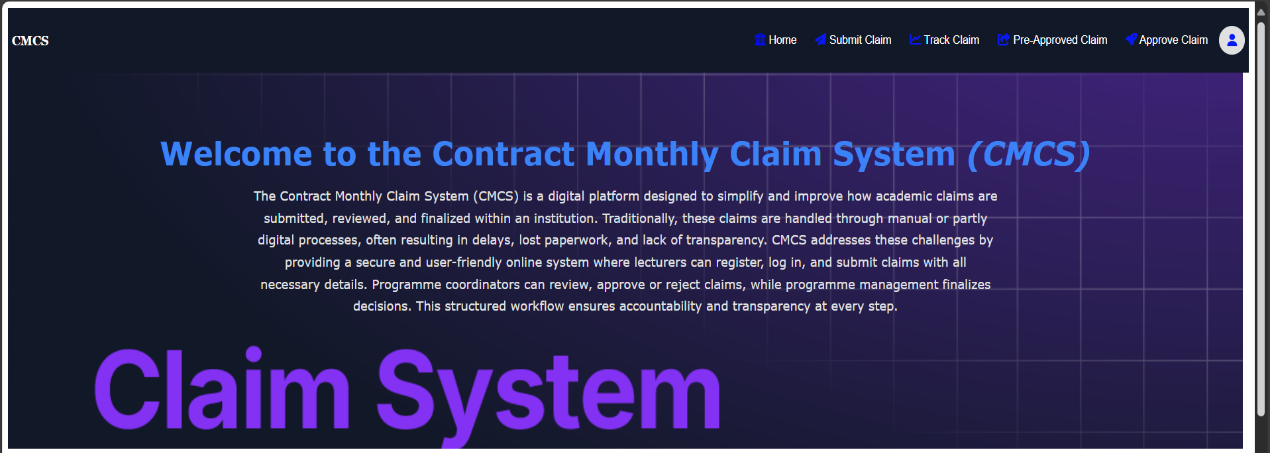
1. The Login Page.



1. The Registration Page.



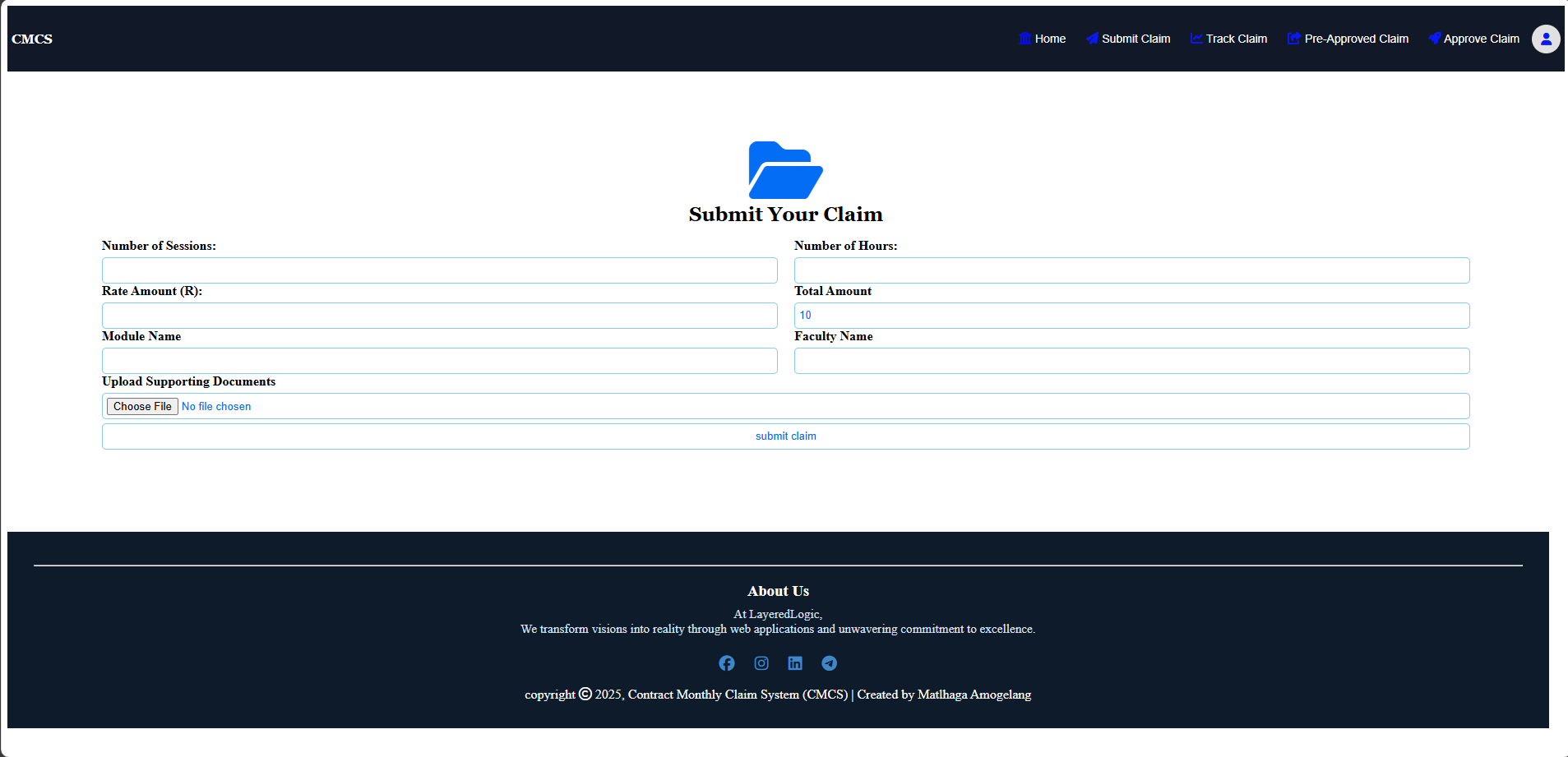
1. The Home Page



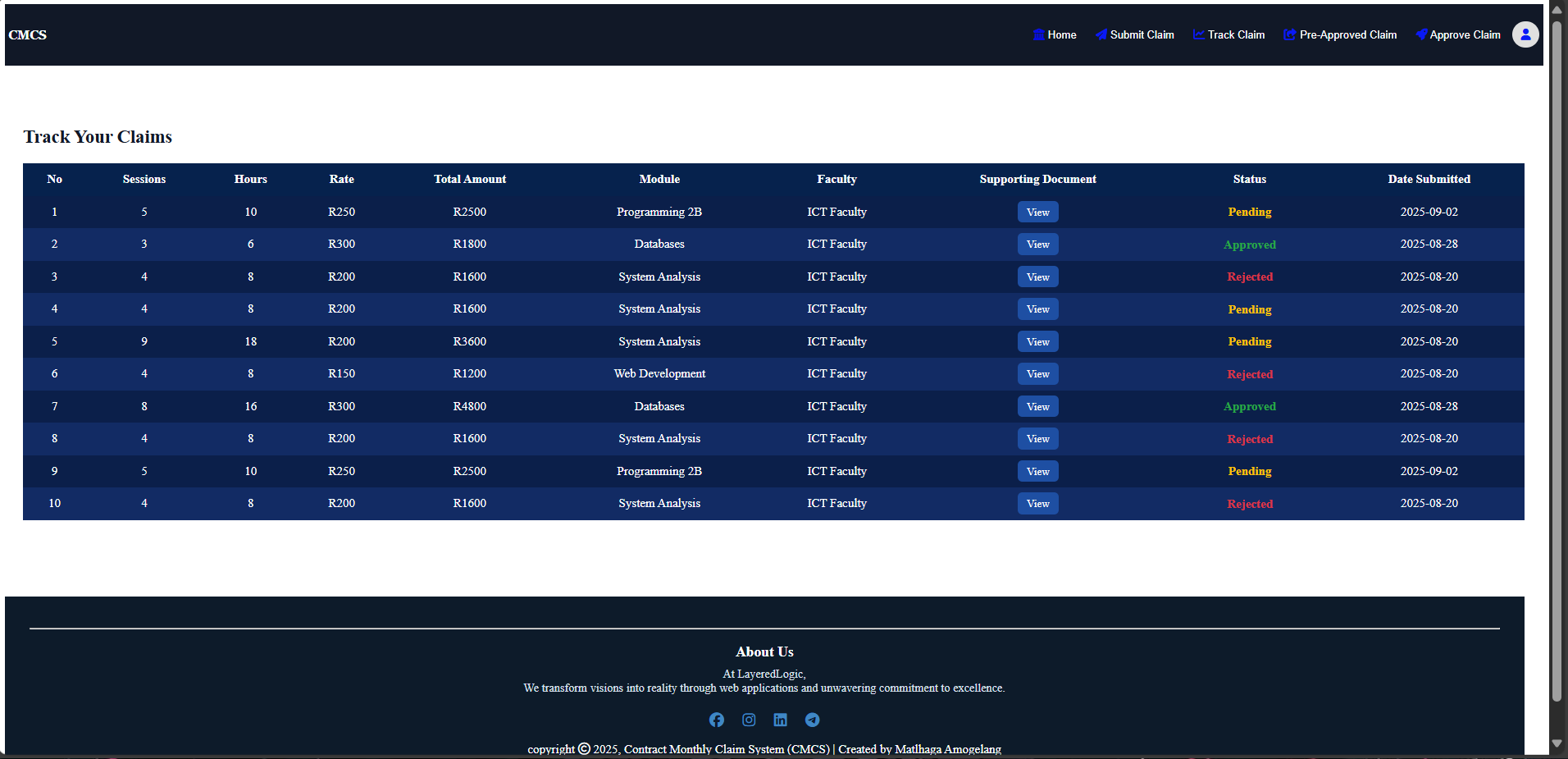
A screenshot of a computer

AI-generated content may be incorrect.

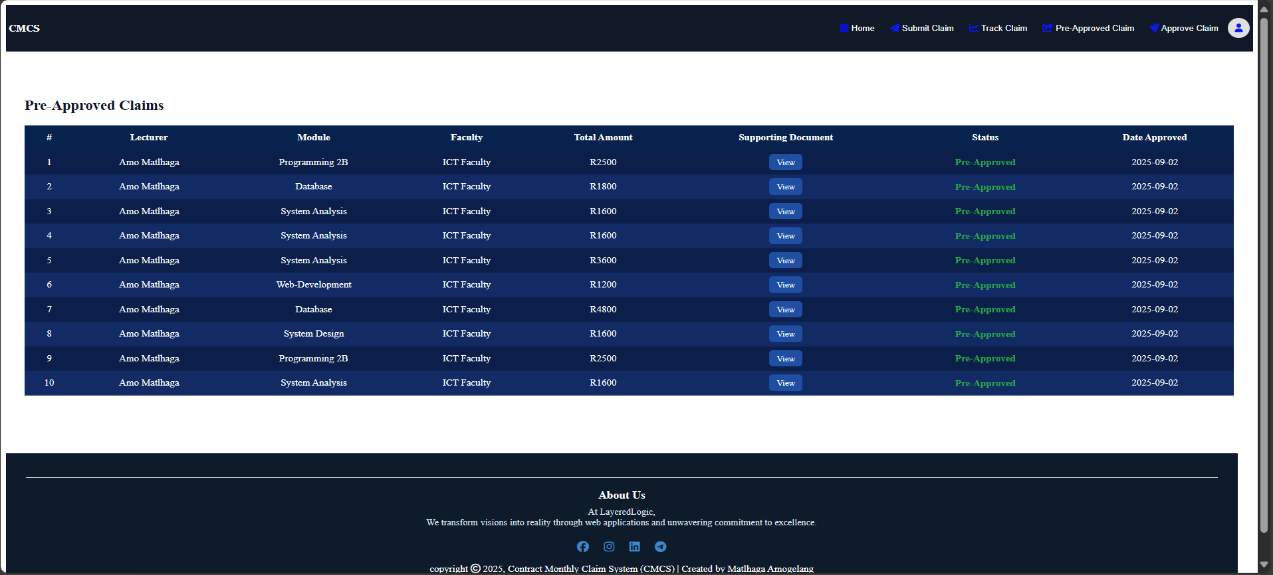
1. The Submit Claim Page



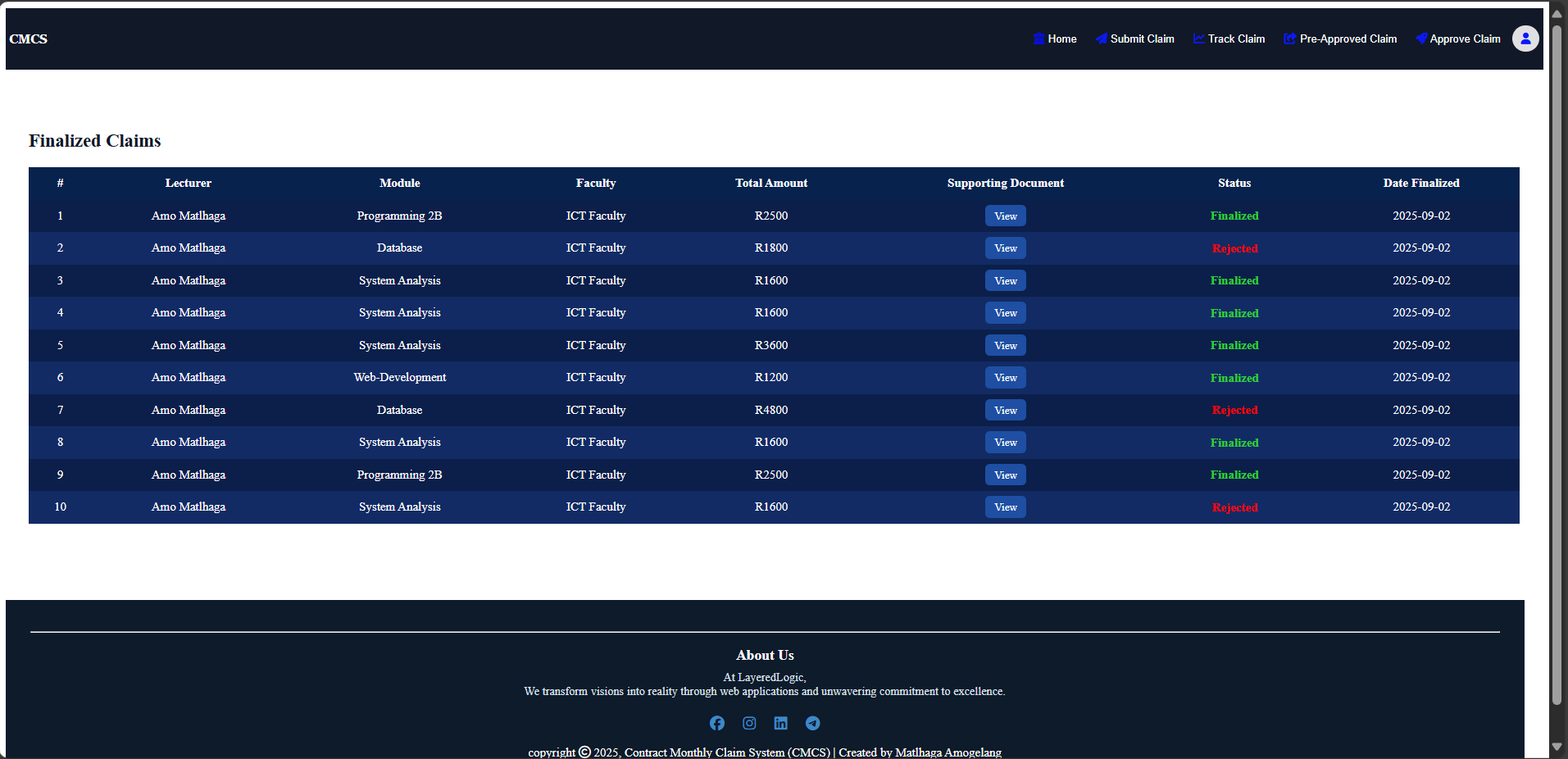
1. The Track Claim Page



1. The Pre-Approved Claim Page



1. The Finalized Claims Page



**Project Plan**

The project follows a phased development approach:

1. **Planning and Requirements (Weeks 1–2):** Define functional and non-functional requirements.
2. **Design (Weeks 2–3):** Database schema, MVC architecture, and wireframes.
3. **Back-End Development (Weeks 3–6):** Implement authentication, authorization, and claims logic.
4. **Front-End Development (Weeks 4–7):** Develop responsive UI and dashboards.
5. **Integration and Testing (Weeks 7–8):** Conduct unit, integration, and user acceptance testing.
6. **Deployment and Review (Week 9):** Deploy prototype and gather feedback.

Dependencies emphasize sequential progression, with testing contingent on integrated front- and back-end components.

**Conclusion**

The CMCS prototype offers a comprehensive solution for managing academic claims, aligning technical design with user needs. Through its MVC architecture, structured database, UML design, and intuitive GUIs, the system enhances efficiency, accountability, and transparency. The project plan ensures systematic implementation, testing, and deployment, positioning CMCS as a reliable and scalable tool for academic institutions.

**Reference List**

1. Lee, H.-Y. and Wang, N.-J., 2019. Cloud-based enterprise resource planning with elastic model–view–controller architecture for Internet realization. Computer Standards & Interfaces, 64, pp.11–23.
2. Philip, M.M., Seshadri, A. and Vijayakumar, B., 2020. Microservices Centric Architectural Model for Handling Data Stream Oriented Applications. Cybernetics and Information Technologies, 20(3), pp.32–44.
3. Dehbozorgi, N. and Norkham, A., 2021. An Architecture Model of Recommender System for Pedagogical Design Patterns. In: Frontiers in Education Conference Proceedings, IEEE.
4. Li, F. and Zhang, J., 2021. Construction and Realization of the Marketing Management Information System for E-commerce Companies Based on SQL Server. In: ACM International Conference Proceedings, pp.389–393.
5. Arias-Serna, M.A. et al., 2022. Development of an application for the calculation of credit risk in entities of the Colombian solidarity sector. Revista Iberoamericana de Sistemas e Tecnologías de Información, 2022(E52), pp.174–188.