Programming Part 1

POE 2025- TERM 2

mR.mOHALE

9 sEPTEMBER 2025

Armalene Tamryn Magerman

Table of Contents

[PART 1- Project Planning and Prototype Development 2](#_Toc208342707)

[1. Documentation 2](#_Toc208342708)

[2. UML Class Diagram for Databases 4](#_Toc208342709)

[3. Project Plan 5](#_Toc208342710)

[4. GUI Design- WPF 6](#_Toc208342711)

# PART 1- Project Planning and Prototype Development

**Contract Monthly Claim System – Prototype Design Report**

## 1. Documentation

The Contract Monthly Claim System is a desktop application designed to help lecturers submit their monthly claims for hours worked and allow Programme Coordinators and Academic Managers to review, verify, and approve these claims. This system also allows lecturers to upload supporting documents, such as timesheets or attendance records, and track the progress of their claims until they are approved and settled.

For the GUI, I have chosen to use Windows Presentation Foundation (WPF) with .NET Core, as it is well-suited for building modern, responsive, and visually clean desktop applications. WPF uses XAML (Extensible Application Markup Language) to design user interfaces and supports data binding, styles, and templates, making it easier to keep the UI consistent and maintainable. The application is structured using the MVVM (Model-View-ViewModel) design pattern, which helps to separate the business logic from the presentation layer. This is a widely-used approach in WPF applications because it improves code readability, reusability, and testing.

In this prototype stage, the focus is on front-end design only. No database connections or business logic are included yet. All screens are static and used to demonstrate what the final application will look like. For now, buttons and inputs are non-functional.

The system includes three types of users:

* Lecturers, who can submit monthly claims and upload documents.
* Programme Coordinators, who can review submitted claims and either approve or reject them.
* Academic Managers, who give final approval to verified claims.

The assumptions we made in designing the system are as follows:

* Each lecturer is allowed to submit one claim per module per month.
* Supporting documents are required for every claim.
* A claim must be approved first by a Programme Coordinator, and then by an Academic Manager.
* Users log in with their email and password, and their role determines what dashboard they see.
* Claims move through four statuses: Pending, Verified by Coordinator, Approved by Manager, and Rejected.

The GUI was designed with simplicity and usability in mind. For example, lecturers see a dashboard with buttons to "Submit New Claim" and "Upload Supporting Documents", along with a table showing the status of past claims. Coordinators and managers see a list of pending claims and can click buttons to approve or reject them. Color coding and labels are used to improve clarity (e.g., green for approved, red for rejected).

We are not implementing any back-end logic in this phase. All forms, buttons, and data tables are purely for display and layout purposes. This will change in the next part of the project, where we will connect the GUI to a real database and add the necessary logic for submitting and approving claims.

## 2. UML Class Diagram for Databases

Lecturer

ManagerID: int  
Name: string  
Email: string

AcademicManager

CoordinatorID: int  
Name: string  
Email: string  
ProgrammeID: int

ProgrammeCoordinator

ProgrammeID: int  
ProgrammeName: str

Programme

DocumentID: int  
ClaimID: int  
FileName: string  
FileType: string  
UploadDate: date

SupportingDocument

0.\* has

ClaimID: int  
LecturerID: int  
SubmissionDate: date  
Amount: decimal  
Status: string

Claim

1.\* Submits

LecturerID: int  
Name: string  
Email: string  
ProgrammeID: int

## 3. Project Plan

The development of the Contract Monthly Claim System prototype is planned over 15 days, divided into several key tasks. The project begins with 2 days of requirement gathering to define the system’s users, features, and structure. Then, 2 days are allocated for designing the UML class diagram, outlining how data such as users, claims, and documents will be stored and related.

Following that, we spend 3 days creating wireframes and designing the GUI layout using WPF in .NET Core. WPF is chosen for its powerful tools for building modern, desktop-based graphical interfaces. We then use 1 day to set up the WPF project structure, organizing the solution into folders like Models, Views, and ViewModels to follow the MVVM design pattern.

The WPF front-end prototype is built over the next 5 days, with XAML pages for lecturers, coordinators, and managers. These pages are non-functional at this stage but show how users will interact with the system. The final 2 days are dedicated to documentation, writing the report and preparing diagrams and screenshots. Throughout the process, GitHub is used for version control, with at least five meaningful commits to track progress and maintain a clean development history.

## A screenshot of a computer screen AI-generated content may be incorrect.4. GUI Design- WPF

A screenshot of a computer

AI-generated content may be incorrect.

## A screenshot of a computer dashboard