



Contract Monthly Claim System


Project Plan

Sajana Bidesi
ST10249843

Table of Contents

Table of Contents	1
INTRODUCTION.....	4
What is a CMCS.....	4
Purpose and Objectives of the CMCS	4
Key Features.....	5
Claim Submission.....	5
Claim Verification and Approval.....	5
Status Monitoring.....	5
Administrative Management	5
Benefits	5
Increased Efficiency:.....	5
Improved Accuracy:	5
Enhanced Transparency:	5
Higher User Satisfaction:.....	5
PROJECT OVERVIEW	6
Target Audience.....	6
SYSTEM REQUIREMENTS.....	7
Hardware Requirements:	7
Software Requirements:	7
TECHNICAL ASPECTS	8
Web-Based Application	8
Built with .NET Core MVC	8
Integration with SQL Server	8
HOW MVC WORKS.....	9
MVC Architecture.....	9
Why MVC is Preferred Over WPF	10
Failure Mitigation	10
Ease of Use.....	10
MVC Scalability.....	10
MVC TECHNOLOGY STACK AND CMCS INTEGRATION.....	11
COLOUR CODE & THEMES	12
UML CLASS DIAGRAM	13
Connections Between Classes	13
DATABASE SCHEMA	14

Tables and Relationships:	14
Users Table:.....	14
Lecturers Table:	14
AcademicManagers Table:	14
ProgrammeCoordinators Table:	14
HR Table:	14
Claims Table:.....	14
Invoices Table:	14
DEPENDENCIES	15
Project Management	15
User Feedback	15
Documentation	15
Compliance and Security.....	15
Training and Support.....	15
USE CASE SCENARIOS.....	16
UI DESIGN	17
WELCOME PAGE	17
REGISTER PAGE.....	18
.....	18
LOGIN PAGE.....	19
ABOUT US PAGE	20
.....	20
SUBMIT CLAIM PAGE	21
.....	21
.....	21
.....	21
VIEW CLAIMS PAGE	22
.....	22
.....	22
INVOICE PAGE.....	23
.....	23
.....	23
.....	23
APPROVE CLAIMS	24
.....	24
GENERATE INVOICE	25



MANAGE LECTURERS	26
UNAUTHORISED ACCESS PAGE.....	27
OUTLINE OF TASKS AND TIMELINE	28
GANTT DIAGRAM.....	30
Assumptions	31
CONSTRAINTS	32
REFERENCES.....	33



INTRODUCTION

What is a CMCS

The Contract Monthly Claim System (CMCS) is a sophisticated web-based application tailored to simplify and enhance the process of managing monthly claims submitted by Independent Contractor (IC) lecturers. This system serves as a vital tool in educational institutions, particularly in managing the often complex and labour-intensive task of processing claims for services rendered by IC lecturers.

Purpose and Objectives of the CMCS

The Contract Monthly Claim System (CMCS) is designed to enhance the submission and approval process, streamlining claims to optimize administrative efficiency and ensure accuracy. Key benefits include improved efficiency through automation of routine tasks, faster claim processing, precise data entry and calculations for maintaining accuracy, and enhanced transparency, offering visibility into the status of claim processing for all stakeholders. This document serves as the project plan for the development of CMCS, outlining key design issues, system requirements, installation procedures, and schedules for each stage of development. It acts as a guide to understanding the scope, dependencies, and deliverables of the CMCS project.



Key Features

Claim Submission

The CMCS indeed provides an intuitive interface that allows a lecturer to submit his or her claim with ease. It captures essential information such as hours worked, hourly rate, and supporting notes through an online form. It also allows for uploading supporting documents like timesheets or invoices to ensure all documentation is readily available for review.

Claim Verification and Approval

The platform introduces automated workflows to enhance the process of verification and approval. Claims are systematically routed to Programme Coordinators and Academic Managers for processing in a manner that is designed to minimize manual intervention. The system enables users to approve or reject claims against predefined criteria, with all decisions being recorded and tracked to ensure comprehensive transparency and accountability.

Status Monitoring

CMCS provides tracking of status in real-time, from submission to final approval or rejection on every claim. This is helpful to both lecturers and administrators to track the progress of claims effectively. Automated notifications enhance communication through prompt provision to users on changes of status and required actions for better response times.

Administrative Management

It grants Human Resources staff and administrators a wide range of data management activities, including handling lecturer profiles, claim histories, and report generation. CMCS can connect seamlessly with existing databases for insights and better decision-making. Furthermore, the platform also creates reports for payment processing, auditing, and compliance, hence assuring efficient administrative operations.

Benefits

Increased Efficiency:

CMCS automates routine tasks and optimizes workflows, greatly reducing the time needed to process claims.

Improved Accuracy:

The system should minimize all errors by carrying out automated calculations and strong data validation to handle the claims accurately.

Enhanced Transparency:

The system offers explicit visibility regarding claim statuses and dispatches automatic notifications, thereby ensuring that all stakeholders remain informed and actively engaged.

Higher User Satisfaction:

The intuitive and responsive interface offered by the system greatly enhances the overall experience of lecturers and administrative staff, providing higher satisfaction for users.



PROJECT OVERVIEW

Target Audience

- The target users are the **lecturers** themselves, who, after all, must go through the system and attach supporting documents to the claims. Their tasks include adding detailed information such as hours worked and hourly rates, which the system then processes to ensure correct calculation and timely submissions.
- This means **Programme Coordinators** are required to scrutinize claims thoroughly by ensuring that all details submitted are accurate and of proper standards, verifying that all claims have substantively met the required criteria before passing them for final approval.
- **Academic Managers** have the final approval of claims. They consider claims verified by Programme Coordinators, make decisions considering guidelines set out, and oversee the entire process of approval. The role undertaken ensures that claims are managed with full accountability and a high level of scrutiny.
- **Human Resources** staff are responsible for the manual administrative work of maintaining the lecturers' data and the processing of claims. They utilize the system to check on claim history, updates, and reporting for payment processing and compliance purposes. Their role is paramount in ensuring the accuracy of recordkeeping and promoting efficiency in administrative operations.



SYSTEM REQUIREMENTS

Hardware Requirements:

Processor: 1 GHz or faster

RAM: 2 GB or more

Hard Disk Space: 500 MB of free space

Display: 1024 x 768 resolution or higher

Software Requirements:

Operating System: Windows 7 or later

.NET Framework: Version 4.7.2 or later (will be installed with the application)

IDE: Visual Studio

Database: SQL Server Management Studio (SSMS)

Version Control: GitHub

Framework: .NET Core for MVC application

TECHNICAL ASPECTS

Web-Based Application

The Contract Monthly Claim System (CMCS) is designed as a web-based application, accessible from any device with an internet connection. This design facilitates ease of use and ensures that users can access the system from various locations without being tied to a specific hardware setup.

Built with .NET Core MVC

CMCS is developed using the .NET Core framework with ASP.NET MVC (Model-View-Controller). This architecture offers robust performance and scalability, making it suitable for handling complex applications efficiently. ASP.NET MVC provides a structured approach to managing user interactions and data processing, enhancing both the development and user experience.

Integration with SQL Server

The application integrates with SQL Server for data storage and management, ensuring reliable data handling and efficient querying. SQL Server supports the application's needs for data integrity and efficient processing of claims.



(google, 2024a)

https://miro.medium.com/v2/resize:fit:684/1*XARDV6_sy7v7GuJA5AGqQg.png

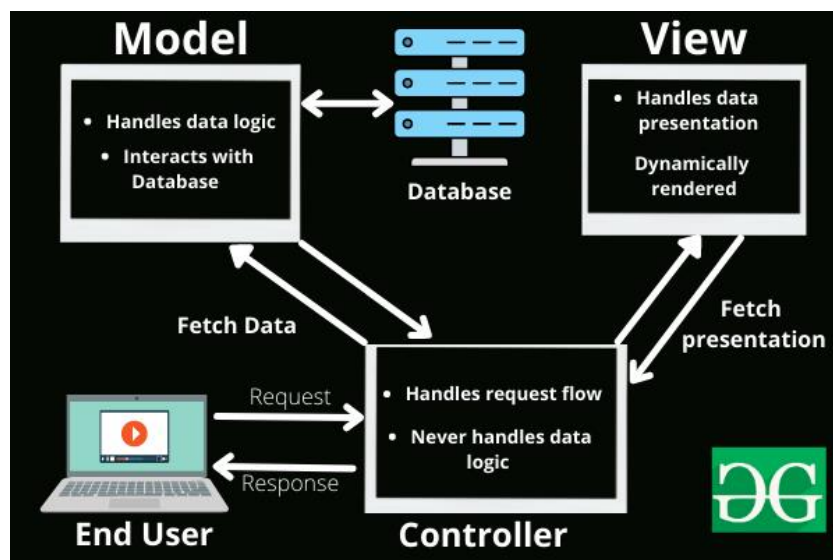
<https://techvify-software.com/wp-content/uploads/2024/01/microsoft-visual-studio-vs-visual-studio-code.jpg>

HOW MVC WORKS

MVC Architecture

The Model-View-Controller (MVC) pattern divides an application into three interconnected components:

- **Model:** Represents the data and business logic of the application. It directly manages the data, logic, and rules of the application.
- **View:** Handles the presentation and display of data. It provides the user interface and ensures that the data is presented in a user-friendly format.
- **Controller:** Manages user input and interactions. It acts as an intermediary between the Model and View, processing input, updating the Model, and selecting the appropriate View for output.



(google, 2024b)

<https://media.geeksforgeeks.org/wp-content/uploads/20220224160807/Model1.png>

Why MVC is Preferred Over WPF

- **Separation of Concerns:** MVC provides a clear separation of concerns by dividing the application into Models, Views, and Controllers. This separation enhances maintainability and allows developers to focus on specific aspects of the application without affecting other parts. If one component fails, the others can continue to function, which improves overall system stability.
- **Better Architecture for Payroll Systems:** MVC's structured approach is well-suited for complex systems like a payroll application, where data management, user interactions, and presentation need to be clearly separated and efficiently managed. This architecture supports scalability and flexibility, crucial for handling various payroll requirements and processes.

Failure Mitigation

MVC's architecture helps in failure mitigation by isolating different components. If a failure occurs in one part of the system (e.g., the Model), it does not necessarily affect the other components (e.g., the View or Controller). This isolation makes it easier to diagnose and address issues without disrupting the entire application. (Quantative, 2022)

Ease of Use

MVC enhances ease of use by providing a structured framework for developing web applications. The separation of concerns simplifies the development process, allowing developers to work on individual components independently. Additionally, the MVC pattern supports a clean and intuitive user interface, making the application more accessible to users.

MVC Scalability

The MVC architecture is highly scalable, allowing applications to grow and adapt to increasing user demands. By separating the application into distinct components, MVC facilitates efficient management of expanding features and functionalities. This scalability is particularly beneficial for applications like CMCS that may evolve over time.

(Troelsen and Japikse, 2021)

MVC TECHNOLOGY STACK AND CMCS INTEGRATION

The MVC framework employs a technology stack comprising HTML, CSS, C#, and JavaScript, each playing a critical role in the functionality and user experience of the Contract Monthly Claim System (CMCS).



(google, 2024c)

<https://www.1training.org/wp-content/uploads/2017/10/6.png>

- **HTML:** Provides the foundational structure and content of web pages in CMCS, ensuring that information is organized and accessible. It defines how content is displayed to users.
- **CSS:** Styles and designs the appearance of CMCS web pages, creating a user-friendly and visually appealing interface. It enhances the overall look and feel of the application.
- **JavaScript:** Manages client-side interactions and dynamic content updates, adding interactivity to the CMCS web pages. It improves user experience by enabling real-time updates and responsive features.
- **C#:** Handles server-side logic and interactions in CMCS. It manages business logic, data processing, and communication with the database, ensuring the smooth operation of the application.

This integration of HTML, CSS, C#, and JavaScript within the MVC framework ensures that CMCS is a robust, scalable, and user-friendly application designed to handle complex claim processing tasks efficiently.

(w3schools, 2019)

The combination of these technologies within the MVC framework ensures that CMCS is a robust, scalable, and user-friendly application capable of efficiently handling complex claim processing tasks. (Troelsen and Japikse, 2021)

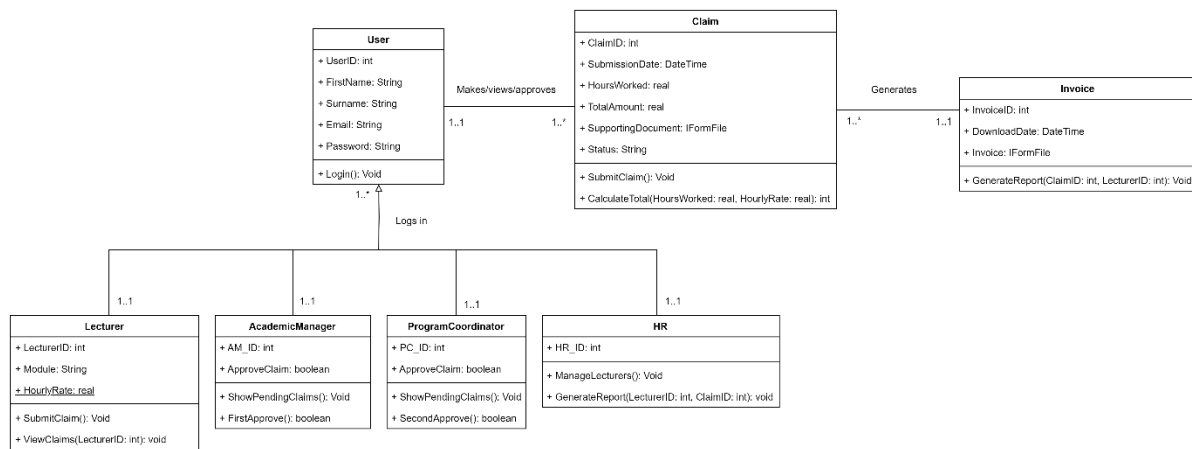
COLOUR CODE & THEMES

The Contract Monthly Claim System (CMCS) is elegantly draped in the alluring essence of the Dark Academia theme, transforming the interface into a canvas of intellectual charm and mystique. This carefully curated aesthetic, with its deep, evocative tones and vintage-inspired allure, invites users into a space where function meets art. The sophisticated palette does more than merely appeal to the eye—it conjures an atmosphere that is both thoughtful and timeless, encouraging focus and enhancing the experience. The Dark Academia style doesn't just bring beauty; it elevates usability by offering a design that minimizes eye strain and gently guides users through the application with a sense of grace and ease.



The CMCS weaves together a harmonious blend of browns, blacks, greens, and whites, each hue playing its part in the tapestry of this scholarly environment. Rich browns ground the design, imbuing it with warmth, stability, and an inviting sense of familiarity, akin to the comforting scent of leather-bound books in an ancient library. Black introduces a bold, modern sophistication—sleek and striking, it frames the interface in clarity and precision, ensuring sharp contrast and easy navigation. Green, the colour of growth and renewal, brings balance and harmony to the design, subtly highlighting key actions and guiding the user's journey. White serves as the clean, neutral backdrop, like the crisp pages of a well-worn manuscript, allowing every element to stand out with clarity and elegance. Together, these tones weave a seamless, immersive experience—one that is as functional as it is enchanting.

UML CLASS DIAGRAM



Link: <https://tinyurl.com/drawiouml>

(draw.io, 2024)

Connections Between Classes

The **User** class is the parent class that contains shared attributes and methods for all users in the system, including `UserID`, `Name`, `Email`, `Password`, and methods like `Login()` and `Register()`. Several child classes inherit from **User**, each adding specific attributes and methods.

- **Lecturer:** Adds `LecturerID`, `Module`, and hourly rate and contains methods like `SubmitClaim()` and `ViewClaims()`.
- **Academic Manager:** Includes `AM_ID` and `ApproveClaim` with methods such as `ShowPendingClaims()` and `FirstApprove()`.
- **Programme Coordinator:** Adds `PC_ID` and `ApproveClaim` and contains methods like `ShowPendingClaims()` and `SecondApprove()`.
- **HR:** Includes `HR_ID` with methods such as `ProcessInvoice()` and `ManageLecturer()`.

Inheritance allows these child classes to reuse the **User** class's common features, such as attributes and basic methods, while extending them with their own specific functions. This design promotes code reuse and consistency across different user roles.

- **Claim:** Contains attributes such as `ClaimID`, `SubmissionDate`, `HoursWorked`, `TotalAmount`, `SupportingDocument`, and `Status` (this could be pending, approved, or denied) along with methods such as `SubmitClaim()`, `CalculateTotal` (automatically calculates the total amount of hours the Lecturer has claimed)
- **Invoice:** Contains attributes such as `InvoiceID`, `DownloadDate`, and `Invoice` (downloaded document) along with methods such as `GenerateReport()`

DATABASE SCHEMA

The database schema is designed to manage the Contract Monthly Claim System (CMCS) efficiently. It includes tables for users (Lecturers, Academic Managers, Programme Coordinators, and HR), claims, and invoices. Each table supports the core functionalities of the system and maintains relationships essential for data integrity and operations. (IBM, n.d.)

Tables and Relationships:

Users Table:

- **Fields:** UserID (Primary Key), Name, Email, Password, Role
- **Purpose:** Central table for storing common user details. The Role field differentiates between Lecturers, Academic Managers, Programme Coordinators, and HR.

Lecturers Table:

- **Fields:** LecturerID (Primary Key, Foreign Key to UserID), UserID
- **Purpose:** Contains specific attributes for lecturers. Linked to the Users table for shared details.

AcademicManagers Table:

- **Fields:** AM_ID (Primary Key, Foreign Key to UserID), UserID, ApproveClaim
- **Purpose:** Holds data specific to Academic Managers. Linked to the Users table and includes an attribute for claim approval.

ProgrammeCoordinators Table:

- **Fields:** PC_ID (Primary Key, Foreign Key to UserID), UserID
- **Purpose:** Stores Programme Coordinators' information, linked to the Users table.

HR Table:

- **Fields:** HR_ID (Primary Key, Foreign Key to UserID), UserID
- **Purpose:** Contains details for HR personnel. Linked to the Users table and includes fields for processing invoices and managing lecturer data.

Claims Table:

- **Fields:** ClaimID (Primary Key), LecturerID (Foreign Key), DateOfSubmission, TotalAmount, Status
- **Purpose:** Manages claim submissions by lecturers. Linked to the Lecturers table.

Invoices Table:

- **Fields:** InvoiceID (Primary Key), ClaimID (Foreign Key), InvoiceDate

DEPENDENCIES

Requirement Analysis: Gather and document requirements. Essential for starting Database Schema Design and GUI Development.

Database Schema Design: Create tables and relationships based on requirements. Necessary before GUI Development and Integration Testing.

GUI Development: Design the user interface based on wireframes and schema. Follows Database Schema Design and leads into Integration Testing.

Integration Testing: Ensure all system components work together. Follows GUI Development and Unit Testing.

(Troelsen and Japikse, 2021)

Project Management

Effective project management involves scheduling, resource allocation, and progress tracking using tools like Gantt charts or project management software.

User Feedback

Collect feedback through usability testing or beta releases to refine design and improve functionality.

Documentation

Create comprehensive documentation, including technical docs, user manuals, and system guides to support development and maintenance.

Compliance and Security

Ensure compliance with relevant standards (e.g., GDPR, HIPAA) and implement data protection measures like encryption and access controls.

Training and Support

Provide training for users and technical support staff for a smooth transition and effective system use.

USE CASE SCENARIOS

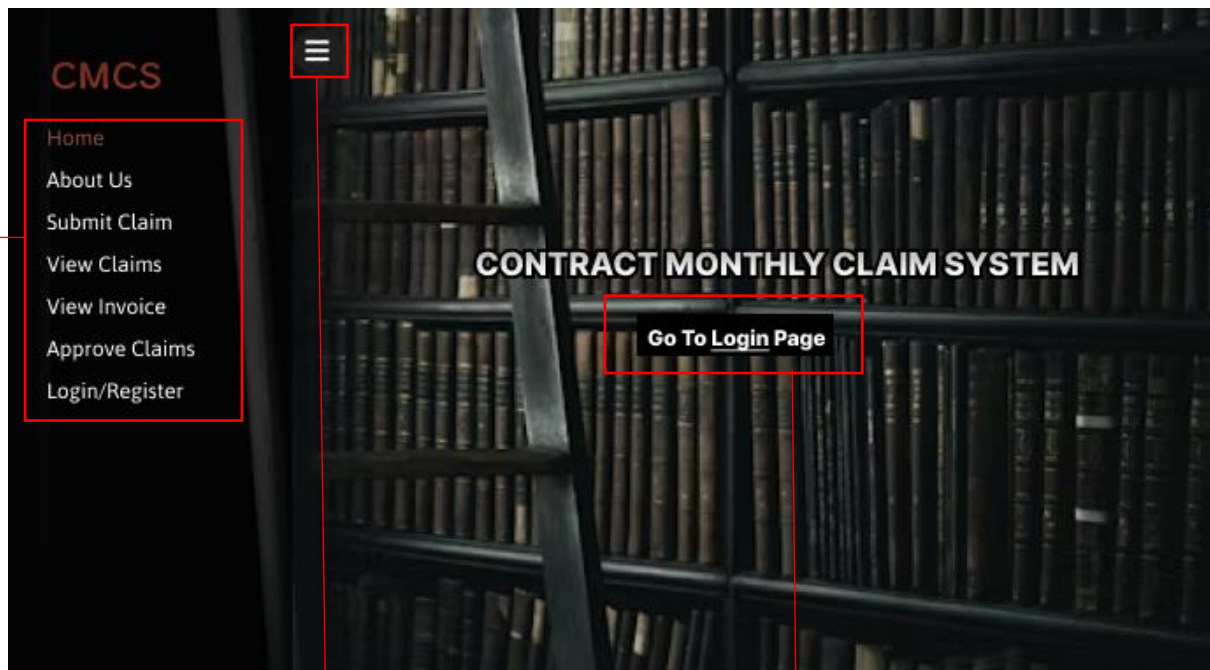
USE CASE	DESCRIPTION
Submit Claim	Lecturers fill out a form with hours worked, hourly rate, and supporting documents. Once submitted, the claim is recorded and awaits review.
Approve/Reject Claim	Admin(Program Coordinators and Academic Managers) review pending claims with details and supporting documentation, then approve or deny them. Status updates are don't automatically.
Upload Supporting Documents	Lecturers can upload files (eg. Invoices, student registers, and timesheets) during claim submissions. Files are linked to claims, with size and format restrictions enforced.
Track Claim Status	Users can view real-time updates on claim statuses('pending', 'approved', or 'denied') that is displayed with labels 'P', 'A', or 'D' respectively.
Generate Report	HR can generate reports and filter reports on claims made by the lecturer name.
Manage Lecturer Data	HR can update and maintain lecturer data, including personal, contact, and payment information, to ensure accurate records.
User Authentication and Authorization	The system verifies user identities and controlls based on roles (Lecturers, Programme Coordinators, Academic Managers, and HR) to ensure appropriate access to functionalities.

UI DESIGN

NB: all users have access to this page.

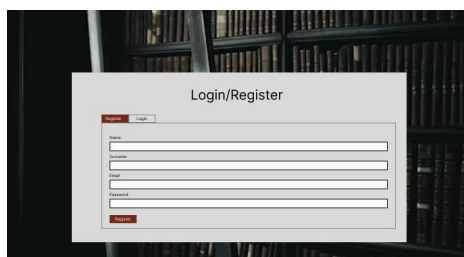
WELCOME PAGE

This is the welcome page. Users that first open the system will be able to see this page first.



Displays the current screen as brown text.

On the left-hand side is my nav bar which can be accessed by clicking the menu button/hamburger sign (\equiv). The layout of the page will change accordingly. This means that the page will move to the right when the menu tab is opened or display as the normal size when the menu is closed.



Clicking on the 'go to login page' button will take the user to the login page where they can enter their credentials in order to access the functionalities of the system.

REGISTER PAGE

NB: all users have access to this page.

The screenshot shows a web form titled "Login/Register" overlaid on a background of bookshelves. The form has two tabs at the top: "Register" (highlighted with a red box) and "Login". Below the tabs are four input fields: "Name", "Surname", "Email", and "Password" (the last three are grouped by a red box). A "Register" button is at the bottom. Red arrows point from the "Register" tab to a text box on the left and from the "Name" and "Surname" fields to a text box on the right.

Login and Register tabs used for easy access to either the register page or login page.

Future development plan:

Regarding the fact that not everyone who has access to the system can register, there will be data validation, and an email verification sent to authorise the user to access the system. This will be done by HR.

When first registering to the system, users must enter their first name and last name along with their verified email, and password.

LOGIN PAGE

NB: all users have access to this page.

The screenshot shows a web interface titled "Login/Register" overlaid on a background of bookshelves. At the top of the form are two tabs: "Register" and "Login". Below these tabs are two input fields labeled "Email" and "Password". A red box highlights the "Register" tab and the "Email" and "Password" input fields. Another red box highlights the "Login" button located below the "Password" field. A red line with arrows points from the "Register" tab to a text box on the left and from the "Login" button to a text box on the right.

Login and Register tabs used for easy access to either the register page or login page.

Future development plan:

Regarding the fact that not everyone who has access to the system can register, there will be data validation, and an email verification sent to authorise the user to access the system. This will be done by HR.

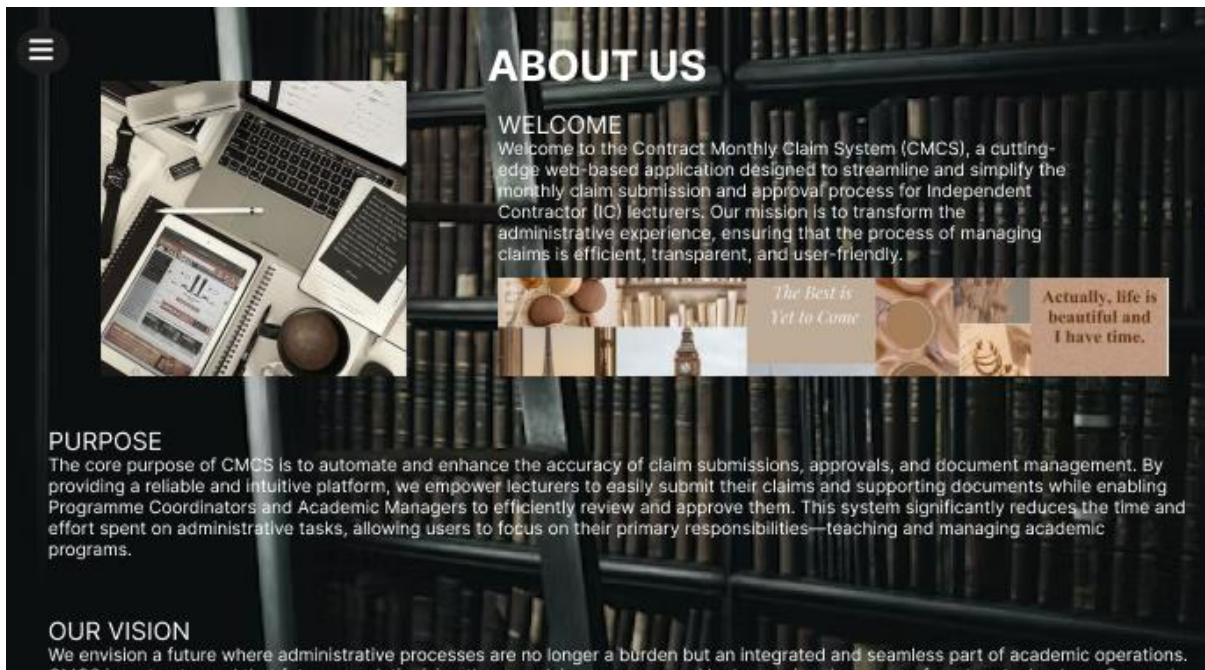
Users can enter their credentials to access the CMCS. The email and password fields will have data validations and error handling.

Emails must contain either: '@admin.co.za' for academic managers, programme coordinators, and HR, or '@iie.co.za' for lecturers.

These will determine the roles and levels of authorization and authentication to which pages certain users are able to access.

ABOUT US PAGE

NB: all users have access to this page.



The purpose of the about us page is to give users a homey and aesthetic feel to the system. Not only does it discuss the purpose, vision, and values, but the user will also have a manual on how to navigate through the system and make claims, approve claims, download invoices, etc.

SUBMIT CLAIM PAGE

NB: only lecturers have access to this page.

The screenshot shows a 'Submit Claim' form with the following fields and buttons:

- Hours Worked:** A text input field.
- Hourly Rate:** A text input field containing 'R 130.00'.
- Total Amount (R):** A text input field.
- Supporting Document:** A section containing a 'Choose File' button and a 'No File Chosen' text.
- Submit Claim:** A red button.
- Reset Values:** A black button.

Red boxes highlight the 'Hourly Rate' field, the 'Total Amount (R)' field, the 'Supporting Document' section, and the 'Submit Claim' and 'Reset Values' buttons. Red arrows point from these boxes to explanatory text boxes below.

'Submit claim' is used to submit the claim to get approved.

'Reset values' will clear the form for resubmission.

The lecturers hourly rate is a final variable and they will not be able to change it.

The lecturers' total amount (hours worked x hourly rate) is automatically calculated and displayed in real time.

Lecturers can submit documents along with their claims, it has to meet the requirements of file size and format.

This will be reviewed by admin.

VIEW CLAIMS PAGE

NB: only lecturers have access to this page. They will only be able to view their own claims. This is done according to their lecturer ID.

[illegible]

Pending,
approved,
or denied
status.

'Date of Submission' is an automatic DateTime Value that will be saved when the lecturer clicks on the 'Submit Claim' button.

Dropdown to view all 1st and 2nd approvers.

INVOICE PAGE


NB: only lecturers have access to this page. They will only be able to view their own claims. This is done according to their lecturer ID.

INVOICE

NAME: SAJANA BIDESI
TITLE: LECTURER
COMPANY: VARSITY COLLEGE DURBAN NORTH

Description	Date	Hour(s) Worked	Total	
SAJANA BIDESI	10/09/2024	24	3 120.00	
SAJANA BIDESI	05/08/2024	19	2 470.00	
GRAND TOTAL				5 590.00

PAYMENT DATE : SEPTEMBER, 25 2024
BANK NAME : ABSA BANK
ACCOUNT HOLDER : S.BIDESI
ACCOUNT NUMBER : 1024984300

DOWNLOAD 

This page displays an invoice for the lecturer based on all their approved claims. They will be able to view their personal details, payment details, as well as their added up total salary for each month and the grand total

Downloads a PDF of the lecturers invoice.

APPROVE CLAIMS

NB: only admin has access to this page.
They will be able to view all claims made by
all lecturers.

Filters by lecturer
name and displays
all their pending
claims.

Immediate changes made
to status in real-time.

Approve Claims

Filter: SAJANA BIDESI

Lecture Name	Date Submitted	Total Amount (R)	Status	Change Status
SAJANA BIDESI	10/09/2024	R 3 120.00	A	APPROVE DENY
SAJANA BIDESI	05/08/2024	R 2 470.00	D	APPROVE DENY

Buttons to approve or
deny claim.

GENERATE INVOICE

Filters by lecturer name and displays all their approved claims.

NB: only admin has access to this page. They will be able to view all claims made by all lecturers.

The screenshot shows a web interface titled 'Generate Invoice'. At the top left is a hamburger menu icon. Below the title is a filter dropdown menu labeled 'Filter: LECTURER NAME' with a downward arrow. Below the filter is a table with the following columns: 'Date Of Submission', 'Hour(s) Worked', 'Hourly Rates', '1st Approver', '2nd Approver', 'Total Amount (R)', 'Status', and an empty column for the 'Generate Report' button. The table contains four rows of data, all with a status of 'A' (Approved). Red boxes highlight the filter dropdown, the '1st Approver' and '2nd Approver' columns, and the 'Generate Report' button column. Red arrows point from the text boxes to these elements.

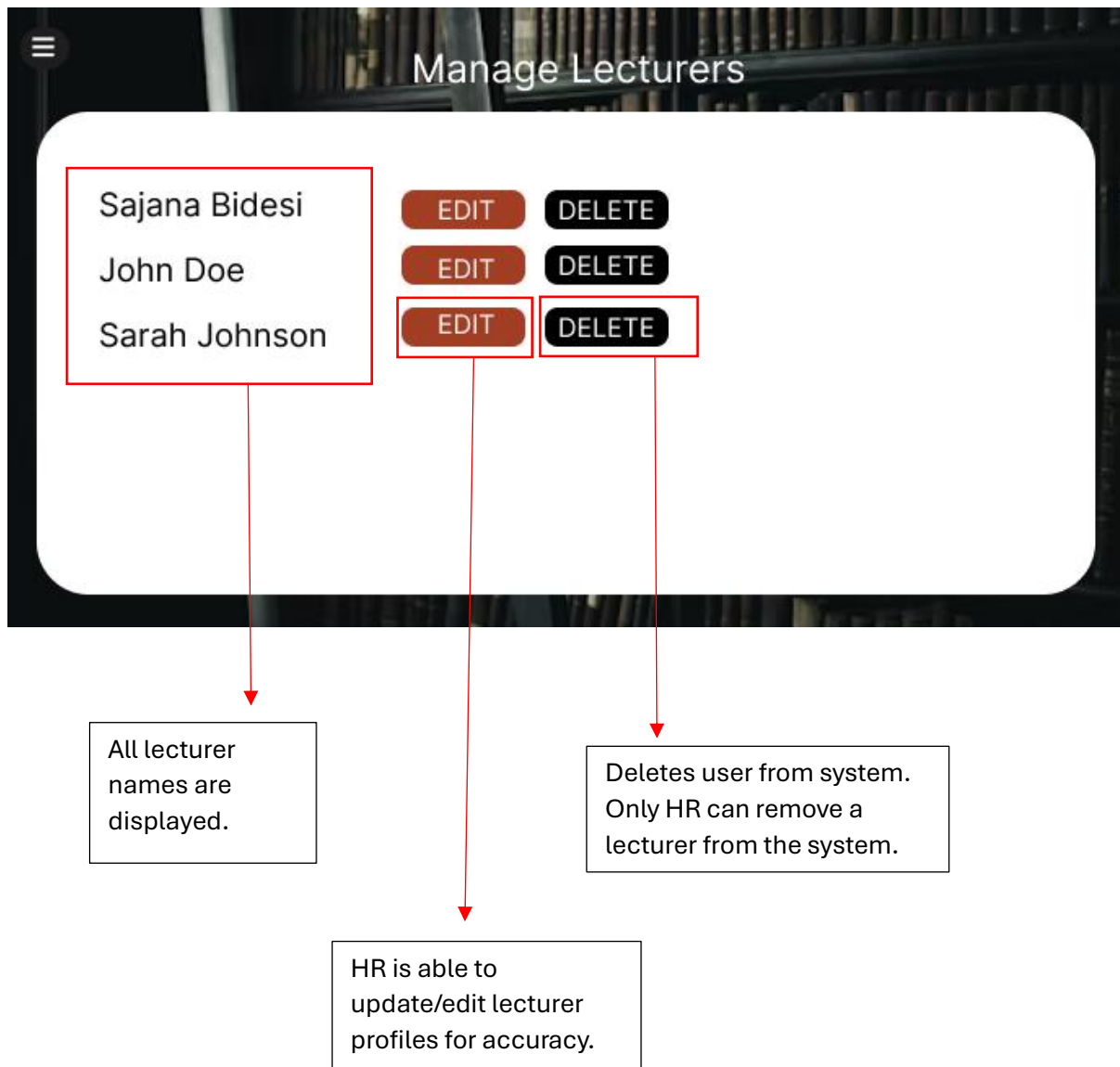
Date Of Submission	Hour(s) Worked	Hourly Rates	1st Approver	2nd Approver	Total Amount (R)	Status	
10/09/2024	24	R 130.00	J. NAICKER	E. ATKISON	3 120.00	A	Generate Report
05/08/2024	19	R 130.00	J. NAICKER	E. ATKISON	2 470.00	A	Generate Report
10/09/2024	10	R 190.00	J. GONZALVES	A. RAMKISSON	1 900.00	A	Generate Report
05/08/2024	16	R 190.00	J. GONZALVES	A. RAMKISSON	3 040.00	A	Generate Report

Button to generate report. This should be done automatically, but if the user requests their invoice, they will be able to go through HR to access it.

First (academic managers) and second approver (program coordinators) names are displayed.

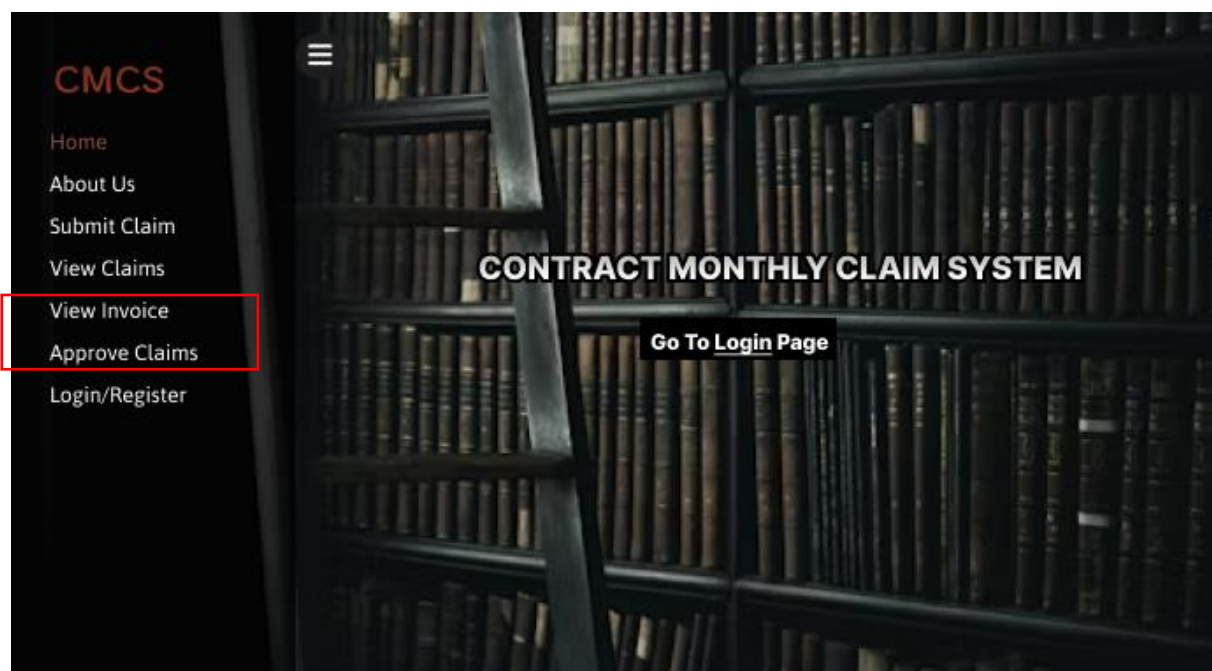
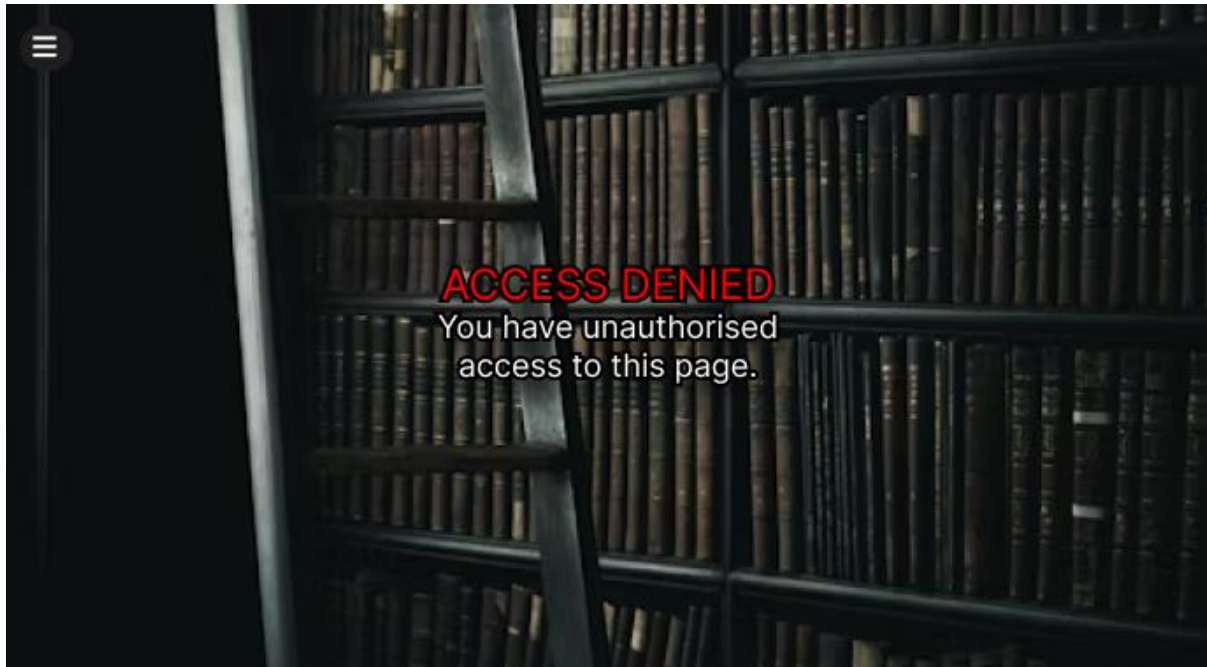
MANAGE LECTURERS

NB: only HR has access to this page.



UNAUTHORISED ACCESS PAGE

NB: lecturers will see this page if they click on certain buttons in the menu



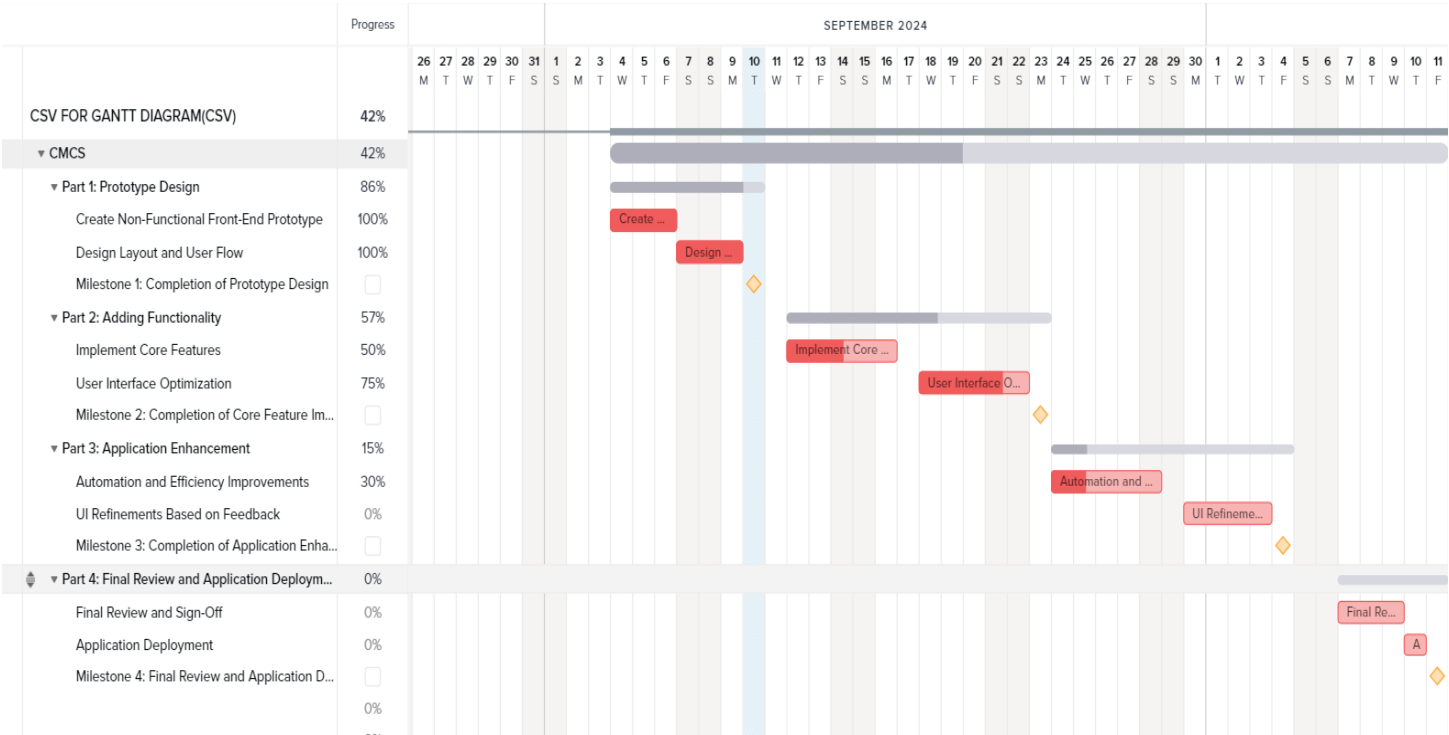
OUTLINE OF TASKS AND TIMELINE

WEEK	TASK	DURATION	DETAILS	MILESTONES
WEEK 1	REQUIREMENT ANALYSIS & INITIAL DESIGN	1 WEEK	CONDUCT DISCUSSIONS WITH STAKEHOLDERS, GATHER AND DOCUMENT SYSTEM AND USER REQUIREMENTS	MILESTONE 1: COMPLETION OF REQUIREMENTS ANALYSIS
	REQUIREMENT ANALYSIS	1 WEEK	ALIGN PROJECT WITH USER NEEDS AND EXPECTATIONS	ALL SYSTEM AND USER REQUIREMENTS DOCUMENTED AND APPROVED
	INITIAL DESIGN SKETCHES & WIREFRAMES	1 WEEK	PREPARE INITIAL DESIGN SKETCHES AND WIREFRAMES FOR REVIEW	INITIAL DESIGN SKETCHES AND WIREFRAMES READY FOR REVIEW
WEEK 2	SYSTEM DESIGN	1 WEEK	DEVELOP UML DIAGRAMS, DESIGN THE DATABASE SCHEMA, FINALIZE WIREFRAMES	MILESTONE 2: COMPLETING OF DESIGN PHASE
	SYSTEM REQUIREMENTS	3 DAYS	DOCUMENT TECHNICAL REQUIREMENTS (HARDWARE, SOFTWARE, SECURITY)	UML DIAGRAMS AND DATABASE SCHEMA FINALIZED AND APPROVED
	USER REQUIREMENTS	4 DAYS	IDENTIFY AND DOCUMENT SPECIFIC NEEDS AND GOALS OF EACH USER GROUP	WIREFRAMES COMPLETE AND READY FOR DEVELOPMENT
WEEK 3	DEVELOPMENT PHASE	1 WEEK	SET UP DEVELOPMENT ENVIRONMENT, CREATE DATABASE, DEVELOP PROJECT STRUCTURE, IMPLEMENT GUI	MILESTONE 3: COMPLETION OF INITIAL GUI DEVELOPMENT
	SETUP DEVELOPMENT ENVIRONMENT	1 DAY	CONFIGURE DEVELOPMENT TOOLS, LIBRARIES, AND FRAMEWORKS	BASIC STRUCTURE OF THE PROJECT IS IN PLACE
	CREATE DATABASE AND TABLES	2 DAYS	IMPLEMENT DATABASE SCHEMA, SET UP INITIAL TABLES	CORE GUI IS FUNCTIONAL, MAJOR FEATURES READY FOR TESTING
	DEVELOP BASIC STRUCTURE OF MVC PROJECT	2 DAYS	SET UP MODEL, VIEW, AND CONTROLLER LAYERS	
	IMPLEMENT GUI ELEMENTS	2 DAYS	DEVELOP INITIAL GUI COMPONENTS, FOCUSING ON USABILITY AND AESTHETICS	

WEEK 4	TESTING PHASE	1 WEEK	CONDUCT UNIT, INTEGRATION, AND USER ACCEPTANCE TESTING: REFINES THE PROTOTYPE	MILESTONE 4: COMPLETION OF TESTING AND PROTOTYPE REFINEMENT
	UNIT TESTING	3 DAYS	TEST INDIVIDUAL COMPONENTS AND FUNCTIONS	ALL TESTING COMPLETE, SYSTEM STABLE, PROTOTYPE REFINES
	INTEGRATION TESTING	2 DAYS	TEST THE INTEGRATION OF DIFFERENT COMPONENTS AND MODULES	PROTOTYPE READY FOR FINAL DEPLOYMENT
	USER ACCEPTANCE TESTING	2 DAYS	VALIDATE THAT THE SYSTEM MEETS USER NEEDS AND PREFORMS AS REQUIRED	
	REFINEMENTS	2 DAYS	MAKE ADJUSTMENTS AND IMPLEMENTS BASED ON TESTING FEEDBACK	

(Troelsen and Japikse, 2021)

GANTT DIAGRAM



(teamgantt, 2024; Troelsen and Japikse, 2021)



Assumptions

- **Use of MVC Architecture:** The system is designed using the Model-View-Controller (MVC) architecture. This structure separates the application into three interconnected components: models (representing data), views (user interface), and controllers (handling input and application logic). This assumption ensures organized code management and separation of concerns.
- **Integration with a Database:** The system relies on a relational database for storing and retrieving data related to claims, users, and other relevant information. This integration supports data persistence, consistency, and facilitates efficient data management.
- **Database Efficiency:** It is assumed that the chosen database management system will efficiently handle the volume of data and maintain data integrity. This includes implementing indexing and normalization to optimize performance and storage.
- **User Accessibility:** All users (lecturers, programme coordinators, academic managers, and HR personnel) will have appropriate access levels to the system features relevant to their roles. This ensures that each user can efficiently perform their tasks.
- **Compliance with Regulations:** The system will adhere to relevant legal and regulatory standards for data protection and privacy.

(Troelsen and Japikse, 2021)

CONSTRAINTS

Hardware and Software Limitations	Development and testing might be constrained by the available hardware and software tools. Limited resources could impact the performance and scalability of the application, necessitating optimization strategies.
Compatibility Issues	There may be compatibility issues with different versions of software tools, libraries, or frameworks used in development. Ensuring that the system functions across various environments can require additional testing and adjustments.
Claim Limits	Lecturers are restricted to claiming a specific maximum number of hours per month, typically around ± 200 hours. This constraint impacts how the system validates and processes claims, requiring accurate tracking and validation mechanisms.
Specified Hourly Rate	Lecturers have a predefined hourly rate, which must be adhered to for calculating claim amounts. The system must enforce this rate and handle any changes in rates as per organizational policies.
Document Size and Format Restrictions	Uploaded supporting documents must adhere to size and format restrictions. This constraint ensures that the system handles file uploads efficiently and avoids performance issues related to large or incompatible files.
Approval Workflow	The approval process for claims involves multiple roles (programme coordinators, academic managers, and HR personnel). The system must support a structured workflow to handle approvals and rejections, ensuring proper authorization at each stage.
User Training and Adoption	Users must be adequately trained to use the system effectively. This constraint involves planning for training sessions and support to ensure smooth adoption and minimize user errors.
System Security	The system must be secure from unauthorized access and data breaches. Implementing robust security measures, such as authentication, authorization, and encryption, is crucial to protect sensitive information.

(Troelsen and Japikse, 2021)

REFERENCES

draw.io (2024). *Flowchart Maker & Online Diagram Software*. [online] Tinyurl.com. Available at: <https://tinyurl.com/4yf827r4> [Accessed 10 Sep. 2024].

Figma (2024). *Figma*. [online] Figma. Available at: <https://www.figma.com/design/kTuwlz1ro4W0eaOWQV6lUJ/CMCS?node-id=19-158&t=D8tD2glOvlMao8Mk-1> [Accessed 10 Sep. 2024].

google (2024a). [online] Techvify-software.com. Available at: <https://techvify-software.com/wp-content/uploads/2024/01/microsoft-visual-studio-vs-visual-studio-code.jpg> [Accessed 10 Sep. 2024].

google (2024b). [online] Geeksforgeeks.org. Available at: <https://media.geeksforgeeks.org/wp-content/uploads/20220224160807/Model1.png>.

google (2024c). [online] 1training.org. Available at: <https://www.1training.org/wp-content/uploads/2017/10/6.png> [Accessed 10 Sep. 2024].

IBM (n.d.). *What is a database schema? / IBM*. [online] www.ibm.com. Available at: <https://www.ibm.com/topics/database-schema>.

Quantative (2022). *OKR Program Failure Prevention Guide / Quantive*. [online] Quantive.com. Available at: <https://community.quantive.com/blog/okr-program-failure-prevention-guide> [Accessed 10 Sep. 2024].

teamgantt (2024). *TeamGantt*. [online] Teamgantt.com. Available at: <https://app.teamgantt.com/projects/gantt?ids=4068680> [Accessed 10 Sep. 2024].

Troelsen, A. and Japikse, P. (2021). *Pro C# 9 with .NET 5 : foundational principles and practices in programming*. Berkeley, Ca: Apress L. P., . Copyright.

w3schools (2019). *HTML JavaScript*. [online] W3schools.com. Available at: https://www.w3schools.com/html/html_scripts.asp.