**Prog6212 : Part 1**

**ST10438554**

**Due Date: 9 September 2025**

**Question 1: Documentation**

The Contract Monthly Claim System (CMCS) was developed in response to the need for a practical, user-friendly solution that accurately reflects the claim management process used by independent contractor lecturers. The ASP.NET Core MVC architecture, the system's foundation, was selected due to its scalability and strong separation of concerns, which guarantee future application growth and maintenance. The data layer and application logic seamlessly integrate with Entity Framework Core for database access, enabling rapid prototyping while adhering to industry standards.ASP offers strong and dependable transactional support since SQL Server is used as the database. Because NET Identity is integrated for secure authentication and role-based authorization, managers, coordinators, and instructors can only access features that are pertinent to their roles.

In order to simulate the process of submitting, approving, and maintaining records, the database structure was created. The Lecturer, Claim, Supporting Document, and Approval Record are core entities that are connected by clear relationships. A lecturer might, for instance, submit several claims, each with a number of supporting documents. A program coordinator reviews claims first, followed by an academic manager. Each decision is recorded in the Approval Record entity for traceability. To guarantee that computations are transparent and auditable, important fields like Hours Worked, Hourly Rate, and Claim Amount are included. Claims are directly linked to supporting documentation, such as proof of work, ensuring that every submission is supported by data.

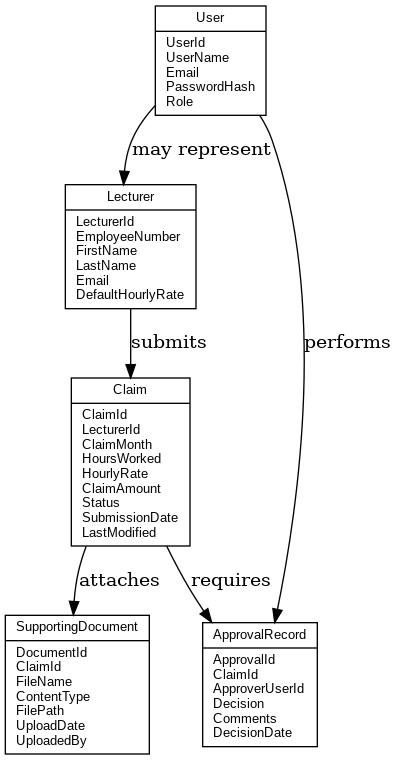
To increase consistency, the layout includes a fixed navigation menu, clear action buttons, inline validation messages, and confirmation prompts for important actions such as rejecting a claim. The graphical user interface (GUI) was created with simplicity, usability, and accessibility in mind to enhance the user experience. The instructor can choose a claim month, enter their work hours, and attach the necessary supporting documentation using the Submit Claim form that is part of their interface. Because the claim amount is automatically determined using the hourly rate, efficiency is increased, and errors are decreased. Lecturers can monitor the status of their submissions with a My Claims section that shows statuses like Draft, Submitted, Approved, or Rejected.

The CMCS was designed with a number of presumptions and limitations in mind. For authentication purposes, it is assumed that all lecturers have access to dependable internet connections and institutional email addresses. Additionally, it is presumed that hourly rates are fixed and seldom alter, which makes claim computation easier. To guarantee accountability and transparency, it is also assumed that each claim must be accompanied by a minimum of one supporting document. To balance usability and storage management, restrictions include a 10 MB file upload size limit per document and a maximum of five attachments per claim.

At this point, the prototype is non-functional and mainly concentrates on the database schema, user interface mock-ups, and structural design, providing a basis for subsequent stages of development. However, managers and coordinators can review, approve, or reject pending claims with comments on an Approval Dashboard. This division of views ensures that each user interacts with only the features relevant to their role, improving the system's overall usability.

All things considered, the design decisions, database schema, and GUI layout demonstrate a careful balancing act between technical viability and user-centric design. The CMCS prototype satisfies the essential requirements of claim submission and approval processes while providing a solid foundation for further advancement by following industry best practices such as role-based access control, the MVC pattern, and structured workflows.

**Question 2: UML Class Diagram**



**Question 3: Project Plan**

The Contract Monthly Claim System (CMCS) prototype development was organized into distinct stages to guarantee methodical advancement and on-time completion. System objectives, stakeholder needs, and functional requirements are determined during the project's first phase, requirements gathering and analysis. This stage makes sure the prototype accurately depicts the actual claim submission and approval procedure. The System Design phase comes next, with an emphasis on database modelling, creating UML class diagrams, and planning GUI layouts. These deliverables directly affect later implementation tasks and lay the system's structural foundation.

After the design is complete, the project moves on to the Prototype Development stage, where the ASP.NET Core MVC framework is used to build the graphical user interface. Entity Framework Core is used to map database objects to application classes after the database schema is created in SQL Server. At this point, the focus is on creating visually accurate but non-functional screens, forms, and navigation flows. Here, dependencies are crucial because GUI development depends on a finalized UML diagram and a stable database schema.

The testing and refinement phase starts after the prototype is constructed. Even though the prototype isn't entirely working, the workflows and design components need to be checked against the specifications. This entails evaluating the GUI's usability, making sure the navigation reflects the intended workflows, and verifying that database tables and class relationships match the UML and ER diagrams. The prototype's usability and clarity are enhanced by incorporating stakeholder feedback into refinements.

The documentation and submission phase, which wraps up the project, entails polishing the written report, adding diagrams, and making sure it complies with the rubric's requirements. As the documentation compiles all deliverables into an organized and readable format, dependencies at this stage include the completion of the design, development, and testing phases.

The project is expected to take six weeks to complete. While Weeks two and three concentrate on system and database design, Week one is devoted to requirements collection and analysis. The creation of the GUI prototype takes place in Week 4, and Week 5 is set aside for testing and improvements. With documentation, diagram integration, and submission preparation, week six brings the project to a close. The completion of design must come before development, and development must come before testing, indicating a sequential dependence of tasks. By doing this, risks are reduced and each deliverable build on the one before it.

The project plan delineates distinct tasks, dependencies, and a practical six-week timeline. The strategy guarantees that the CMCS prototype can be developed methodically and successfully by breaking the work down into manageable phases with logical dependencies.

**Question 4: GUI/UI**

**Login Screen**

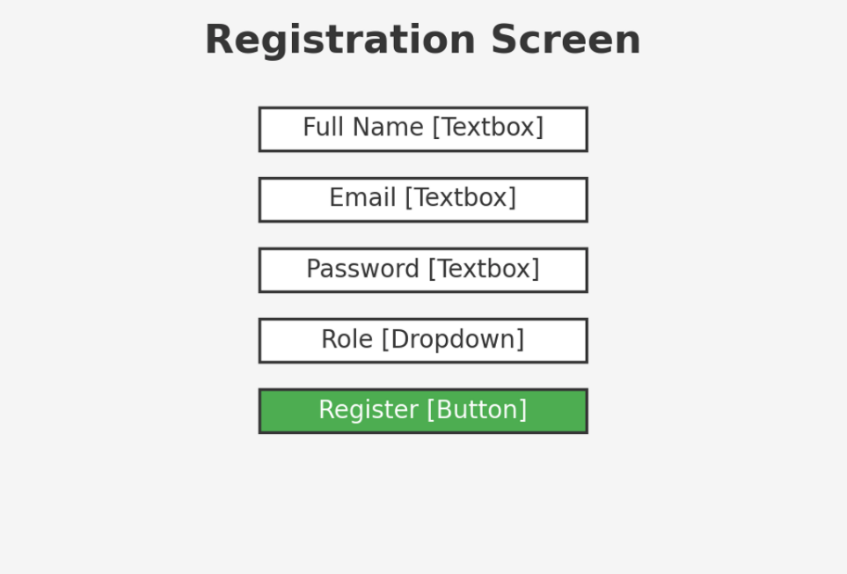
Users (managers, coordinators, or lecturers) can safely access the Contract Monthly Claim System through the login screen. It has a Login button for authentication and text boxes for the password and email. If necessary, a "Forgot Password?" option can help users reset their login information. The system is easily accessible thanks to its straightforward layout.

**A screenshot of a login screen

AI-generated content may be incorrect.**

**Registration Screen**

New users can create an account in the system through the registration screen. Important information like your full name, email address, and password are gathered. After completing the form, users can create their account by clicking the Register button. The basis for safe system access is this screen, which makes sure that only authorized instructors and staff are registered.



**Document Upload Screen**

Lecturers can efficiently submit supporting claim documents using the document upload screen. Users can upload files using the Browse button or drag-and-drop, enter the name of the document, and click Upload to submit. The system guarantees consistent and legitimate submissions by explicitly defining the permitted file types (PDF, DOCX, XLSX) and the maximum file size (5MB). This feature improves transparency and makes claim verification easier.

A screenshot of a computer

AI-generated content may be incorrect.

**Question 5: GitHub Repository**