CASSANDRA NONTOBEKO MOYO

ST10393195

PROGRAMMING 2B PART 1

PROG6212

Nhlakanipho Mavundla

09 SEPTEMBER 2024

Design Choice and Structure for Contract Monthly Claim System (CMCS)

The design choice

The MVC Design Pattern splits an application into three parts

1. Model – according to (Pham, 2023) the model handles “application data and business logic”. It is the function that separates the “user interface and the user actions”. Its functions include “data retrieval, manipulation, and validation”.
2. View - (Pham, 2023) states that this the view functions to “display the application(data) user interface to the user”.
3. Controller - (Pham, 2023) explains that the controller is the go between the model and the view. The controller “handles user input and updates the model accordingly”, any changes in the model are updated by the controller and reflected in the view.

(Pham, 2023) shares a few pros in choosing MVC as a design choice that algin with programming principles. This motivated me to choose MVC.

1. Compartmentalizing of code – this allows for an understandable, maintainable code. I believe this is training for collaboration skills. I will be able to evaluate my strong and weak areas.
2. Reusability – the MCV design model “promotes code efficiency”, “the separation of the model and view allow for reuseable at different parts of the code”.
3. Scalability – Our POE has been broken down into 3 submissions, added code and improvements made to features will not impact my entire database.
4. Although this is not a collaborative POE, in the end, I will be able to understand business requirements and fulfill front end, database and back-end code requirements.

The database structure

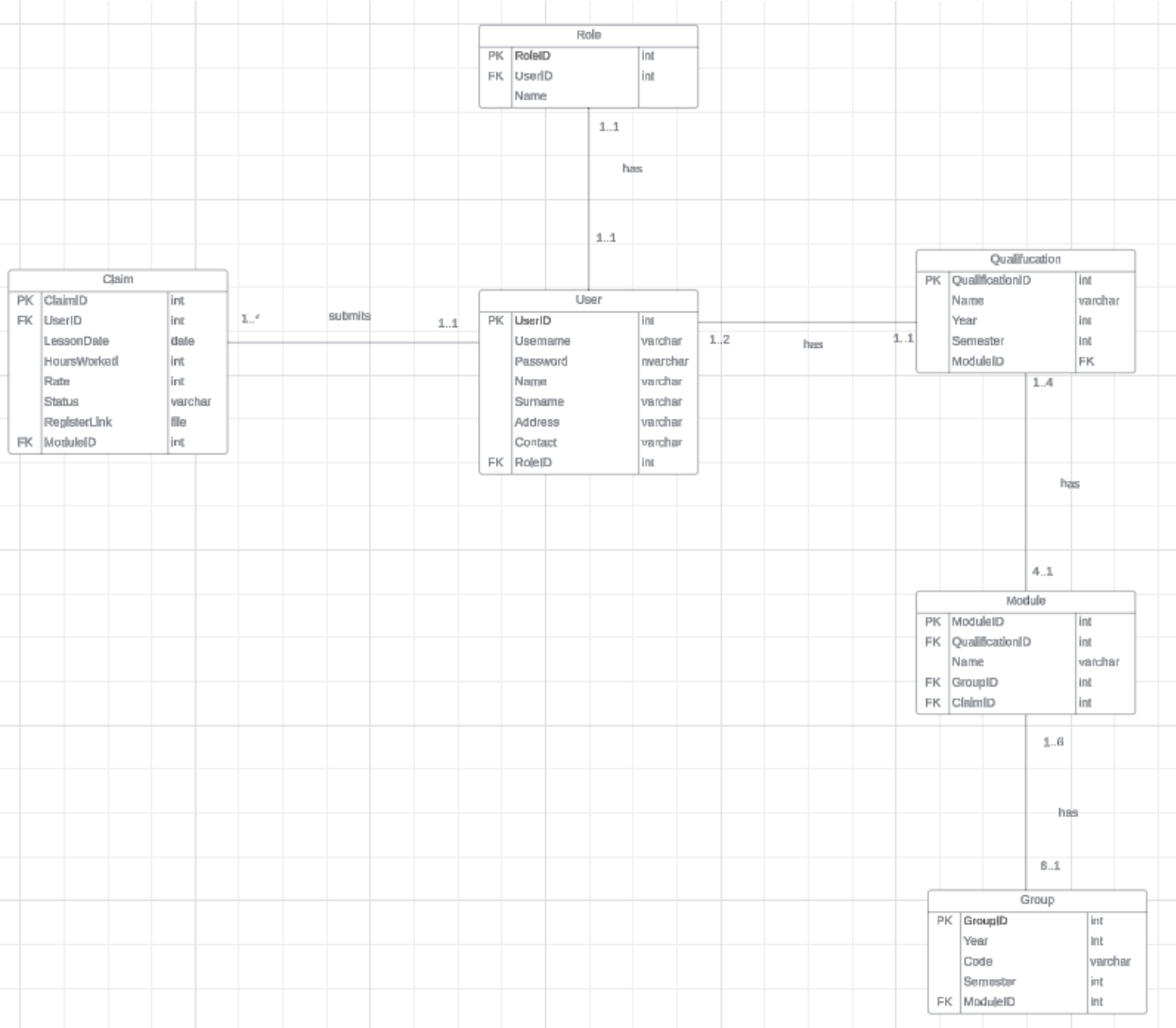
- I will be using the Entity Framework Core Database First Approach; my database will be managed by Microsoft SQL Server Management.
- After creating a .NET Core Project,
- I will install EF Core NuGet Packages (Microsoft.EntityFrameworkCore.SqlServer for SQL Server)
- Generate my models utilizing DbContext allowing for CRUD functions in database.

Assumptions and Constraints

Assumptions	Constraints
• Assumptions will be that the Independent Contractors, Programme Coordinators and Academic Managers are under the same institution. • They have received appointment letters with assigned IDs, qualifications, modules, groups and ICs are aware of their PCs. PCs are aware of AMs • The institution recognizes and gives the Program Coordinates and Academic Manger rights to verify and approve submitted claims respectively.	Integrating HR system to verify user information captured on the CMCS. At no point can an unrecognized user access the system. All information is verified against the HR system. Users must be employed by the institution to submit claims. Intergration to Payroll to process approved claims for payment
• Parties who will use the application are not agnostic and open to learning how to use the application. • Authenticates users have access to the systems, there should be a record of who made what modification to a claim. • The system will consistently provide accurate information on demand.	
• There will be strict time constraints. This process should take 7 working days.	

UML Class Diagram for Contract Monthly Claim System (CMCS)

This part of the document is used to plan out database tables and referential integrity.



Below are the entities, attribute and relationship in my database.

Entities and Attributes:

- **Role**: RoleID (PK),UserID(FK), Name
- **User**: UserID(PK), RoleID(FK), Username, Password, Name, Surname, Address, Contract, ModuleID (FK)
- **Claim**: ClaimID(PK), UserID(FK), LessonDate, HoursWorked, Rate, Status, RegisterLink, ModuleID
- **Qualification**: QualificationID (PK), Name, Year, Semester, Module (FK)
- **Module**: ModuleID (PK), Name, QualificationID (FK), ClaimID(FK), ClaimID(FK)
- **Group**: GroupID (PK), Year, Code, Semester, QualificationID (FK), ModuleID (FK)

Relationships:

- Each **User** will be assigned a role, their views and interaction with the system will depend on their role
- Each **User** can only be one of three roles Independent Contractor Lecturer, Project Coordinator, Academic Manager
- **Users** will enter their Name, Username (email address), Password, Name, Surname, Address, Contract and assigned role.
- Each **User (Independent Contractor Lecturer) can** submit many **Claims.**
- Each **Claim** will contain **ClaimID, UserID, LessonDate, HoursWorked, Rate, Status, RegisterLink, ModuleID**
- Each **Qualification** has 4 **Modules** *per 2 semesters*.
- Each **Qualification** contains QaulificationID, Name, Year, Semester, ModuleID.
- Each **Module** belongs to a **Qualification**.
- Each **Module** contain ModuleID, QualificationID, Name, GroupID, ClaimID
- Each **Module** can be assigned to many **Qualifications**.
- Each **Group** is in a certain year, doing a specific code during a certain semester under a specific **Qualification**.
- Each **Group** contains GroupID, Year, Code, Semester, ModuleID
- **Lecturers** submit a **Claim** for every lesson.
- Each **Qualification** has a **Program Coordinator** who verifies claim details.
- The **Academic Manager** approves verified claims.

Project Plan for Contract Monthly Claim System (CMCS)

1. Value Proposition

The CMCS aims to digitize the process of submitting and approving monthly claims for independent contractor lecturers. It ensures:

- Efficient claim submission and approval.
- Easy verification and approval by Programme Coordinators and Academic Managers.
- Transparent tracking of claim status.
- Consistent and reliable information.

2. Stakeholders

- My marking lecturer
- Independent Contractors
- Programme Coordinators
- Academic Managers

3. Nature of Work and Effort Required

1. Requirement Gathering: A thorough reading of the requirement detailed in the POE.
2. Design: Creating wireframes and design documents.
3. Development: Coding the frontend and backend.
4. Testing: Conducting unit tests, integration tests, and user acceptance tests.

4. Work Breakdown Structure (WBS)

TaskID	Task Description	Dependencies	Duration (Number of days)	Strat Date	End Date	Milestone
1	Information gathering	Design	2	28/08/2024	30/08/2024	Define project scope. Identify stakeholders
2	Design	Development	3	02/09/2024	08/09/2024	Creating wireframes and design documents
3	Development	Testing		10/09/2024	10/10/2024	Coding the frontend, database and backend
4	Testing	Deployment		15/10/2024	24/10/2024	Conducting unit tests, integration tests, and user acceptance tests.
5	Deployment	Final Submission		01/11/2024	22/11/2024	Submit final POE a

5. Resource Allocation

- Human Resources: Myself(student)
- Technical Resources: Figma for wire frames Development tools (Visual Studio, .NET framework), Testing tools, possible servers for deployment.

Contract Monthly Claim System (CMCS) GUI Layout

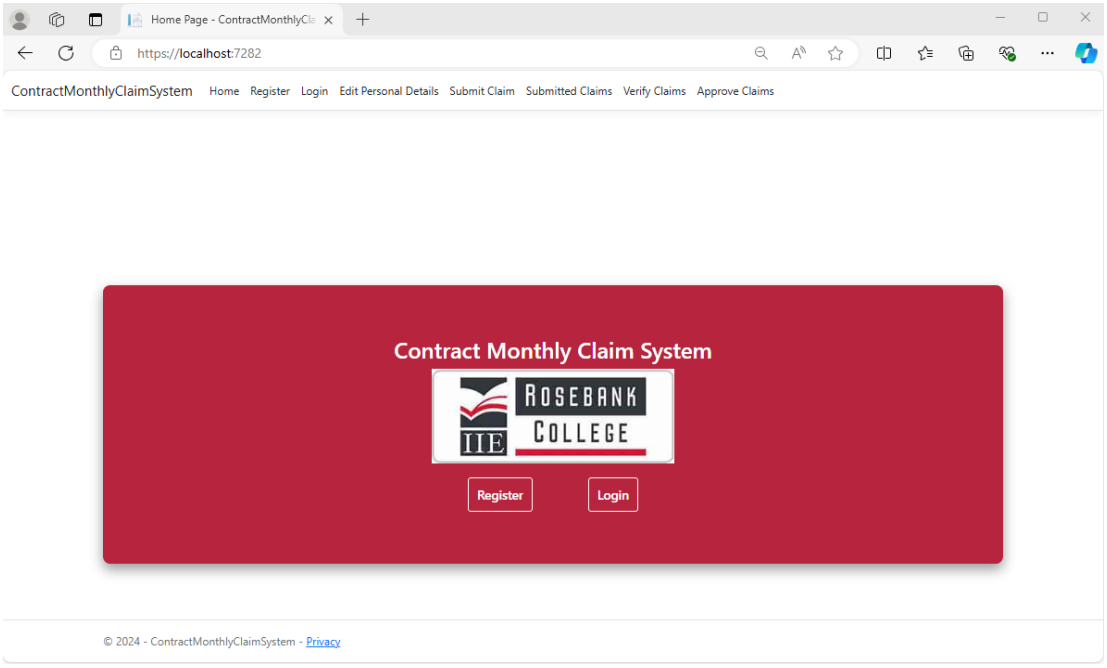
This part of the documentation will give context to how the user will interact with the CMC system.

Color palette #32373b #b6243d #303136

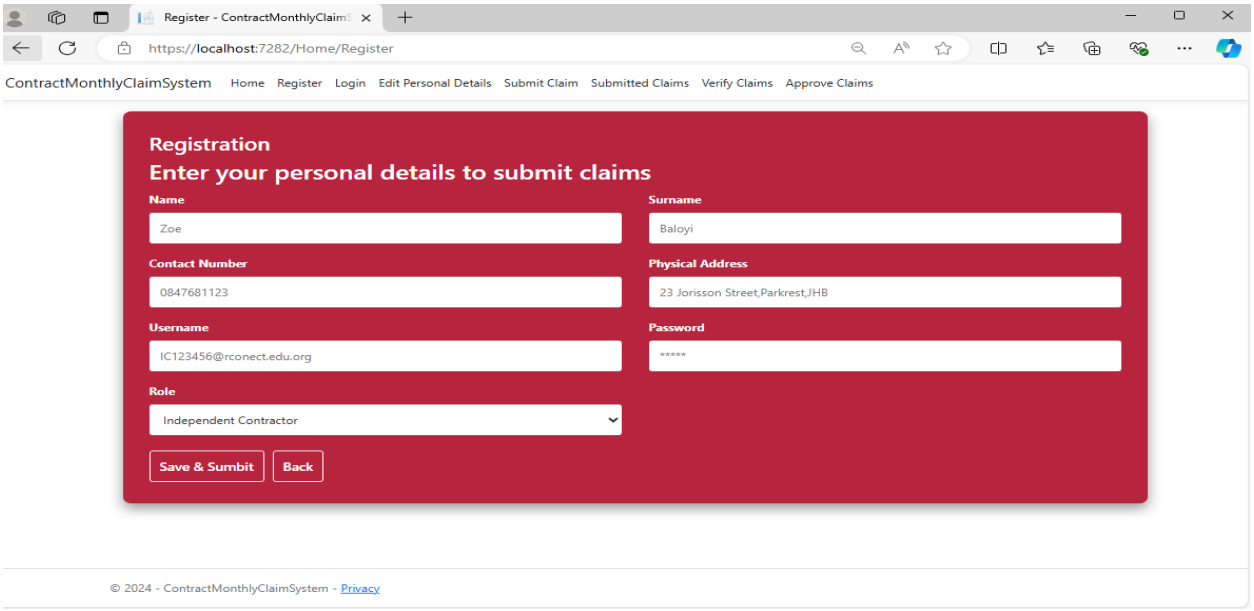
This will be a timed system allowing for efficient claim submission.

User authorization from Action Filters will assist in tracking user interaction with the system. (applied to a controller actions) (Rick-Anderson, 2022).

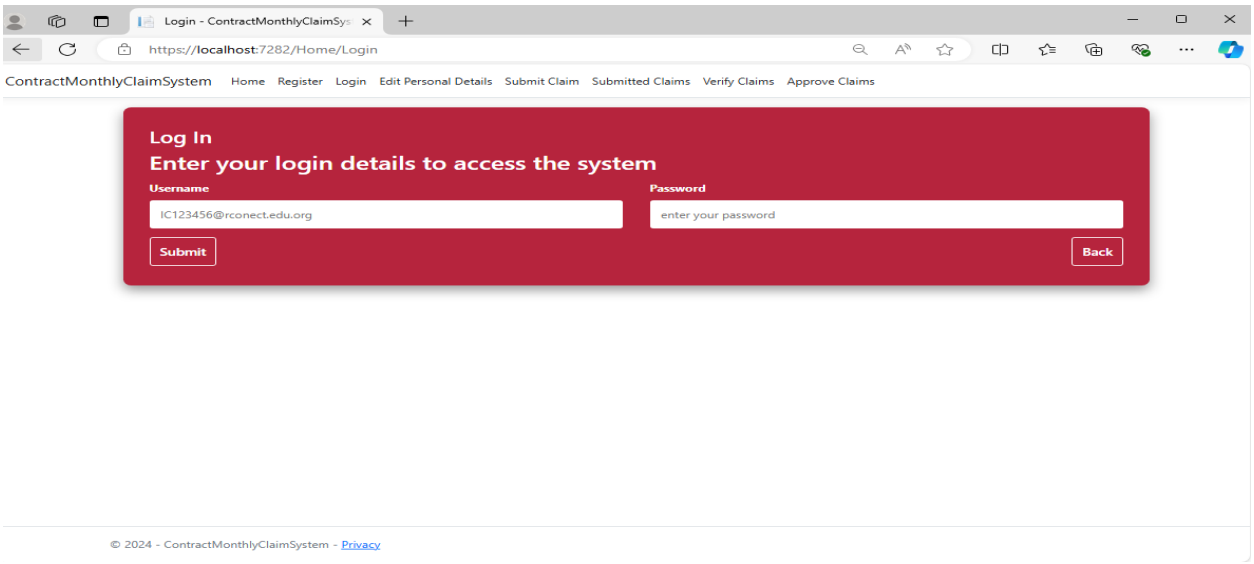
The Landing page with the login and register button for users to create a profile that will enable them to use the system.



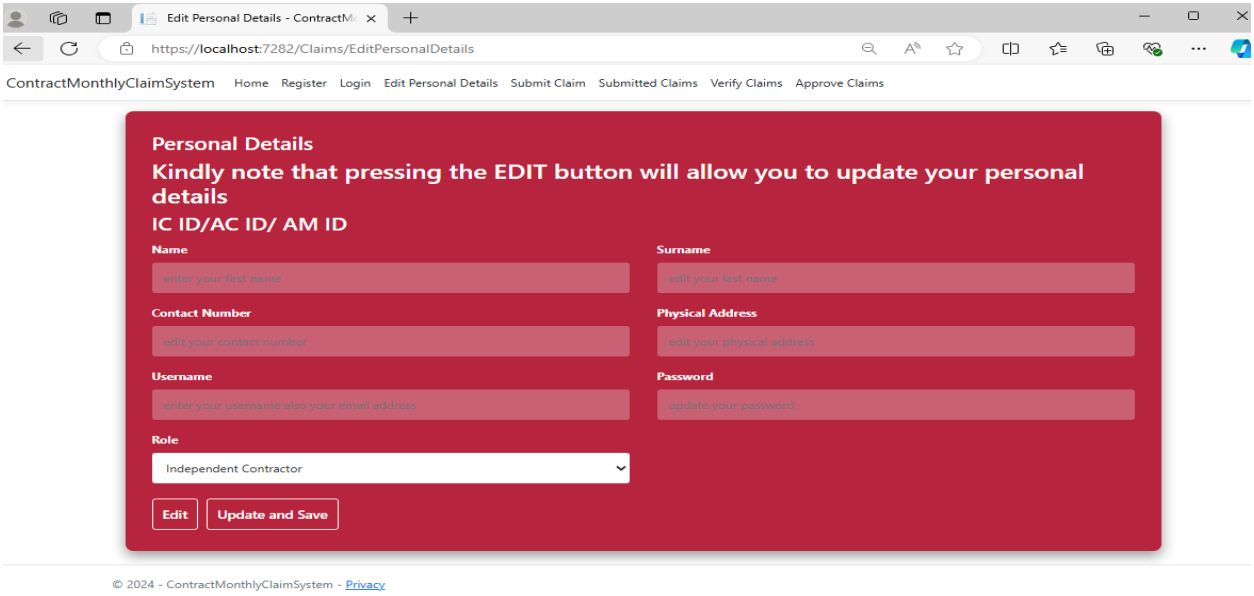
The Register form for that will capture use details, users will need to enter/confirm their role to access functionality.



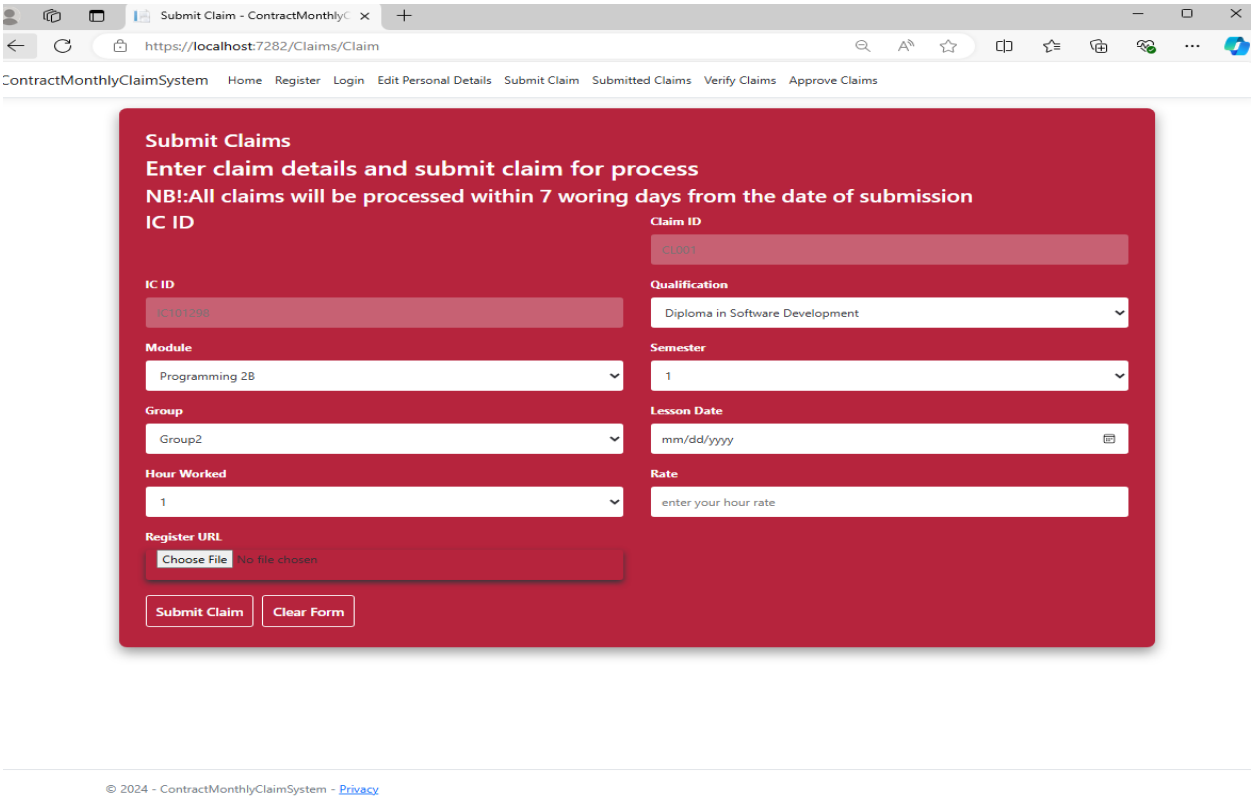
The Login form once user details have been verified.



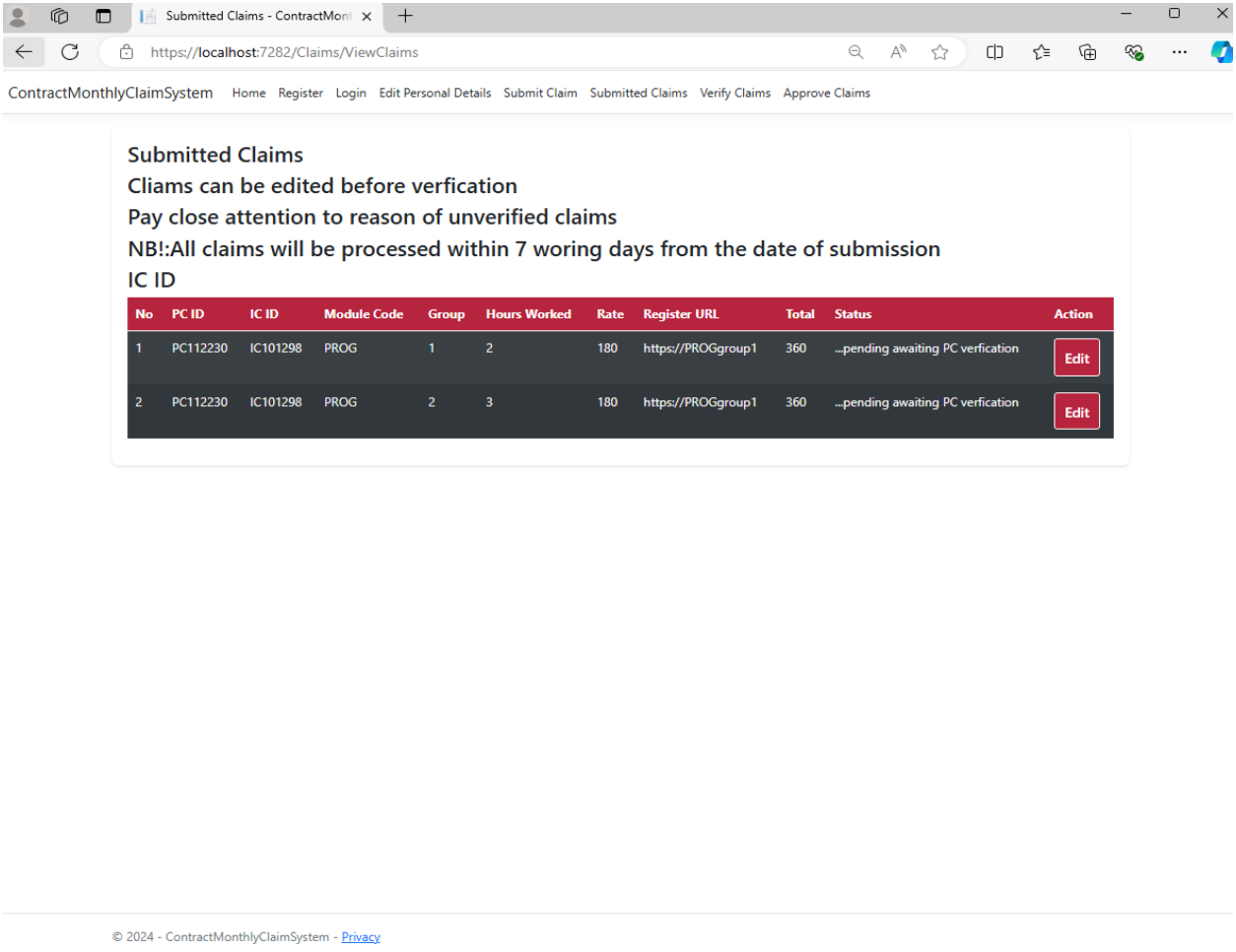
This is the Edit form for users who will access this form should they wish to update their details.



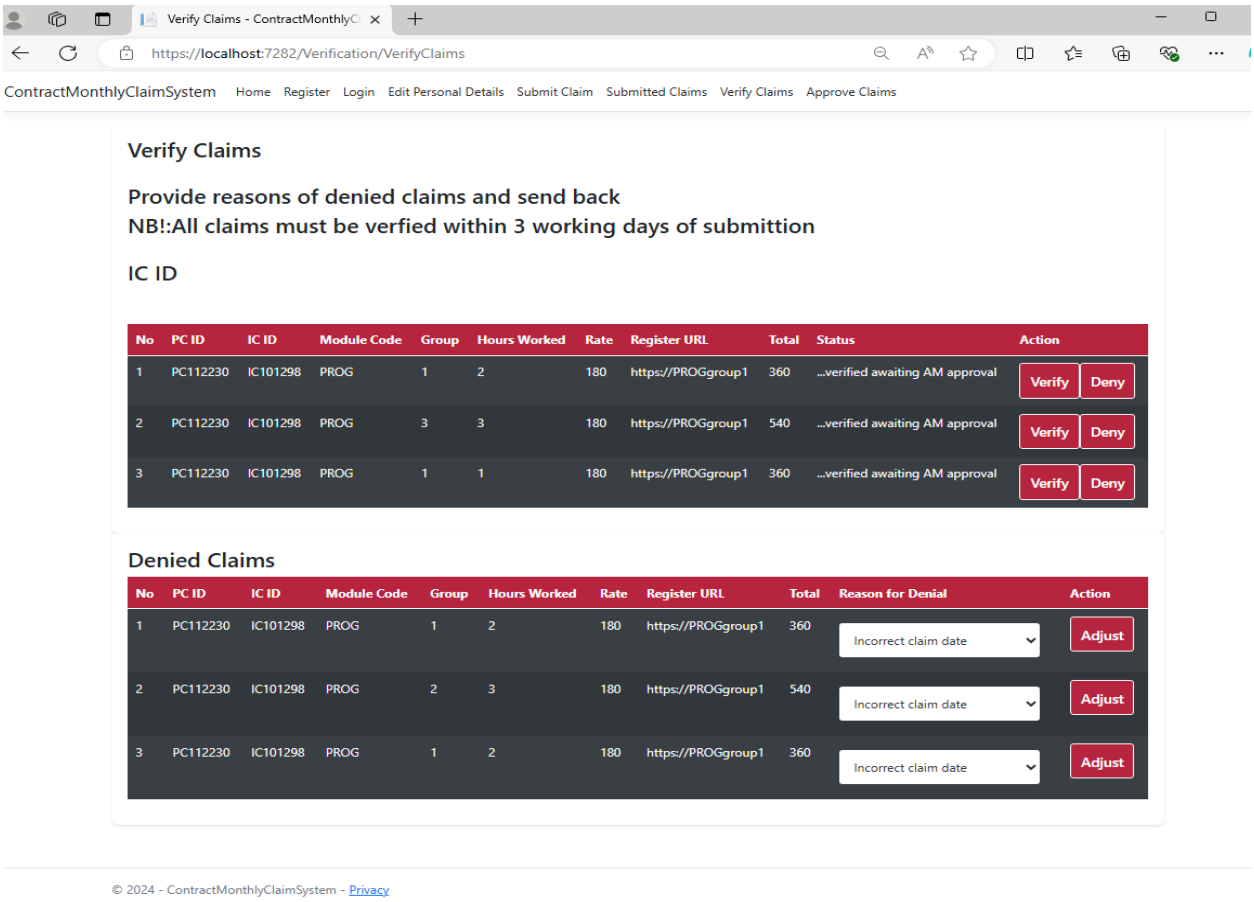
The Claim form, ICs will see their assigned ID, qualifications, programs, modules, groups as per their appointment letters. ICs enter their hourly rate and attach their register URLs.



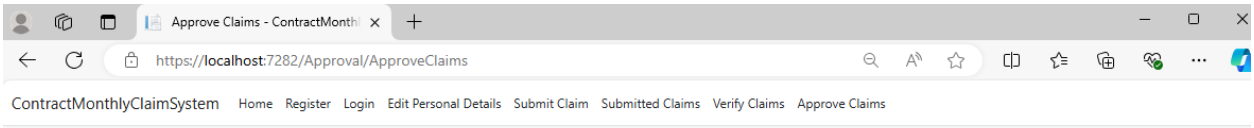
ICs will view their history of submitted claims. ICs will be able to edit claims within a specified timeframe.



PCs will verify claims against submitted registers. Decline claims will be sent back to ICs with reasons as a guide on what to adjust on claim.



The AM must only approve claims ,there should be no mistakes on claims at this stage.



Approve Claims

NB!:All claims must be approved within 2 working days of verification submittions

AM ID

No	AM ID	PC ID	IC ID	Module Code	Group	Hours Worked	Rate	Register URL	Total	Status	Action
1	AM2034761	PC112230	IC101298	PROG	1	2	180	https://PROGgroup1	360	...pending waiting your approval	Approve
2	AM2034761	PC112230	IC101299	PROG	1	3	180	https://PROGgroup1	540	...pending waiting your approval	Approve
3	AM2034761	PC112233	IC101296	DATA	1	1	180	https://PROGgroup1	180	...pending waiting your approval	Approve
4	AM2034761	PC112230	IC101288	PROG	1	2	180	https://PROGgroup1	360	...pending waiting your approval	Approve

Reference List

Pham, L. (2023). *The MVC Design Pattern: A Timeless Approach to Web Development*. [online] Medium. Available at: https://medium.com/@phamtuanchip/the-mvc-design-pattern-a-timeless-approach-to-web-development-c132ff3afd37.

Rick-Anderson (2022). *Understanding Action Filters (C#)*. [online] learn.microsoft.com. Available at: https://learn.microsoft.com/en-us/aspnet/mvc/overview/older-versions-1/controllers-and-routing/understanding-action-filters-cs.