Contract Monthly Claim System could look and function:

1. Lecturer’s Claim Submission Form:

**Layout**:

A clean, intuitive form at the center of the screen.

Left-aligned labels next to each input field for clarity.

Grouped sections with borders for different inputs like hours worked, rate, and notes.

**Fields**:

Hours Worked: Numeric input field.

Hourly Rate: Automatically populated or manually inputted.

Additional Notes: A text box for extra information.

**File Upload**:

A button labeled “Upload Document” with accepted formats (.pdf, .docx, .xlsx).

Once a file is uploaded, its name should be displayed in the form.

Option to remove and re-upload the document.

**Buttons**:

A large, prominent "Submit" button at the bottom of the form.

Clear Form button for easy resetting of fields.

**Color Scheme**:

A calm, professional color palette—such as light blue or soft grey for the background—with contrasting buttons (e.g., green for submit, red for clear).

Error messages displayed in red, while success messages in green.

**User Flow**:

Upon clicking Submit, a success message or error notification (if required fields are missing) appears at the top.

After submission, users can view a summary of the claim.

2. Coordinators and Managers’ Claim Review Page:

**Layout**:

A grid/table view listing all claims, with columns for details like:

Lecturer name, hours worked, hourly rate, total amount, status, document link, etc.

Clear status labels on each row (e.g., "Pending", "Approved", "Rejected").

At the end of each row, two buttons: Approve (green) and Reject (red).

**Filters**:

A sidebar or top row allowing managers to filter claims by status (Pending, Approved, etc.).

**Buttons**:

Approve and Reject buttons are easy to access next to each claim row.

When a button is clicked, a confirmation modal pops up to confirm the action.

**Color Scheme**:

Similar to the lecturer’s form but with added red/green visual cues for approvals/rejections.

3. File Upload Feature:

**Lecturers**:

The "Upload Document" button is located next to the "Additional Notes" section.

Uploaded files will be listed below the button with their names displayed and an option to remove the file if needed.

A size and format warning displayed next to the button (e.g., “Max size: 5MB. Accepted formats: .pdf, .docx, .xlsx”).

4. Claim Status Tracking:

**Lecturers’ View**:

After submitting, a status label is displayed on their claim overview page (e.g., "Pending," "Approved," or "Rejected").

Alternatively, a progress bar could show the claim's journey from submission to approval/rejection.

**Real-Time Updates**:

When a coordinator approves/rejects a claim, the lecturer’s view should update in real-time to reflect the status change.

5. Error Handling:

If there are issues during submission (e.g., missing fields or file errors), meaningful error messages will be shown below the relevant fields.

General errors (e.g., server errors) should appear at the top of the form in a red box.

6. Unit Testing and Error Handling:

Ensure tests cover claim submission (valid and invalid data), file upload, approval workflows, and status updates.

Error messages are displayed clearly to guide users if something goes wrong.

UML Diagram

A diagram of a program

Description automatically generated

This diagram appears to be an Entity-Relationship Diagram (ERD) that models the relationships between different entities in a system, such as:

1. Rate: Identified by rateID, it has a 1-to-1 relationship with Session.
2. Programme: Contains information about various programs, with a 1-to-many relationship to both Session and Module.
3. Session: A central entity linked to Programme, Module, Group, and Rate. It contains foreign keys for each related entity.
4. Module: Linked 1-to-1 with Programme, but also linked to Session.
5. Group: Defines group details, connected in a 1-to-many relationship with Session.

Breakdown of the attributes in the diagram

Rate:

rateID: The primary key, unique identifier for each rate.

Programme:

programmeID: Primary key for identifying each program.

programmeName: Name of the program.

Session:

SessionID: Unique identifier for each session (Primary key).

programmeID: Foreign key linking to the Programme entity.

moduleID: Foreign key linking to the Module entity.

groupID: Foreign key linking to the Group entity.

Module:

moduleID: Primary key for identifying each module.

programmeID: Foreign key linking to Programme.

Group:

groupID: Unique identifier (Primary key) for each group.

groupName: Name of the group.

Project plan for developing the Contract Monthly Claim System for lecturers:

Week 1-2: Planning & Initial Setup

Requirements gathering: Finalize features, roles (lecturer, coordinator, manager).

Design mockups: Create UI wireframes for both lecturer and manager views.

Technology setup: Set up project environment, databases (SQL/EF Core), repository.

Week 3-4: Database & Backend Development

Design database schema (lecturers, claims, sessions, etc.).

Implement backend functionality for claim submissions and approval workflow.

Set up file upload functionality and secure document storage.

Week 5-6: Frontend & User Interface

Build and integrate lecturer’s claim submission form.

Implement manager/coordinator view with filtering and approval/rejection options.

Test and refine UI based on user feedback.

Week 7: Integration & Testing

Integrate backend with frontend.

Implement real-time updates for claim status tracking.

Write unit tests to ensure functional coverage.

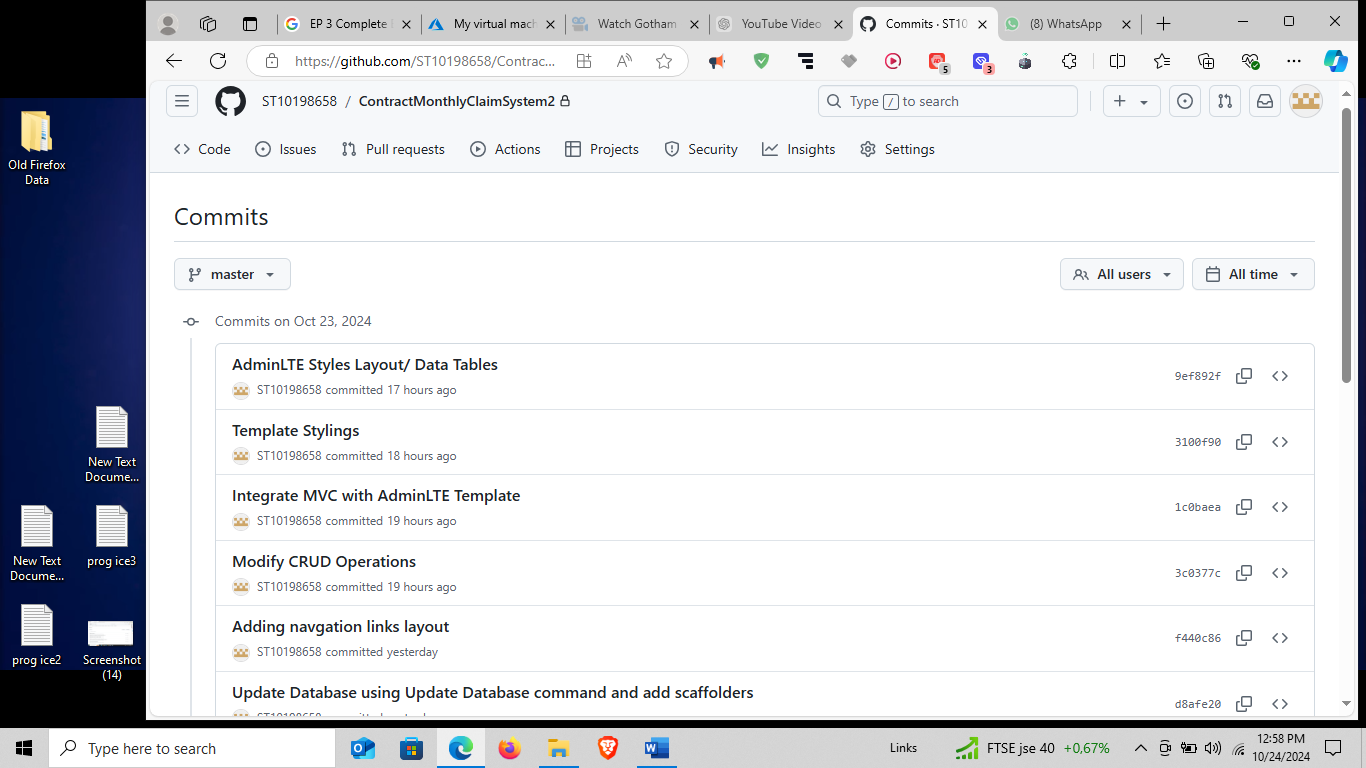
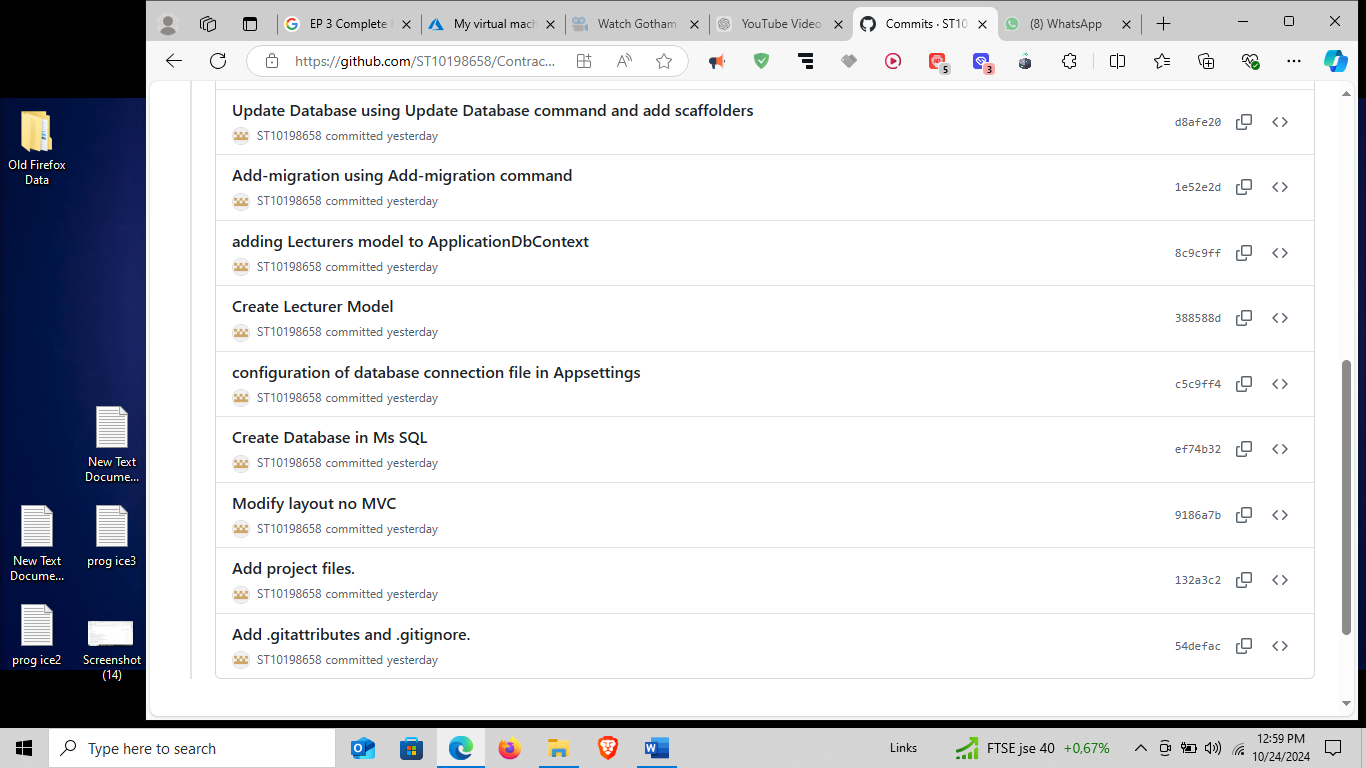
Week 8: Finalization & Deployment

Final testing and debugging.

Error handling and user-friendly messaging.

Deploy the system and prepare user documentation/training.

https://github.com/ST10198658/ContractMonthlyClaimSystem2.git



Coffin, C., Mary Jane Curry, Goodman, S., Hewings, A., Lillis, T. and Swann, J. (2005). *Teaching Academic Writing*. Routledge.

‌ O’Brien, S. (2012). Translation as human–computer interaction. *Translation Spaces A multidisciplinary, multimedia, and multilingual journal of translation*, [online] 1, pp.101–122. doi:https://doi.org/10.1075/ts.1.05obr.

‌ Berry, M.W., Azlinah Hj. Mohamed and Bee Wah Yap (2016). *Soft Computing in Data Science*. Springer.

‌ Dinesh Maidasani (2005). *Learning Computer Fundamentals, Ms Office and Internet & Web Tech.* Firewall Media.

‌ Mortiboys, A. (2010). *How To Be An Effective Teacher In Higher Education*. McGraw-Hill Education (UK).