CEBU INSTITUTE OF TECHNOLOGY UNIVERSITY

COLLEGE OF COMPUTER STUDIES

Software Design Description

for

Software Project Management Plan Evaluator

Signature

Change History

Preface

Table of Contents

1. Introduction

1.1. Purpose

This Software Design Description (SDD) provides a detailed architectural and design specification for the Software Project Management Plan Evaluator (SPMP Evaluator). The document defines the system's structural design, data flow, module interactions, and component-level logic supporting automated evaluation of SPMP documents for IEEE 1058 standard compliance. It serves as a guide for software developers, testers, and system administrators throughout implementation, integration, and maintenance phases..

1.2. Scope

The SPMP Evaluator is a web-based evaluation platform that automates the assessment of Software Project Management Plans (SPMPs) against IEEE 1058 standards. The system will extract document structure from uploaded PDF/DOCX files, evaluate compliance using a configurable rules engine, generate compliance scores, and provide descriptive feedback.

The solution supports two primary user roles—students and professors—each with specific access levels and dashboards. Core components include:

- Authentication and Authorization Module for secure user access
- Document Upload and Parsing Module for structured data extraction.
- Evaluation Engine that applies IEEE 1058 rules and produces scores.
- Results and Feedback Module for reporting and visualization.
- Admin Console for managing users, roles, and evaluation rules.

1.3. Definitions and Acronyms

- SPMP — Software Project Management Plan
- IEEE 1058 — IEEE standard for Software Project Management Plans
- Parser — Module that extracts structural components from SPMP documents
- Evaluator — Core engine applying rule-based and weighted scoring logic
- AI — Artificial Intelligence (optional component for semantic analysis)

- DB — Database
- SRS — Software Requirements Specification
- SDD — Software Design Description

1.4. References

- IEEE Std 1058-1998 (and subsequent revisions) — *Software Project Management Plan Standard*
- IEEE Std 1016-2009 — *Software Design Description Standard*
- Google Document AI & Vertex AI documentation (for proposed parsing integration)
- Apache POI documentation (for DOCX handling)
- PDF.js documentation (for PDF parsing and text extraction)
- Software Requirements Specification prepared by the proponent team, September 2025

2. Architectural Design

Client / Frontend

- React / Next.js Framework
- Tailwind CSS / Material UI for styling
- Axios / Fetch API for REST communication
- Supabase Auth (JWT-based login)
- Browser Compatibility: Chrome, Firefox, Edge

Server / Backend

- Node.js (Express Framework)
- RESTful API endpoints
- Multer / Busboy for file uploads
- Evaluation Engine (IEEE 1058 Rules)
- Report Generation (PDF / HTML)

File Parsing & AI Modules

- Apache POI – DOCX parsing
- PDF.js / PDFBox – PDF extraction
- Google Document AI – Optional structure parsing
- Vertex AI (optional) – Semantic feedback enhancement

Database & Storage

- Supabase PostgreSQL – User, score, and feedback data
- Supabase Storage – Encrypted file storage
- Supabase Auth – Token-based authentication

Hosting & Security

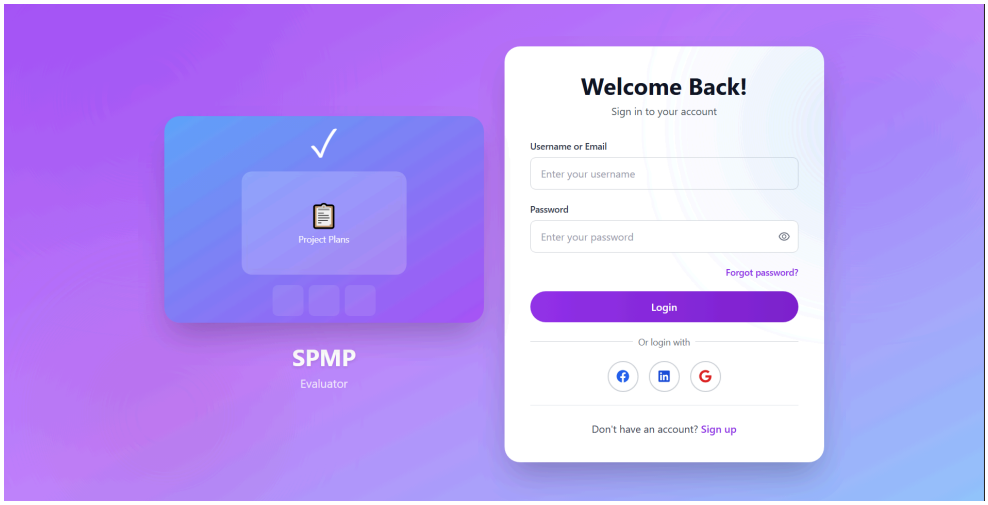
- Render – Node.js backend hosting
- HTTPS / TLS Encryption
- Free-tier cloud infrastructure
- 50–100 concurrent evaluations target

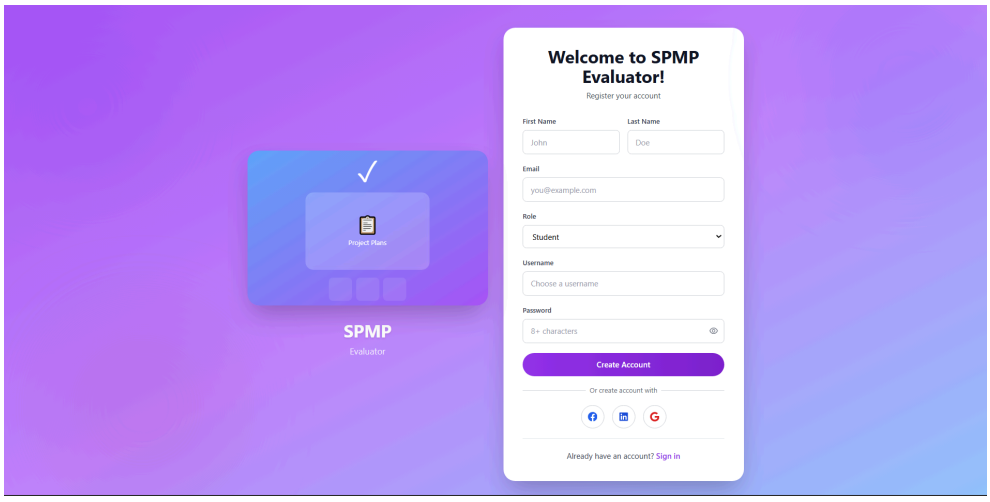
3. Detailed Design

Module 1 - Basic Login and Security System

1.1 User Registration and Login

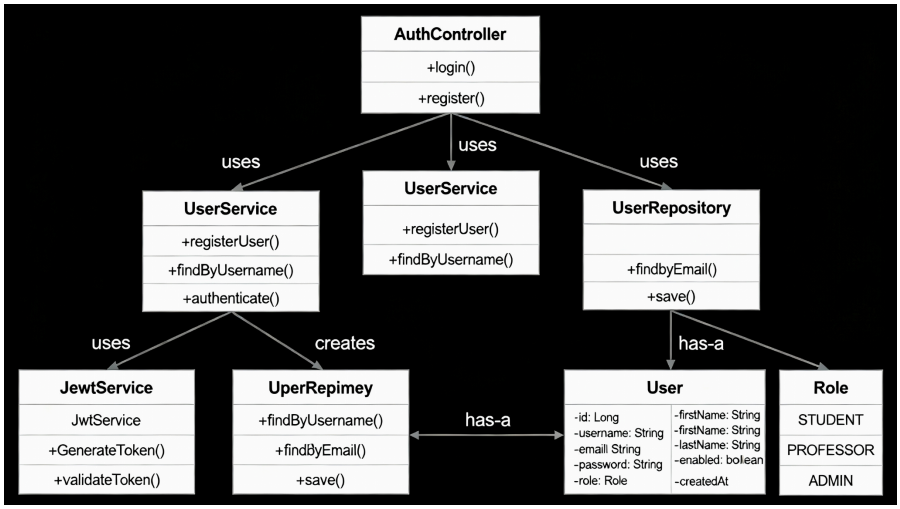
- User Interface Design



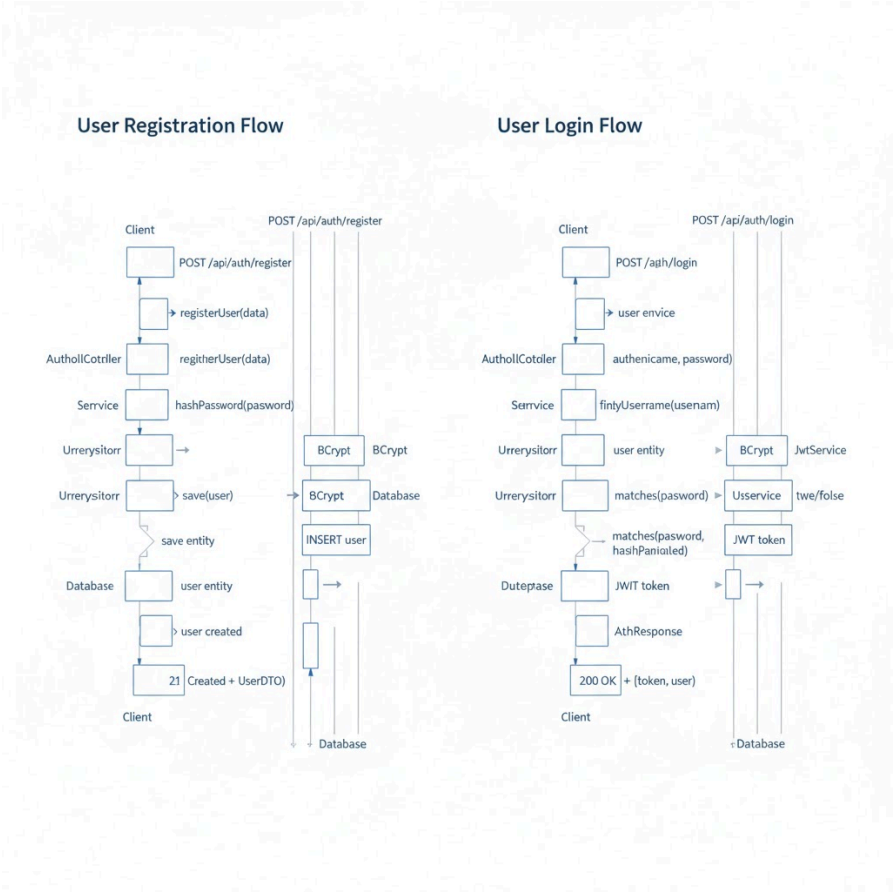


- Front-end component(s)
    - Component Name - AuthPage.jsx
        - Description and purpose - Dual-mode authentication page for user login and registration with role-based access (Student, Professor, Admin). Handles form validation, error messages, and JWT token management through AuthContext.
        - Component type or format - React Functional Component (JSX) - Authentication form with login/signup toggle, API integration via apiService, and Tailwind CSS styling.

- Back-end component(s)
    - Component Name - Login and Registration
        - Description and purpose - Handles user authentication, registration, JWT token generation, role-based access control, and password encryption using BCrypt. Manages user sessions and enforces security policies across the application.
        - Component type or format - Spring Boot REST API with the following components:
            - AuthController.java - REST endpoints for login/register
            - UserService.java - Business logic for user management
            - JwtService.java - JWT token generation and validation
            - SecurityConfig.java - Security configuration and filters
            - JwtFilter.java - Request authentication filter
            - User.java - User entity with role-based permissions
            - Role.java - Enum (STUDENT, PROFESSOR, ADMIN)
            - UserRepository.java - JPA repository for database operations
- Object-Oriented Components
    - Class Diagram -



- Sequence Diagram -



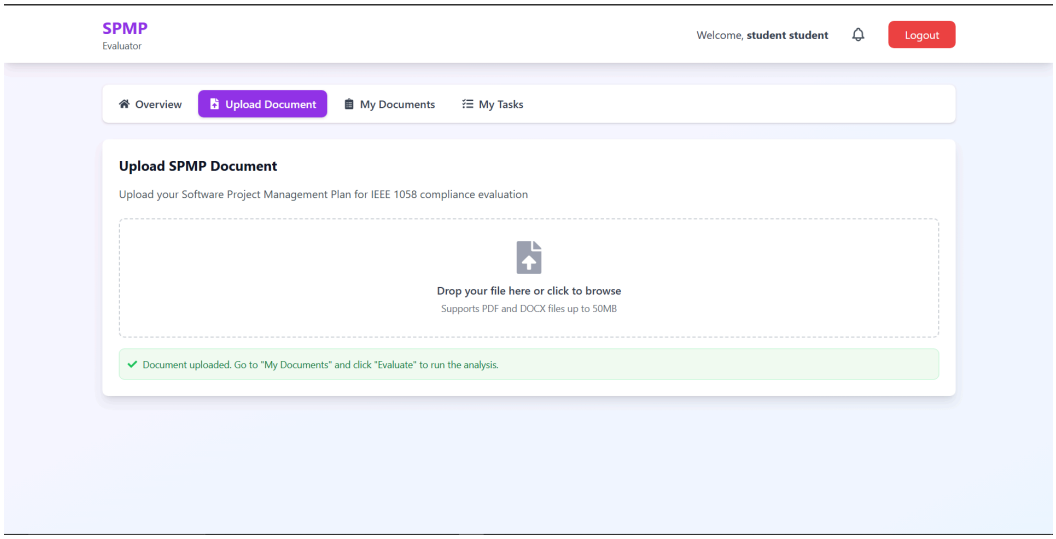
- Data Design
  - ERD or schema -

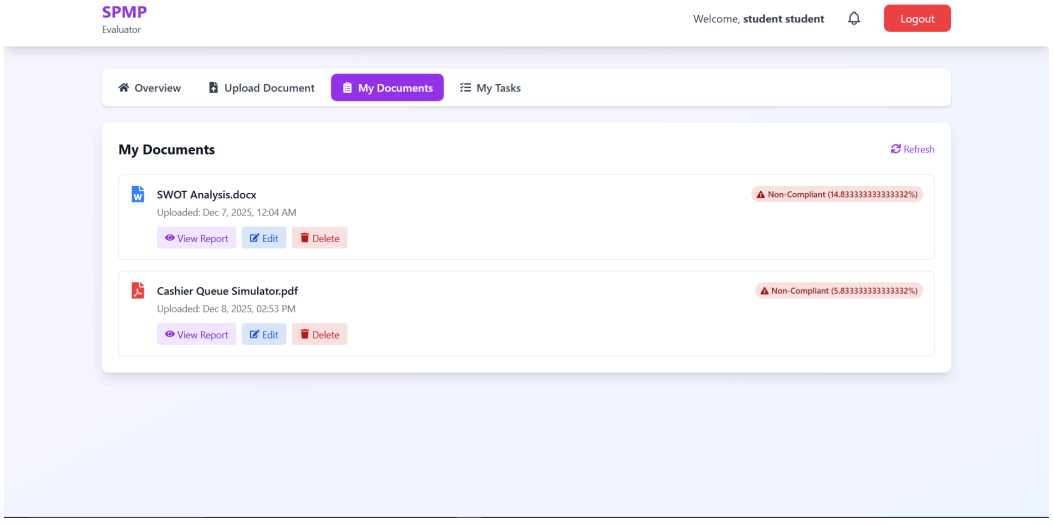
```
CREATE TABLE users (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(255) NOT NULL UNIQUE,
    email VARCHAR(255) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    role ENUM('STUDENT', 'PROFESSOR', 'ADMIN') NOT NULL,
    first_name VARCHAR(255),
    last_name VARCHAR(255),
    enabled BOOLEAN NOT NULL DEFAULT TRUE,
    created_at DATETIME,
    updated_at DATETIME
);
```

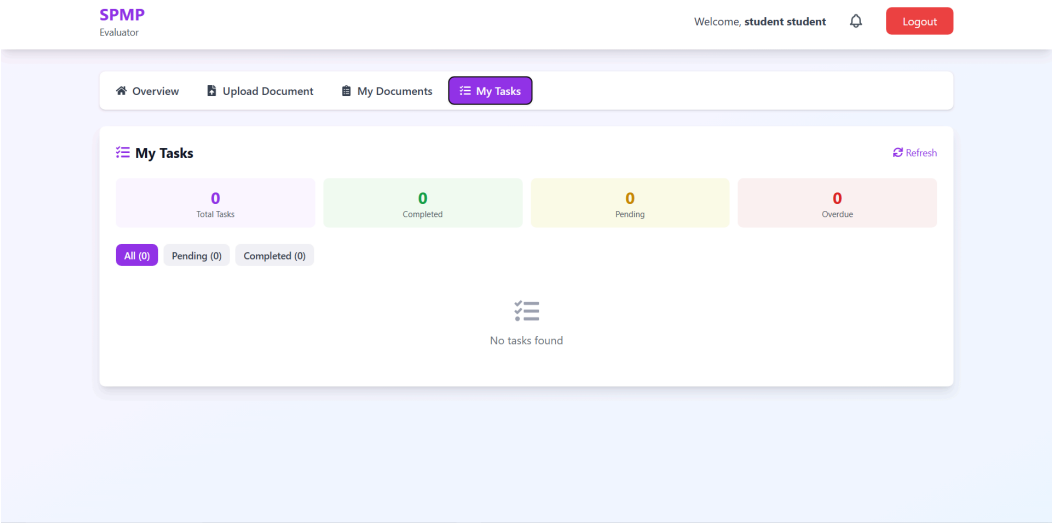
Module 2 - Role-Based User Interface Transactions

***2.1** Student*

- User Interface Design



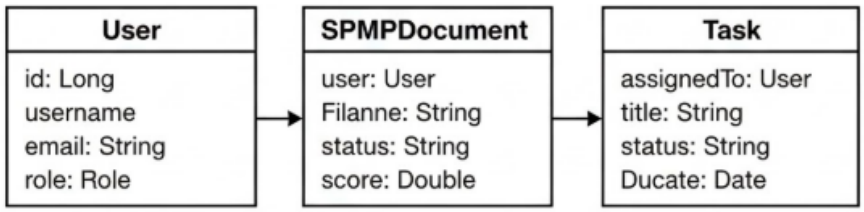




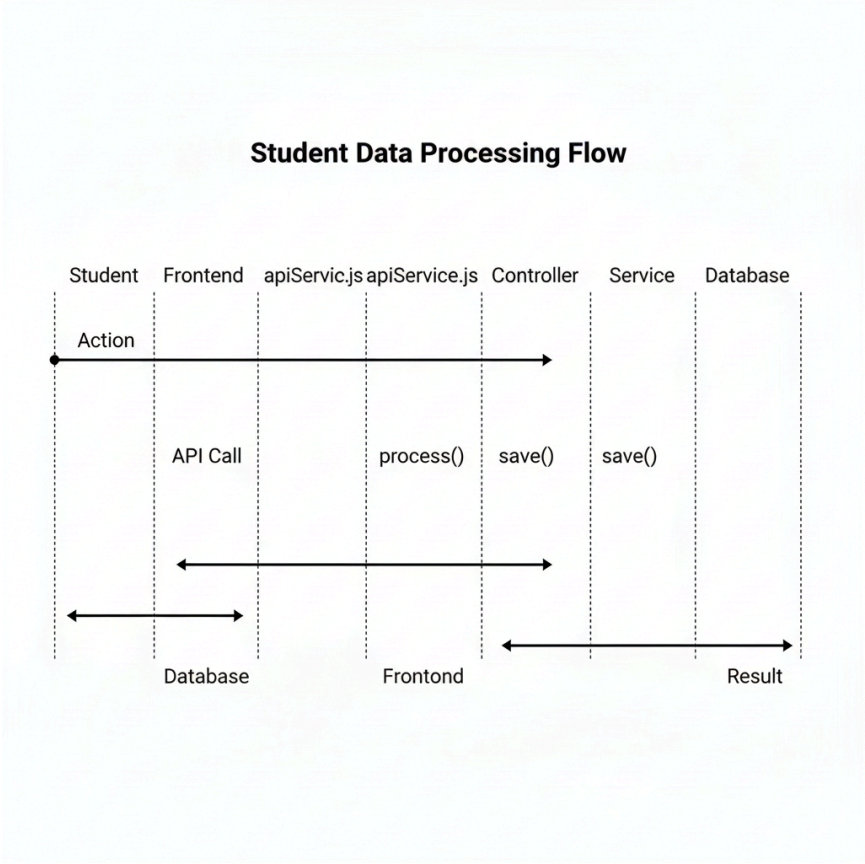
- Front-end component(s)
    - Component Name - DocumentUpload.jsx, DocumentList.jsx, TaskTracker.jsx
        - Description and purpose - Student dashboard modules for SPMP document management and task tracking. DocumentUpload handles drag-and-drop file uploads with format validation (PDF/DOCX). DocumentList displays uploaded documents with View Report, Edit, and Delete actions. TaskTracker shows assigned tasks with status filters (Pending, Completed, Overdue) and progress statistics.
        - Component type or format - React Functional Components (JSX) - Dashboard panels with state management via useState/useEffect hooks, API integration via apiService.js, and Tailwind CSS styling.
- Back-end component(s)
    - Component Name - Document & Task Management API
        - Description and purpose - Handles SPMP document upload, storage, retrieval, evaluation, and deletion. Manages task assignment, status tracking, and deadline monitoring. Enforces role-based access control ensuring students can only access their own documents and assigned tasks.
        - Component type or format - Spring Boot REST API with the following components:
            - DocumentController.java - REST endpoints for upload, retrieve, update, delete documents
            - DocumentService.java - Business logic for document processing and evaluation TaskController.java - REST endpoints for task management
            - TaskService.java - Business logic for task assignment and tracking
            - SPMPDocument.java - Document entity with file metadata and evaluation scores
            - Task.java - Task entity with assignment and status tracking

- DocumentRepository.java - JPA repository for document operations
- TaskRepository.java - JPA repository for task operations

- Object-Oriented Components
  - Class Diagram -



  - Sequence Diagram -

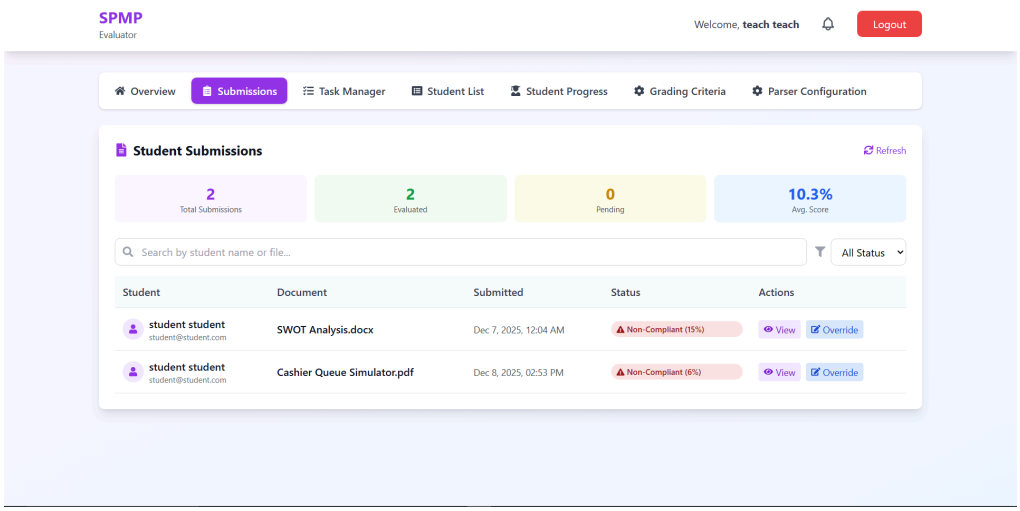


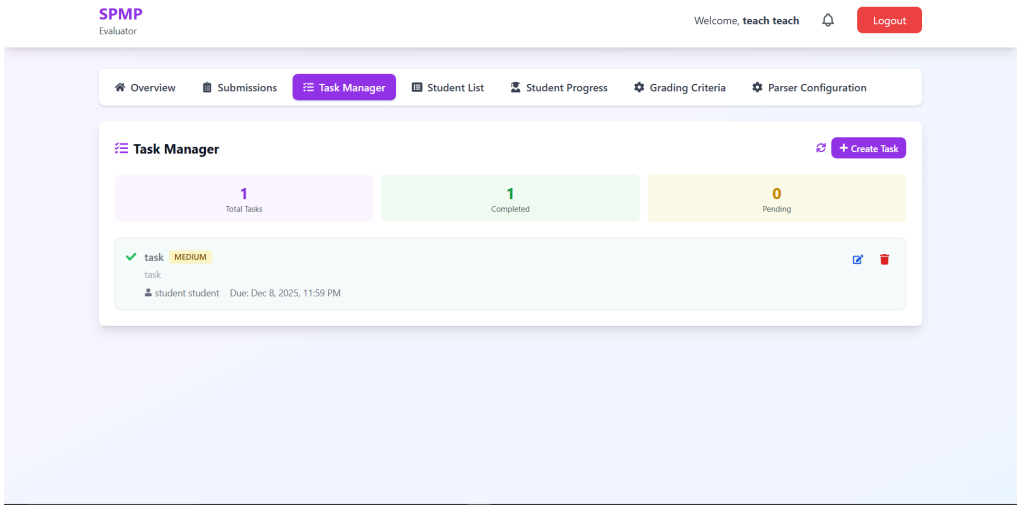
- Data Design
  - ERD or schema -

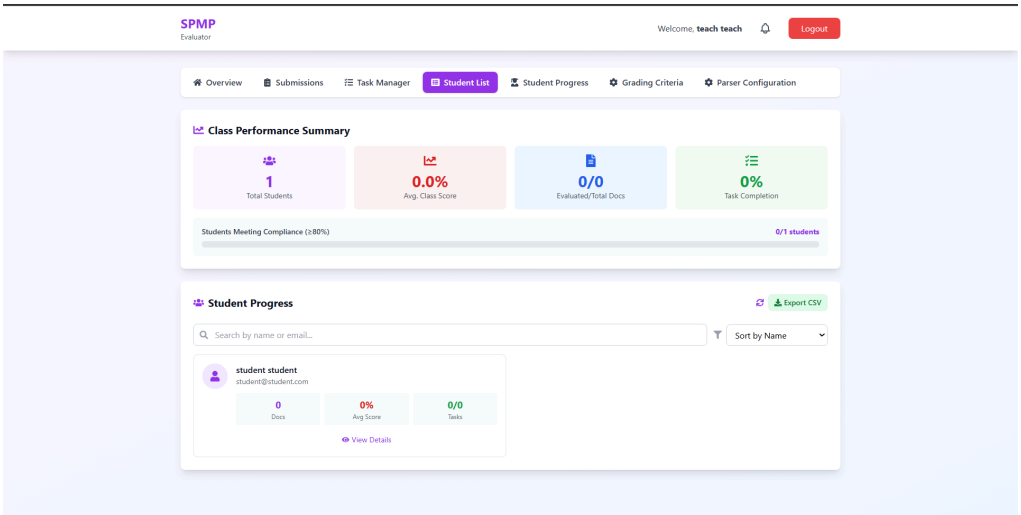
```
CREATE TABLE spmp_documents (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    user_id BIGINT NOT NULL,
    file_name VARCHAR(255) NOT NULL,
    file_path VARCHAR(500) NOT NULL,
    file_type VARCHAR(50),
    file_size BIGINT,
    status ENUM('PENDING', 'EVALUATED', 'REJECTED') DEFAULT 'PENDING',
    score DOUBLE,
    professor_notes TEXT,
    uploaded_at DATETIME,
    evaluated_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE tasks (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    assigned_to BIGINT NOT NULL,
    created_by BIGINT NOT NULL,
    status ENUM('PENDING', 'IN_PROGRESS', 'COMPLETED') DEFAULT 'PENDING',
    due_date DATETIME,
    completed BOOLEAN DEFAULT FALSE,
    created_at DATETIME,
    updated_at DATETIME,
    FOREIGN KEY (assigned_to) REFERENCES users(id),
    FOREIGN KEY (created_by) REFERENCES users(id)
);
```

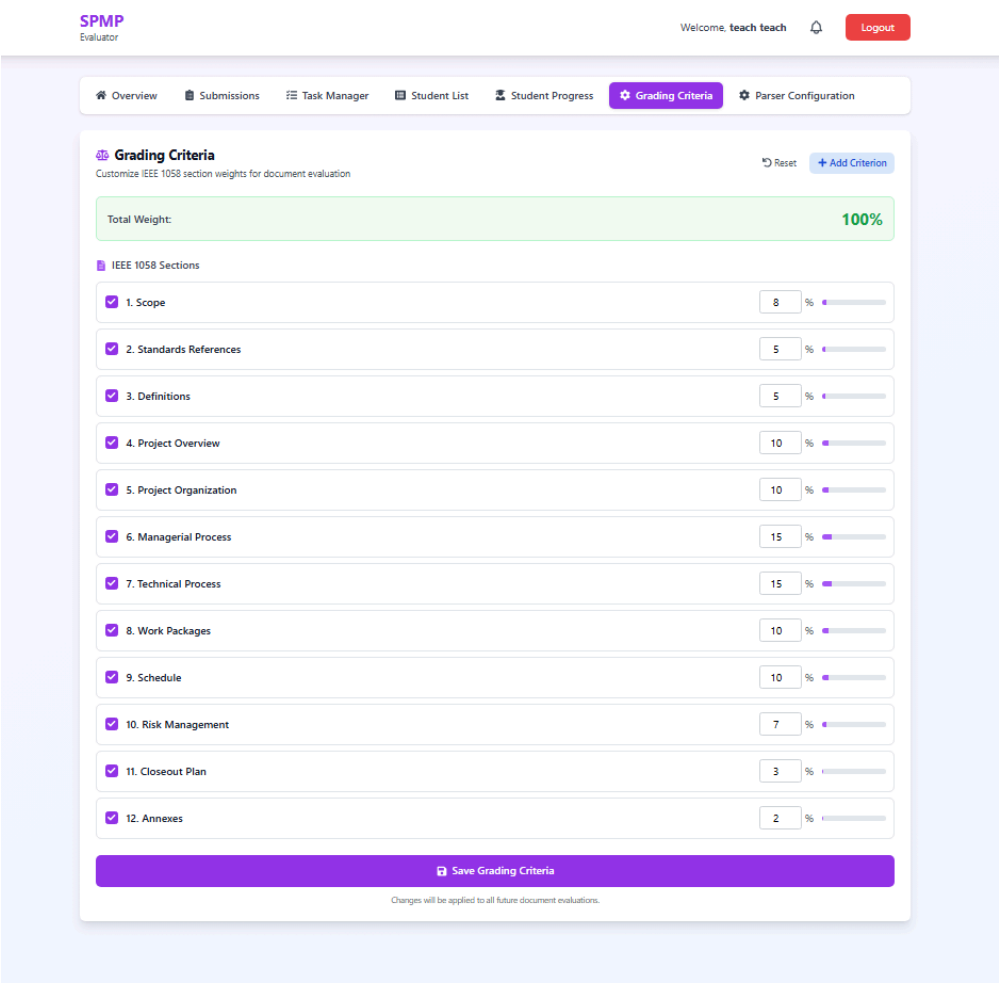
2.2 Professor

- User Interface Design



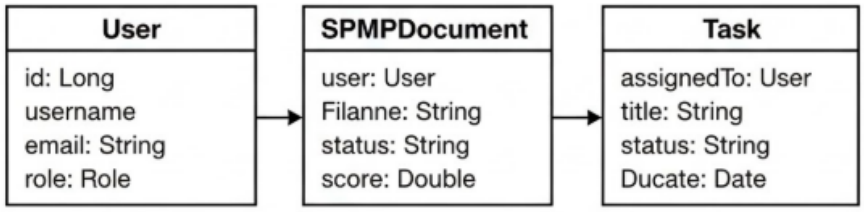



SPMP
Evaluator
Welcome, teach teach
Logout
Overview
Submissions
Task Manager
Student List
Student Progress
Grading Criteria
Parser Configuration
Class Performance Summary
1
Total Students
0.0%
Avg. Class Score
0/0
Evaluated/Total Docs
0%
Task Completion
Students Meeting Compliance (≥80%)
0/1 students
Student Progress
Export CSV
Search by name or email...
Sort by Name
student student
student@student.com
0
Docs
0%
Avg Score
0/0
Tasks
View Details

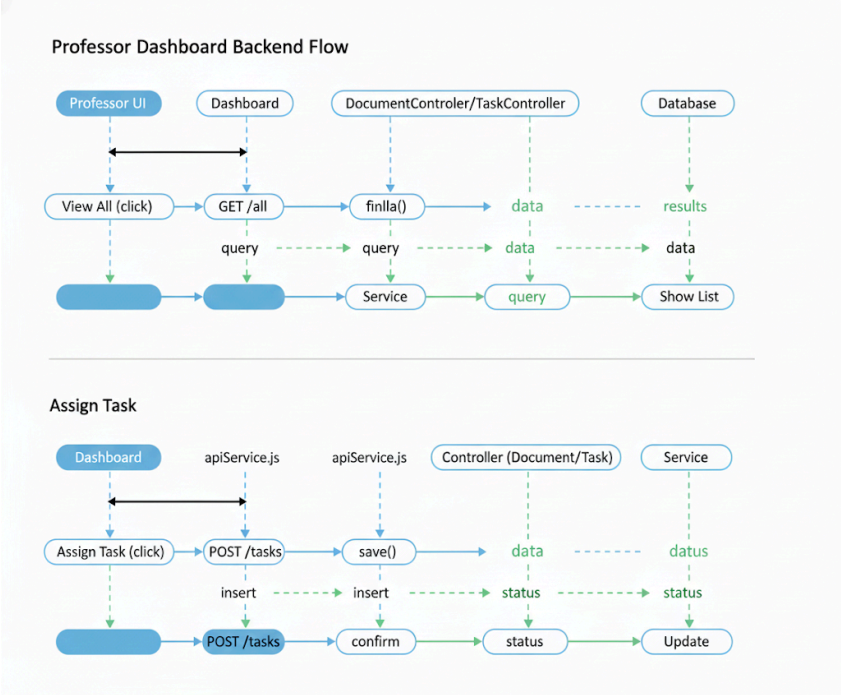

SPMP
Evaluator
Welcome, teach teach
Logout
Overview
Submissions
Task Manager
Student List
Student Progress
Grading Criteria
Parser Configuration
Grading Criteria
Customize IEEE 1058 section weights for document evaluation
Reset
+ Add Criterion
Total Weight:
100%
IEEE 1058 Sections
1. Scope
8
%
2. Standards References
5
%
3. Definitions
5
%
4. Project Overview
10
%
5. Project Organization
10
%
6. Managerial Process
15
%
7. Technical Process
15
%
8. Work Packages
10
%
9. Schedule
10
%
10. Risk Management
7
%
11. Closeout Plan
3
%
12. Annexes
2
%
Save Grading Criteria
Changes will be applied to all future document evaluations.

- Front-end component(s)
    - Component Name - DocumentList.jsx, TaskTracker.jsx
        - Description and purpose - Professor dashboard modules for managing and monitoring student SPMP documents and assigned tasks. DocumentList allows professors to view all student submissions, access detailed reports, override evaluation scores, and delete or download documents. TaskTracker enables professors to assign new tasks to students, update task statuses, set deadlines, and monitor completion rates and overdue items.
        - Component type or format - React Functional Components (JSX) – Dashboard panels with state management via useState/useEffect hooks, API integration via apiService.js, and Tailwind CSS styling.
- Back-end component(s)
    - Component Name - Document & Task Management API
        - Description and purpose - Provides professors with endpoints to retrieve all student SPMP document submissions, view and override evaluation results, delete or download documents, and manage all tasks (create, assign, update, and track status for students). Enforces role-based access control so only professors can access all student data and perform administrative actions.
        - Component type or format - Spring Boot REST API with the following components:
            - DocumentController.java – REST endpoints for professors to retrieve, override, and delete student documents, and download reports.
            - DocumentService.java – Business logic for professor actions on document processing, evaluation, and overrides.
            - TaskController.java – REST endpoints for professors to create, assign, update, and monitor student tasks.
            - TaskService.java – Business logic for professor-driven task assignment and tracking.
            - SPMPDocument.java – Document entity with file metadata, evaluation scores, and professor notes.
            - Task.java – Task entity with assignment, status, and deadline tracking.
            - DocumentRepository.java – JPA repository for professor document operations.
            - TaskRepository.java – JPA repository for professor task operations.

- Object-Oriented Components
  - Class Diagram -



  - Sequence Diagram -



- Data Design
  - ERD or schema -

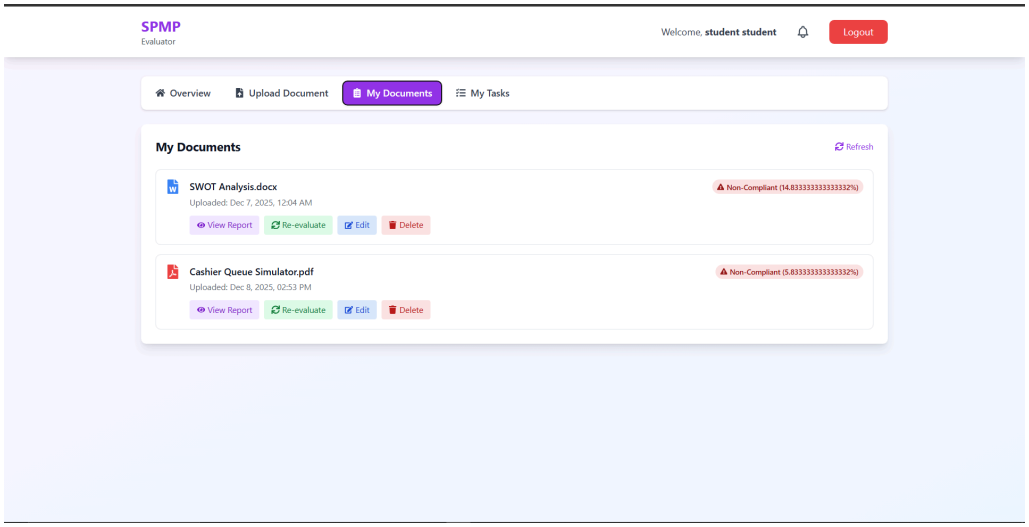
```
CREATE TABLE spmp_documents (
  id BIGINT AUTO_INCREMENT PRIMARY KEY,
  user_id BIGINT NOT NULL,
  file_name VARCHAR(255) NOT NULL,
  file_path VARCHAR(500) NOT NULL,
  file_type VARCHAR(50),
```

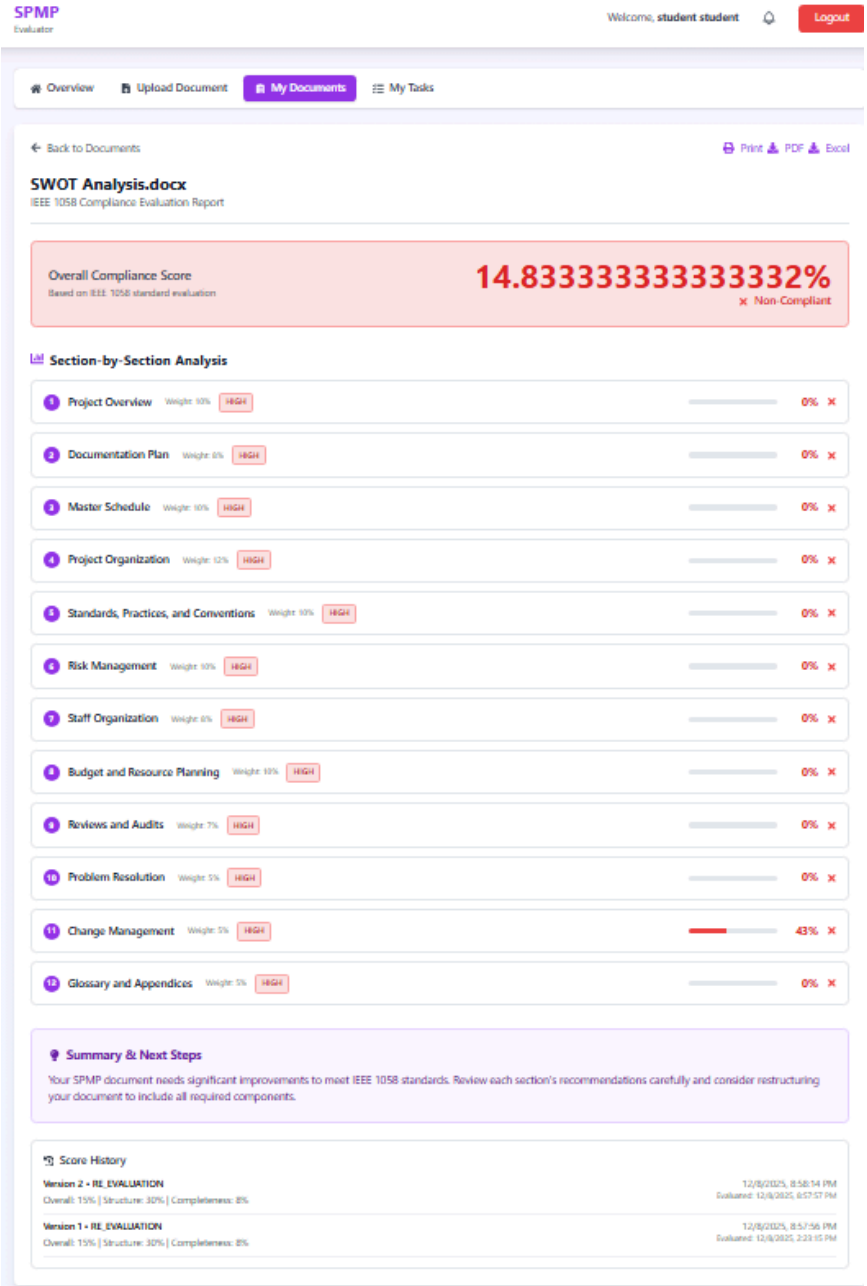
```
    file_size BIGINT,
    status ENUM('PENDING', 'EVALUATED', 'REJECTED') DEFAULT 'PENDING',
    score DOUBLE,
    professor_notes TEXT,
    uploaded_at DATETIME,
    evaluated_at DATETIME,
    FOREIGN KEY (user_id) REFERENCES users(id)
);

CREATE TABLE tasks (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    assigned_to BIGINT NOT NULL,
    created_by BIGINT NOT NULL,
    status ENUM('PENDING', 'IN_PROGRESS', 'COMPLETED') DEFAULT 'PENDING',
    due_date DATETIME,
    completed BOOLEAN DEFAULT FALSE,
    created_at DATETIME,
    updated_at DATETIME,
    FOREIGN KEY (assigned_to) REFERENCES users(id),
    FOREIGN KEY (created_by) REFERENCES users(id)
);
```

Module 3 - Document Parsing

3.1 *Automated Parser Module*

- User Interface Design -



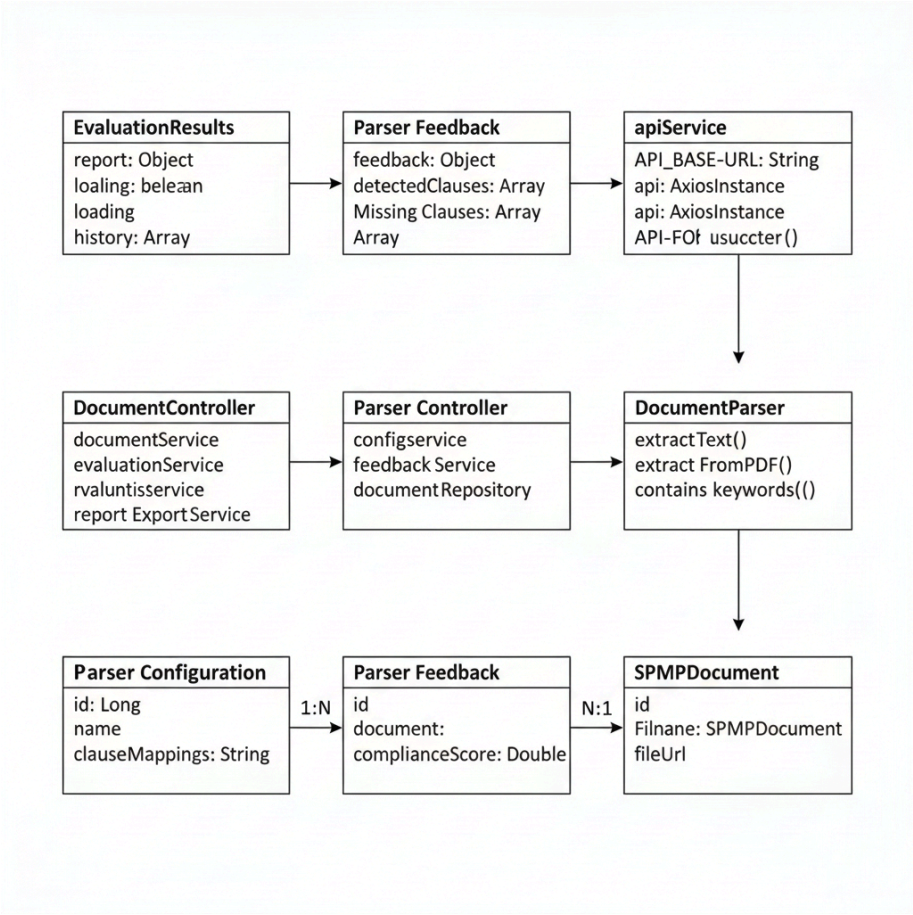


- Front-end component(s)
    - Component Name - EvaluationResults.jsx, ParserFeedback.jsx, apiService.js
        - Description and purpose - Dashboard modules for IEEE 1058 compliance evaluation and report viewing. EvaluationResults displays the compliance report with overall score, section-by-section analysis with severity indicators (HIGH/MEDIUM/LOW), expandable findings and recommendations, and export options (PDF/Excel). ParserFeedback shows AI parser analysis including detected IEEE 1058 clauses with location and scores, missing/incomplete clauses with severity ratings, prioritized AI recommendations, and detailed analysis report.

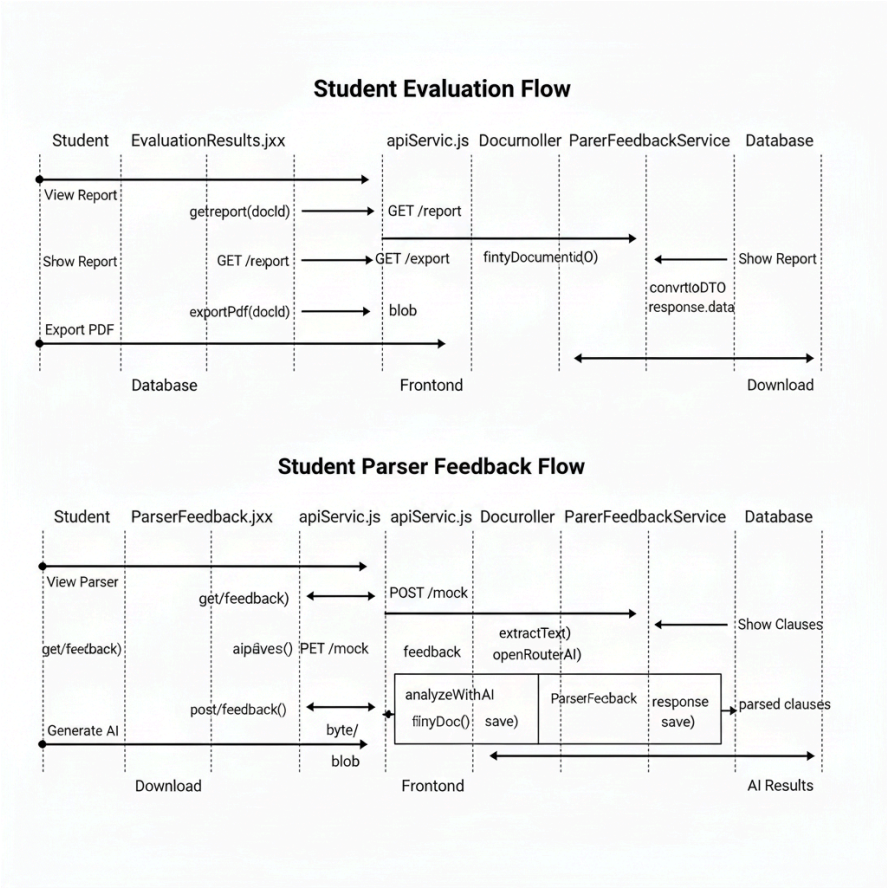
apiService.js provides the API integration layer with documentAPI methods (evaluate, reEvaluate, getReport, exportPdf, exportExcel) and direct api calls for parser feedback endpoints.

- Component type or format - React Functional Components (JSX) - Report viewer panels with state management via useState/useEffect hooks, JSON parsing for clause data, API integration via Axios-based apiService.js with JWT token interceptors, and Tailwind CSS styling with color-coded severity indicators.

- Back-end component(s)
  - Component Name - Document Parsing & Feedback API
    - Description and purpose - Extracts text content from uploaded SPMP documents (PDF/DOCX) using Apache PDFBox and Apache POI libraries. Integrates with OpenRouter AI service for IEEE 1058 clause detection and compliance analysis. Manages parser configurations with custom clause mappings and evaluation rules. Stores detected clauses, missing clauses, and AI-generated recommendations. Professors can create and manage parser configurations with custom keyword mappings.
    - Component type or format - Spring Boot REST API with the following components:
      - ParserController.java - REST endpoints for parser configuration CRUD and feedback retrieval (/api/parser/config, /api/parser/feedback)
      - ParserFeedbackService.java - Business logic for AI document analysis, clause detection, and recommendation generation
      - DocumentParser.java - Utility for extracting text from PDF (PDFBox) and DOCX (Apache POI) files with keyword matching
      - OpenRouterService.java - Integration with OpenRouter AI for intelligent document analysis
      - ParserConfiguration.java - Entity storing clause mappings, custom rules, and configuration metadata
      - ParserFeedback.java - Entity storing compliance score, detected/missing clauses (JSON), recommendations, and analysis report
- Object-Oriented Components
  - Class Diagram -



- Sequence Diagram -



- Data Design
  - ERD or schema -

```
CREATE TABLE parser_configurations (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description VARCHAR(1000),
    clause_mappings TEXT,
    custom_rules TEXT,
    created_by_user_id BIGINT,
    created_at DATETIME NOT NULL,
    updated_at DATETIME,
    is_active BOOLEAN DEFAULT TRUE,
    is_default BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (created_by_user_id) REFERENCES users(id)
);
```

```
CREATE TABLE parser_feedbacks (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    document_id BIGINT NOT NULL,
    parser_config_id BIGINT,
    compliance_score DOUBLE NOT NULL,
    detected_clauses TEXT,
    missing_clauses TEXT,
    recommendations TEXT,
    analysis_report TEXT,
    analyzed_at DATETIME NOT NULL,
    parser_version VARCHAR(50),
    status ENUM('PENDING', 'IN_PROGRESS', 'COMPLETED', 'FAILED') DEFAULT
'PENDING',
    error_message TEXT,
    FOREIGN KEY (document_id) REFERENCES spmp_documents(id),
    FOREIGN KEY (parser_config_id) REFERENCES parser_configurations(id)
);
```

Module 4

4.1 *Scoring Mechanism Module*

- User Interface Design
- Front-end component(s)
    - Component Name
        - Description and purpose
        - Component type or format
- Back-end component(s)
    - Component Name
        - Description and purpose
        - Component type or format
- Object-Oriented Components
    - Class Diagram

    - Sequence Diagram
- Data Design
    - ERD or schema