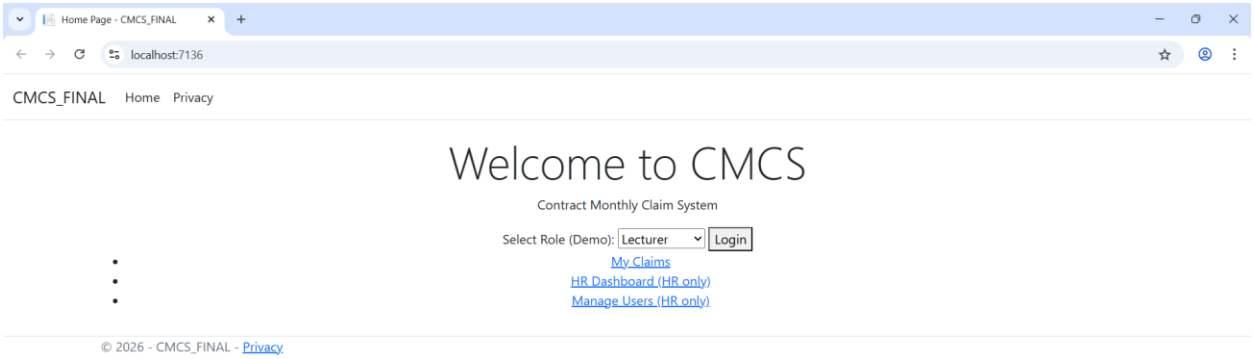
Name/Surname: **JADEEL KISTEN**

Student Number: **ST10339718**

Module Code: **PROG6212**

POE -Part Three Resubmission



Page 2: **Table of Contents**

Page 3: **System Architecture**

Contract Monthly Claim System (CMCS) Documentation

Design Choice Explanation for Contract Monthly Claim System (CMCS)

System Architecture

The chosen architecture is ASP.NET Core MVC with .NET 8.0. MVC separates concerns: Models handle data, Views manage UI, and Controllers process logic. This is beneficial for CMCS as it promotes maintainability, scalability, and testability – allowing independent updates to claim submission (Controllers), data models (Models), or user interfaces (Views).

Model Layer:

Handles data entities and business logic. Includes Claim, Lecturer, Contract, SupportingDocument, and Approval models with validation attributes (e.g., [Required], [Range]). Uses InMemoryDataService for data persistence instead of a database, ensuring quick prototyping and no external dependencies.

View Layer:

Renders user interfaces using Razor pages. Includes polished views like Claims/Index (table with badges/icons), Create (form with validation), and HR dashboard. Enhanced with Bootstrap 5 for responsive design, cards, alerts, and icons.

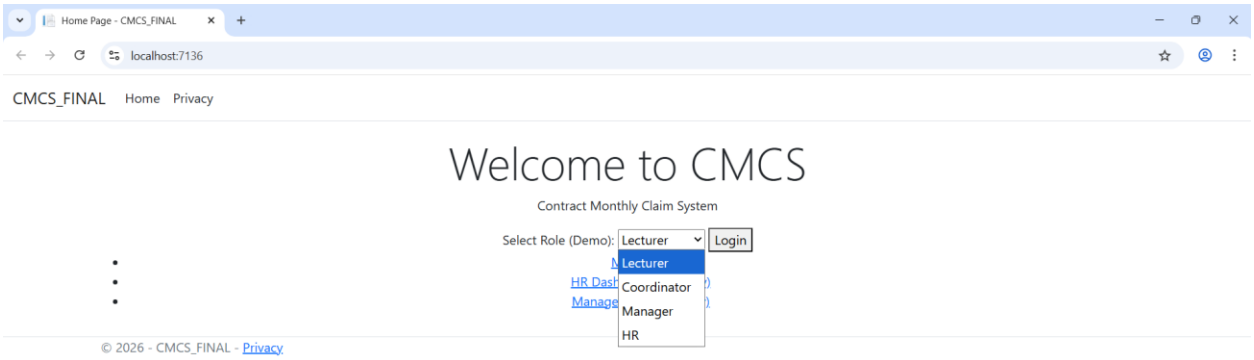
Controller Layer:

Manages requests and responses. Includes ClaimsController for submission/validation/approval, HRController for reporting/user management. Uses dependency injection for services like FileEncryptionService (AES encryption).

Page 4: **GUI Layout Explanation**

GUI Layout Explanation for Contract Monthly Claim System (CMCS)

Login View



(Screenshot of Home/Index with role selection dropdown)

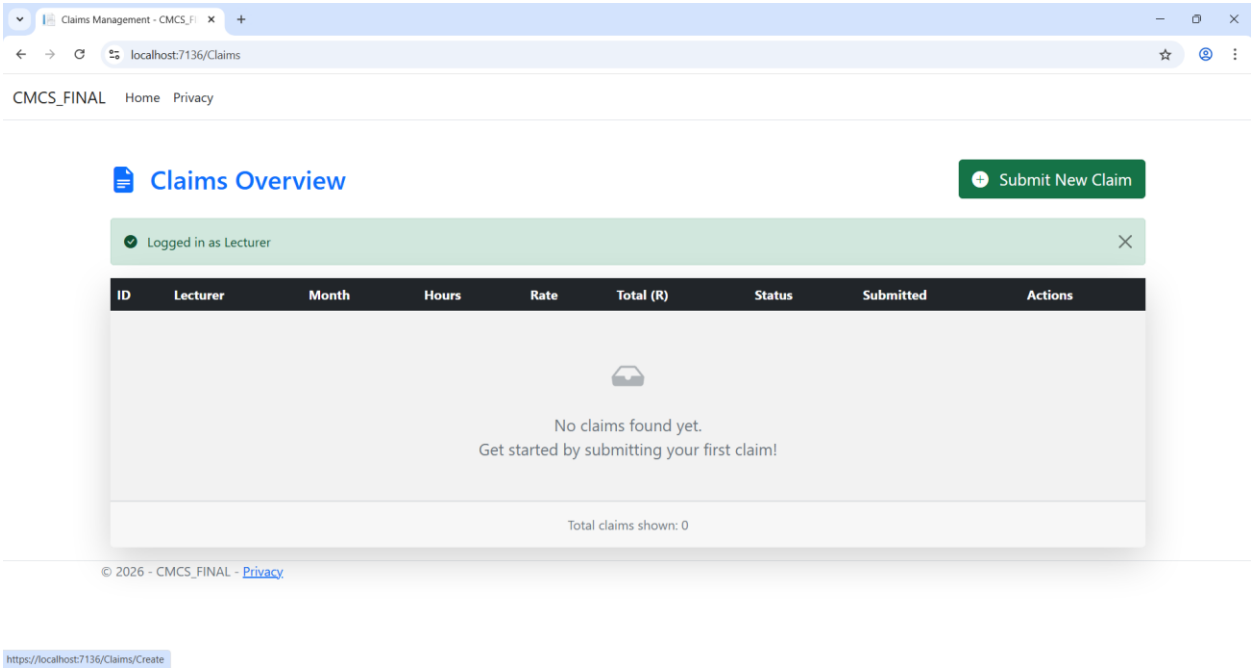
How the layout enhances user experience:

Clean card-based design with centered form, Bootstrap styling for mobile-responsiveness, and simple role selection. Icons and alerts provide visual feedback, making it intuitive and easy to use.

How the layout supports system functionality:

Supports role-based access control (RBAC) by simulating login via session, directing users to appropriate dashboards (e.g., Lecturer to claims submission).

Claims Submission View



(Screenshot of Claims/Create form)

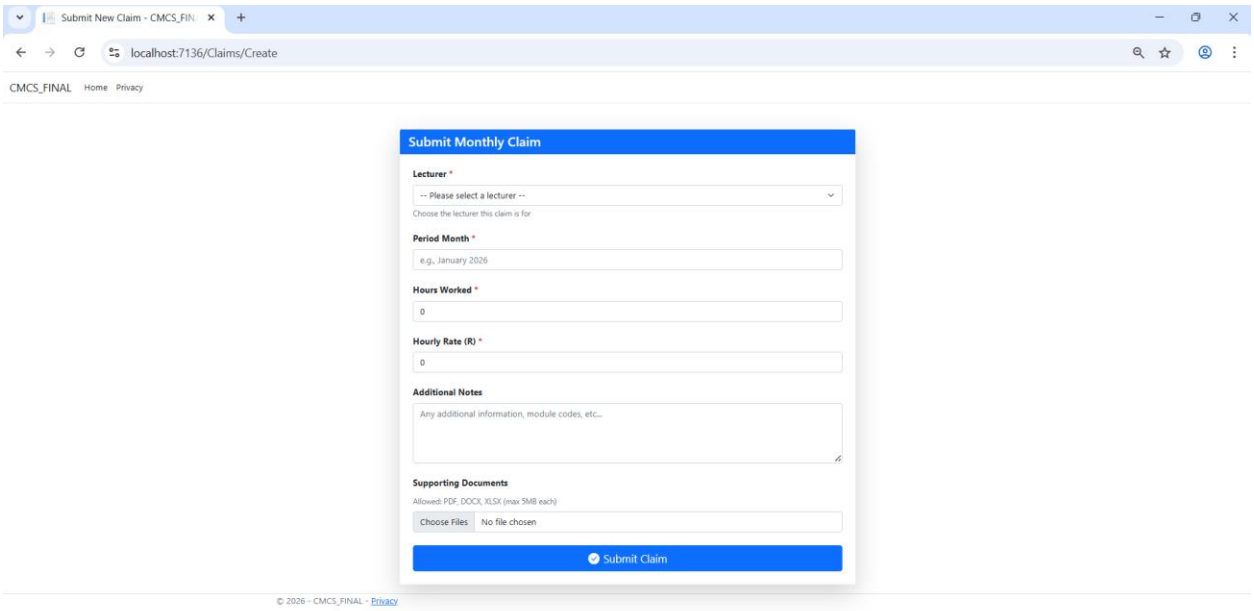
How the layout enhances user experience:

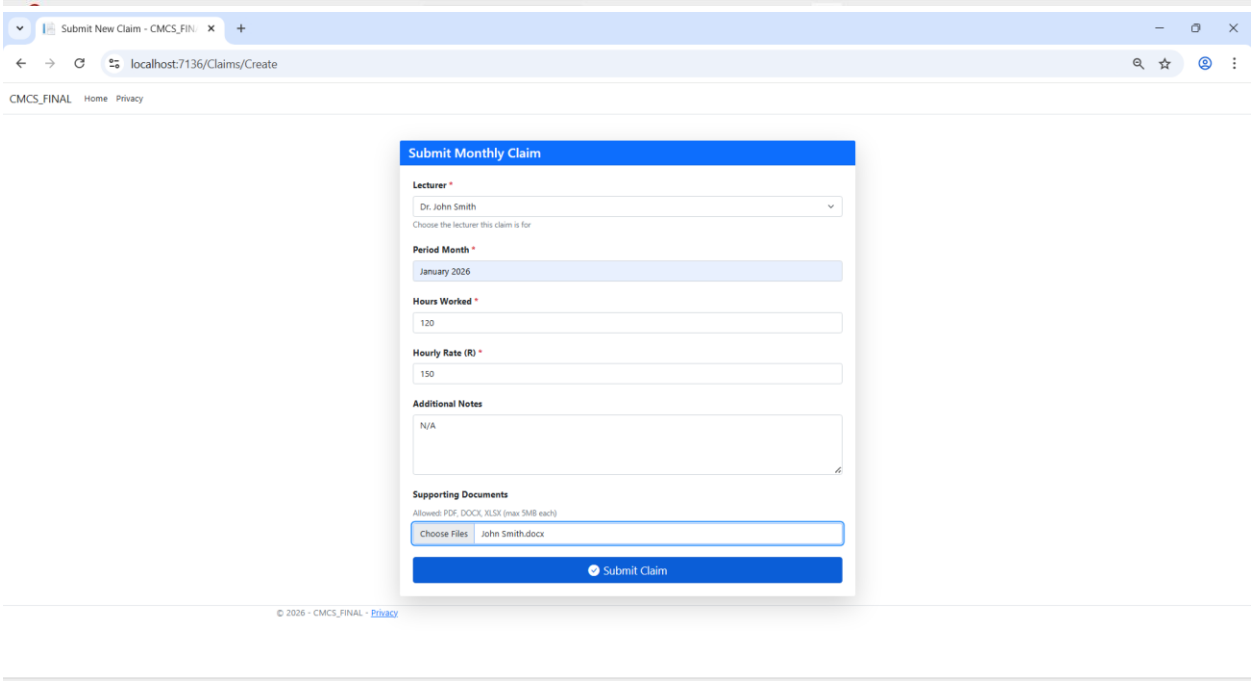
Card layout with required field indicators (*), real-time validation hints, and file upload previews. Success/error alerts guide users, reducing frustration.

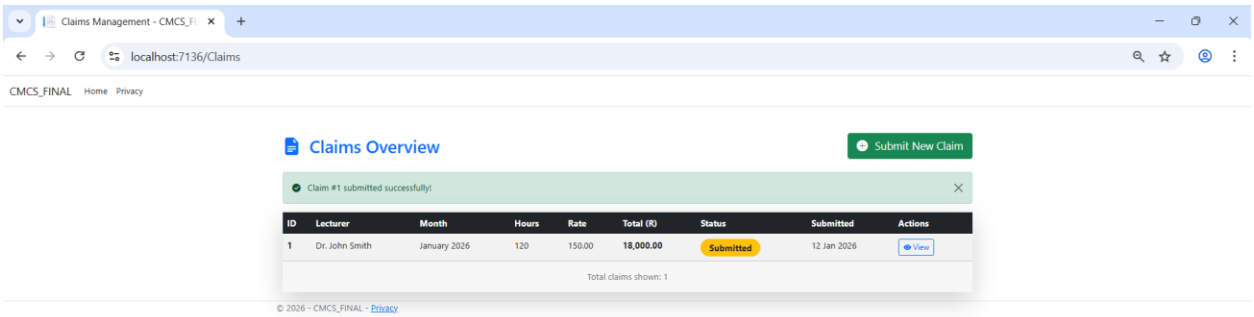
How the layout supports system functionality:

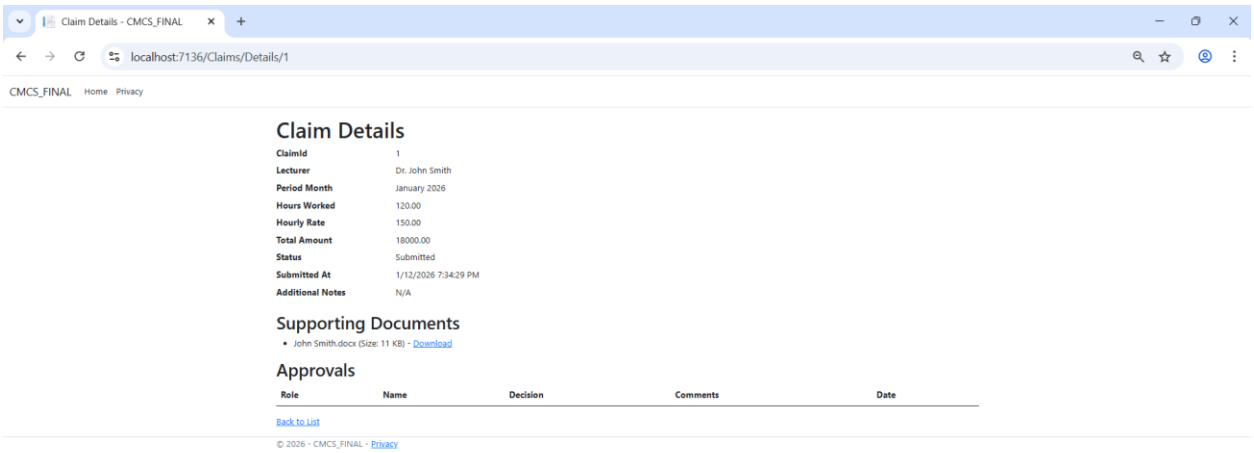
Enables claim submission with auto-calculation, document encryption, and validation – core to lecturer workflow.

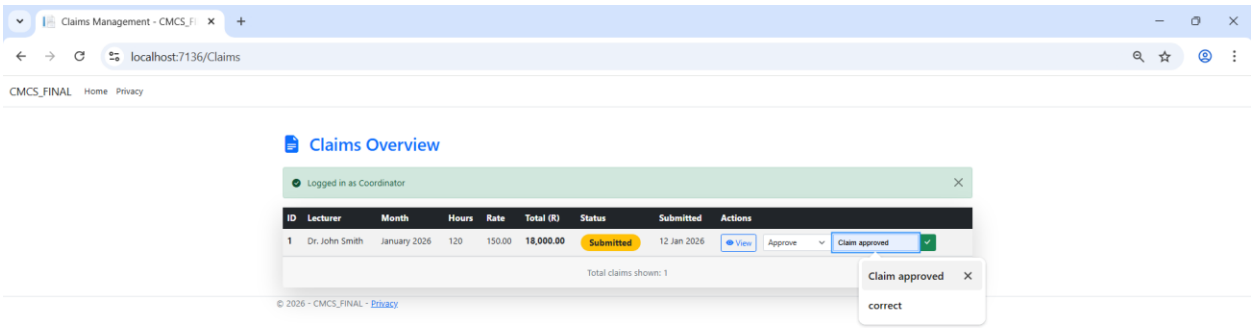
Claims Overview View

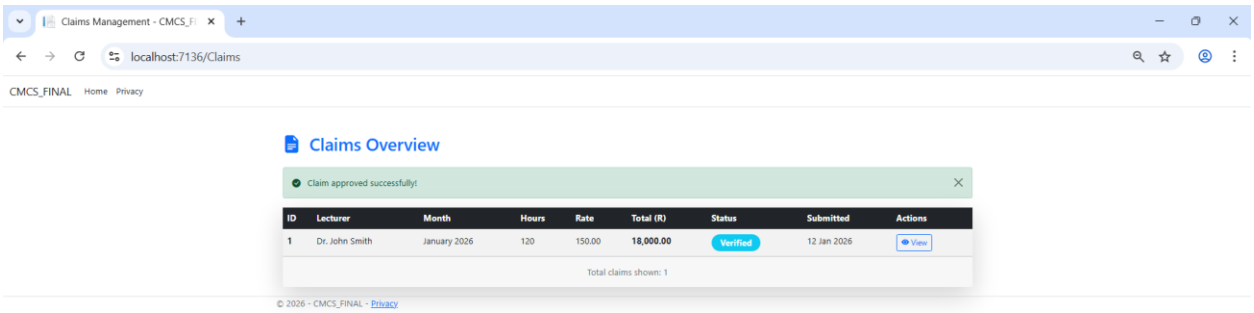

Submit New Claim - CMCS_FIN
localhost:7136/Claims/Create
CMCS_FINAL Home Privacy
Submit Monthly Claim
Lecturer *
-- Please select a lecturer --
Choose the lecturer this claim is for
Period Month *
e.g., January 2026
Hours Worked *
0
Hourly Rate (R) *
0
Additional Notes
Any additional information, module codes, etc...
Supporting Documents
Allowed: PDF, DOCX, XLSX (max 5MB each)
Choose Files
No file chosen
Submit Claim
© 2026 - CMCS_FINAL - Privacy

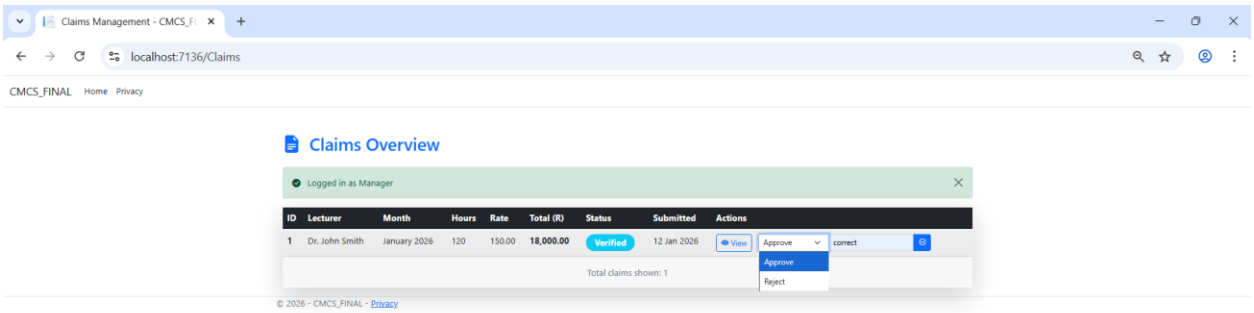

Submit New Claim - CMCS_FIN
localhost:7136/Claims/Create
CMCS_FINAL Home Privacy
Submit Monthly Claim
Lecturer *
Dr. John Smith
Choose the lecturer this claim is for
Period Month *
January 2026
Hours Worked *
120
Hourly Rate (R) *
150
Additional Notes
N/A
Supporting Documents
Allowed: PDF, DOCX, XLSX (max 5MB each)
Choose Files
John Smith.docx
Submit Claim
© 2026 - CMCS_FINAL - Privacy

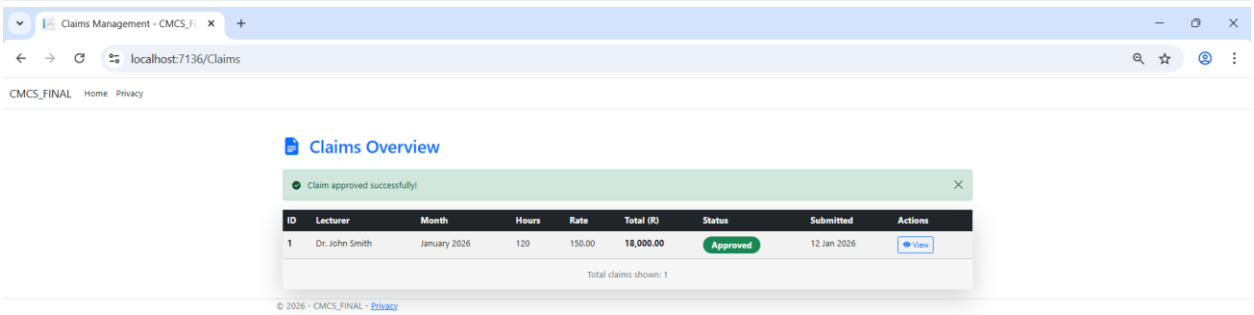

Claims Management - CMCS_FI
localhost:7136/Claims
CMCS_FINAL Home Privacy
Claims Overview
Submit New Claim
Claim #1 submitted successfully!
ID Lecturer Month Hours Rate Total (R) Status Submitted Actions
1 Dr. John Smith January 2026 120 150.00 18,000.00 Submitted 12 Jan 2026 View
Total claims shown: 1
© 2026 - CMCS_FINAL - Privacy

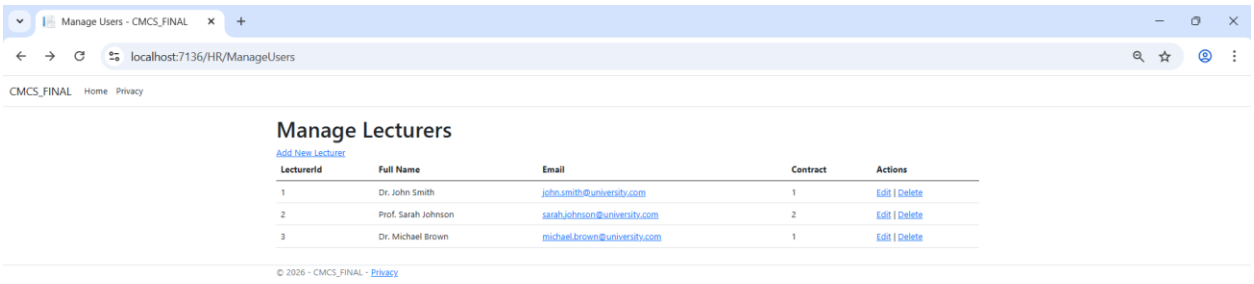

Claim Details - CMCS_FINAL
localhost:7136/Claims/Details/1
CMCS_FINAL Home Privacy
Claim Details
ClaimId 1
Lecturer Dr. John Smith
Period Month January 2026
Hours Worked 120.00
Hourly Rate 150.00
Total Amount 18000.00
Status Submitted
Submitted At 1/12/2026 7:34:29 PM
Additional Notes N/A
Supporting Documents
John Smith.docx (Size: 11 KB) - Download
Approvals
Role Name Decision Comments Date
Back to List
© 2026 - CMCS_FINAL - Privacy

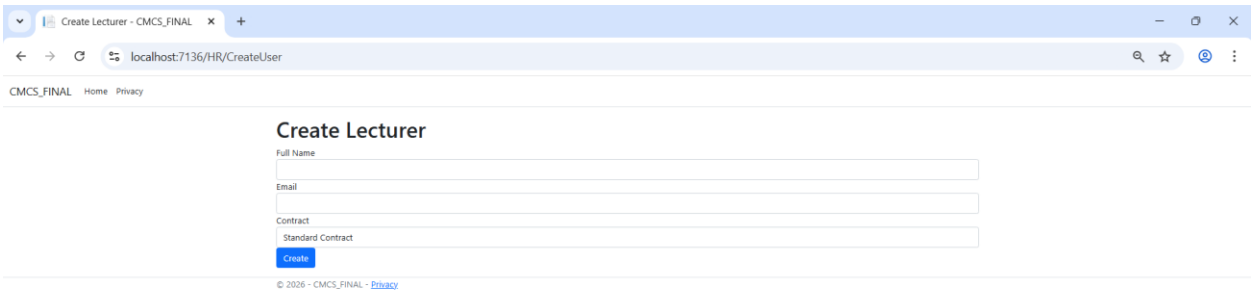

Claims Management - CMCS_FI
localhost:7136/Claims
CMCS_FINAL Home Privacy
Claims Overview
Logged in as Coordinator
ID Lecturer Month Hours Rate Total (R) Status Submitted Actions
1 Dr. John Smith January 2026 120 150.00 18,000.00 Submitted 12 Jan 2026 View Approve Claim approved
Total claims shown: 1
Claim approved
correct
© 2026 - CMCS_FINAL - Privacy

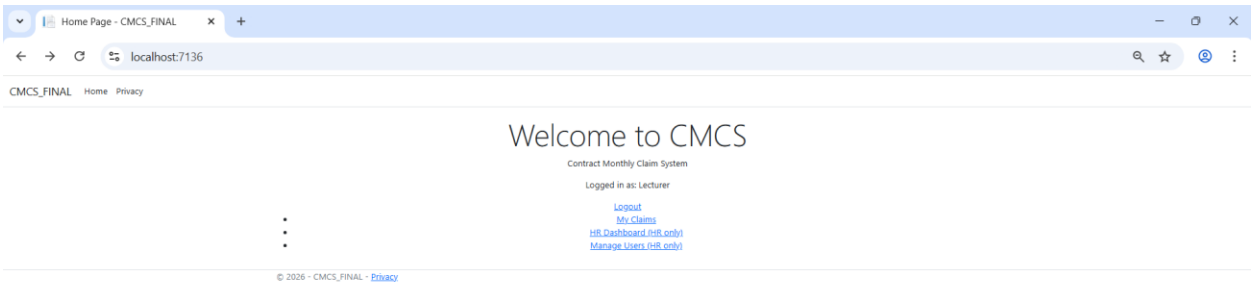

Claims Management - CMCS_FI
localhost:7136/Claims
CMCS_FINAL Home Privacy
Claims Overview
Claim approved successfully!
ID Lecturer Month Hours Rate Total (R) Status Submitted Actions
1 Dr. John Smith January 2026 120 150.00 18,000.00 Verified 12 Jan 2026 View
Total claims shown: 1
© 2026 - CMCS_FINAL - Privacy

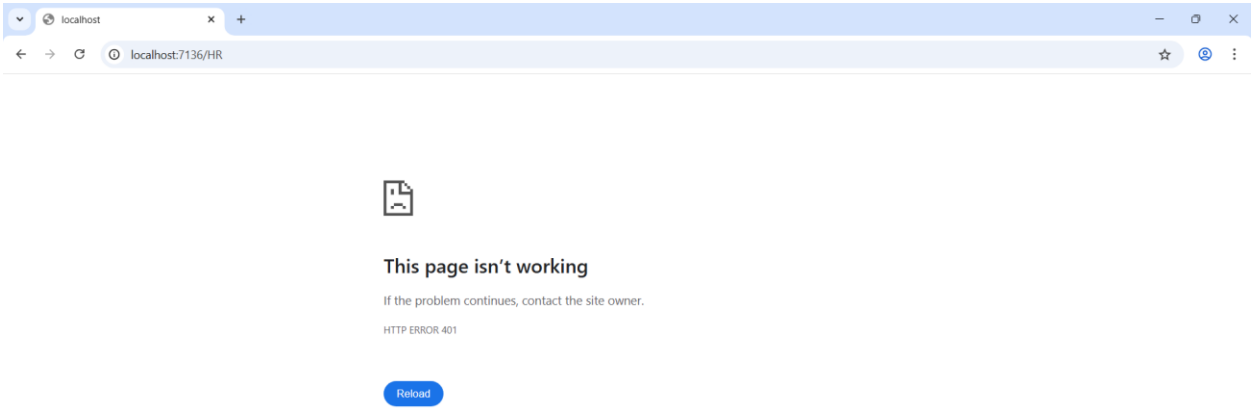

Claims Management - CMCS_FI
localhost:7136/Claims
CMCS_FINAL Home Privacy
Claims Overview
Logged in as Manager
ID Lecturer Month Hours Rate Total (R) Status Submitted Actions
1 Dr. John Smith January 2026 120 150.00 18,000.00 Verified 12 Jan 2026 View Approve correct
Approve
Reject
Total claims shown: 1
© 2026 - CMCS_FINAL - Privacy


Claims Management - CMCS_FI
localhost:7136/Claims
CMCS_FINAL Home Privacy
Claims Overview
Claim approved successfully!
ID Lecturer Month Hours Rate Total (R) Status Submitted Actions
1 Dr. John Smith January 2026 120 150.00 18,000.00 Approved 12 Jan 2026 View
Total claims shown: 1
© 2026 - CMCS_FINAL - Privacy


Manage Users - CMCS_FINAL
localhost:7136/HR/ManageUsers
CMCS_FINAL Home Privacy
Manage Lecturers
Add New Lecturer
LecturerId Full Name Email Contract Actions
1 Dr. John Smith john.smith@university.com 1 Edit | Delete
2 Prof. Sarah Johnson sarah.johnson@university.com 2 Edit | Delete
3 Dr. Michael Brown michael.brown@university.com 1 Edit | Delete
© 2026 - CMCS_FINAL - Privacy


Create Lecturer - CMCS_FINAL
localhost:7136/HR/CreateUser
CMCS_FINAL Home Privacy
Create Lecturer
Full Name
Email
Contract
Standard Contract
Create
© 2026 - CMCS_FINAL - Privacy



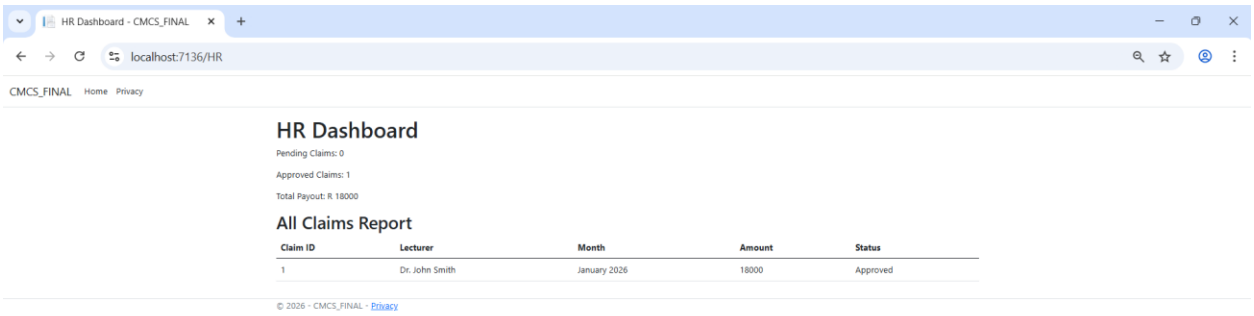


How the layout enhances user experience:

Responsive table with colored badges, icons, and empty state messaging. Search/filter options (if added) make navigation quick.

How the layout supports system functionality: Displays all claims with role-based actions (e.g., approve/reject buttons for Coordinator/Manager).

HR Dashboard View



(Screenshot of HR/Index)

How the layout enhances user experience:

Summary cards for stats, export buttons, and clean tables. Visual indicators for pending/approved claims.

How the layout supports system functionality:

Enables reporting, lecturer management, and invoice generation (via export).

Page 13: **Assumptions and Constraints**

Assumptions and Constraints for Contract Monthly Claim System (CMCS)

Assumptions and Constraints for Contract Monthly Claim System (CMCS)

Area Assumptions

System The system will allow users to upload files/documents.

Constraints

The system can only accept certain file types, such as PDFs, DOCX, and XLSX, with an upload limit of 5MB per file. Encrypted storage uses AES, but keys are hardcoded (production would use secure key management).

Area Assumptions

Data In-memory data persists during runtime; seeded with sample lecturers and contracts.

Constraints

No persistent database; data resets on application restart. Limited to small-scale use and demonstration purposes.

Area Assumptions

Security Role-based access is simulated via session; users follow the intended workflow.

Constraints

No real authentication (demo only); potential session hijacking risks in a production environment.

Area Assumptions

UI	Bootstrap 5 is available for styling; users have modern browsers.

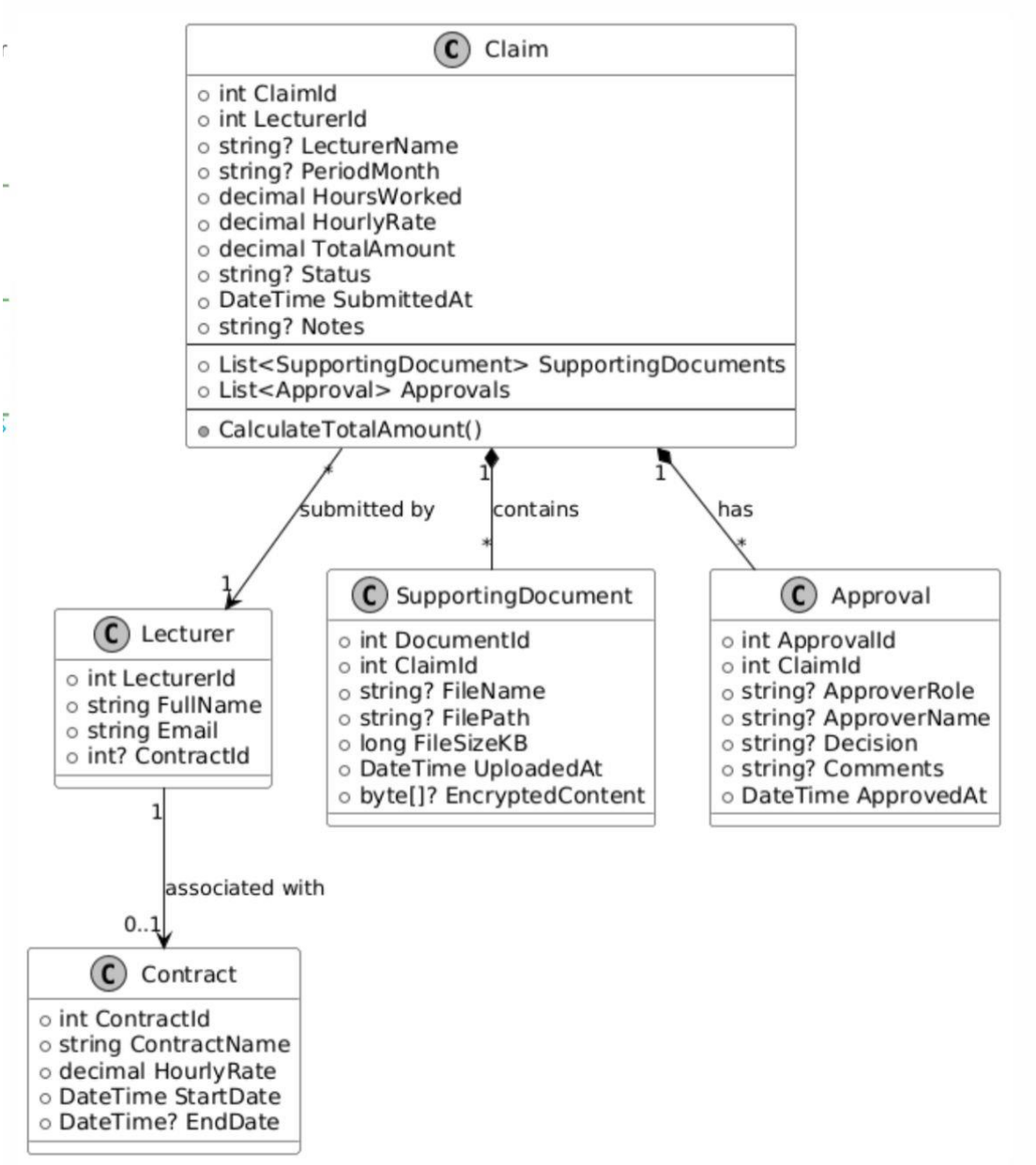
Constraints

No support for Internet Explorer or legacy browsers; assumes HTTPS for production deployment.

Area	Assumptions
Performance	The application handles a small user load; quick in-memory operations.

Constraints

Not optimized for high traffic; in-memory storage limits scalability.



Page 16: **Database Structure Description Database Structure Description for Contract Monthly Claim System (CMCS) Tables and Key Fields Lecturer Table Purpose**

– Stores lecturer information for claim association. Key Fields: LecturerId: int (PK)

– Unique identifier for lecturers. FullName: string

– Lecturer’s full name. Email: string – Contact email, validated as [EmailAddress]. ContractId: int?

– Foreign key to Contract. Relationships: Used in Claim (LecturerId FK); links to Contract (ContractId FK). Claim Table Purpose

– Core table for monthly claims submissions. Key Fields: ClaimId: int (PK)

– Unique claim identifier. LecturerId: int

– FK to Lecturer. PeriodMonth: string – Claim month (e.g., “January 2026”). HoursWorked: decimal

– Hours claimed, [Range(0.1, 744)]. HourlyRate: decimal

– Rate, [Range(0.01, 10000)]. TotalAmount: decimal – Auto-calculated (Hours * Rate). Status: string

– “Submitted”, “Verified”, “Approved”, “Rejected”. SubmittedAt: DateTime

– Submission timestamp. Notes: string – Additional comments. Relationships: Links to Lecturer (LecturerId FK); 1:* with SupportingDocument and Approval. Contract Table Purpose

– Stores contract details for rate association. Key Fields: ContractId: int (PK)

– Unique contract ID. ContractName: string

– Name (e.g., “Standard”). HourlyRate: decimal

– Base rate. StartDate: DateTime

– Contract start. EndDate: DateTime?

– Optional end. Relationships: Referenced by Lecturer (ContractId FK). SupportingDocument Table Purpose

– Stores uploaded claim documents. Key Fields: DocumentId: int (PK)

– Unique document ID. ClaimId: int

– FK to Claim. FileName: string

– Original name. FilePath: string

– Storage path. FileSizeKB: long

– Size in KB. UploadedAt: DateTime

– Upload time. EncryptedContent: byte[]

– Encrypted data. Relationships: 1:* with Claim (ClaimId FK). Approval Table Purpose

– Tracks verification/approval history. Key Fields: ApprovalId: int (PK)

– Unique approval ID. ClaimId: int

– FK to Claim. ApproverRole: string

– “Coordinator”/“Manager”. ApproverName: string

– Approver name. Decision: string

– “Approved”/“Rejected”. Comments: string

– Notes. ApprovedAt: DateTime

– Timestamp. Relationships: 1:* with Claim (ClaimId FK).

Page 18: **Relationships, Structure, Support Relationships Explain the relationship between the entities in your UML Diagram.**

For example, Claim has 1:* relationship with SupportingDocument and Approval, allowing multiple documents/approvals per claim. Lecturer has 1:1 with Contract for rate lookup.

Explanation of the Structure The UML Diagram structure uses classes for entities with properties and methods (e.g., Claim.CalculateTotalAmount). In-memory service simulates database with lists, supporting document upload via byte[] encryption. Support for CMCS Requirements The design supports CMCS by enabling modular data handling (e.g., in-memory for quick claims), role-based workflows, and reporting via LINQ queries on collections.

Page 19: **Project Plan Project Plan for Contract Monthly Claim System (CMCS) Project**

Phase 1 – Planning and Setup

Phase 2 – Core Development

Phase 3 – Enhancement and Testing

Phase 4 – Documentation and Polish Project Tasks, Dependencies, and Timeline Phase 1 Planning and Requirements Gathering (5 Days) Task 1.1

– Project Setup (2 days) Create ASP.NET Core MVC project, add folders (Models/Services/Views). Dependency – None Task 1.2 – Model Design (3 days) Define Claim, Lecturer, etc., models with validation. Dependency – Completion of task 1.1

Phase 1 milestone – Basic structure ready.

Phase 2 Core Development (10 Days) Task 2.1 – Services Implementation (4 days) Create InMemoryDataService and FileEncryptionService. Dependency – Phase 1 completion Task 2.2 – Controllers and Views (6 days) Implement ClaimsController, HRController, and Razor views. Dependency – Task 2.1 Phase 2 milestone – Claim submission and approval working.

Phase 3 Enhancement and Testing (7 Days) Task 3.1 – UI Polish and Validation (3 days) Add Bootstrap, icons, error handling. Dependency – Phase 2 Task 3.2 – Unit Testing (4 days) Write xUnit tests for models/services. Dependency – Task 3.1 Phase 3 milestone – Full functionality tested.

Phase 4 Documentation and Polish (3 Days) Task 4.1 – Documentation (2 days) Write README, update report. Dependency – Phase 3 Task 4.2 – Final Fixes (1 day) Address bugs, add commits. Dependency – Task 4.1 Phase 4 milestone – Ready for submission.

Page 20: **Summary Timeline Summary Timeline**

Phase 1 – 5 Days

Phase 2 – 10 Days

Phase 3 – 7 Days

Phase 4 – 3 Days Total project duration: 25 Days (approximately 5 weeks)

Page 21: **REFERENCE LIST**

Microsoft (2024) Routing in ASP.NET Core. Available at: https://learn.microsoft.com/en us/aspnet/core/fundamentals/routing?view=aspnetcore-9.0 (Accessed: 21 November 2025).

Microsoft (2024) Dependency injection in ASP.NET Core. Available at: https://learn.microsoft.com/en-us/aspnet/core/fundamentals/dependency injection?view=aspnetcore-9.0 (Accessed: 21 November 2025).

Microsoft (2024) LINQ (Language Integrated Query). Available at: https://learn.microsoft.com/en-us/dotnet/csharp/linq/ (Accessed: 21 November 2025).
Microsoft (2024) System.Security.Cryptography Namespace. Available at: https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography?view=net-9.0 (Accessed: 21 November 2025).

Microsoft (2024) AES Class - Advanced Encryption Standard. Available at: https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography.aes?view=net-9.0 (Accessed: 21 November 2025).

Microsoft (2024) File uploads in ASP.NET Core. Available at: https://learn.microsoft.com/en us/aspnet/core/mvc/models/file-uploads?view=aspnetcore-9.0 (Accessed: 21 November 2025).

Microsoft (2024) Model validation in ASP.NET Core MVC. Available at: https://learn.microsoft.com/en-us/aspnet/core/mvc/models/validation?view=aspnetcore-9.0 (Accessed: 21 November 2025).

Bootstrap (2024) Bootstrap Icons. Available at: https://icons.getbootstrap.com/ (Accessed: 21 November 2025).

jQuery Validation Plugin (2024) jQuery Validation Plugin Documentation. Available at: https://jqueryvalidation.org/documentation/ (Accessed: 21 November 2025).

Microsoft (2024) Unit testing in .NET. Available at: https://learn.microsoft.com/en us/dotnet/core/testing/ (Accessed: 21 November 2025).

Microsoft (2024) Unit testing C# with xUnit. Available at: https://learn.microsoft.com/en us/dotnet/core/testing/unit-testing-with-dotnet-test (Accessed: 21 November 2025).

Microsoft (2024) Dependency Injection Pattern. Available at: https://learn.microsoft.com/en us/dotnet/core/extensions/dependency-injection (Accessed: 21 November 2025).

Microsoft (2024) Entity Framework Core. Available at: https://learn.microsoft.com/en us/ef/core/ (Accessed: 21 November 2025).

Microsoft (2024) Collections (C#). Available at: https://learn.microsoft.com/en us/dotnet/csharp/programming-guide/concepts/collections (Accessed: 21 November 2025).

Mozilla (2024) MDN Web Docs - JavaScript. Available at: https://developer.mozilla.org/en US/docs/Web/JavaScript (Accessed: 21 November 2025).

W3C (2024) HTML Standard. Available at: https://html.spec.whatwg.org/ (Accessed: 21 November 2025). Microsoft (2024) Exception Handling (C#). Available at: https://learn.microsoft.com/en us/dotnet/csharp/fundamentals/exceptions/ (Accessed: 21 November 2025).

Microsoft (2024) Secure coding guidelines. Available at: https://learn.microsoft.com/en us/dotnet/standard/security/secure-coding-guidelines (Accessed: 21 November 2025).

Microsoft (2024) Background tasks with hosted services. Available at: https://learn.microsoft.com/en-us/aspnet/core/fundamentals/host/hosted services?view=aspnetcore-9.0 (Accessed: 21 November 2025).

YouTube Link: https://youtu.be/diWSk1cxw2I

GitHub Link: https://github.com/VCDN-2025/prog6212-poe-supplementary-resubmission-JadeelK