🎓 AI Teaching Assistant

AI-powered lecture summarization platform built with **Spring Boot**, **React**, **Apache PDFBox**, and pluggable **LLM integration** (OpenAI / Claude / Gemini).

📁 Project Structure

```
ai-teaching-assistant/
├── backend/                          # Spring Boot (Java 17)
│   ├── pom.xml
│   └── src/main/java/com/ai/teachingassistant/
│       ├── TeachingAssistantApplication.java
│       ├── controller/LectureController.java
│       ├── service/
│       │   ├── LectureService.java
│       │   ├── PdfExtractionService.java
│       │   └── SummarizationService.java
│       ├── client/LlmClient.java       ← Strategy pattern: OpenAI / Claude / Gemini
│       ├── dto/
│       │   ├── LectureRequest.java
│       │   └── SummaryResponse.java
│       ├── model/Lecture.java
│       └── exception/GlobalExceptionHandler.java
│
└── frontend/                         # React 18 + Vite
    ├── package.json
    ├── vite.config.js
    ├── index.html
    └── src/
        ├── App.jsx                   ← Root state management
        ├── main.jsx
        ├── index.css
        ├── components/
        │   ├── UploadLecture.jsx     ← Drag-and-drop upload
        │   └── SummaryView.jsx       ← Structured summary display
        └── services/api.js           ← Backend API communication
```

⚙️ Setup Instructions

Prerequisites

- Java 17+

- Maven 3.8+
- Node.js 18+ & npm

---

1. Backend Setup

```bash
cd backend
```

Set your LLM provider API key as an environment variable:

```bash
# For OpenAI (default)
export OPENAI_API_KEY=sk-your-key-here

# For Claude
export CLAUDE_API_KEY=sk-ant-your-key-here

# For Gemini
export GEMINI_API_KEY=your-key-here
```

Switch provider in `src/main/resources/application.properties`:

```properties
llm.provider=openai    # Options: openai | claude | gemini
```

Run the backend:

```bash
mvn spring-boot:run
```

Backend will start on **http://localhost:8080**

---

2. Frontend Setup

```bash
cd frontend
npm install
npm run dev
```

Frontend will start on **http://localhost:5173**

🚀 API Endpoints

Method	Endpoint	Description
POST	`/api/lecture/summarize`	Upload PDF → AI Summary (JSON)
GET	`/api/lecture/health`	Health check + active provider

Sample Response

json


```
{
  "title": "Introduction to Machine Learning",
  "keyPoints": [
    "Supervised learning uses labeled training data",
    "Neural networks are inspired by the human brain"
  ],
  "definitions": [
    "Gradient Descent: An optimization algorithm that minimizes loss functions",
    "Overfitting: When a model performs well on training data but poorly on unseen data"
  ],
  "examPoints": [
    "Know the difference between supervised, unsupervised, and reinforcement learning",
    "Understand bias-variance tradeoff"
  ],
  "fileName": "ml_lecture.pdf",
  "provider": "openai",
  "generatedAt": "2026-02-19T10:30:00",
  "pageCount": 12
}
```

🔧 Configuration

All settings in `backend/src/main/resources/application.properties`:

Property	Default	Description
`llm.provider`	`openai`	Active AI provider
`openai.model`	`gpt-4`	OpenAI model version
`claude.model`	`claude-sonnet-4-6`	Claude model version
`server.port`	`8080`	Backend port
Max upload size	`10MB`	Configured in application.properties

✅ Functional Requirements Coverage

Requirement	Status
FR-1.1 Accept PDF only	✅ Validated in controller + service
FR-1.2 Max 10MB	✅ Spring multipart + manual validation
FR-1.3 File integrity check	✅ PDFBox load + encrypted check
FR-2.1 Extract all pages	✅ PDFTextStripper full document
FR-2.2 Multi-column layout	✅ setSortByPosition(true)
FR-2.3 Corrupted PDF detection	✅ IOException + encrypted check
FR-3.1 Key concepts list	✅ [KEY_CONCEPTS] section
FR-3.2 Important definitions	✅ [DEFINITIONS] section
FR-3.3 Exam-focused points	✅ [EXAM_POINTS] section
FR-3.4 Multiple LLM providers	✅ Strategy pattern in LlmClient
FR-4.1 Display title	✅ SummaryView title
FR-4.2 Formatted key points	✅ Numbered list in SummaryView
FR-4.3 Processing indicator	✅ Spinner + progress dots

Non-Functional Requirements Coverage

Requirement	Implementation
NFR-1.1 < 30s for 10-page PDF	LLM timeout set to 60s; extraction is sub-second
NFR-2.1 HTTPS	Configure reverse proxy (Nginx/AWS ALB) for production
NFR-2.2 API keys in env vars	`${OPENAI_API_KEY}` in application.properties
NFR-2.3 Input validation	Controller + PdfExtractionService validation
NFR-3.2 Retry logic	Extend LlmClient with Spring Retry if needed
NFR-3.3 Error logging	SLF4J @Slf4j across all services
NFR-4.3 Dependency injection	@RequiredArgsConstructor across all services