Lecturer Claim Automation System

ST10442580
PROG6121 Final POE

Contents

Title and Profession

Introduction to slide

Design and Architecture

All the designs and Understanding of **MVC Pattern** and **Separation of Concerns** (Service Layer).

functionality

Calculation Logic,User Views/Tracking,

Automation and Conclusion

For more info:
SLIDESGO | SLIDESGO AI TOOLS | BLOG | FAQs

You can edit the master slides easily.
For more info, click **here**

You can visit our sister projects:
FREEPIK AI TOOLS | FREEPIK LIBRARY RESOURCES

You can delete this slide when you're done editing the presentation

Slide 1:

I am presenting my Portfolio of Evidence for
the **Lecturer Claim Automation System (L-CAS)**, a web application built for PROG6121.
This system was designed to replace a manual, paper-based process, transforming it into a secure, automated digital workflow."
"My presentation will focus on the architectural decisions, the implementation of the core Functional Requirements (F1-F4), and crucially, the **Process Automation (P3)** features I developed.

Slide 2: Design

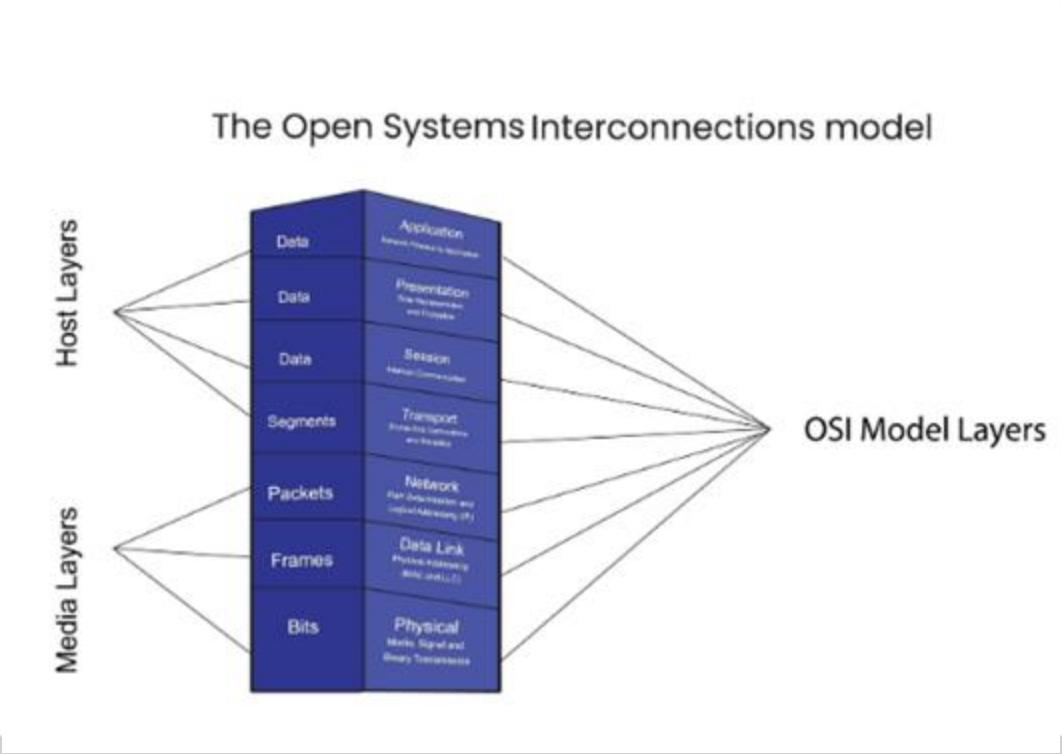
"The foundation of L-CAS is the **ASP.NET Core MVC** pattern, using C# and Entity Framework Core for data management."

"The key design decision was enforcing **Separation of Concerns**. We have the Controller to handle traffic, and the Database for persistence."

"But critically, the **Service Layer** is where all the POE requirements are fulfilled. It executes F1, F2, and P3 logic. This isolation makes the system maintainable and highly testable."

"*(Referencing the Diagram)*: This diagram shows the clean flow: The Controller delegates business tasks to the Service Layer, which then interacts with the database."

Slide 2:Design -image:



Slide 3:Functional Requirements:

Starting with basic requirements, **F3** is the **Claim Submission** form. This form uses Data Annotations to ensure data validation, such as checking for valid emails and requiring all fields."
"Once submitted, the lecturer needs to track the claim (**F4**). The system ensures strict security by filtering the list to show **only** the claims associated with the lecturer's email.

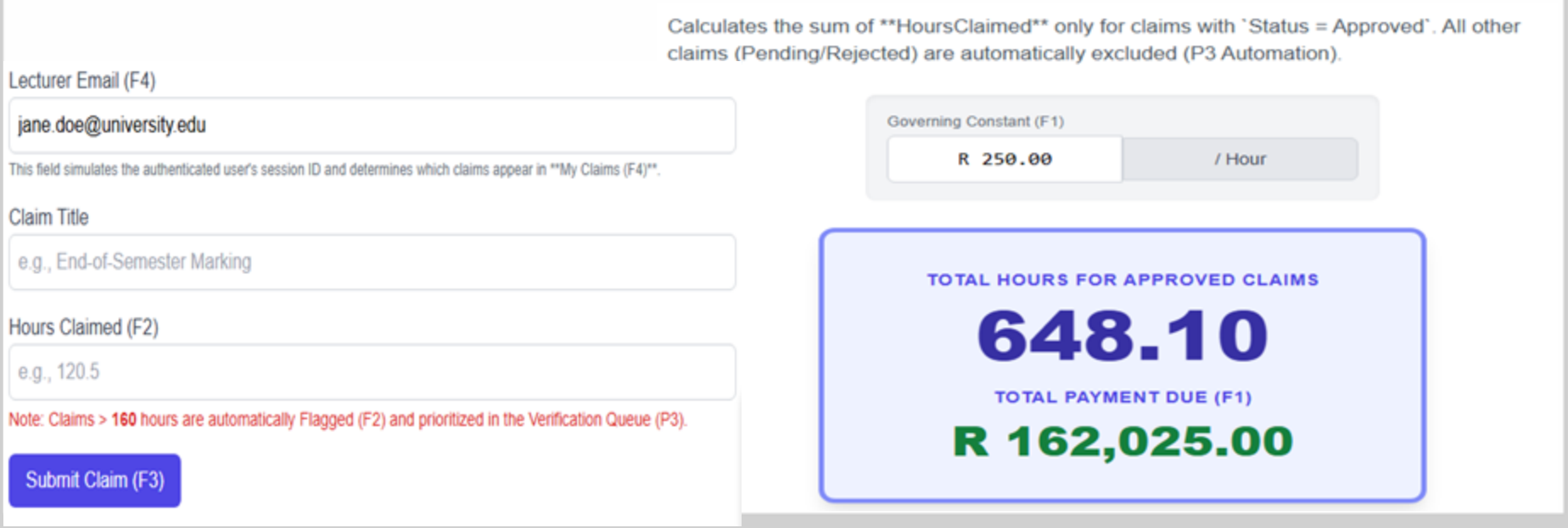
Showing claims for: **Freda@rcconnect.edu**

TITLE (ID)	HOURS CLAIMED	STATUS	SUBMISSION DATE	FINAL PAYMENT (F1)
Logistic erros (#1006)	89.50	Pending	11/21/2025	R 22,375.00
Rewrite (#1007)	50.20	Pending	11/21/2025	R 12,550.00
transportation (#1008)	67.60	Pending	11/21/2025	R 16,900.00

Slide 4: Functional Requirements:

Automatic Calculation is simple: the Service Layer calculates the Final Payment Amount by multiplying the hours claimed by the standard hourly rate *before* the claim is saved. No manual calculation is needed."

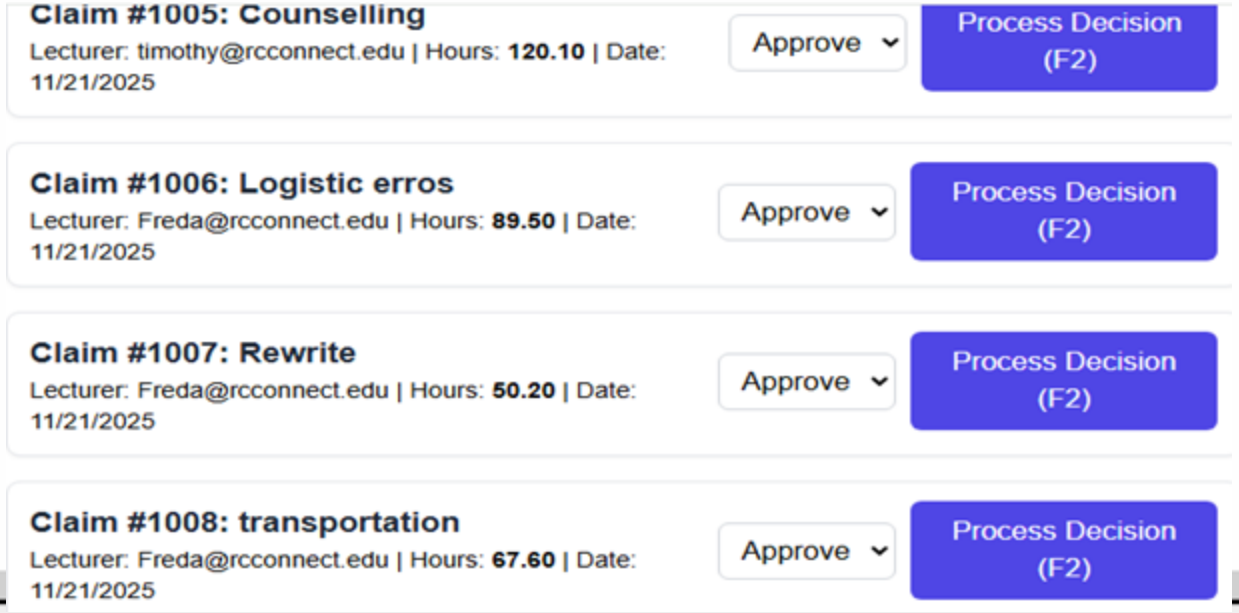
"**F2 (Manager Flagging)** is our initial risk control. If a claim exceeds **160 hours**, the Service Layer automatically sets a `RequiresManagerFlag` to true. This proactively highlights claims that may indicate fraud or budget issues.



Slide 5: Automation:

This is the first part of **P3 Automation**—making the Coordinator's job efficient. Instead of reviewing claims randomly, the system implements a **Priority Queue**.

"The sorting logic is two-tiered: **1. High-Risk claims (Flagged)** are always moved to the top of the queue. **2. Within that group, we use FIFO (First-In, First-Out)** based on the submission date.



Slide 6: Payroll

The second part of **P3** is the **Payroll Report**. Its core function is financial security."

"The system aggregates the total hours **only** for claims that have {Status} = {Approved}. All Pending or Rejected claims are automatically excluded from the final payment total.

Conclusion:

In summary, L-CAS successfully integrates all required functional (F1-F4) and process automation (P3) requirements, delivering a tested, robust solution."

"Looking forward, the next critical step would be implementing **ASP.NET Identity** to manage secure user roles (Lecturer vs. Coordinator) and adding an **Audit Trail** to track exactly who is approved.