Contract Monthly Claim System

Streamlining Lecturer Claim Submissions & Approvals

PROG6212 Final Project Presentation

Presented by Naoyuki Christopher Higaki

Project Overview: What is the CMCS?

Our vision is to create a web-based platform that fully automates the monthly claim process for contract lecturers. This system addresses the inefficiencies of manual, paper-based workflows.

Problem Solved:

The CMCS replaces a cumbersome, error-prone manual process with a digital, efficient, and transparent workflow.

Core Value:

The system saves significant time, drastically reduces administrative errors, and provides full transparency throughout the claims lifecycle.

1	2	3
Lecturers Submit claims and track their status in real-time.	**Programme Coordinators** Perform first-level review and approval of claims.	**Academic Managers** Provide final approval and maintain oversight of all

System Architecture & Technology

The Contract Monthly Claim System is built on a modern and robust technical stack, ensuring reliability and scalability.

Framework

ASP.NET Core MVC provides a powerful foundation for building dynamic web applications.

Language

C# is utilized for its strong typing, performance, and extensive libraries within the .NET ecosystem.

Data Storage

A JSON-based Text File System was implemented to meet a specific project requirement, demonstrating adaptability beyond traditional databases.

Architecture

The Model-View-Controller (MVC) Pattern ensures clear separation of concerns and maintainable code.

Frontend

HTML, CSS (with Grid/Flexbox), and JavaScript deliver a responsive and intuitive user interface.

Authentication

Robust, Session-Based with Role Management provides secure access control tailored to user roles.

Part 1: Core Prototype (4 Weeks)

Establishing the Foundation

The initial phase focused on building the foundational elements and core functionality of the CMCS.

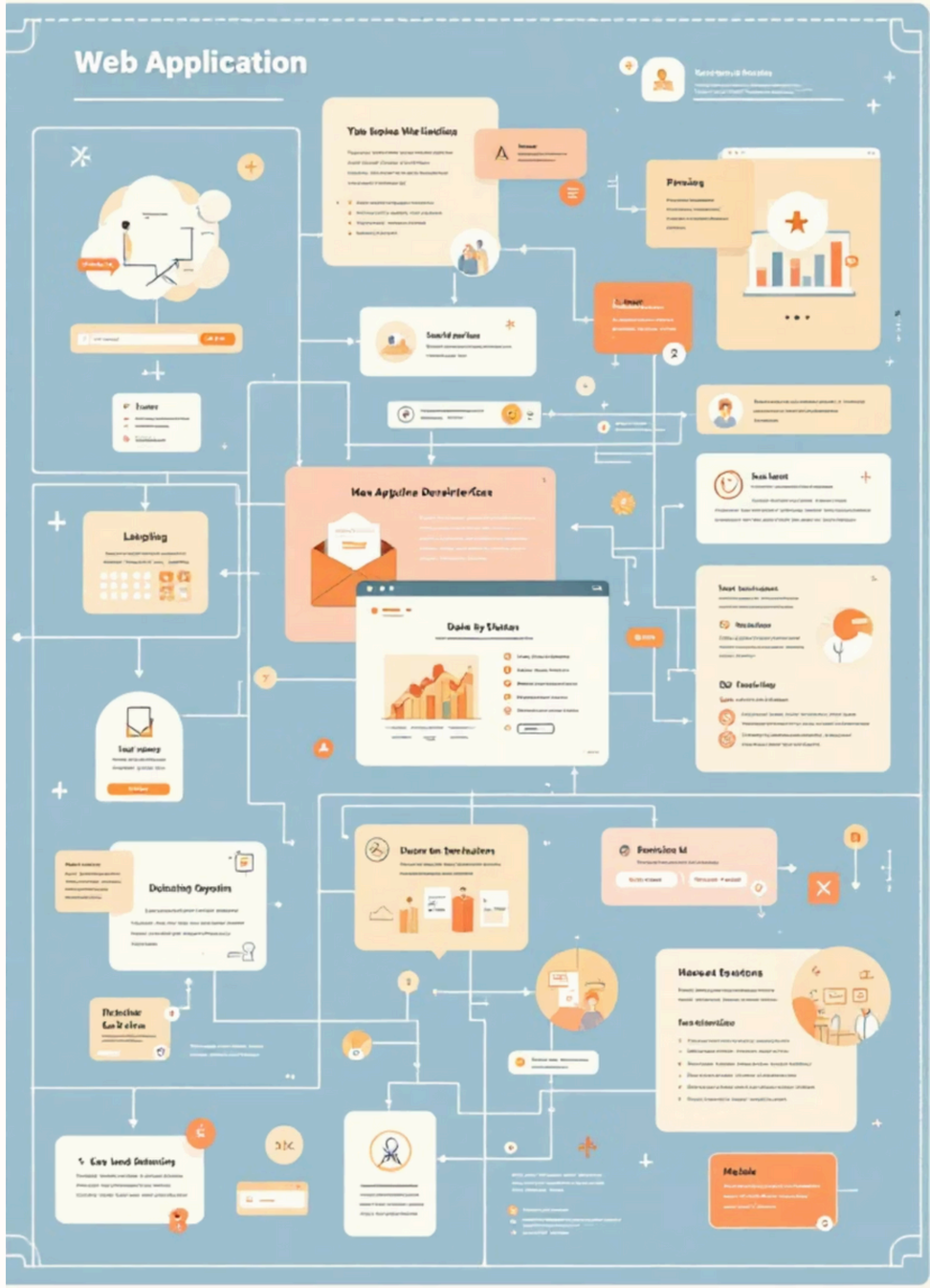
1 Weeks 1-2: Requirements & Design

- Detailed analysis of lecturer and administrative needs.
- Creation of a comprehensive UML Class Diagram to map system components.
- Initial wireframing and user flow definition.

2 Weeks 3-4: Core Development

- Setup of the ASP.NET Core MVC Framework.
- Implementation of secure Session Authentication mechanisms.
- Development of the basic claim submission and approval workflow.

This phase successfully delivered a functional prototype, demonstrating the essential user journey and validating the system's core concept.



Part 2: Feedback & Enhancement (3 Weeks)

Addressing Feedback | Grade: 99%

Following the initial prototype, this phase involved critical enhancements based on feedback, refining the system's robustness and user experience.

- ✅ **Database Integration:**
  Successfully replaced with a fully functional text-file data layer, demonstrating efficient data handling without a traditional DBMS.
- ✅ **Document Upload:**
  Implemented secure file handling with robust validation for file types and sizes, preventing unauthorized uploads.
- ✅ **UI/UX Refinement:**
  The user interface was refined with a minimalist design, focusing on clarity and ease of use.
- ✅ **Error Handling:**
  Comprehensive client-side and server-side validation were added, significantly improving system stability and user feedback.

The result is a robust, high-fidelity application that not only met but exceeded all specified requirements, achieving an outstanding grade of 99%.

Part 3: The Roadmap (3 Weeks)

Future Development Plan

My future development plan outlines the next steps to evolve the CMCS into a comprehensive and production-ready system.

Week 8: Advanced Features

- Develop a Reporting Dashboard for Managers, offering key insights.
- Implement Enhanced Data Filtering & Export functionalities.
- Integrate an In-App Notification System for timely updates.

Week 9: Security & Performance

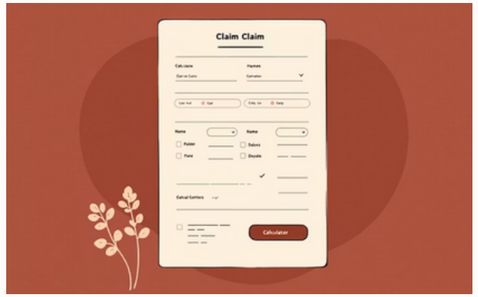
- Integrate BCrypt Password Hashing for enhanced security.
- Optimize File I/O and implement Caching strategies for performance.
- Prepare the system for seamless deployment to a live environment.

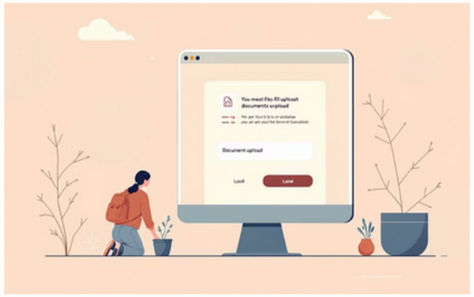
Week 10: Polish & Deployment

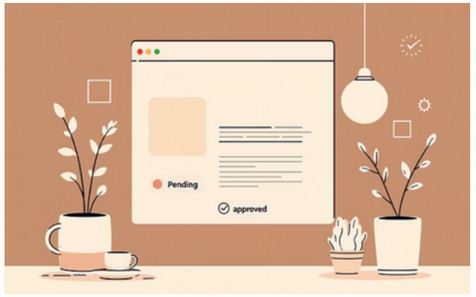
- Conduct Comprehensive Testing across all modules.
- Finalize all project documentation for future reference.
- Execute Production Deployment, making the system live.

Key Features Demonstrated

A Glimpse of the Application in Action







Role-Based Dashboard

Tailored homepages providing relevant information for each user type.

Dynamic Claim Form

Automated calculation of claim amounts based on hours and rates.

Document Upload

Secure multiple file uploads with robust type and size validation.

Dual-Level Approval

Streamlined workflows for Programme Coordinators and Academic Managers.

Status Tracking

Clear visual indicators for claim statuses: Pending, Approved, Rejected.

Fully Responsive Design

Seamless functionality and optimal viewing experience across all devices.

Technical Challenges & Solutions

Developing the CMCS presented several technical hurdles, each overcome with innovative and strategic solutions.

Challenge: Text File as a Database

A key requirement was to avoid traditional database systems. Our solution involved developing a custom TextFileDataService class.

Using JSON serialization for all CRUD operations, we ensured data integrity and efficient persistence without a full DBMS.

Challenge: Session State Management

Maintaining secure and consistent session state across various user roles was complex.

We implemented custom session extensions and rigorous role-based checks, creating a secure and seamless user experience.

Challenge: File Upload Security

Ensuring the security and integrity of uploaded documents was paramount.

Our solution included server-side validation, GUID-based renaming to prevent conflicts, and secure storage outside the web root.

Learning Outcomes & Conclusion

Summary & Reflection on the CMCS Project

This project has been an invaluable learning experience, fostering both technical proficiency and professional growth.

Technical Skills Enhanced:

- Deepened understanding of the ASP.NET Core MVC lifecycle and its advanced features.
- Mastered the development of custom data persistence layers using non-traditional storage methods.
- Significantly improved skills in UI/UX design principles and client-side integration techniques.

Professional Growth:

- Gained hands-on experience in interpreting diverse user feedback and translating it into functional features.
- Developed the ability to adhere to a structured, agile project timeline, managing iterations effectively.
- Enhanced problem-solving abilities when faced with complex technical constraints.

In conclusion, the Contract Monthly Claim System is a successfully delivered, fully-functional, and well-architected web application. It not only proves the core concept but also stands ready for future development and deployment.

Thank You!

Questions & Discussion

We appreciate your time and attention to my project. We are now open to any questions you may have about the Contract Monthly Claim System.

Thank you for evaluating my final project.