PROGRAMMING 2B

PROG POE PART 1

Aaliyah Allie ST10212542

2024

Contents

[Introduction 2](#_Toc176614767)

[1.Documentation 3](#_Toc176614768)

[Design Choices Explanation: 3](#_Toc176614769)

[Design Background: 3](#_Toc176614770)

[Database Structure: 3](#_Toc176614771)

[GUI Layout: 5](#_Toc176614772)

[Assumptions for programs design: 6](#_Toc176614773)

[Constraints for program design: 6](#_Toc176614774)

[2. UML Class Diagram 7](#_Toc176614775)

[3.Project Plan 8](#_Toc176614776)

[Tasks Breakdown: 9](#_Toc176614777)

[Part 1 POE 9](#_Toc176614778)

[Milestone Summary 10](#_Toc176614779)

[PROJECT PLAN 11](#_Toc176614780)

[4.GUI 13](#_Toc176614781)

[Main Window (Launch Window) 13](#_Toc176614782)

[Sign- In (Lecturer) Window 14](#_Toc176614783)

[Sign-In (Programme Coordinator/Academic Manager) 14](#_Toc176614784)

[Create Account Window 15](#_Toc176614785)

[Lecturer Dashboard Window 16](#_Toc176614786)

[Submit Claim Window 16](#_Toc176614787)

[Claim Status Window 17](#_Toc176614788)

[Coordinator Dashboard Window 17](#_Toc176614789)

[Git Hub Repo Link 18](#_Toc176614790)

[GIT HUB COMMIT HISTORY 18](#_Toc176614791)

[Conclusion 19](#_Toc176614792)

[Bibliography 20](#_Toc176614793)

# Introduction

This semester, we are tasked with creating a Contract Monthly Claim System (CMCS) for our college. The goal of this project is to simplify the process of submitting and approving monthly claims for Independent Contractor lecturers. This system will help us understand how to tackle real-world challenges in a professional setting, where accuracy and efficiency are key.

As part of this module, we will dive into .NET GUI development using C#, focusing on creating an interactive and user-friendly interface. The CMCS isn’t just about submitting claims—it involves calculating hours worked and ensuring that these claims are thoroughly reviewed by Programme Coordinators and Academic Managers. This adds a layer of complexity that highlights the importance of precision in administrative tasks.

This specific report consists of design explanations, constructing database structures, and creating and managing project plans and tasks, as well as designing a graphical user interface for users to interact with. By the end of the semester, we will have developed a functional prototype that integrates essential features like document uploads and status tracking, ultimately enhancing user satisfaction and reducing the chances of errors.

# 1.Documentation

## Design Choices Explanation:

### Design Background:

For the design of the GUI Prototype, I used Windows Presentation Foundation (WPF). The selection of WPF was based on its extensive control set, adaptability in UI design, and interaction with.NET, which facilitates a smooth C# development process.

### Database Structure:

Important elements like Users (with roles like Lecturer or Program Coordinator/Academic Manager), Claims, and Supporting Documents are captured in the database design. Every entity has its properties translated to a table so that the data for the system can be managed and stored.

For the database structure I will be using the software Microsoft SQL Server to design the database that I will be using in the POE (P.Atkins and R.Vieria, 2012). My database design makes use of relationships between all tables to ensure a flow that relates to the flow of the program I am creating.

I also used applications like Lucid Charts for the designing of the UML structure for linking relationships between entities of the program.

Below is the SQL DATABASE STRUCTURE I will be using throughout the POE to save information that will be captured within the program.

CREATE DATABASE PROG2BPOE;

USE PROG2BPOE;

-- Table: Lecturer

CREATE TABLE Lecturer (

LecturerID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100),

Faculty VARCHAR(100),

PhoneNumber VARCHAR(20),

PasswordHash VARCHAR(255) NOT NULL,

AccountType VARCHAR(50) NOT NULL

);

-- Table: Claims

CREATE TABLE Claims (

ClaimsID INT PRIMARY KEY,

LecturerID INT,

SubmissionDate DATE,

NoOfSessions INT,

HourlyRatePerSession DECIMAL(10, 2),

ClaimStatus VARCHAR(50),

FOREIGN KEY (LecturerID) REFERENCES Lecturer(LecturerID)

);

-- Table: Programme Coordinator/Academic Manager

CREATE TABLE ProgrammeCoordinator (

CoordinatorManagerID INT PRIMARY KEY,

ClaimsID INT,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100),

Faculty VARCHAR(100),

PhoneNumber VARCHAR(20),

PasswordHash VARCHAR(255) NOT NULL,

AccountType VARCHAR(50) NOT NULL

FOREIGN KEY (ClaimsID) REFERENCES Claims(ClaimsID)

);

-- Table: Supporting Documents (to store uploaded documents)

CREATE TABLE SupportingDocuments (

DocID INT PRIMARY KEY,

ClaimsID INT,

DocName VARCHAR(100), -- Name of the uploaded document

FilePath VARCHAR(255), -- Path where the document is stored

SubmissionDate DATE, -- Date the document was uploaded

FOREIGN KEY (ClaimsID) REFERENCES Claims(ClaimsID)

);

-- Table: Claims Status

CREATE TABLE ClaimsStatus (

ClaimStatusID INT PRIMARY KEY,

ClaimsID INT,

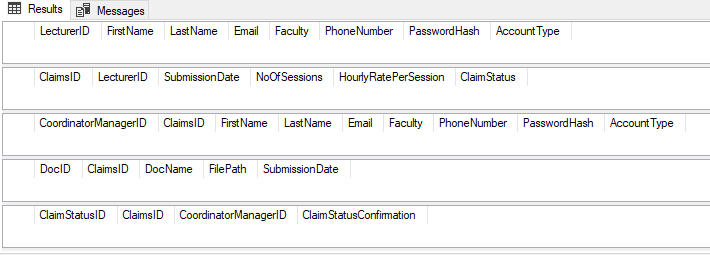
CoordinatorManagerID INT,

ClaimStatusConfirmation VARCHAR(50),

FOREIGN KEY (ClaimsID) REFERENCES Claims(ClaimsID),

FOREIGN KEY (CoordinatorManagerID) REFERENCES ProgrammeCoordinator(CoordinatorManagerID)

);



### GUI Layout:

The designs will be shown later in this document to demonstrate what the GUI design looks like, the following is to explain what the designs do.

**Main Window:** This window welcomes users to the system and provides options to sign in as either a Lecturer or a Programme Coordinator/Academic Manager.

**Sign In/Create Account Windows**: These windows contain clearly labeled text fields for entering user credentials and account creation information. A checkbox is included to allow users to select the type of account they are creating, ensuring that the correct role is assigned.

**Lecturer Dashboard:** This window allows lecturers to submit claims, upload supporting documents, and track the status of their claims.

**Coordinator/Manager Dashboard:** Designed for Programme Coordinators and Academic Managers, this window facilitates the rejection and approval of claims.

**Submit Claims Window:** Lecturers can enter details for each class they taught, along with the number of sessions, and the system automatically calculates the total amount based on the fixed hourly rate of R105 per hour.

**Claim Status Window:** Provides lecturers with real-time updates on the status of their claims, ensuring transparency throughout the approval process.

## Assumptions for programs design:

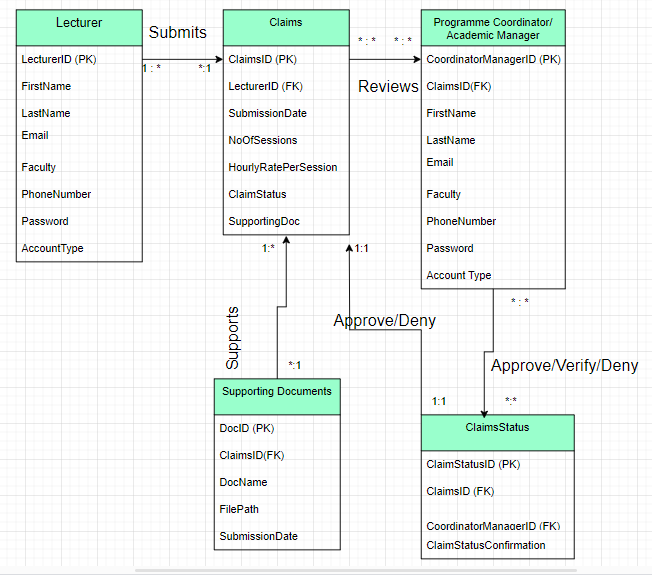
* Users are expected to have basic computer literacy.
* The application is intended for use within a secure institutional network.
* Claims are calculated based on a fixed hourly rate of R105, with each session representing one hour of work.
* Program must be cost effective.
* All the resources are necessary for the project life cycle (projectpractical.com, 2024).

## Constraints for program design:

* **Legal and regulatory constraints:** The system must adhere to institutional policies regarding data protection and privacy.
* **Technical constraints:** The focus of this stage is on GUI design using WPF and database structure; no backend functionality is implemented yet.
* **Financial constraints:** Work with budget given (uxpin.com, n.d.).

The use of WPF allows for a modern, responsive interface that is easy to navigate, enhancing the overall user experience. The design decisions reflect a focus on usability, security, and adherence to project constraints, ensuring the system is both effective and user-friendly.

# 2. UML Class Diagram



(LucidCharts, n.d.)

# 3.Project Plan

Project Plan: Contract Monthly Claims System Development

(The project plan dates only apply to weekdays and do not include weekends in the timeline).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task No. | Task Description | Timeline | Milestone | Start Date | End Date |
| 1 | Requirements Gathering | 2 days |  | 19th Aug 2024 | 21st Aug 2024 |
| 1.2 | Task 1: Identify key functionalities and user requirements. | 16 hours |  | 20th Aug 2024 | 21st Aug 2024 |
|  |  |  | Completion of requirements document. | 20th Aug 2024 | 21st Aug 2024 |
| 2 | Design Phase | 5 days |  | 22nd Aug 2024 | 27th Aug 2024 |
| 2.1 | Task 1: Create UML Class Diagram | 2 days |  | 22nd Aug 2024 | 23rd Aug 2024 |
| 2.2 | Task 2: Design the GUI layout using WPF. | 2 days |  | 23rd Aug 2024 | 26th Aug 2024 |
| 2.3 | Task 3: Design Database Structure | 1 day |  | 27th Aug 2024 | 27th Aug 2024 |
|  |  |  | Finalized UML Class Diagram, GUI layout, Database Design | 22nd Aug 2024 | 27th Aug 2024 |
| 3 | Prototype Development | 7 days |  | 28th Aug 2024 | 6th Sept 2024 |
| 3.1 | Task 1: Implement the database scheme in a mock database | 2 days |  | 28th  Aug 2024 | 30th  Sept 2024 |
| 3.2 | Task 2: Build WPF forms | 4 days |  | 30th Aug 204 | 5th Sept 2024 |
|  | Task 3: Upload Code to GIT | 1 day |  | 6th Sept 2024 | 6th Sept 2024 |
|  |  |  | Milestone 3: Completed prototype with working database schema and initial WPF forms. | 28th Aug 2024 | 6th Sept 2024 |
| 4 | Documentation | 2 days |  | 6th Sept 2024 | 9th Sept 2024 |
| 4.1 | Task 1: Compile the design documentation, including assumptions, constraints, and detailed explanations. |  |  |  |  |
|  |  |  | Milestone 4: Submission of complete design documentation |  | 9th Sept 2024 |
| 5 | Review and Feedback |  |  |  |  |
| 5.1 | Task 1: Submit Part 1. |  |  |  | 9th Sept 2024 |
|  |  |  |  | Milestone 5: Wait for lecturer feedback and apply it to part 2. |  |

## Tasks Breakdown:

### Part 1 POE

**1. Requirements Gathering**

* **Task:** Identify the key functionalities and user requirements.
* **Timeline:** 2 days
* **Milestone:** Complete requirements document with key functionalities and user requirements outlined.

**2. Design Phase**

* **Task 1:** Create UML Class Diagram.
* **Task 2:** Design the GUI layouts using WPF.
* **Task 3:** Design Database Structure
* **Timeline:** 5 days
* **Milestone:** Finalized UML Class Diagram and GUI layout designs.

**3. Prototype Development**

* **Task 1:** Implement the database schema in a mock database.
* **Task 2:** Build the WPF forms.
* **Task 3:** Upload code to GIT repo.
* **Timeline:** 7 days
* **Milestone:** Completed prototype with working database schema and initial WPF forms/MVC views.

**4. Documentation**

* **Task:** Compile the design documentation, including assumptions, constraints, and detailed explanations.
* **Timeline:** 4 days
* **Milestone:** Submission of complete design documentation.

**5. Review and Feedback**

* **Task 1:** Submit PART 1.
* **Task 2:** Present the prototype and documentation for lecturer feedback.
* **Timeline:** 2 days
* **Milestone:** Receive and incorporate feedback from lecturer (www.atlassian.com, 2024).

## Milestone Summary

Milestone 1: Complete requirements gathering documents with key functionalities and user requirements outlined.

* Start Date: 20th August 2024
* End Date: 21st August 2024

Milestone 2: Finalized UML Class Diagram, GUI layout, Database Design

* Start Date: 22nd August 2024
* End Date: 27th August 2024

Milestone 3: Completed prototype with working database schema and initial WPF forms.

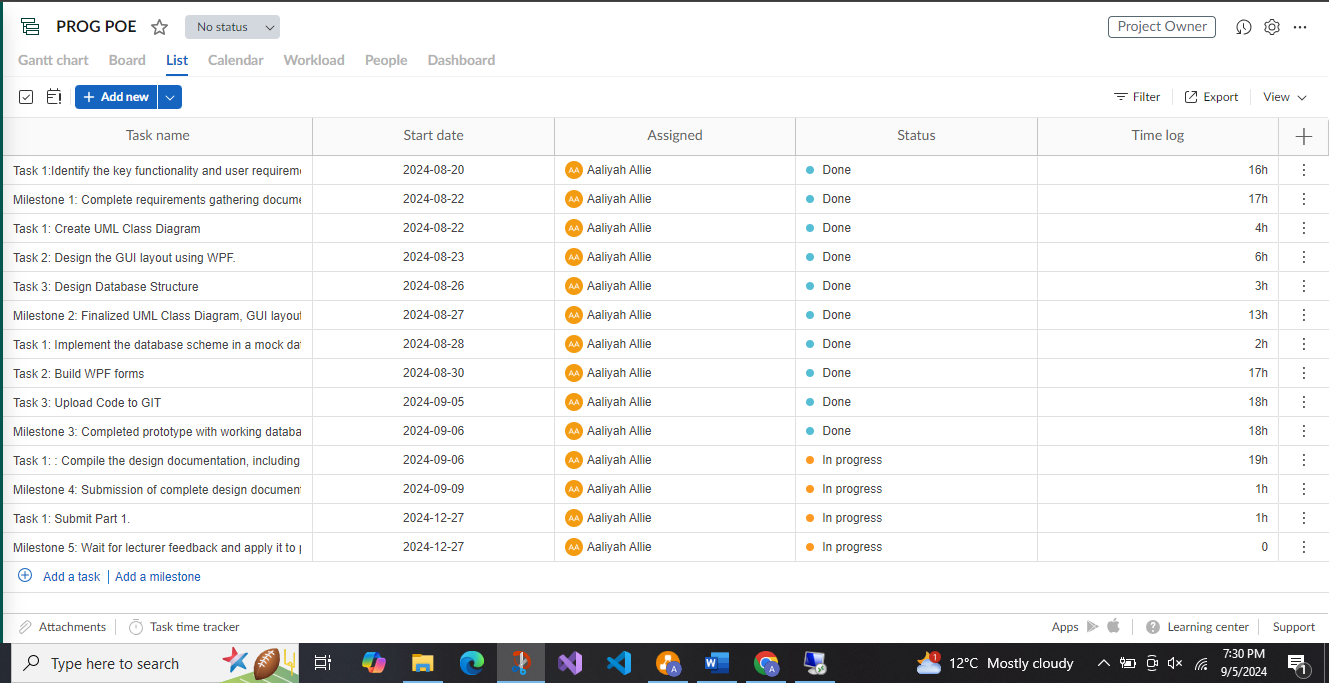
* Start Date: 28th August 2024
* End Date: 6th September 2024

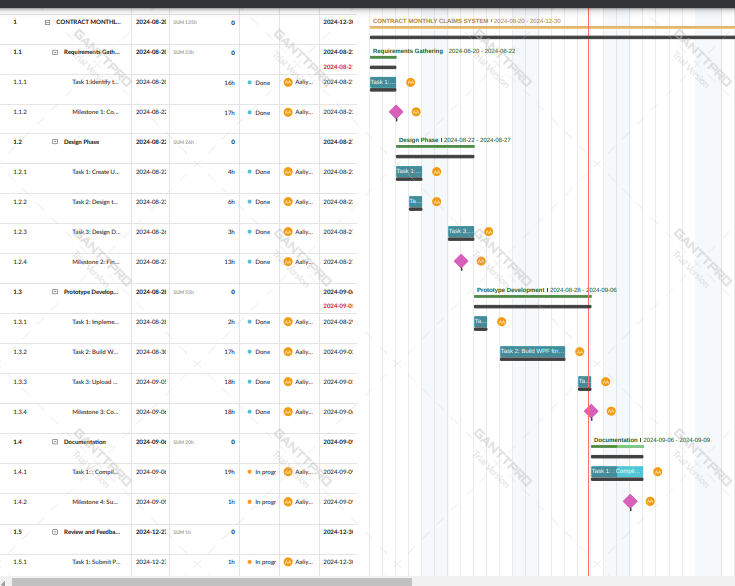
Milestone 4: Submission of complete design documentation.

* Start Date: 6th September 2024
* End Date: 9th September 2024

Milestone 5: Wait for lecturer feedback and apply it to part 2.

# PROJECT PLAN

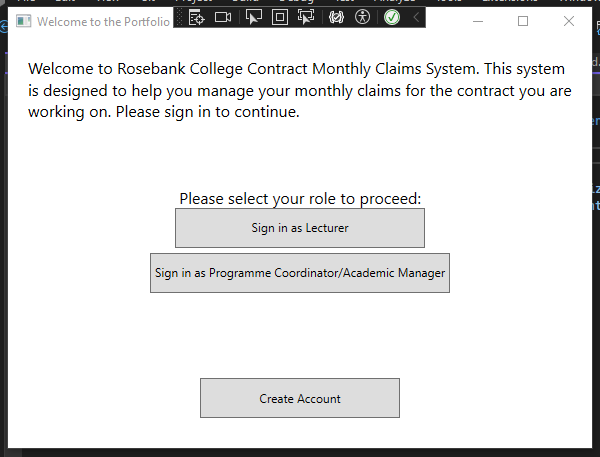




(app.ganttpro.com, n.d.)

# 4.GUI

## Main Window (Launch Window)



The main window consists of a welcome message informing the user what the program is about.

The main window consists of 3 buttons, the 1st button will take the user to a sign in screen for lecturers if the user is a lecturer, the 2nd button will take the user to a sign in screen for programme coordinators / academic managers if that is their occupation and they have an account under that role.

The last button on the screen allows a user to create an account as either a lecturer or programme coordinator / academic manager.

## Sign- In (Lecturer) Window

A screenshot of a computer

Description automatically generated

If in the main window a user chose to sign in as a lecturer this is the window it will take them to.

The window prompts the user to enter their username and password and to login they must press the login button which takes them to the lecturer dashboard.

The create account button will let them create an account if they don’t have one.

## Sign-In (Programme Coordinator/Academic Manager)

A screenshot of a computer

Description automatically generated

If in the main window a user chose to sign in as a programme coordinator/academic manager this is the window it will take them to.

The window prompts the user to enter their username and password and to login they must press the login button which takes them to the lecturer dashboard.

The create account button will let them create an account if they don’t have one.

## Create Account Window

A screenshot of a computer

Description automatically generated

If in the main window or sign-up windows the user decided to create an account, it will redirect them to this screen/window.

It consists of capturing their first name, last name, email address, phone number, faculty, username, password and what role they play in the system.

The submit button saves their account details into the database.

The cancel button goes back to the main screen.

## Lecturer Dashboard Window

A screenshot of a computer

Description automatically generated

The lecturer dashboard allows them to submit claims with the submit claims button, they also can upload supporting documents under the upload supporting documents button, and they can view previous submitted claims and the status of their claims on this screen.

## Submit Claim Window

A screenshot of a computer

Description automatically generated

This screen captures what classes the lecturer has taught, the number of sessions and calculates it based on the hourly rate, once calculated it will submit it for review by programme coordinators / academic managers.

## Claim Status Window

A screenshot of a computer

Description automatically generated

Lecturers can view the status of their claims under this window.

## Coordinator Dashboard Window

A screenshot of a computer

Description automatically generated

This allows for programme coordinators to approve or reject claims submitted by all lecturers.

# Git Hub Repo Link

<https://github.com/AaliyahAllie/AaliyahAllie_ST10212542_PROG6212_PART1.git>

## GIT HUB COMMIT HISTORY

A screenshot of a computer

Description automatically generated

# Conclusion

In conclusion, this part of the POE consists of a report defining, the implementation of the Contract Monthly Claims System (CMCS) and illustrates how software design principles are applied in practice, starting from collecting initial requirements to building a working prototype. The CMCS seeks to rationalise the process of submitting, verifying, and tracking claims for Independent Contractor lecturers by prioritizing user experience and system efficiency. Utilizing WPF for the graphical user interface, combined with a properly organized SQL database, guarantees a responsive and easy-to-use interface. This project has not just increased our knowledge of .NET development but also emphasized the significance of accuracy and client-focused design in professional software solutions. As we progress, continued feedback and iterations will improve the system, making it more functional and better suited to meet real-world requirements.

# Bibliography

app.ganttpro.com, n.d. *https://app.ganttpro.com/.* [Online]   
Available at: https://app.ganttpro.com/  
[Accessed 1 September 2024].

LucidCharts, n.d. *LucidCharts.* [Online]   
Available at: https://lucid.app/documents#/documents?folder\_id=recent  
[Accessed 22 August 2024].

projectpractical.com, 2024. *https://www.projectpractical.com/.* [Online]   
Available at: https://www.projectpractical.com/project-assumptions-examples-and-how-to-manage-them/  
[Accessed 6 September 2024].

uxpin.com, n.d. *www.uxpin.com/studio/blog/constraints-in-design.* [Online]   
Available at: https://www.uxpin.com/studio/blog/constraints-in-design/#:~:text=Common%20types%20of%20design%20constraints%20include%3A%20Technical%20constraints%3A,must%20follow%20Organizational%20constraints%3A%20culture%2C%20structure%2C%20policies%2C%20bureaucracy  
[Accessed 6 September 2024].

Vieria, P. A. R., 2012. *Beginning Microsoft SQL Server 2012 Programming.* 1st ed. s.l.:s.n.

www.atlassian.com, 2024. *www.atlassian.com/blog/project-management.* [Online]   
Available at: https://www.atlassian.com/blog/project-management/write-an-effective-project-plan  
[Accessed 3 September 2024].