St10317969 kgotatso phetoe

Prog6212

Part 1

User Interface (UI) Design:

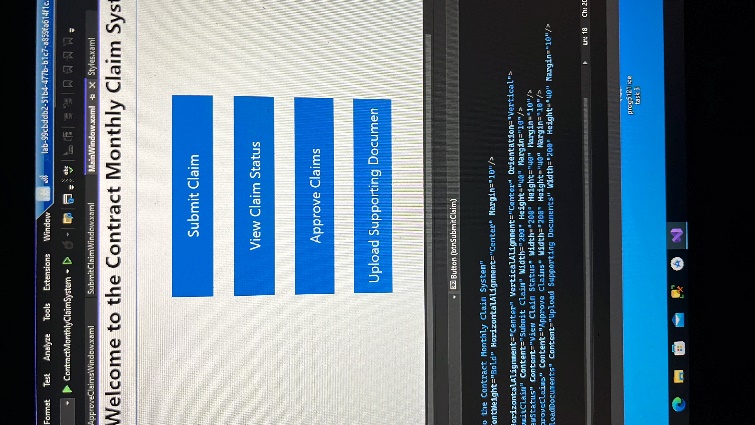
- I designed the ui to be clean and straightforward, minimizing the learning curve for users. Buttons are clearly labeled, and the layout is consistent across different windows, making navigation intuitive.

- Role-Based Navigation: The main window (dashboard) presents options based on the user's role. This approach streamlines the workflow by only showing relevant features, reducing the chance of user error.

- Feedback Mechanisms: Message boxes are used to provide immediate feedback to users for actions like submitting a claim or uploading documents. This ensures users know the status of their interactions with the system.

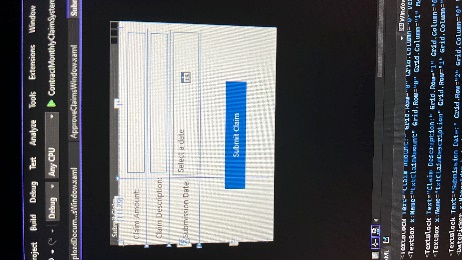
Window Layout

- Main Window:



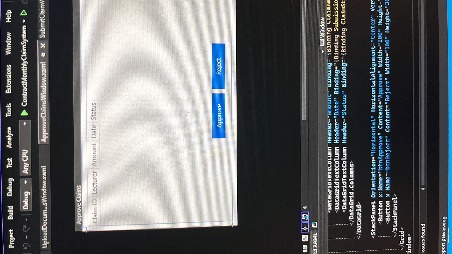
Acts as the central hub with buttons leading to different functional areas: submitting claims, viewing status, approving claims, and uploading documents. This centralization of options makes it easy for users to easily find and perform their tasks

- Claim Submission Window:



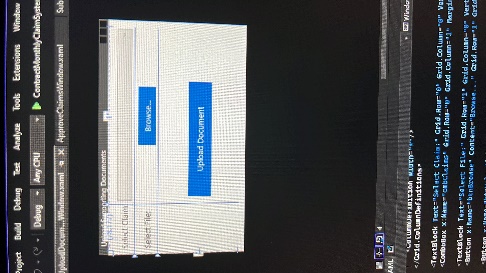
Includes fields for claim amount, description, and submission date. These fields are essential for capturing the basic details of a claim.

- Claim Approval Window:



Displays a dataGrid for showing claim details, which allows approvers to view, select, and approve or reject claims. The DataGrid is ideal for displaying tabular data, making it easy to compare multiple claims.

- Upload Documents Window:



Provides a simple interface for uploading files related to claims. The use of a ComboBox and file dialog makes selecting a claim and uploading documents straightforward.

Database Structure

I designed the database structure to support the core functionalities of the Contract Monthly Claim System. It includes tables for storing information about users (lecturers, coordinators, managers), claims, claim statuses, and uploaded documents.

- User Table:

- Stores user information such as `UserID`, `Name`, `Role`, and `Email`.

- Roles include Lecturer, Programme Coordinator, and Academic Manager.

Claim Table:

- Stores details about each claim, such as `ClaimID`, `UserID` (foreign key referencing `User`), `ClaimAmount`, `Description`, and `SubmissionDate`.

- A `Status` field stores the current status of the claim (e.g., Pending, Approved, Rejected).

Document Table:

- Stores information about uploaded documents with fields like `DocumentID`, `ClaimID` (foreign key referencing `Claim`), `FilePath`, and `UploadDate`.

ClaimStatus Table:

- Tracks changes in the status of claims over time, with fields like `StatusID`, `ClaimID`, `Status`, and `DateChanged`.

More about the ui for my part 1

- The system has three primary user roles: Lecturer, Programme Coordinator, and Academic Manager.

- Each claim is submitted by one lecturer and can be approved or rejected by a Programme Coordinator or an Academic Manager.

- Documents uploaded are assumed to be in PDF format for consistency.

- The prototype is not functional; therefore, no backend processing, data validation, or database connectivity is implemented.

- The gui is designed with basic navigation and interaction in mind, suitable for demonstration purposes but not for actual deployment.

UML Diagram

Classes

User:

- Attributes: `UserID`, `Name`, `Role`, `Email`

- Relationships: One-to-Many with Claim

Claim

- Attributes: `ClaimID`, `UserID`, `ClaimAmount`, `Description`, `SubmissionDate`, `Status`

- Relationships: One-to-Many with Document and One-to-Many with ClaimStatus

Document

- Attributes: `DocumentID`, `ClaimID`, `FilePath`, `UploadDate`

- Relationships: Many-to-One with Claim

ClaimStatus

- Attributes: `StatusID`, `ClaimID`, `Status`, `DateChanged`

- Relationships: Many-to-One with Claim

