ONTLAMETSE MOGAPI

PROG6212 POE PART 2

ST10244738

Documentation and screenshots

**Project Overview**

The **Contract Monthly Claim System (CMCS)** is a .NET Core-based web application designed to manage the submission and approval of monthly claims from independent contractor lecturers. The system streamlines the process by allowing lecturers to submit claims, and Programme Managers (PMs) to approve or reject them. It includes functionality for tracking claim status, viewing submitted claims, and providing feedback for rejected claims.

**Application Pages Overview**

Here is a detailed description of the different pages and functionalities in your CMCS application:

**1. Home Page (Home/Index)**

* **Purpose**: The landing page of the application, typically providing navigation links for different users (e.g., lecturers and Programme Managers).
* **Key Features**:
  + Displays links for submitting new claims (for lecturers) or viewing pending claims (for PMs).
  + Contains buttons for navigating to different functionalities (submit claims, view claims, etc.).

**2. Submit Claim Page (Home/SubmitClaim)**

* **Purpose**: Allows lecturers to submit their monthly claims, including information such as the amount, date submitted, and any necessary documentation.
* **Key Features**:
  + A form where lecturers enter their details (e.g., name, amount, and supporting documents).
  + On submission, the claim is saved in the database with a status of "Pending" for further action by the PM.
  + Data validation to ensure proper submission (e.g., required fields, amount formats).

**3. Approve Claims Page (Home/ApproveClaims)**

* **Purpose**: Allows Programme Managers to view all claims that are pending approval and decide whether to approve or reject them.
* **Key Features**:
  + Displays a list of all submitted claims with key information (lecturer name, amount, submission date, etc.).
  + Provides action buttons (Approve/Reject) next to each claim.
  + Once a decision is made, the claim's status is updated in the database to either "Approved" or "Rejected", along with the date and comments from the PM.

**4. View Claim Page (Home/ViewClaim)**

* **Purpose**: Allows lecturers to view the status of the claims they submitted.
* **Key Features**:
  + Displays the claim details such as lecturer name, amount, date submitted, and current status.
  + If a claim is rejected, a reason or comment from the PM will be visible.
  + Claim history is stored, showing when the claim was approved or rejected.

**5. Claim Submission Form (Home/SubmitClaimForm)**

* **Purpose**: The form where lecturers fill in the details to submit a new claim.
* **Key Features**:
  + Includes fields for entering the lecturer's name, date, amount, and uploading any relevant supporting documentation.
  + The claim is saved as a "Pending" entry upon submission.

**Database Structure**

The application's database is structured to store the following information:

1. **Claims Table**:
   * ClaimId: Unique identifier for each claim.
   * LecturerName: Name of the lecturer submitting the claim.
   * DateSubmitted: Date when the claim was submitted.
   * Amount: Amount of the claim.
   * Status: Status of the claim (e.g., Pending, Approved, Rejected).
   * ApprovalDate: Date when the claim was approved or rejected.
   * ProgrammeManagerComments: Any comments provided by the Programme Manager upon approval or rejection.

**Functionality Description**

1. **Lecturer Workflow**:
   * Lecturers log in to the system and submit their claims via the **Submit Claim Page**.
   * They can view the status of their claims on the **View Claim Page**.
   * If a claim is approved, the status will update to "Approved". If it is rejected, the lecturer will see the PM's comments explaining the rejection.
2. **Programme Manager Workflow**:
   * PMs log in and navigate to the **Approve Claims Page** to see all pending claims.
   * PMs can approve or reject claims, providing comments if needed. The system updates the claim's status accordingly.

**Explanation of Tests**

**Test Initialization:**

* A mock logger is created to pass into the HomeController.
* The controller instance is created for each test.

**Tests:**

* Each test checks a specific action in the HomeController.
* The tests use assertions to verify the action returns the expected type (like ViewResult or RedirectToActionResult).
* The user registration and login tests simulate the behavior of the insert\_user and login\_user methods to confirm the controller logic works as expected.

**Conclusion**

The **Contract Monthly Claim System** is designed to efficiently manage the submission, review, and approval of claims for independent contractor lecturers. The system ensures transparency, quick feedback, and easy access to claim statuses. The goal is to make the process of handling claims faster and more user-friendly, benefiting both lecturers and Programme Managers.

**Screenshots**







