# **SINDISWA NOMAKHOLWA MADLIWA**

# **ST10240725**

# **PROGRAMMING 2B**

# **POE PART 1**

# **DUE: 10 SEPT 2024**

GIT HUB LINK:

<https://github.com/IIEMSA/poe-part-1-SindiswaN.git>

# **DOCUMENTATION**

**Design choices:**

**Separation of Concerns**: The project is structured to separate different concerns, such as handling user interactions, managing claims, and defining data. Each class has a clear responsibility:

* **LecturerForm**: Manages user interactions for lecturers submitting claims.
* **MainWindow**: Acts as the entry point and routes users to different parts of the system.
* **CoordinatorView**: Allows the coordinator to review, approve, or reject claims.
* **ViewClaims**: Displays a list of claims that have been submitted.
* **Claim**: Represents the data model for a claim.
* **ClaimsManager**: Manages the lifecycle of the claims in the system, such as adding new claims.
* **Modularity and Extensibility**: The design allows for future expansion. Each part of the system (lecturer interaction, claim viewing, coordinator approval) is encapsulated in its own class. If future requirements emerge, such as adding additional roles or complex features, they can be incorporated without rewriting the entire system.
* **Static Management of Claims**: Claims are stored in a static list within the ClaimsManager or LecturerForm. This simplifies the flow and allows different parts of the system to access claims without the need for complex database interactions at this stage. However, this structure can be extended to use a proper database in the future.

**Database structure:**

**Claims Table**:

* **LecturerName (string)**: The first name of the lecturer.
* **LecturerSurname (string)**: The surname of the lecturer.
* **SubmissionDate (DateTime)**: The date the claim was submitted.
* **HoursWorked (double)**: The total hours worked.
* **HourlyRate (double)**: The hourly rate of the lecturer.
* **ClaimAmount (double)**: Calculated as HoursWorked \* HourlyRate.
* **Status (string)**: The status of the claim (e.g., Pending, Approved, Rejected).

**GUI Layout:**

**1. MainWindow (Menu)**

* The main window presents three options for the user:
  1. Submit Claim
  2. View Claims
  3. Coordinator Approval
* It serves as the central navigation point, and the button placement ensures accessibility.

2. **LecturerForm (Submit Claim)**

* The lecturer enters their name, surname, date, hours worked, and hourly rate.
* The form includes fields for supporting documents, although the file upload logic is minimal for now.
* Upon clicking "Submit," the claim details are validated and calculated, and a confirmation message is displayed.
* The layout focuses on ease of use, with all input fields clearly labelled and easy to locate.

**3.** **ViewClaims**

* A simple grid layout displays claims in a table format. This makes it easy to compare claims at a glance.
* Columns include essential information like Lecturer Name, Surname, Submission Date, Hours Worked, Hourly Rate, Claim Amount, and Status.

**4. CoordinatorView**

* Similar to the ViewClaims window, but with additional buttons for approving or rejecting claims.
* The list of claims is displayed, and the coordinator can click to approve or reject individual claims.

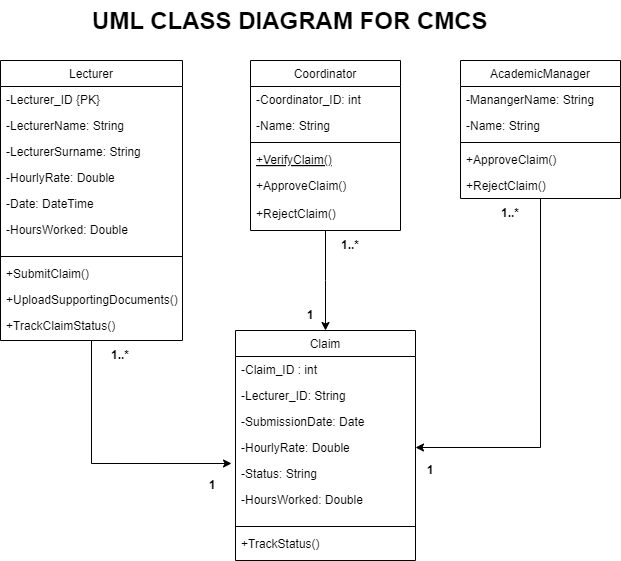
**Assumptions and constrains  
  
Assumptions**:

* Claims are entered by lecturers who will correctly input their data (e.g., valid names, correct hours worked).
* Coordinators have full authority to approve or reject claims.
* The lecturer will only submit claims once per month, reducing the likelihood of duplicate claims.
* Lecturers have a fixed hourly rate.

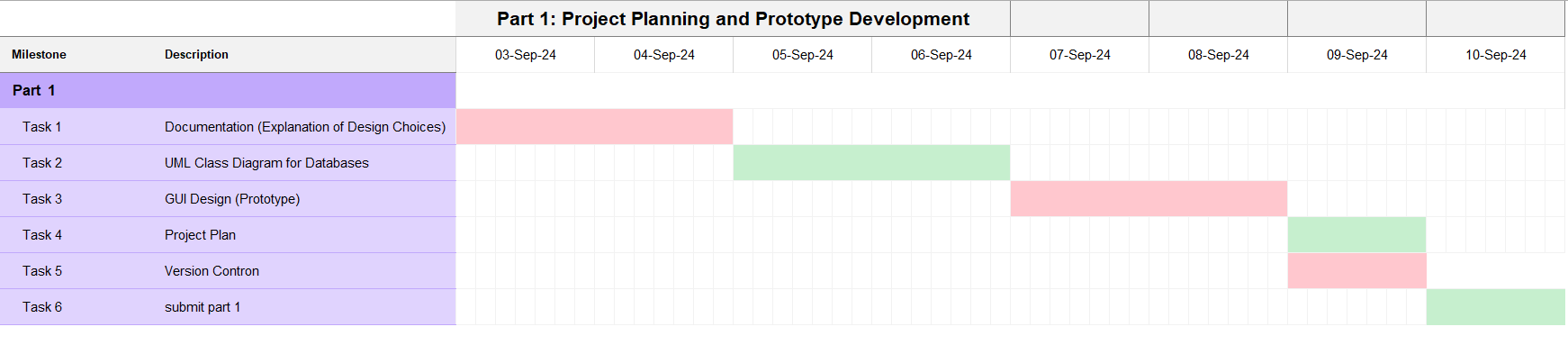
**Constraints**:

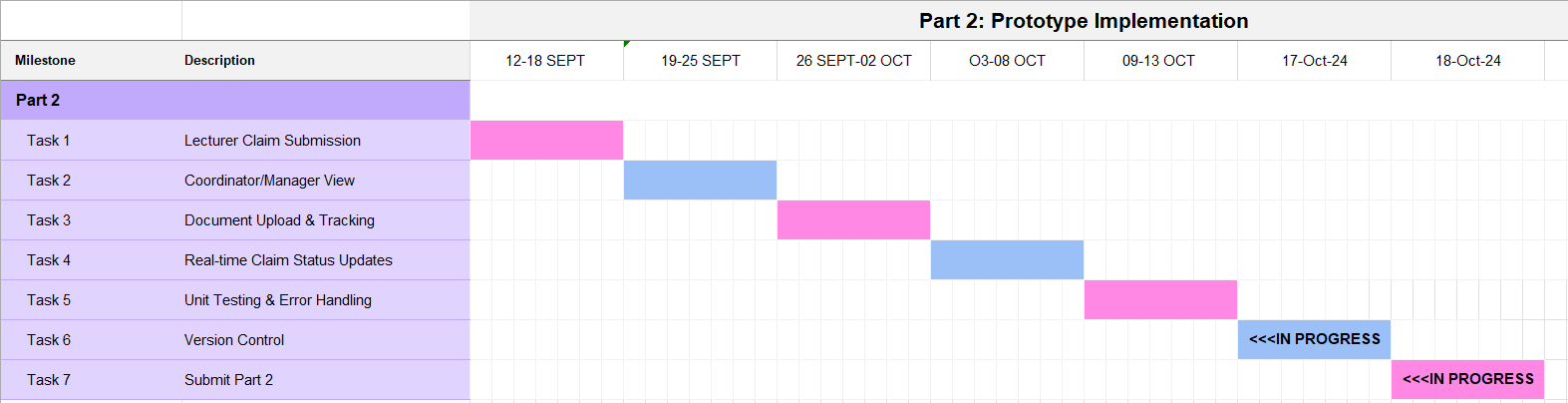
* **Validation**: Currently, validation is minimal (e.g., ensuring non-empty fields). Further checks (like data type validation, ensuring positive values for hours and rate) could be added.
* **Static Data Storage**: At this stage, claims are stored in memory and will be lost when the application closes. Moving to persistent storage (e.g., a database) would be a necessary future improvement.
* **File Upload**: The file upload functionality is basic, with no actual file handling implemented yet.
* **Role-Based Access**: In the current structure, the coordinator view and lecturer submission are separate without authentication mechanisms to control access. Introducing a login system would address this.

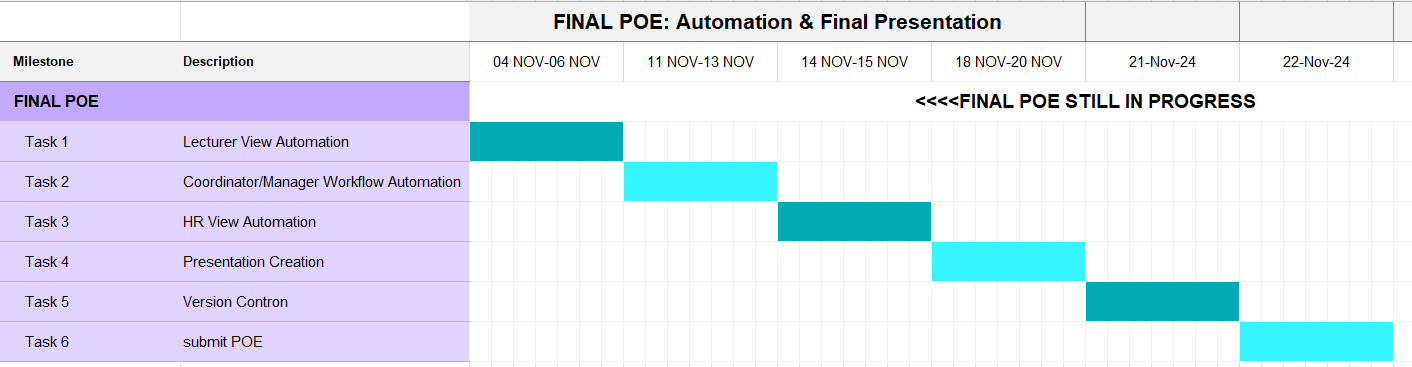
# **UML CLASS DIAGRAM FOR CMCS**

****

# **PROJECT PLAN**

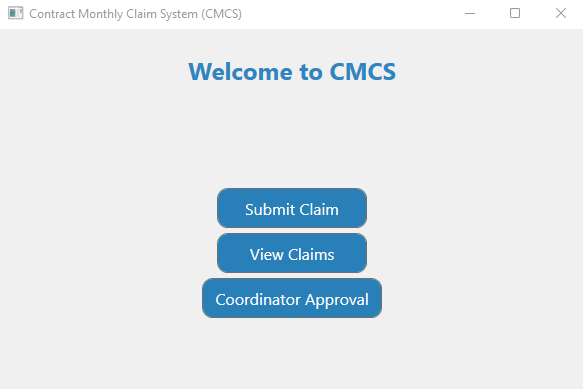
**part 1: project plan**

**Part 2: project plan**

**Part 3: project plan**

# **GUI DESIGN**

1. **MainWindow:** The main navigation hub for the application, allowing users to access different functionalities.

****

1. A screenshot of a computer

   Description automatically generated**LecturerForm:** This page is used by lecturers to submit their monthly claims.
2. **View Claims:** This page is for viewing all submitted claims, regardless of their status (pending, approved, or rejected).

**A screenshot of a computer

Description automatically generated**

1. **Coordinator View:** This page is used by coordinators to review and approve or reject submitted claims from lecturers.

A screenshot of a computer

Description automatically generated