System Design Document (SDD)

Project: Online Course and Exam Platform
Tech Stack: React (Frontend) + Spring Boot (Backend) + PostgreSQL (Database)
Use Case: Enables digital learning with automated testing, progress tracking, live and recorded classes, daily practice problems, and certificate generation.

1. Introduction

The **Online Course and Exam Platform** is a web application designed to facilitate **digital learning** for students, with **instructors creating courses and exams**. The platform includes:

- **Live classes** for real-time learning.
- **Recorded classes** for on-demand access.
- **Daily Practice Problems (DPP)** for continuous learning.
- **Certificate generation** after course completion.

The platform is **role-based**, responsive, and supports automated MCQ exams with timers and progress tracking.

2. Requirements

2.1 Functional Requirements

Admin Module

- Manage users (Students, Instructors)
- Assign courses and exams
- Generate system-wide reports

Instructor Module

- Create and manage courses and exams
- Schedule live classes
- Upload recorded classes
- Add daily practice problems (DPP)
- View student performance

Student Module

- Browse and enroll in courses
- Attend live classes
- Access recorded classes
- Attempt exams and DPP
- Track progress and download certificates

Exam Engine

- Timer-based MCQ evaluation
- Auto-grading of exams and DPP
- Prevent multiple attempts unless allowed

Reporting & Certificate Module

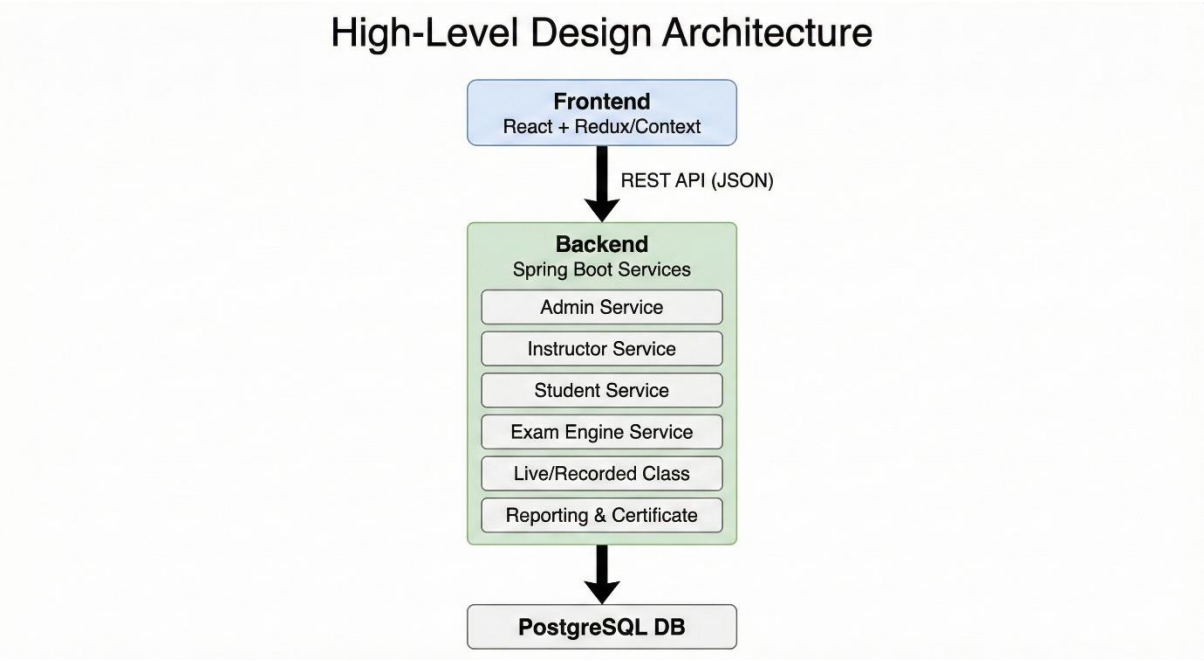
- Generate performance reports for students and courses
- Automatically generate PDF certificates upon course completion

2.2 Non-Functional Requirements

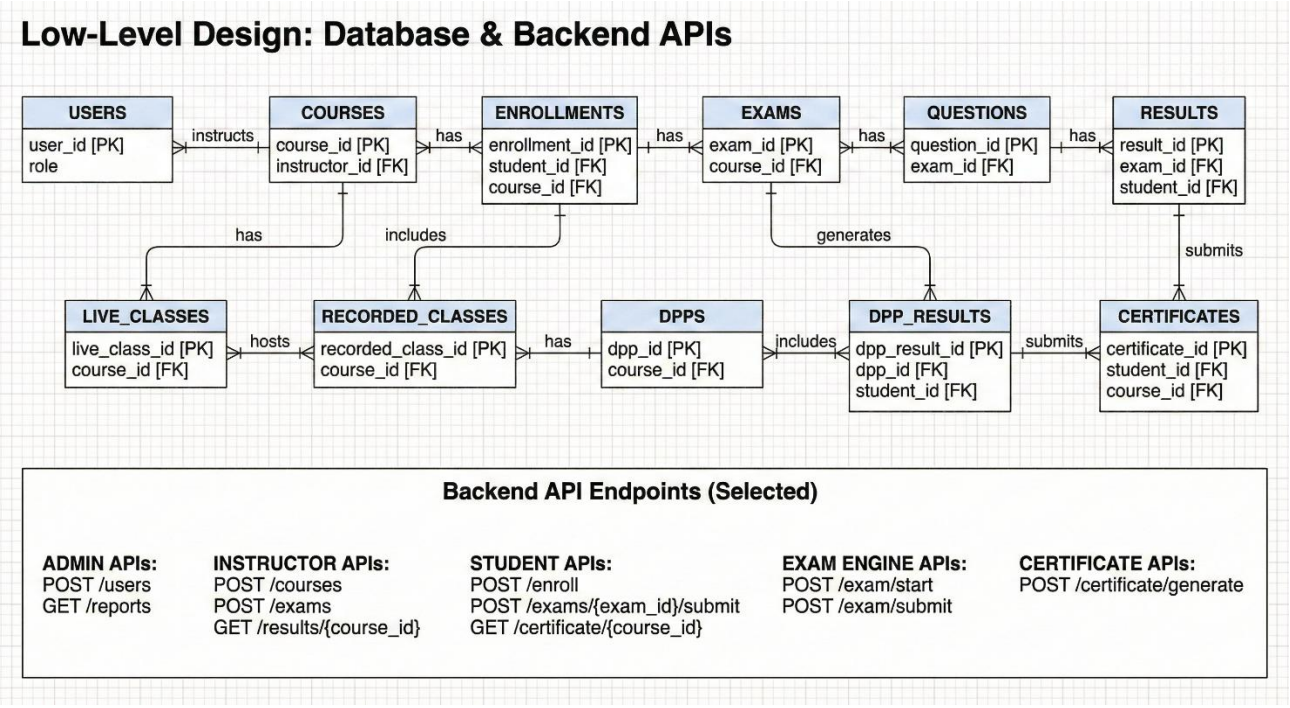
- **Scalability:** Handle 10,000+ concurrent students.
- **Performance:** Accurate exam timers and fast DPP grading.
- **Security:** Role-based access, encrypted passwords, secure streaming URLs.
- **Responsiveness:** Works on desktop, tablet, and mobile.
- **Reliability:** Data persistence with PostgreSQL and backup support.

3. High-Level Design (HLD)

3.1 Architecture Overview

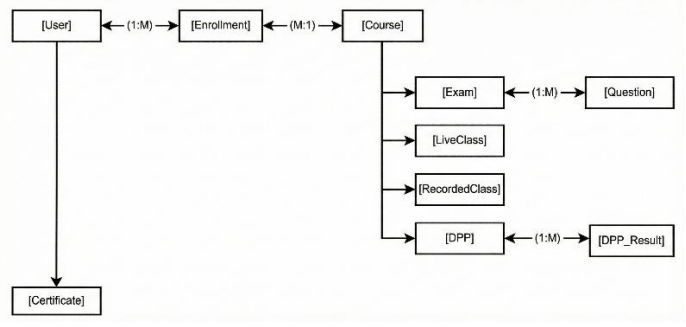


4. Low-Level Design (LLD)



4.1 Database Design (Tables)

Table	Columns	Description
User	user_id, name, email, password, role	Stores all users (Admin, Instructor, Student)
Course	course_id, name, description, instructor_id	Stores course details
Enrollment	enrollment_id, course_id, student_id, progress	Tracks student enrollment and progress
Exam	exam_id, course_id, title, duration, max_marks	Stores exam details
Question	question_id, exam_id, question_text, options(JSON), correct_option	Stores MCQs for exams
Result	result_id, exam_id, student_id, score, status	Stores exam results
LiveClass	live_class_id, course_id, instructor_id, title, url, date_time	Stores scheduled live classes
RecordedClass	recorded_class_id, course_id, title, video_url, upload_date	Stores uploaded recorded sessions
DPP	dpp_id, course_id, title, questions(JSON), publish_date	Stores daily practice problems
DPP_Result	dpp_result_id, dpp_id, student_id, score, submission_time	Tracks DPP submissions and scores
Certificate	certificate_id, student_id, course_id, issue_date, pdf_url	Stores PDF certificate info for completed courses



5. Backend API Design

1. User Management APIs

- **Purpose:** Handle login, registration, and role-based access.
- **Key Actions:**
    - Create, update, delete, and fetch users.
    - Authenticate users (Admin, Instructor, Student).
- **Flow Example:**
    - Admin adds a student → Frontend sends user details → Backend creates user → Confirmation returned.

2. Course Management APIs

- **Purpose:** Manage courses and student enrollments.
- **Key Actions:**
    - Instructor creates courses.
    - Student enrolls in courses.
    - Track student progress per course.
- **Flow Example:**
    - Instructor creates a course → Backend saves course → Students see the new course in their dashboard.

3. Exam Management APIs

- **Purpose:** Handle all exam-related functionality.
- **Key Actions:**
    - Create exams with MCQs and timers.
    - Student starts and submits exams.
    - Auto-grade and return results.
- **Flow Example:**
    - Student starts exam → Backend starts timer → Student submits answers → Backend calculates score → Result displayed.

4. Live & Recorded Class APIs

- **Purpose:** Enable online teaching and access to recorded lectures.
- **Key Actions:**
    - Schedule live classes (Instructor).
    - Join live class (Student).
    - Upload and fetch recorded sessions.
- **Flow Example:**
    - Instructor schedules class → Students see class in dashboard → Click "Join" → Open video link → Attendance tracked.

5. DPP (Daily Practice Problem) APIs

- **Purpose:** Allow instructors to post daily practice questions and track student performance.
- **Key Actions:**
    - Create DPP questions.
    - Students attempt DPP → Auto-grade → Result stored.
- **Flow Example:**
    - Instructor posts DPP → Student attempts → Backend grades → Student dashboard updated.

6. Certificate & Reporting APIs

- **Purpose:** Generate reports and certificates automatically.
- **Key Actions:**
    - Generate PDF certificate when student completes all course requirements.
    - Generate course or student performance reports.
- **Flow Example:**
    - Student completes all exams/DPP → Backend generates certificate → Student downloads from dashboard.

6. Modules

Frontend Modules (3)

Module	Features
Admin Dashboard	Manage users, generate reports, analytics charts
Instructor Dashboard	Create courses/exams, upload recorded classes, schedule live classes, add DPP
Student Dashboard	Attend live classes, access recorded classes, attempt exams & DPP, track progress, download certificates

Backend Modules (5)

Module	Features
User Management (Auth/Admin)	Login, registration, role-based access
Course Management	CRUD courses, track progress, enrollments
Exam Management (Exam Engine)	Timer-based MCQs, auto-grading, DPP
Live/Recorded Class Module	Schedule live classes, upload & fetch recorded sessions
Reporting & Certificate Module	Generate performance reports, create PDF certificates after course completion

6. Sequence Flows

6.1 Student Attending Live Class

1. Student selects course → fetch live class list via API.
2. Student clicks **Join** → Opens live streaming link.
3. Backend tracks attendance.

6.2 Student Accessing Recorded Class

1. Student selects course → fetch recorded class list.
2. Click video → React Player plays video.

6.3 Student Submitting DPP

1. Instructor uploads DPP → saved in DB.
2. Student attempts DPP → backend auto-grades.
3. Results stored in DPP_Result table → visible on dashboard.

6.4 Certificate Generation

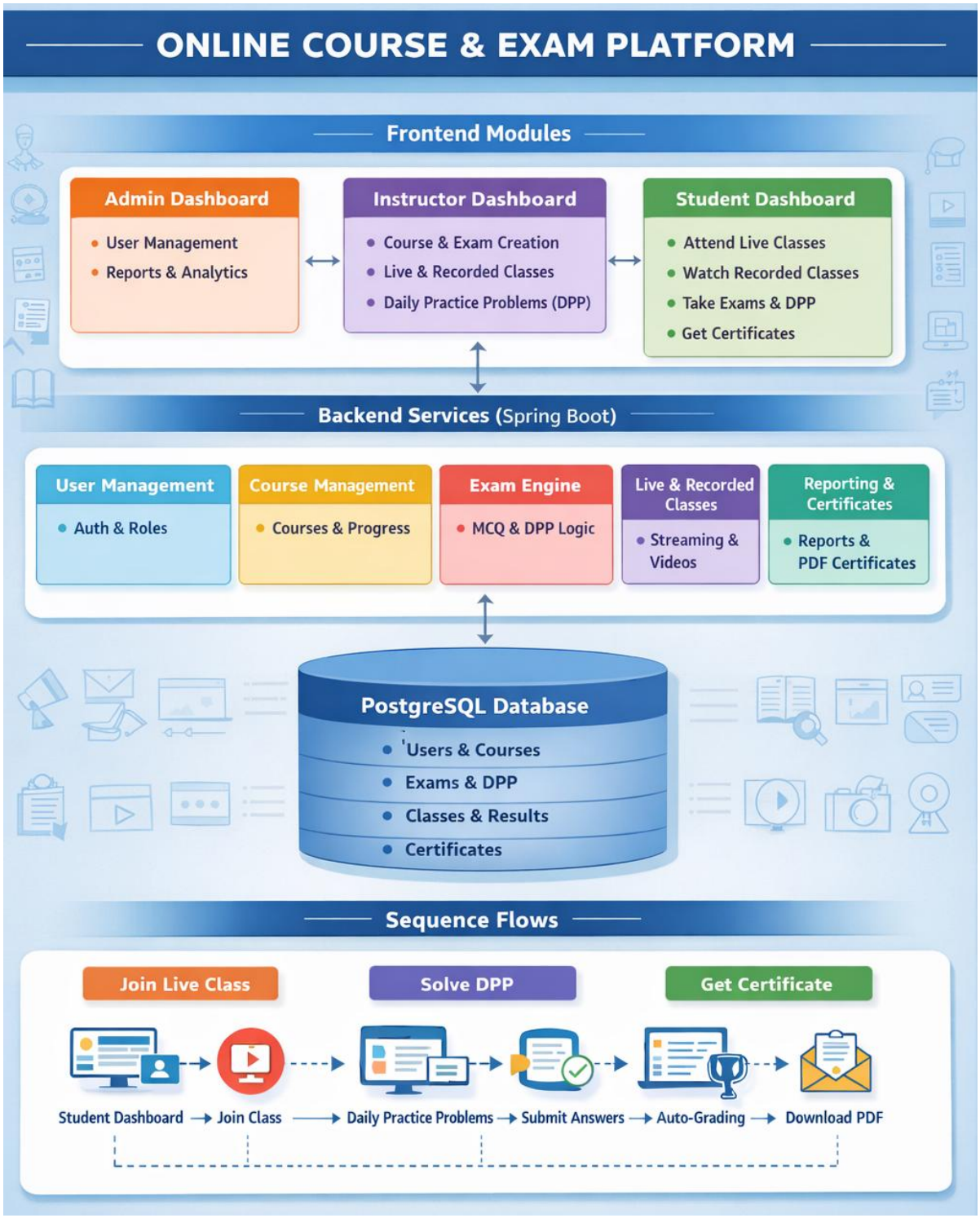
1. Backend checks course progress → all exams/DPP completed.
2. Generate PDF certificate using **iText / Apache PDFBox**.
3. Save URL in Certificate table → student can download.

7. Technology / Tools

- **Frontend:** React, Redux/Context API, Axios, Material UI, React Player (video)
- **Backend:** Spring Boot, Spring Security, JWT, REST APIs, iText/Apache PDFBox (PDF generation)
- **Database:** PostgreSQL
- **Version Control:** GitHub

8. Additional Notes

- **React Router** for SPA navigation.
- **PostgreSQL JSON type** for MCQ and DPP options.
- Unit and integration tests for backend using JUnit.
- Secure streaming URLs for live and recorded classes.
- PDF certificates downloadable only for enrolled students who completed the course.



“Figure : System Architecture and Module Flow for Online Course & Exam Platform”