DoodleOnMoodle

AI-Powered Learning Management System (LMS) Module

Mayank A, Sakshi C, Ghanshyam D, Sandeep K

Team ZeroDowntime

September 13, 2025

Agenda

1. Problem Statement
2. Purpose & Goals
3. User Personas & Pain Points
4. Project Scope (Key Features)
5. Requirements
6. Architecture
7. Cloud Deployment
8. Testing Plan
9. Weekly Milestones
10. Acceptance Criteria

The Problem

- **For Educators:**
  - Significant time spent on administrative tasks.
  - Tiresome assessment creation process.
  - Lack of data-driven insights into student weaknesses.

- **For Students:**
  - Generic, one-size-fits-all learning experience.
  - Inefficient studying due to a lack of personalization.
  - Lack of targeted feedback and tailored materials.

Purpose and Goals

Purpose

To develop an intelligent Moodle plugin that streamlines teaching and learning by automating assessment creation, providing personalized feedback, and generating adaptive learning paths.

Key Goals

- **Professors:** Reduce workload and gain insights.
- **Students:** Achieve a personalized learning journey.
- **System:** Create a scalable and integrable module.

User Personas

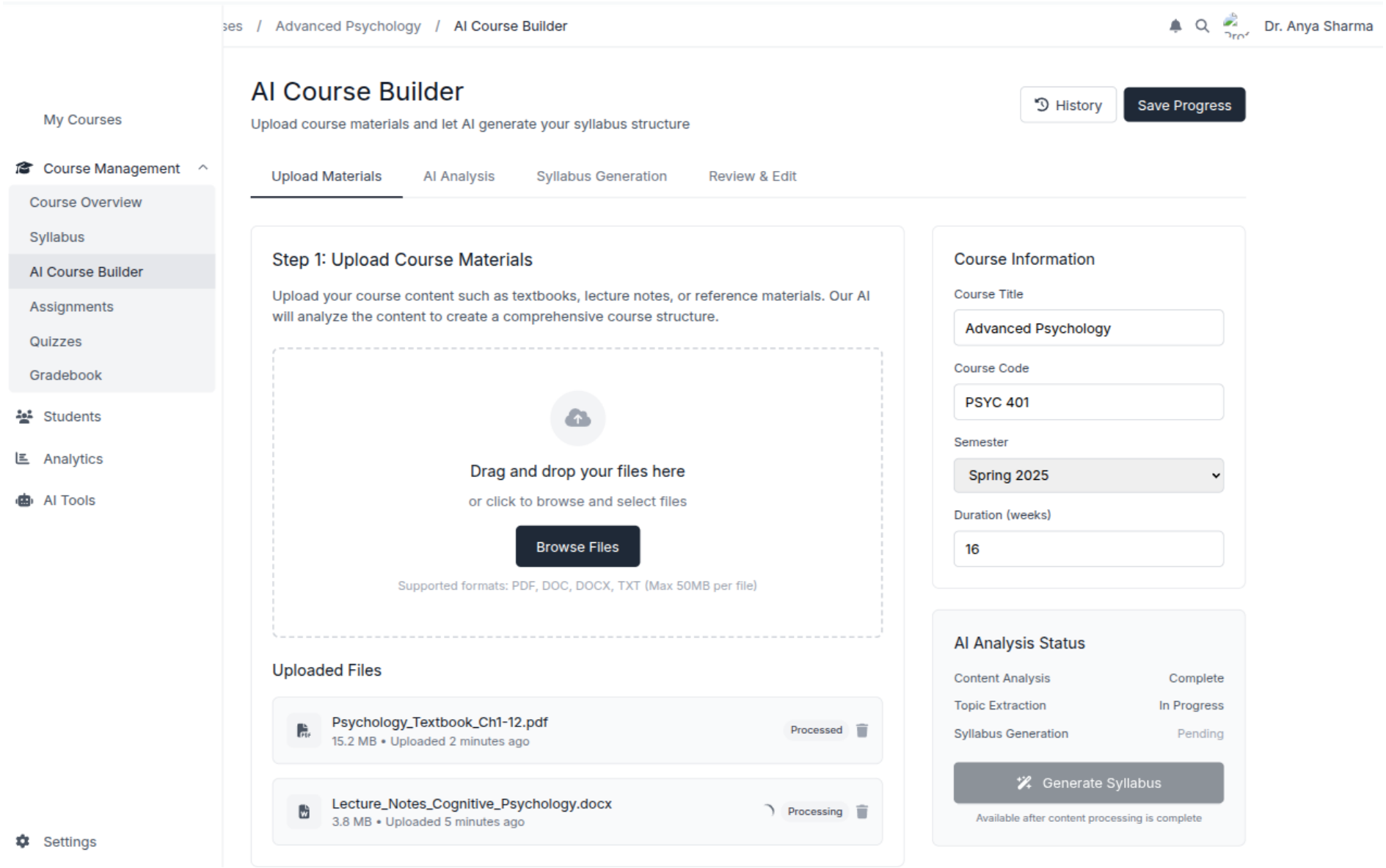
Dr. Anya Sharma (Professor)

- **Goals:** Timely assessment of students, provide constructive feedback based on class level analysis.
- **Pain Points:**
  - Assessment creation.
  - Lack of data-driven insights.

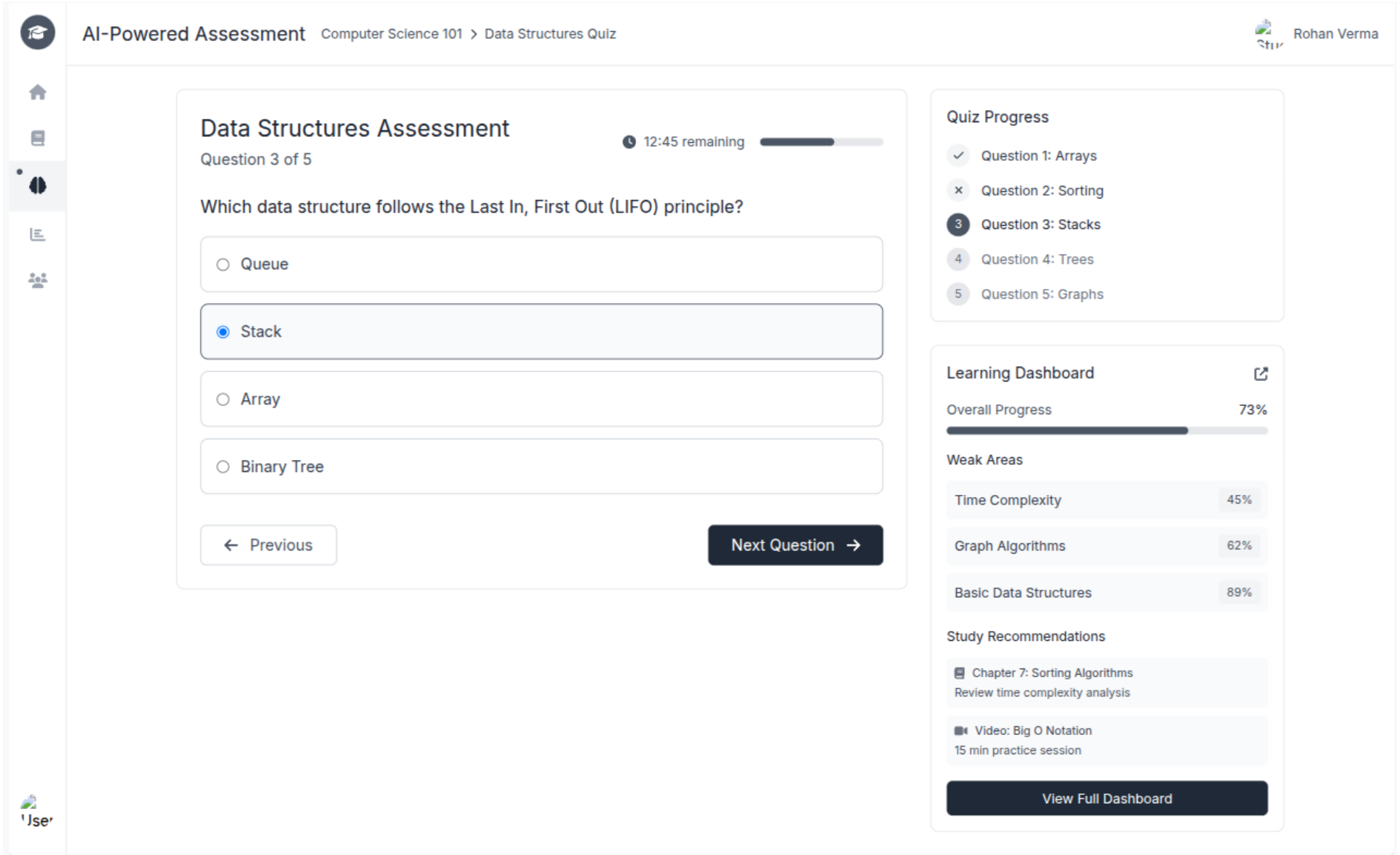
Rohan Verma (Student)

- **Goals:** Understand weak points and improve.
- **Pain Points:**
  - No personalized feedback.
  - Inefficient studying.
  - No progress visibility.

Professor UI Workflow



Student UI Workflow



Feature Scope

- **FC-01:** AI-Powered Course Assessment Generation
  - Upload course materials and segregate it into topics and subtopics.
- **FC-02:** Automated Assessment Generation
  - LLM generates a question bank consisting of MCQs, MSQs, short-answer questions, etc.
- **FC-03:** Adaptive Student Learning Module
  - System identifies student's weaknesses based on performance.
- **FC-04:** Personalized Study Recommendations (RAG)
  - RAG-based recommendations from uploaded materials.
- **FC-05:** Professor Analytics Dashboard
  - View class-wide performance and generate custom exams.

Key Requirements

Functional Requirements (FR)

- **FR-01:** Upload document to segregate content into topics and subtopics syllabus.
- **FR-02:** Edit and save the generated syllabus outline.
- **FR-03:** Request specific number and type of questions.
- **FR-04:** Get study material recommendations after an incorrect answer.
- **FR-05:** Review weak areas identified in submitted assessments.
- **FR-06:** View a dashboard showing difficult topics for the class.
- **FR-07:** Generate a final exam for a set of topics.

Key Requirements

Non-Functional Requirements (NFR)

- **NFR-01:** Performance: Syllabus generation in ≈10 min; quiz creation in ≈5 min.
- **NFR-02:** Scalability: System should support 100 concurrent users.
- **NFR-03:** Availability: 95% uptime.
- **NFR-04:** Usability: Intuitive UI requiring minimal training.
- **NFR-05:** Security: Data encrypted in transit and at rest.
- **NFR-06:** Extensibility: Modular architecture for future LMS integration.

Proposed Architecture

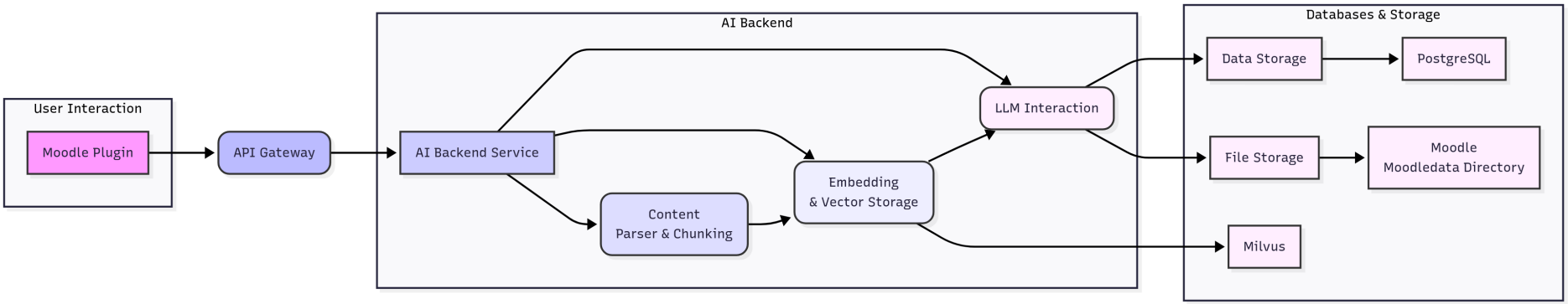


Figure: System Architecture Flow

Technology Stack

Component	Category	Technology Choice
Moodle Plugin	Core Language	PHP 8.x
Moodle Plugin	Frontend UI	React.js
AI Microservice	Language	Python / FastAPI
AI Microservice	LLM API	OpenAI API (GPT-4o)
Vector Database	Data Storage	Milvus
Primary DB	Data Storage	PostgreSQL 15+

Cloud Deployment Strategy

- **Source Control:** Git, hosted on GitHub.
- **CI/CD Pipeline:** GitHub Actions will be used to automate the build, test, and deployment process.
  - On code push, it will run linters and tests.
  - It will build a Docker image and push it to a container registry.
  - The new image will be deployed to the AWS Fargate service.
- **Infrastructure as Code (IaC):** Terraform will manage all cloud resources, including Fargate, RDS (PostgreSQL), and S3.
- **Monitoring & Logging:** AWS CloudWatch will be used to collect logs, monitor resource utilization, and set up alarms for performance issues or service unavailability.
- **Health Checks:** A health endpoint on the FastAPI service will allow Fargate to monitor the application's health and automatically restart unhealthy containers.

Testing Plan

- **Unit Testing:** We will use **pytest** to test individual functions
- **API & Web-App Testing:** Using Postman and PlayWright.
- **Integration Testing:** We will verify that all components, from the Moodle plugin to the AI backend and databases, work together seamlessly.
- **Performance Testing:** We will use **locust.io** to simulate 100 concurrent users to ensure the system remains fast and responsive under load.


AM I TESTING THIS PROGRAM
OR IS IT TESTING ME?

Project Milestones

- **Week 1:** Project Kickoff & Foundation
- **Week 2:** Core Parsing & Embedding
- **Week 3:** Syllabus Content Segregation & UI
- **Week 4:** Midterm Review Prep (RAG)
- **Week 5:** Assessment Generation
- **Week 6:** Professor Analytics
- **Week 7:** Custom Exam & Integration
- **Week 8:** Final Submission & Polish

Acceptance Criteria

- All core features implemented (FC-01 to FC-05).
- All functional requirements (FR-01 to FR-07) met.
- All non-functional requirements (NFR-01 to NFR-06) met.
- A final presentation must demonstrate a successful end-to-end workflow.

Thank You!

Questions?