SDS DOCUMENT

BPA PROJECT

BY:

HUZAIFA IMRAN (033)

HASEEB IRFAN (029)

Software Design Specification (SDS)

Project Name: FYP Management System (BPA-FYP)

Version: 1.0 **Status:** Completed

Executive Summary

The **FYP Management System (BPA-FYP)** is a comprehensive web-based platform designed to digitize, automate, and streamline the entire Final Year Project lifecycle for university students and faculty. The system addresses the inefficiencies of manual project tracking by providing a centralized environment for **Students, Coordinators, Supervisors, and External Examiners**.

The application manages the project through four distinct phases: **Proposal Submission (Phase 1)**, **Initial Defense (Phase 2)**, **Development & Midterm (Phase 3)**, and **Finalization & Grading (Phase 4)**. Key functionalities include secure document uploads (Proposals, PPTs, SRS/SDS, Final Reports), automated scheduling for defenses, digital weekly progress logs, and a complex **Composite Grading System** that automatically calculates final results based on weighted inputs from multiple evaluators (ID 15%, MD 15%, FD 30%, SE 30%, CE 10%).

Built on the **MERN Stack (MongoDB, Express.js, React.js, Node.js)**, the system ensures real-time status updates, data integrity through schema validation, and role-based access control. This solution significantly reduces administrative overhead, enhances transparency, and ensures accurate academic record-keeping.

Table of Contents

2. System Architecture

- 2.1 Architectural Pattern
- 2.2 Technology Stack

3. Database Design

- 3.1 User Collection (`users`)
- 3.2 Project Collection (`projects`)
- 3.3 Grading Schema Structure

4. Module Specifications

- 4.1 Student Portal (Dashboard & Workflow)
- 4.2 Coordinator Portal (Management & Scheduling)
- 4.3 Supervisor Portal (Consent & Monitoring)
- 4.4 FYP Board Portal (Panel Evaluation)

5. Interface Design (API Endpoints)

- 5.1 Project Management Routes
- 5.2 Evaluation & Grading Routes
- 5.3 Administrative Routes

6. Non-Functional Requirements

- 6.1 Data Integrity & Validation
- 6.2 Scalability & Performance
- 6.3 Usability & User Experience
- 6.4 Security & Access Control

---

1. Introduction

1.1 Purpose

The purpose of the FYP Management System is to automate the manual processes involved in the Final Year Project lifecycle at the university. It serves as a centralized platform connecting Students, Supervisors, Coordinators, and the FYP Board (Panel/External Examiners) to manage proposals, scheduling, documentation, and grading.

1.2 Scope

The system covers four distinct phases of the FYP lifecycle:

- **Phase 1:** Project Proposal submission, Coordinator review, and Supervisor consent.
- **Phase 2:** Initial Defense (ID) scheduling, presentation upload, and panel grading.
- **Phase 3:** Development phase involving SRS/SDS submission, weekly logging, and Midterm Defense (MD).
- **Phase 4:** Finalization, code implementation, final report submission, and Final Defense (FD).

1.3 Definitions & Acronyms

- **SRS:** Software Requirements Specification
- **SDS:** Software Design Specification
- **ID:** Initial Defense (15%)
- **MD:** Midterm Defense (15%)
- **FD:** Final Defense (30%)
- **SE:** Supervisor Evaluation (30%)
- **CE:** Coordinator Evaluation (10%)

---

2. System Architecture

2.1 Architectural Pattern

The system follows the **MERN Stack** (MongoDB, Express.js, React.js, Node.js) architecture, utilizing a **Client-Server** model.

- **Frontend (Client):** React.js (Single Page Application) using Tailwind CSS for styling. Handles user interactions and file uploads.
- **Backend (Server):** Node.js with Express.js. Serves RESTful APIs to handle business logic, file storage (Multer), and database operations.
- **Database:** MongoDB Atlas (Cloud NoSQL). Stores user profiles, project metadata, and grading structures.

2.2 Technology Stack

Component	Technology	Description
Frontend	React.js, Vite	UI Rendering and State Management
Backend	Node.js, Express	API Routing and Server Logic
Database	MongoDB Atlas	Cloud Data Storage
File Storage	Local (`/uploads`)	Storage for Proposals, PPTs, SRS, SDS
Authentication	JWT (JSON Web Tokens)	Secure User Login and Session Management

---

3. Database Design

The database consists of two primary collections: `Users` and `Projects`.

3.1 User Collection (`users`)

Stores credentials and role-based access control.

Field	Type	Description
`_id`	ObjectId	Unique Identifier
`name`	String	Full Name
`email`	String	Unique (Optional for students)
`enrollment`	String	Unique Student ID (e.g., 01-131232-033)
`password`	String	Encrypted Password

Field	Type	Description
`role`	String	Enum: `['student', 'coordinator', 'supervisor', 'board']`

3.2 Project Collection (`projects`)

The core collection tracking the entire lifecycle.

Field	Type	Description
`leaderId`	ObjectId	Reference to Student User
`status`	String	Current State (e.g., `Scheduled for Defense`, `Defense Cleared`)
Phase 1 Data		
`documentUrl`	String	Path to Proposal File
`supervisorId`	ObjectId	Reference to Assigned Supervisor
`supervisorConsent`	String	`Pending`, `Approved`, `Rejected`
Phase 2 Data		
`defenseDate`	Date	Date assigned by Coordinator
`defenseRoom`	String	Room/Lab assigned for defense
`presentationUrl`	String	Path to PPT File
Phase 3 Data		
`srsUrl`	String	Path to SRS Document
`sdsUrl`	String	Path to SDS Document
`weeklyLogs`	Array	List of objects `{weekNo, content, date}`
Phase 4 Data		
`finalReportUrl`	String	Path to Final Report
`finalCodeUrl`	String	Path to Source Code Zip

3.3 Grading Schema (Embedded in `projects`)

Grading is calculated automatically using Mongoose pre-save hooks.

Metric	Weight	Structure	Description
ID	15%	`{supervisor, coordinator, panel, total}`	Split evenly (5+5+5)
MD	15%	`{supervisor, coordinator, panel, total}`	Split evenly (5+5+5)
FD	30%	`{supervisor, coordinator, panel, total}`	Split evenly (10+10+10)
SE	30%	Number (Max 30)	Supervisor Evaluation
CE	10%	Number (Max 10)	Coordinator Evaluation
Total	100%	Number	Sum of all above

---

4. Module Specifications

4.1 Student Portal

Key Features:

- **Dynamic Dashboard:** UI changes based on `project.status` (e.g., Phase 2 card is hidden until Phase 1 is complete).
- **Unified Upload Interface:** Uses `handleUpload(docType)` to upload Proposals, PPTs, SRS, SDS, and Final Reports.
- **Weekly Logs:** Interface to submit progress logs (`weekNo`, `content`).
- **Results View:** Real-time view of grades as they are entered by evaluators.

4.2 Coordinator Portal

Key Features:

- **Proposal Review:** View pending proposals, Approve/Reject.
- **Supervisor Allocation:** Assign supervisors to approved groups.
- **Defense Scheduling:** Assign `defenseDate` and `defenseRoom` for ID, MD, and FD.
- **Grading:** Input "Coordinator Share" of marks for all defenses (ID, MD, FD) and the final CE marks.

4.3 Supervisor Portal

Key Features:

- **Consent Management:** Accept or Reject supervision requests.
- **Grading:** Input "Supervisor Share" of marks for defenses and the specific SE (30%) marks.
- **Monitoring:** View uploaded SRS/SDS and Weekly Logs of supervised students.

4.4 FYP Board (Panel) Portal

Key Features:

- **Phase-Based Tabs:** Toggle between `Proposal`, `Interim`, and `Final` defense lists.
- **Document Access:** One-click download for PPTs (Phase 2), SRS/SDS (Phase 3), and Final Reports (Phase 4).
- **Grading System:** Input "Panel Share" of marks.
- **Feedback System:** Submit textual feedback (`defenseFeedback`) that appears on the Student Dashboard.

---

5. Interface Design (API Endpoints)

5.1 Project Management Routes

- `GET /api/projects/my-project/:studentId` - Fetches full project details.
- `POST /api/projects/upload-doc` - Universal upload handler.
  - **Body:** `studentId`, `docType` ('proposal', 'ppt', 'srs', 'final').
- `POST /api/projects/submit-log` - Appends entry to `weeklyLogs` array.

5.2 Evaluation & Grading Routes

- `GET /api/projects/evaluation-list/:type` - Fetches projects filtered by phase (`proposal`, `interim`, `final`).

- `PUT /api/projects/grade-defense/:id` - The core grading logic.
    - **Body:** `{ phase: 'id'|'md'|'fd', role: 'panel'|'supervisor', marks: 5 }`
    - **Logic:** Updates the specific mark field. Checks if all 3 evaluators have graded. If yes, updates status to `Defense Cleared` or `Interim Cleared`.

5.3 Administrative Routes

- `PUT /api/projects/assign-defense/:id` - Updates `defenseDate`, `defenseRoom`, and sets status to `Scheduled for Defense`.
- `PUT /api/projects/assign-supervisor/:id` - Links a Supervisor ID to the project.

---

6. Non-Functional Requirements

1. **Data Integrity:** The system uses MongoDB `schema validation` (e.g., `max: 5`) to prevent incorrect mark entry (e.g., entering 15 marks for a 5-mark component).
2. **Scalability:** Hosted on MongoDB Atlas (Cloud) to support multiple concurrent connections.
3. **Usability:** React components use conditional rendering to only show relevant tasks (e.g., "Upload PPT" button only appears after the date is scheduled).
4. **Security:** Passwords stored (conceptually) or managed via secure Auth routes; Backend routes validate IDs before updates.