|  |  |
| --- | --- |
| PROG part 2  Programming POE:1st part | By ST10440580 |

**Index**

[**1.Documentation** 2](#_Toc208315094)

[**3.PROJECT PLAN** 9](#_Toc208315095)

[**Gnatt Chart** 13](#_Toc208315096)

[**4)Screenshots of prototype** 13](#_Toc208315097)

[**Reference List** 19](#_Toc208315098)

# **1.Documentation**

* **Design choices**

My designs will be designed with simplicity, usability and professionalism, thus complying with the Contract Monthly Claim System.

* **Login and Registration**

These pages will be laid out to include input fields for the Lecturer Name, Programme Coordinator Name and Academic Manager Name, Email and Password. These pages will also include explicit navigation to enable the system to be user-friendly to first time users.

* **Role Dashboards**

The lecturer ,coordinator and academic dashboards will be arranged with clear tables and action buttons that will enable the users to see and handle claims without any complexities whatsoever

**How I will improve user experience when using the interface:**

To improve user interface and make it look professional, I will apply a uniform structure, balanced text, and contemporary design(Gradients and rounded buttons).Then when designing the claim submission, approval and tracking buttons I will ensure that they are represented in the GUI design to indicate the real business process and to ensure that the interface is user friendly, as well as accommodating to the actual goals and requirements of the system.

* **Structure of database**

**Lecturer Table**

| Field | Type | Key | Description |
| --- | --- | --- | --- |
| LecturerID | int | PK | Unique identifier for each lecturer |
| Name | string |  | Lecturer’s full name |
| Email | string |  | Lecturer’s email address |
| HourlyRate | double |  | Rate per teaching hour |
| Claim | string |  | Claim of lecturer |

**Claim Table**

| Field | Type | Key | Description |
| --- | --- | --- | --- |
| ClaimID | int | PK | Unique identifier for each claim |
| LecturerID | int | FK | Unique identifier for each Lecturer |
| Month | DateOnly |  | Claim month |
| SubmittedOn | DATETIME |  | Date claim was submitted |
| TotalAmount | double |  | Calculated total for the claim |
| Details | string |  | Details of claim from lecturer |

**Programme Coordinator Table**

| Field | Type | Key | Description |
| --- | --- | --- | --- |
| CoordinatorID | int | PK | Unique identifier for each coordinator |
| ClaimId | int | FK | Unique identifier for each claim |
| Email | string |  | Coordinator’s email address |
| ClaimState | string |  | Check state of claim |
| TotalAmount | double |  | Calculated total for the claim |
| Name | string |  | Coordinator’s full name |

**Academic Manager Table**

| Field | Type | Key | Description |
| --- | --- | --- | --- |
| ManagerID | int | PK | Unique identifier for each academic manager |
| ClaimId | int | FK | Unique identifier for each claim |
| Email | string |  | Academic Manager’s email address |
| ClaimState | string |  | Checks state of claim |
| Name | string |  | Academic Manager’s full name |
| TotalAmount | double |  | Calculated total for the claim |

* **Layout of GUI**
* **LecturerDashboard**

Lecturer dashboard will give the busy buttons (Submit Claim, Upload Document), and a history timeline of previous claims.

Period, module, hours, rate are entered on the Claim form, and an overall calculated amount will be displayed as read-only.

* **Coordinator Programme and Academic Manager Dashboard**

The Coordinator/Manager perspective will provide some filters (Month, Lecturer, State), approvals queue with Approve/Reject buttons and a details panel. Navigation remains the same through a common design.

* **The entire prototype as a whole**

The prototype will be front-end only: buttons and forms will be inert and act as a visual template upon which to base subsequent implementation.

* **Assumptions and constraints**

**Claim assumptions**

* Claims are allowed to be submitted by only contracted lecturers only. Prices are pre-set in per lecturer/module.

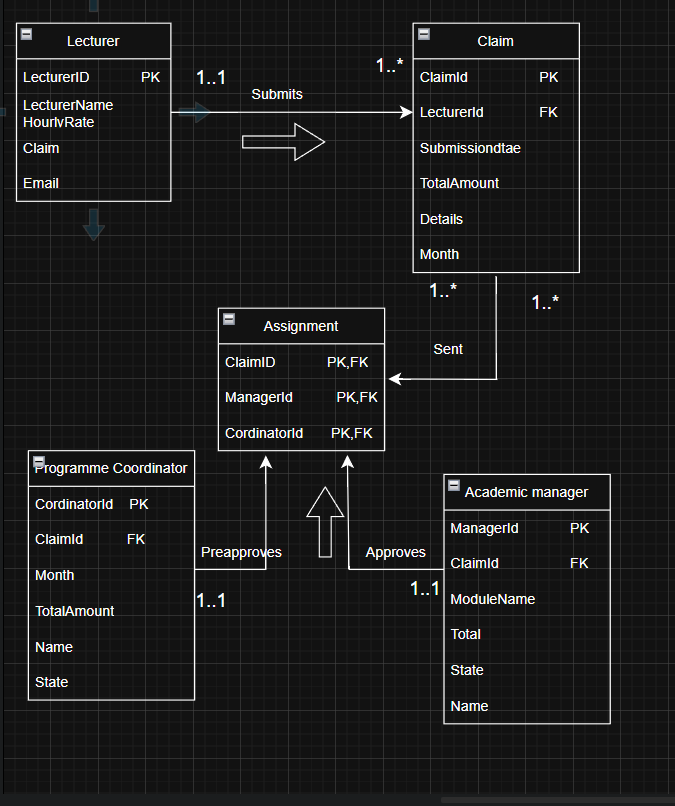
**Coordinator Programme and Academic Manager assumptions**

* States go through Draft - Submitted - Verified and Approved/Rejected - Settled. Time is documented in hours (decimal). Authentication/authorization will be implemented later; in this prototype, we only deal with UI.

**Constraints:**

At this stage system isn’t functional whatsoever

**2.UML Class Diagram**



(Drwawio, n.d.)

**Business Rules:**

* A lecturer can submit 1 or more claim.
* A claim is sent to a programme coordinator and they preapprove the claim.
* A claim is sent to a academic manager and they approve the claims.

**Relationships:**

Lecturer🡪Submits🡪Claim

Claims🡪Sent🡪Programme Coordinator

Claims🡪Sent🡪Academic Manager

Programme Coordinator🡪Preapproves🡪Claim

Academic Manager 🡪Approves🡪Claim

## **3.PROJECT PLAN**

(Lecture), 2025)**Guidelines from Mr.Tshibiti**

Contract Monthly Claim System (CMCS) – Part 1 Report

* **Project Overview**

Project Name: Contract Monthly Claim System   
Project Manager: Tshepang Ramohapi   
Start Date: 01 September 2025  
End Date: 30 November 2025  
Version: 1.0  
Objective: To develop a digital platform that enables lecturers to enter monthly claims, coordinators to pre-approve the claims and academic managers to finalize or reject them. The aim is to gain the removal of manual paperwork, better transparency and safe records keeping.

* **Scope Statement**

**In Scope:**

* User registration and login (Lecturer, Programme Coordinator, Academic Manager).
* Claim submission with hours, modules, and rates.
* Upload of supporting documents (PDF/JPG/PNG ≤10MB).
* Claim tracking with real-time status updates.
* Approval workflow (Submitted → Verified → Approved/Rejected → Finalized).

**Out of Scope:**

* Integration with payroll/finance systems.
* Mobile application (web version only).
* Automated email/SMS notifications (future enhancement).

**Assumptions:**

* Users have stable internet access.
* Hourly rates are preconfigured per lecturer/module.
* Roles and responsibilities (lecturers, coordinators, managers) are clearly defined.

**Constraints:**

* Development limited to 3 months.
* One-person development team.
* Only localhost deployment for prototype stage.

(Lecture), 2025)

* **Team Roles and Responsibilities(one man project)**
* Tshepang Ramohapi(ST10440580) – Team Lead & Backend Developer (manages system workflow logic in C# and database connections).Frontend Developer & Designer (responsible for user interface design, HTML/CSS, and responsiveness).Database & QA Specialist (designs database schema, ensures data integrity, and conducts testing/QA). (Lecture), 2025)
* **Timeline Table**

|  |  |  |
| --- | --- | --- |
| **WEEK** | **TASK** | **Milestone and Deliverable** |
| **1** | Project Initiation | 05 September 2025  Scope and role approvals |
| **2** | Design Completion | 15 September 2025  Completion of mock-ups, database schema and navigation structure. |
| **3-6** | Development Complete | 15 October 2025  Completion of frontend and backend functionality. |
| **7** | Testing and QA | 01 November 2025  System bug, usability and workflow validation |
| **8-9** | Deployment | 30 November 2025  Project developed and pushed to localhost/production environment. |

(Lecture), 2025)

* **Development Phases**

Requirements Gathering:

* Interview lecturers, coordinators and program managers.
* Document system specifications (position, claim workflow, status changes).

Design:

* Design database schema that includes (Users, Claims, Documents).
* Design GUI for Login, Register, Submit Claim, Track Claim, Approve/Reject pages).

Development:

* MVC structure in ASP.NET Core will be implemented as well as
* developing role-based views and claim workflows after all these
* Styling will be added(CSS).

Testing:

* Perform Unit tests for submitting lecture claims as well as the logic for approving and preapproving requests.
* Manually test the system as a whole by ensuring its user friendly and well structured.

Deployment:

* System will be deployed to localhost (demo).
* System will possibly be deployed to a hosted server.

(Lecture), 2025)

* **Tools and Technologies**
* In the **front end** part of the system HTML5, and C# will be implemented.
* In the **back end** part of the system ASP.NET Core (C# MVC) will be implemented.
* In the database part of the system MySQL (relational, with referential integrity) will be implemented.
* Tools that will be implemented to help develop the CMS are Visual Studio, GitHub (for version control), MS Project (for planning).

(Lecture), 2025)

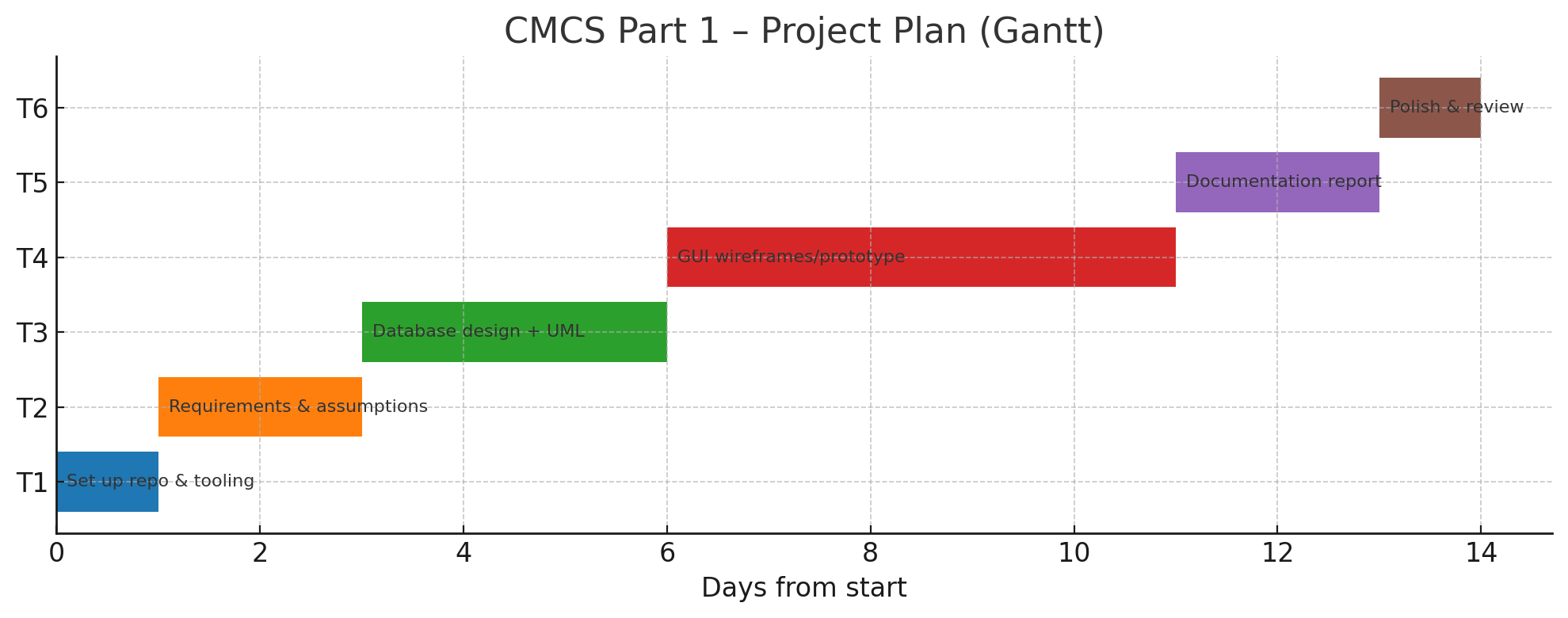
* **Communication Plan**
* How I will ensure system meets requirements: Consult with lecturer, Whatsapp group messagesr to update on daily activities, Github commits to track the progress.
* Client: Lecturer-Checks if everything system meets requirements.

**Risk Management**

| **Risk** | **Impact** | **Probability** | **Mitigation** |
| --- | --- | --- | --- |
| Slow development as a result of workload | High | Medium | Make a tight weekly plan and adhere to project plan |
| Loss or corruption of data | High | Low | Daily backup or Save file on different hard drive |
| Misinterpretation of needs | Medium | Medium | Arrange schedule feedback meetings with client |
| Security issues with user login | High | Medium | Deploy password hashing and role-based access in subsequent step. |

(employee, 2025)

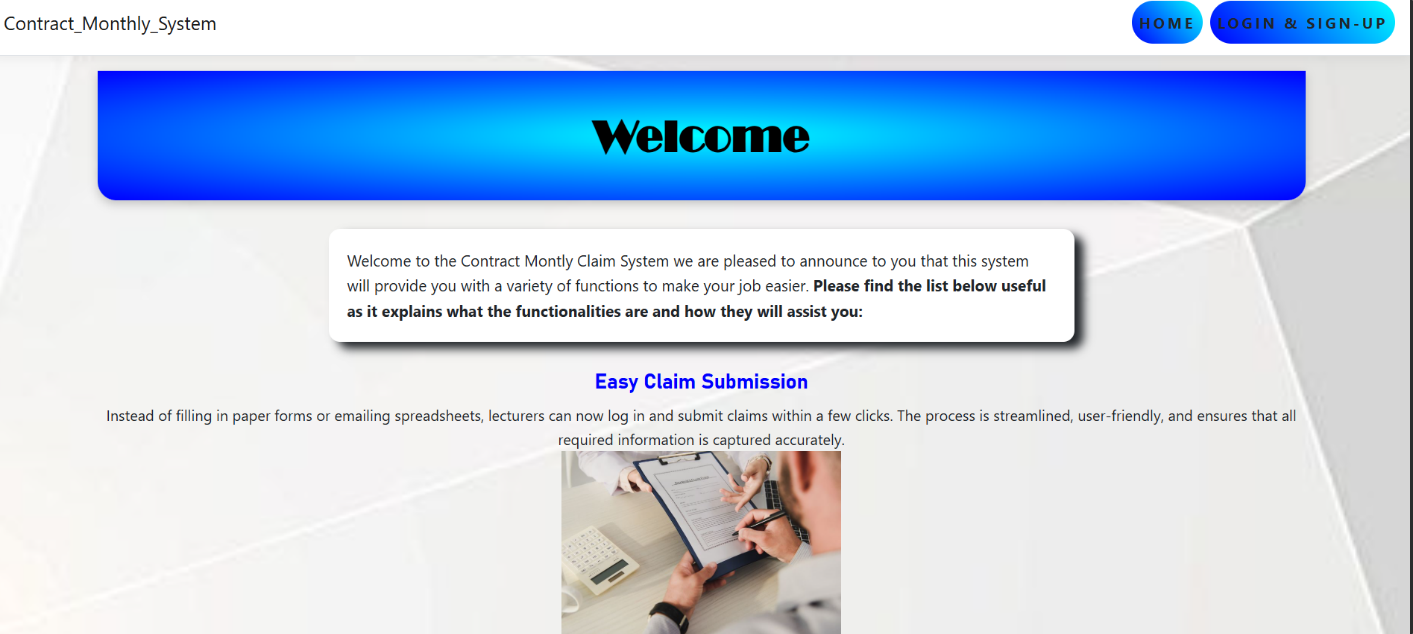
## **Gnatt Chart**



**\*Used excel\***

## **4.Screenshots of prototype**

**Home page**



**Explanation**: Homepage with pictures and explanation of the features that the system provides

**Registration page**



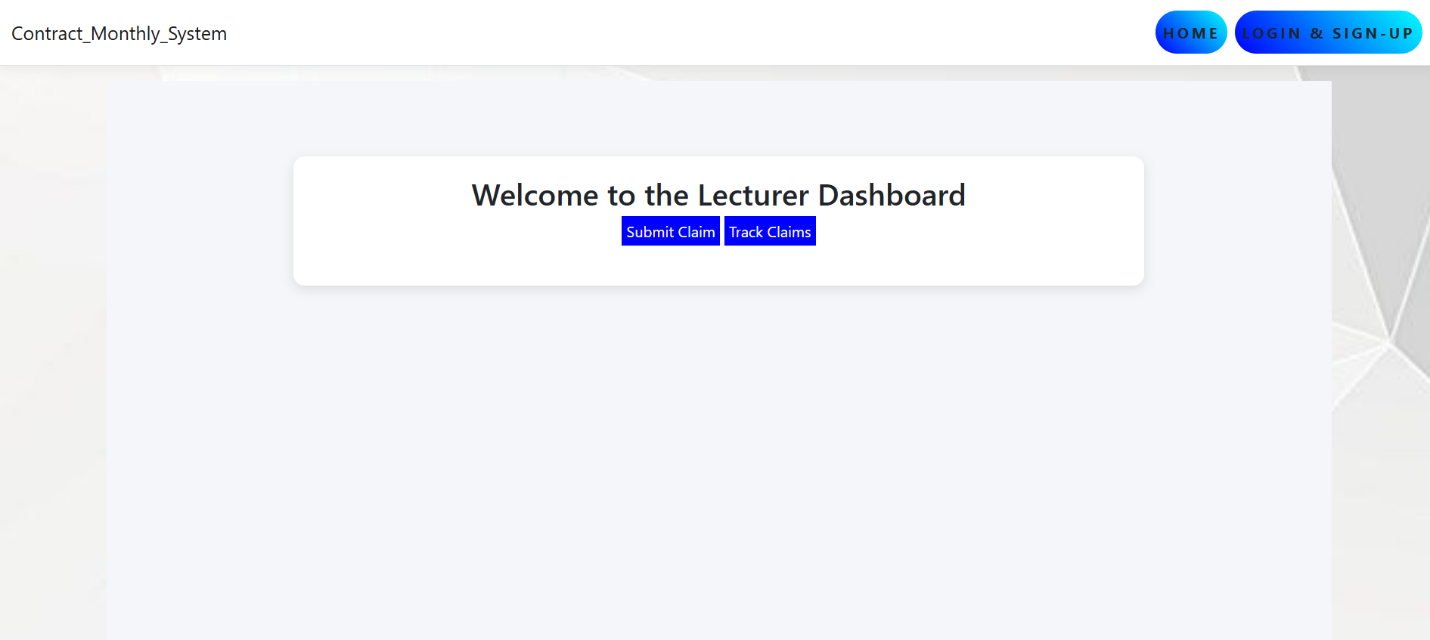
**Explanation:** Once user clicks on Login and Sign Up they will be taken straight to the registration webpage. Where they will fill in their details and hopefully successfully create an account.

**Login page**

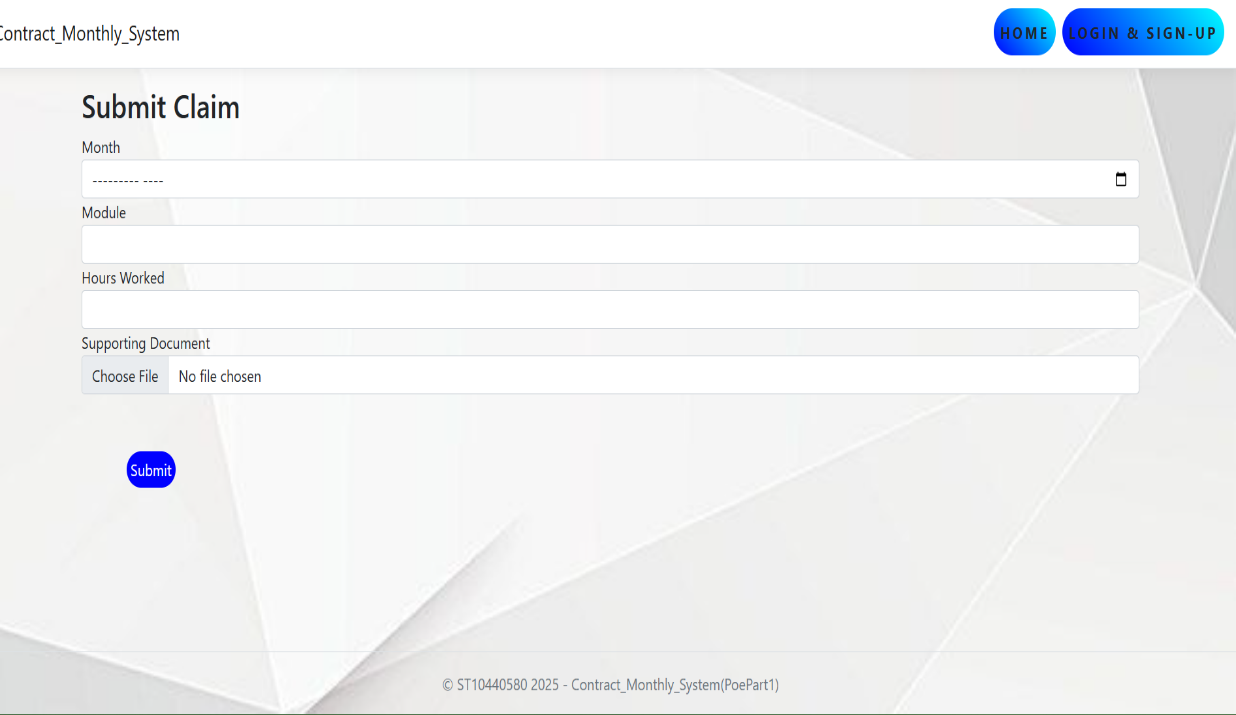
****

**Explanation:** Once a user has created an account or if they click on already have an account they will be taken straight to the Login webpage. This is where user will fill in their existing details and pick a role.

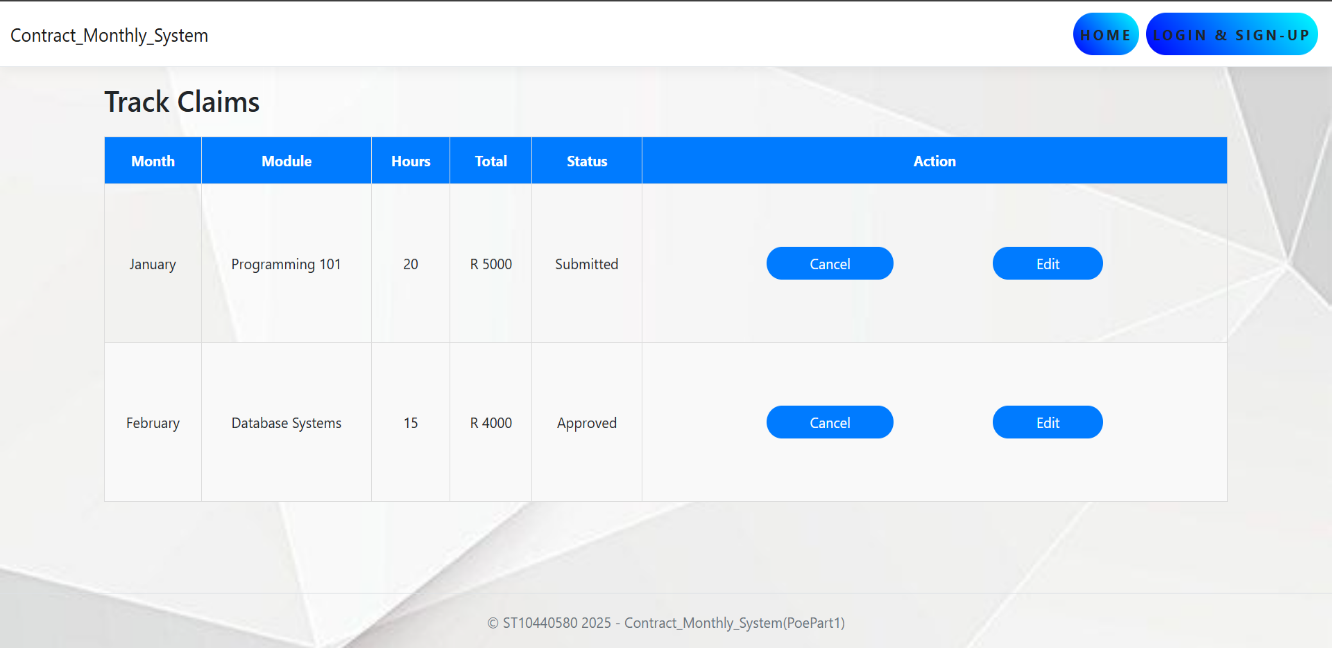
**Lecturer Dashboard**

****

**Explanation:** Once user says their role is a Lecturer then they will be redirected to the lecturer dashboard, where there are two buttons for submitting claims and tracking claims

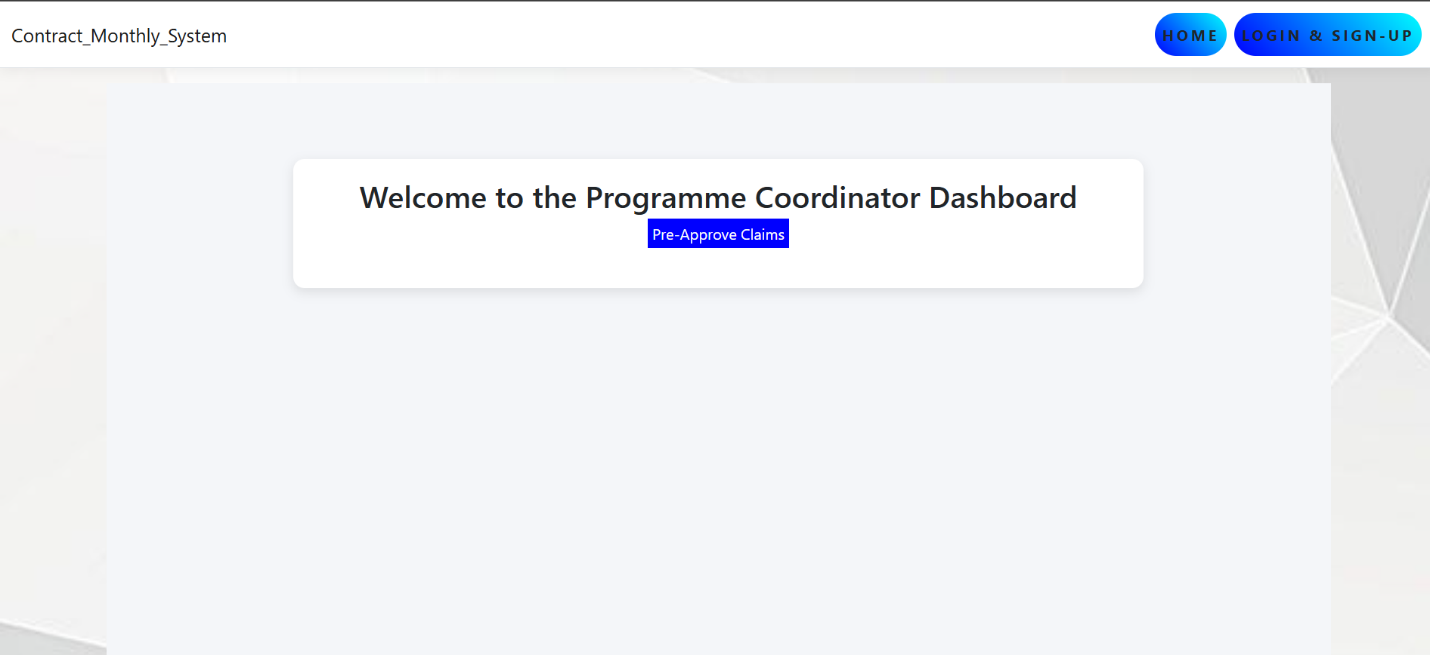
****

**Explanation:** Once submit claim button is clicked you will be redirected to the webpage, where you will include details about your claim

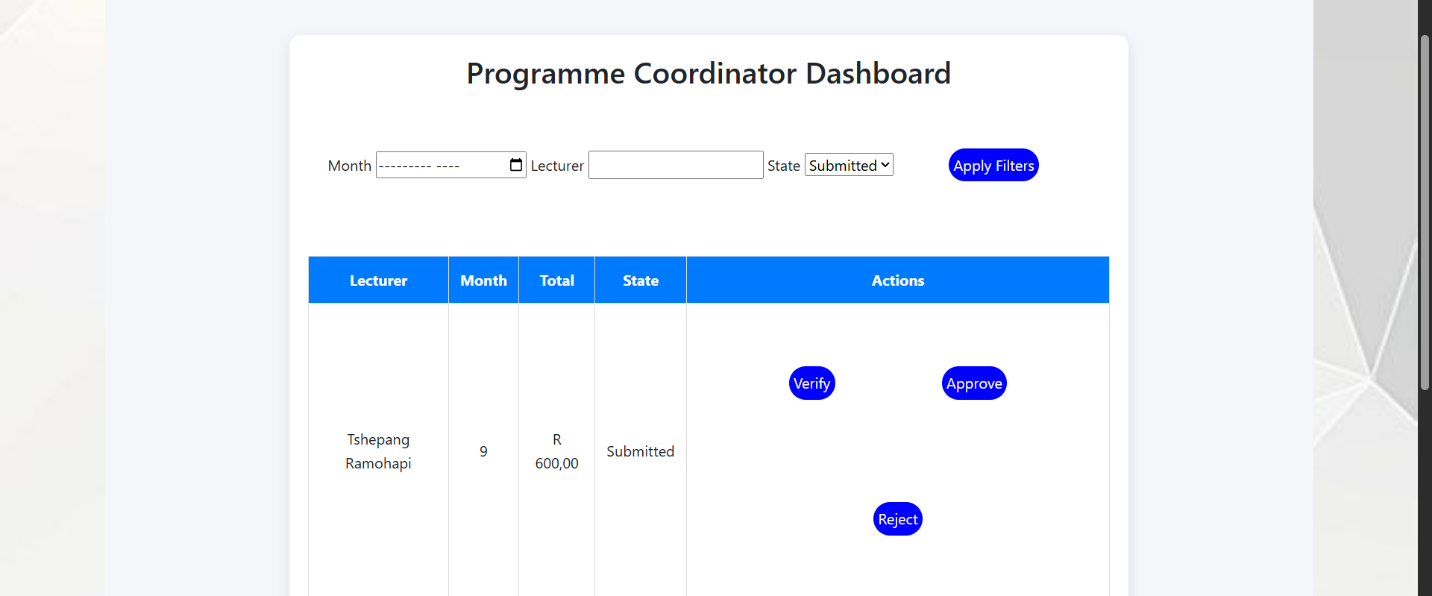
****

**Explanation:** Once user clicks on Track Claims they will be redirected to this webpage where they will see the status of their claims. The user will also be allowed to edit their claim if a mistake has been made.

**Coordinator Dashboard**

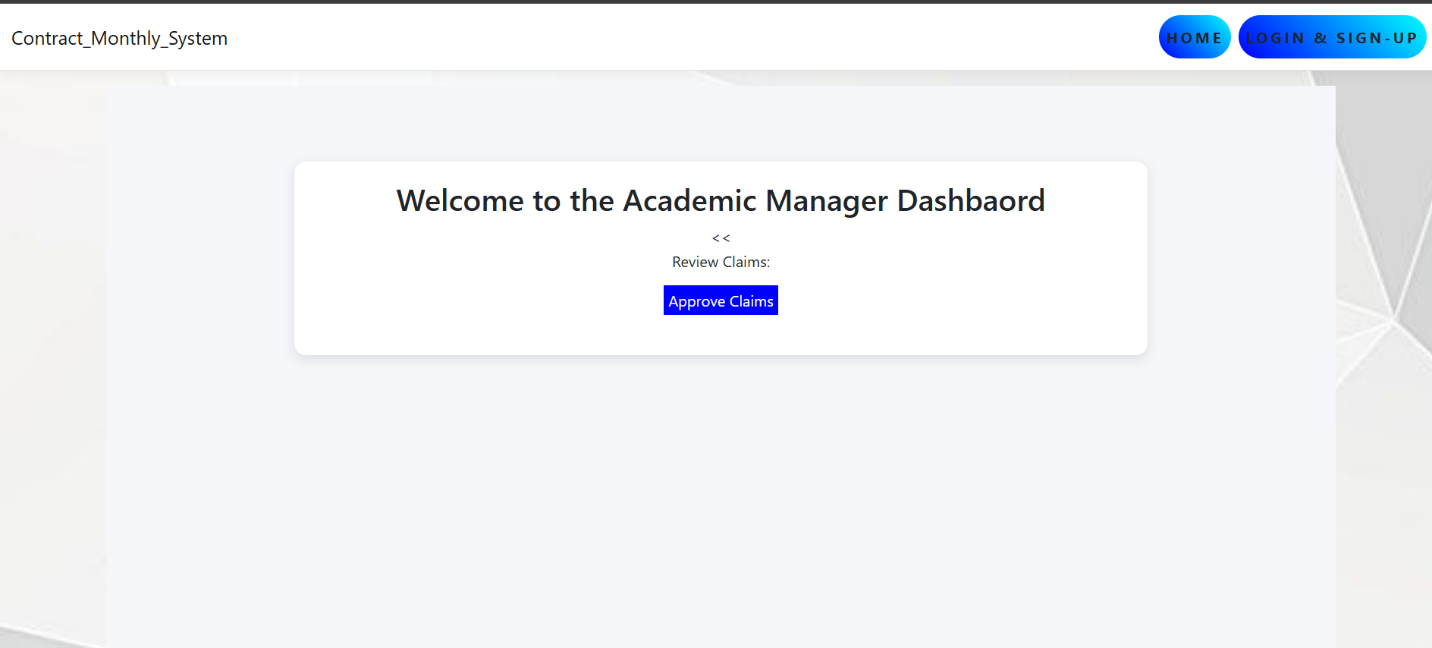
****

**Explanation:** Once user chooses Programme Coordinator as their role they will be redirected to the Programme Coordinator Dashboard. User will also see a pre approve claim button as it is their duty to pre approve lecturer claims.

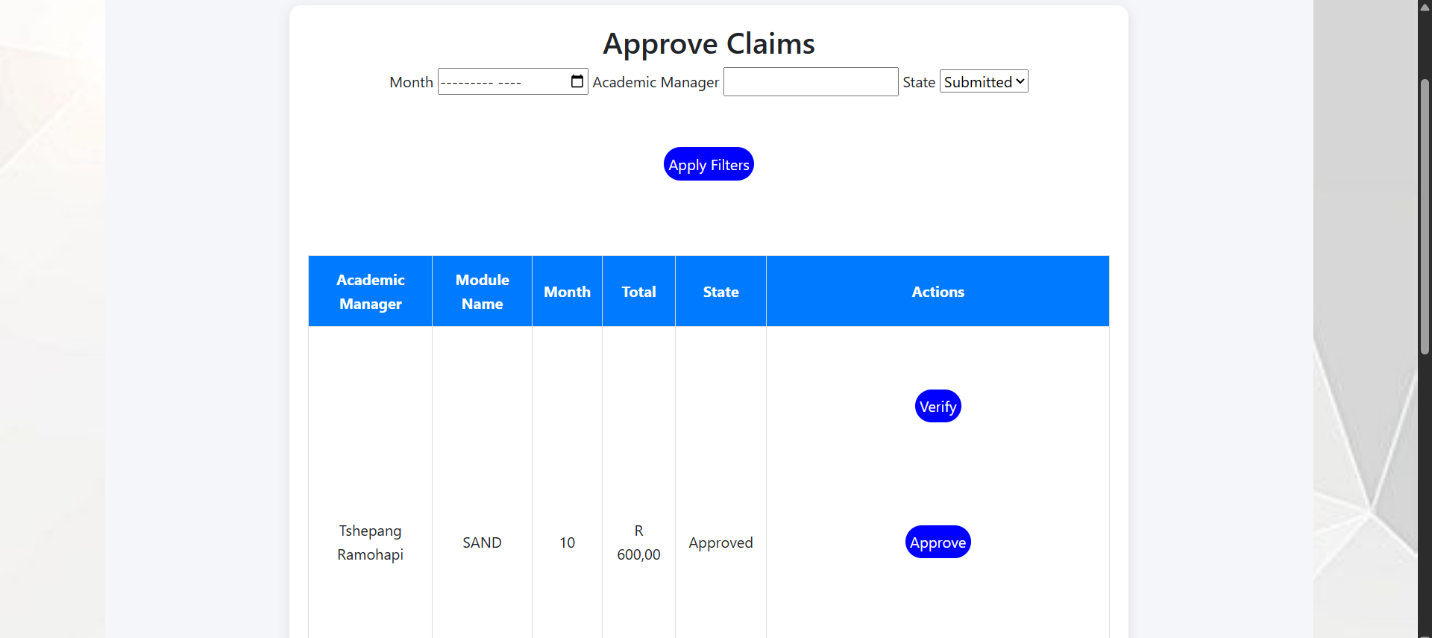


**Explanation**: Once user clicks on the Pre-Approve Claims they will be directed to this webpage, where they will see all the claims made by different lecturers.

**Academic manager dashboard**



**Explanation**: Once user picks Academic manager as their role they will be redirected to the Academic dashboard where they will see a button called Approve claims.



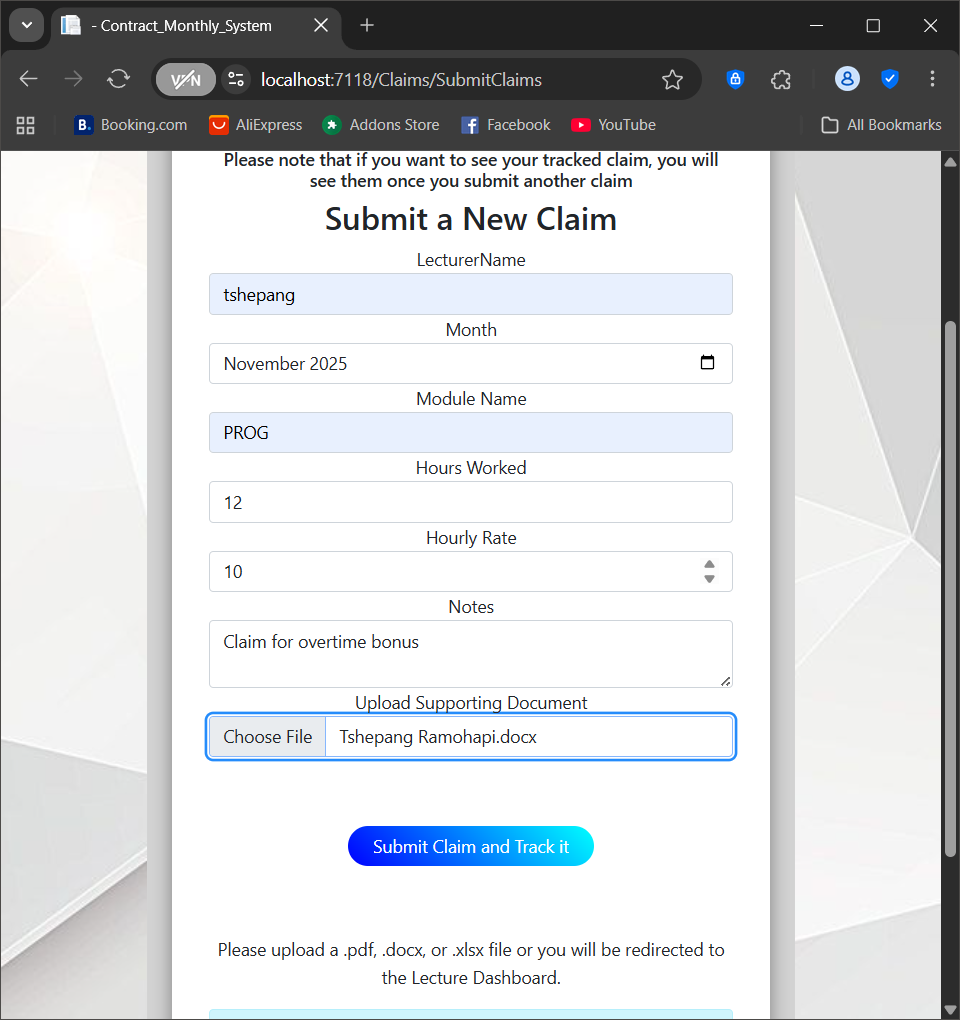
**Explanation**: Once user clicks on Approve Claims they will be redirected to this webpage where they can verify if all the approved claims are valid.

**Part 2**

**Lecture Dashboard**

* **Submitting claims:**

Lecturer can submit claims by simply filling in form at hand, once submitted the system calculates total being the hourly rate \* the hours worked. The claim is also stored in the database as it links it to the lecturer id.



* **Uploading document:**

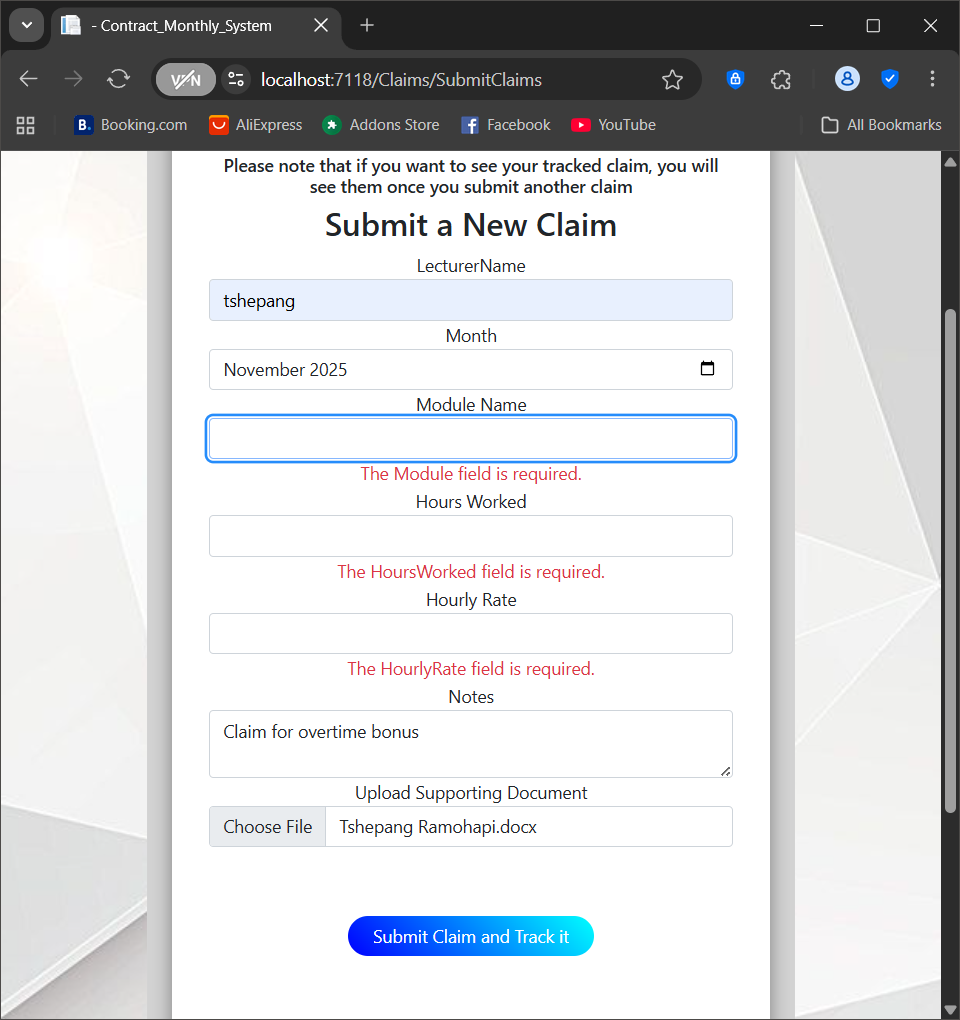
Lecturers can also upload a supporting document, the document must be no more than 5mb.The document is also saved securely in the wwwroot/uploads folder within the programs folder.

A screenshot of a computer

AI-generated content may be incorrect.

* **Error Handling:**

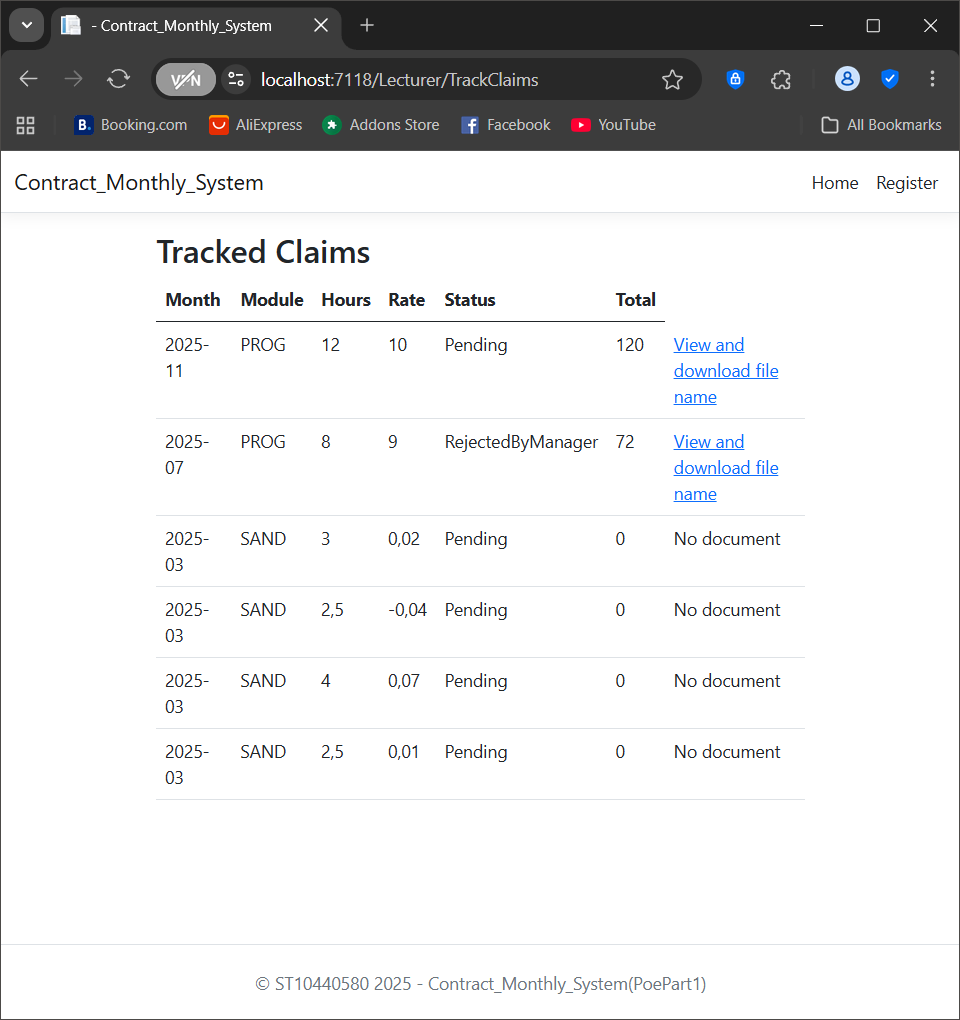
Missing fields trigger validation messages to keep the app stable and user friendly.



* **Track claim:**

Claims are filtered by their session based on lectured.

Also each claim shows the module, month, hours ,total and the status of the claim

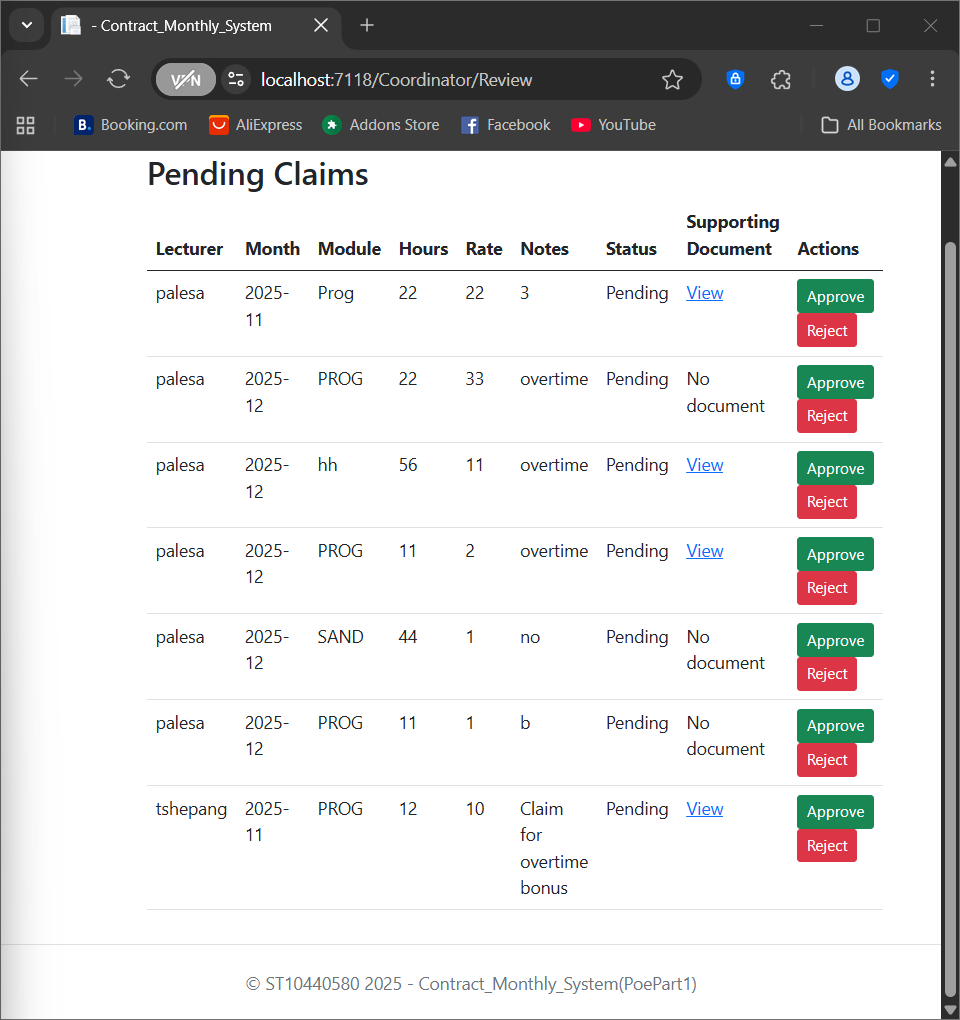


**Programme Coordinator Dashboard**

* **PreApprove Claims :**

The programme coordinator can view all pending claims submitted by the lecture.

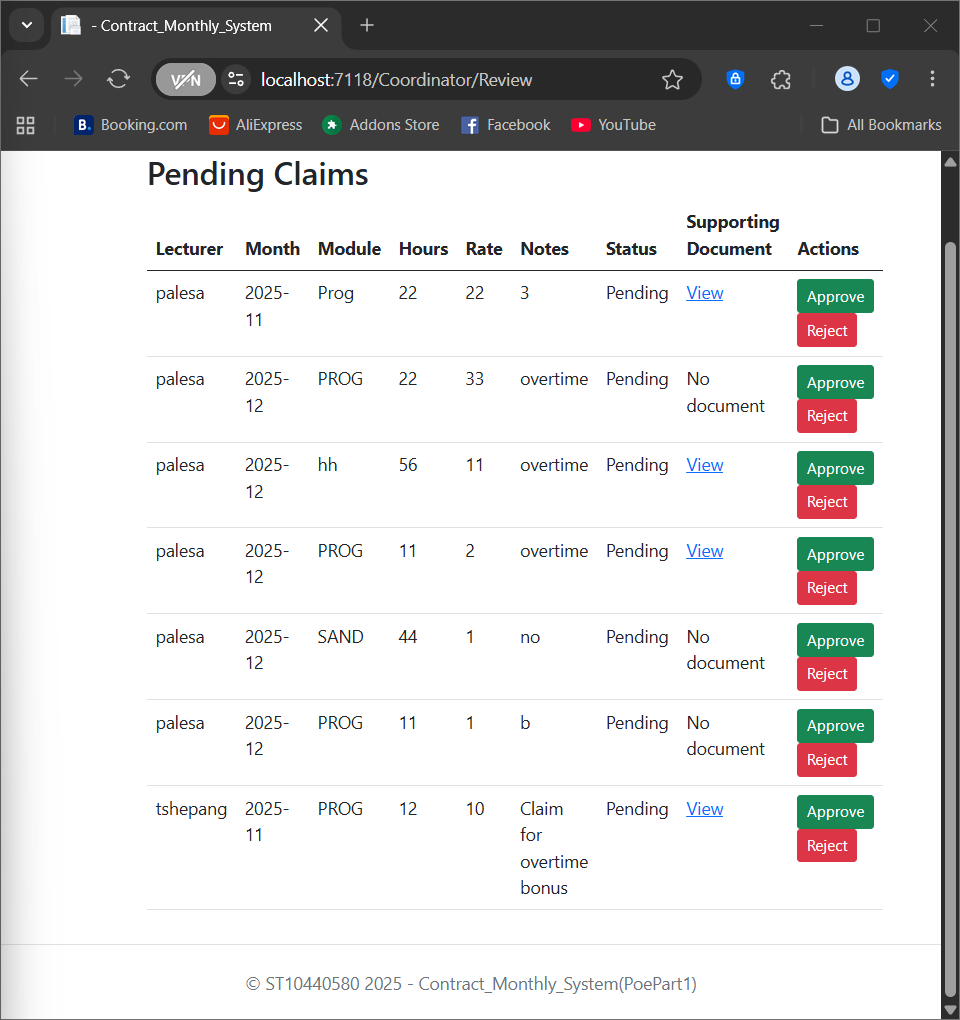
In this case they can approve or reject a claim.



* **Viewable Document:**

The programme coordinator can also choose to view the document submitted by the lecture.

Once they view they will see the filename of the document as its download to them.



**Academic Manager Dashboard**

* **Approve Claims and Reject Claims:**

Academic Manager is at the top of the hierarchy as they are able to see all claims that were approved and rejected by the programme coordinator.

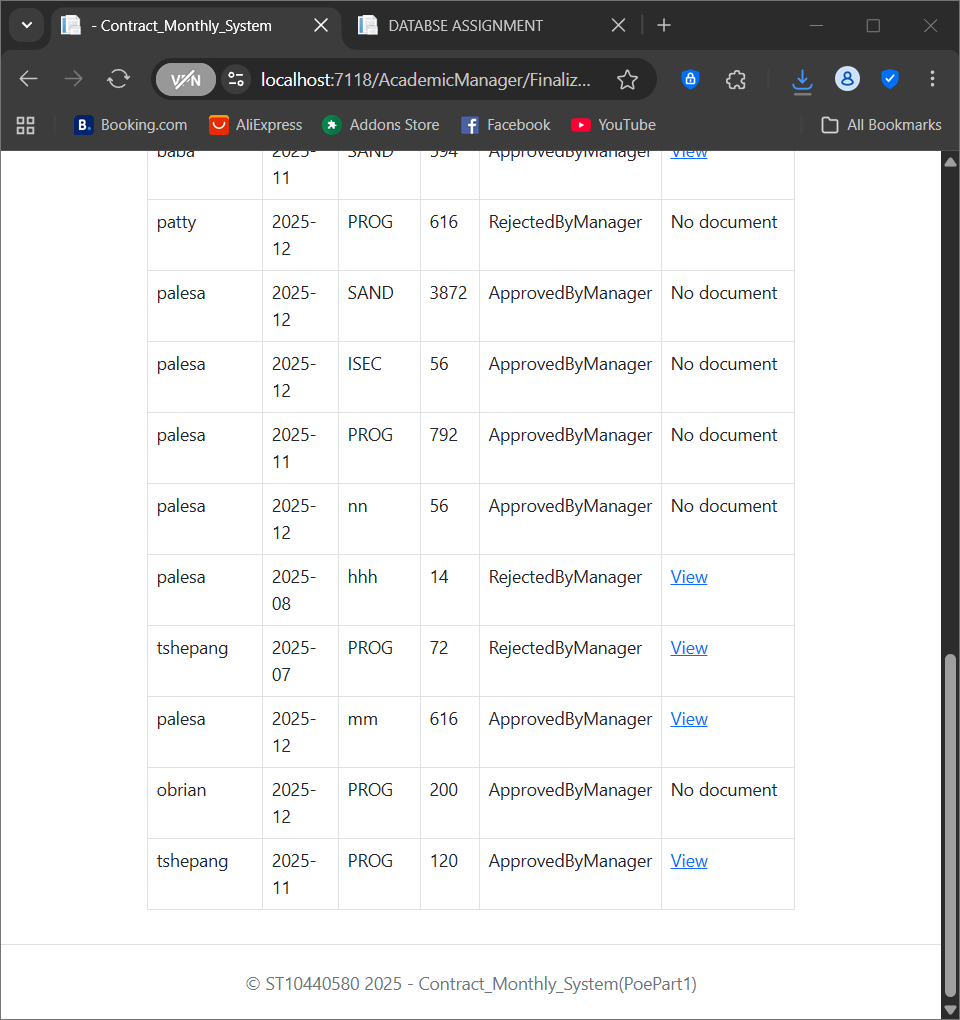
They make the final decisions and the decisions are stored in the Finalized document view.

A screenshot of a computer

AI-generated content may be incorrect.

* **Finalized document:**

The claim becomes read only as final decision has been made.



**GitHub Link**: <https://github.com/ST10440580/Contract_Monthly_System>

# 

# **Reference List**

Drwawio, n.d. *Drawio.* [Online]   
Available at: https://app.diagrams.net/?src=about  
[Accessed 2 September 2025].

employee, Q., 2025. *Role Based Access Control Implementation: From Planning to Deployment,* s.l.: QOHASH.

Lecture), M. C., 2025. *Software Development Project Plan.* [Online]   
Available at: https://mystudies.iie.edu.za/content/enforced/49239-PROG6212\_2025\_FT\_DISD0601\_RCGPBR\_Term2\_GR02/Software\_Development\_Project\_Plan\_Presentation.pptx  
[Accessed 1 September 2025].