🎓 Case Study: E-Learning Management System (ELMS)

1. Overview

The **E-Learning Management System (ELMS)** is an online learning platform that enables **students, instructors, and admins** to collaborate digitally.
It supports **video lessons, quizzes, assignments, grading, certificates, and progress tracking** with **secure JWT + OAuth authentication**.

2. Scope

- **Admin**
  - Manage courses, instructors, students
  - Approve/reject instructor applications
  - Monitor platform usage reports
- **Instructor**
  - Create and publish lessons (video/reading material)
  - Create assignments & quizzes
  - Grade student submissions
- **Student**
  - Enroll in courses
  - Submit assignments & take quizzes
  - Track learning progress
  - Download certificates

3. Tech Stack

- **Backend**: Spring Boot, Spring Security (JWT + OAuth 2.0 for Google/Facebook login), Swagger
- **Frontend**: React + Bootstrap (responsive UI), Axios
- **Database**: MongoDB (document-based for flexible course/quiz content)

- **File Storage**: AWS S3 / Local Storage (for video lessons & documents)
- **Reports**: Chart.js / Recharts

4. System Architecture

- **React Frontend** → Axios → **Spring Boot REST API** → MongoDB
- **Authentication**: JWT (email/password login) + OAuth (Google/Facebook)
- **Swagger**: Auto-generated API docs
- **Certificate Generation**: PDF/Badge generation from backend

5. Key Features with API Integration

🔷 Authentication & Security

- **Register (Student/Instructor)** → `POST /auth/register`
- **Login (JWT)** → `POST /auth/login`
- **OAuth Login (Google/Facebook)** → `POST /auth/oauth`
- JWT attached to every secured API call.

🔷 Course Management

- **Create Course (Instructor/Admin)** → `POST /api/courses`
- **Get All Courses** → `GET /api/courses`
- **Get Course by ID** → `GET /api/courses/{id}`
- **Enroll Student** → `POST /api/courses/{id}/enroll`
- **Delete Course (Admin)** → `DELETE /api/courses/{id}`

🔷 Lesson & Content Management

- **Upload Lesson (Video/PDF)** → `POST /api/courses/{id}/lessons`
- **Get Lessons** → `GET /api/courses/{id}/lessons`
- **Update Lesson** → `PUT /api/lessons/{id}`

- **Delete Lesson** → `DELETE /api/lessons/{id}`

🔷 Assignments

- **Create Assignment (Instructor)** → `POST /api/assignments`
- **Submit Assignment (Student)** → `POST /api/assignments/{id}/submit`
- **Grade Assignment (Instructor)** → `PUT /api/assignments/{id}/grade`
- **View Grades (Student)** → `GET /api/assignments/{id}/grades`

🔷 Quizzes

- **Create Quiz** → `POST /api/quizzes`
- **Attempt Quiz (Student)** → `POST /api/quizzes/{id}/attempt`
- **Get Quiz Results** → `GET /api/quizzes/{id}/results`

🔷 Progress & Certificates

- **Track Progress (Student)** → `GET /api/students/{id}/progress`
- **Generate Certificate** → `POST /api/certificates/{courseId}/generate`
- **Download Certificate** → `GET /api/certificates/{id}/download`

🔷 Reports

- **Admin – Usage Report** → `GET /api/reports/platform-usage`
- **Instructor – Course Report** → `GET /api/reports/courses/{id}`
- **Student – Progress Dashboard** → `GET /api/reports/students/{id}`

6. Swagger Integration

- Spring Boot integrated with **springdoc-openapi-ui**
- Accessible at: `http://localhost:8080/swagger-ui.html`

- Auto-generates **API documentation** with request/response schemas.

7. Frontend (React + Bootstrap)

- **Login/Register Page** (JWT + OAuth login options)
- **Dashboard**:
    - Admin → Manage users & courses
    - Instructor → Course creation, grading
    - Student → Enrolled courses, assignments, progress
- **Course Detail Page**: Lessons, assignments, quizzes
- **Progress Dashboard**: Recharts/Chart.js visualizations
- **Certificate Download Page**

8. Sample Folder Structure

📂 Backend (Spring Boot + JWT + OAuth + Swagger)

```
elms-backend/
 ├── controller       # REST APIs
 ├── service          # Business logic
 ├── repository       # Mongo Repositories
 ├── model            # Entities (Course, Lesson, User,
Assignment, Quiz, Certificate)
 ├── security         # JWT + OAuth configs
 ├── exception        # Custom exception handling
 └── resources
     └── application.yml
```

📂 Frontend (React + Bootstrap)

```
elms-frontend/
 ├── src/
 │   ├── components/
 │   │   ├── Login.js
 │   │   ├── Dashboard.js
 │   │   ├── CourseList.js
 │   │   ├── LessonViewer.js
 │       │         ├── Assignments.js
 │       │         ├── Quizzes.js
 │       │         └── Reports.js
 │       ├── services/ (Axios API calls)
 │       └── App.js
```

9. Security Implementation

- **JWT Authentication** for email/password login
- **OAuth 2.0** for Google/Facebook single sign-on
- **Role-based access**:
    - `/api/admin/**` → Admin only
    - `/api/instructor/**` → Instructor only
    - `/api/student/**` → Student only

10. Benefits

✅ Flexible **MongoDB schema** for lessons/quizzes (dynamic content)
✅ **JWT + OAuth** ensures secure access and social login support
✅ **Swagger API docs** for quick API exploration
✅ **Bootstrap + React** UI for responsive dashboards
✅ **Real-time progress tracking** with reports
✅ **Automatic certificate generation** upon course completion