Product Design

Team-4 : Arshiya Noureen, Sanyam Agarwal, Kushal Mangla, Vishak Kashyap, Shreyas Deb

Design Model

Class No.	Class Name	Class State(Attributes)	Class Behavior

1	User	`userId`
		`email`
		`password`
		`name`
		`mobileUser`
		`role`
		`preferences`
		`createdAt`
		`updatedAt`
			`comparePassword` – Compares the given password with the stored password.
			
			`changePassword` – Updates the password after verifying the old one.
			
			`login` – Authenticates the user and returns the user details.
			
			`signup` – Registers a new user with the given details.
			
			`editProfile`– Updates user profile information.

2	Coach	`(Inherits from User)`
		`coursesTaught`
		`learners`
			`uploadContent`– Uploads new course material.
			
			`updateContent` – Modifies existing course content.
			
			`viewAssignedStudents` – Lists all students enrolled in the coach's courses.
			
			`trackStudentProgress` – Retrieves a student's progress report.
			
			`gradeAssignment` – Assigns grades to a student's submission.

3	Learner	(Inherits from User) enrolledCourses	`viewProfile` – Displays the learner's profile. `updateProfile` – Allows the learner to edit their profile details. `viewEnrolledCourses` – Returns a list of courses the learner is enrolled in. `accessCourseDetails` – Fetches course details.

			`viewAssignments` – Lists assignments for a given course. `submitAssignment` – Submits an assignment for grading. `viewGrades` – Displays grades for completed assignments.
4	Admin	(Inherits from User)	`createUser` – Creates a new user account. `manageUser`– Updates user information. `removeUser` – Deletes a user account. `viewAnalytics` – Retrieves system-wide analytics. `configureSystem`– Updates platform settings. `viewContentPerformance` – Analyzes the performance of uploaded content. `manageCategories`– Organizes content into categories. `generateReports` – Generates system or user reports.

5	Course	`courseId`
		`title`
		`description`
		`coachId`
		`moodleURL`
		`content`
		`enrolledUsers`
			`getCourseDetails` – Returns course metadata and enrollment details.
			
			`getContent` – Fetches all content associated with the course.
			
			`getAssignments` – Lists all assignments within the course.

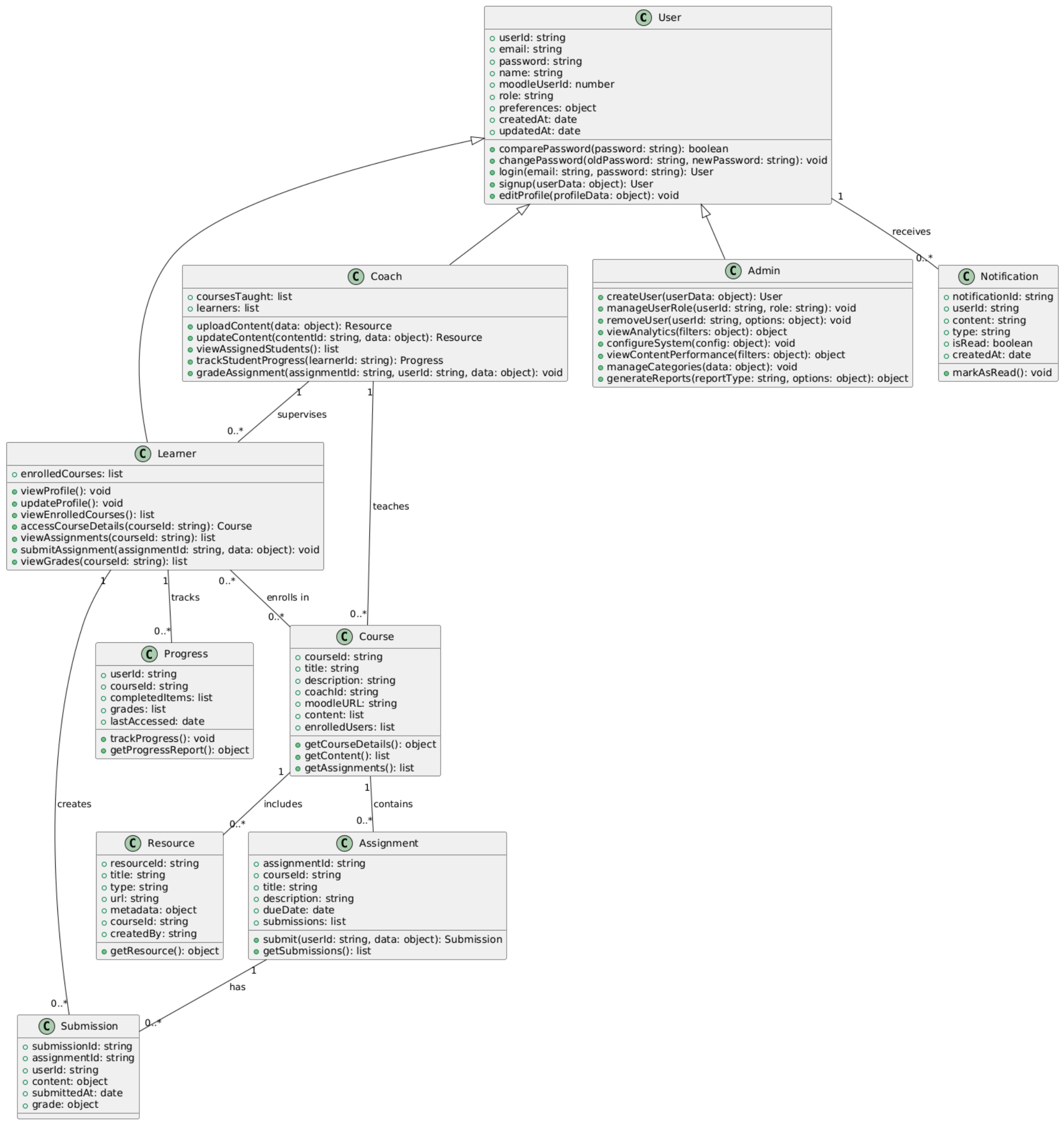
6	Resource	`resourceId`
		`title`
		`type`
		`url`
		`metadata`
		`courseId`
		`createdBy`
			`getResource` – Fetches details of a specific resource.dding resources or content.

7	Progress	`userId`
		`courseId`
		`completedItems`
		`grades`
		`lastAccessed`
			`trackProgress` – Updates the learner's progress in a course.
			
			`getProgressReport` – Returns a detailed report of completed modules and grades.

8	Assignment	`assignmentId`
		`courseId`
		`title`
		`description`
		`dueDate`
		`submissions`
			`submit`– Saves the user's assignment submission.
			
			`getSubmissions` – Retrieves all submissions for the assignment.

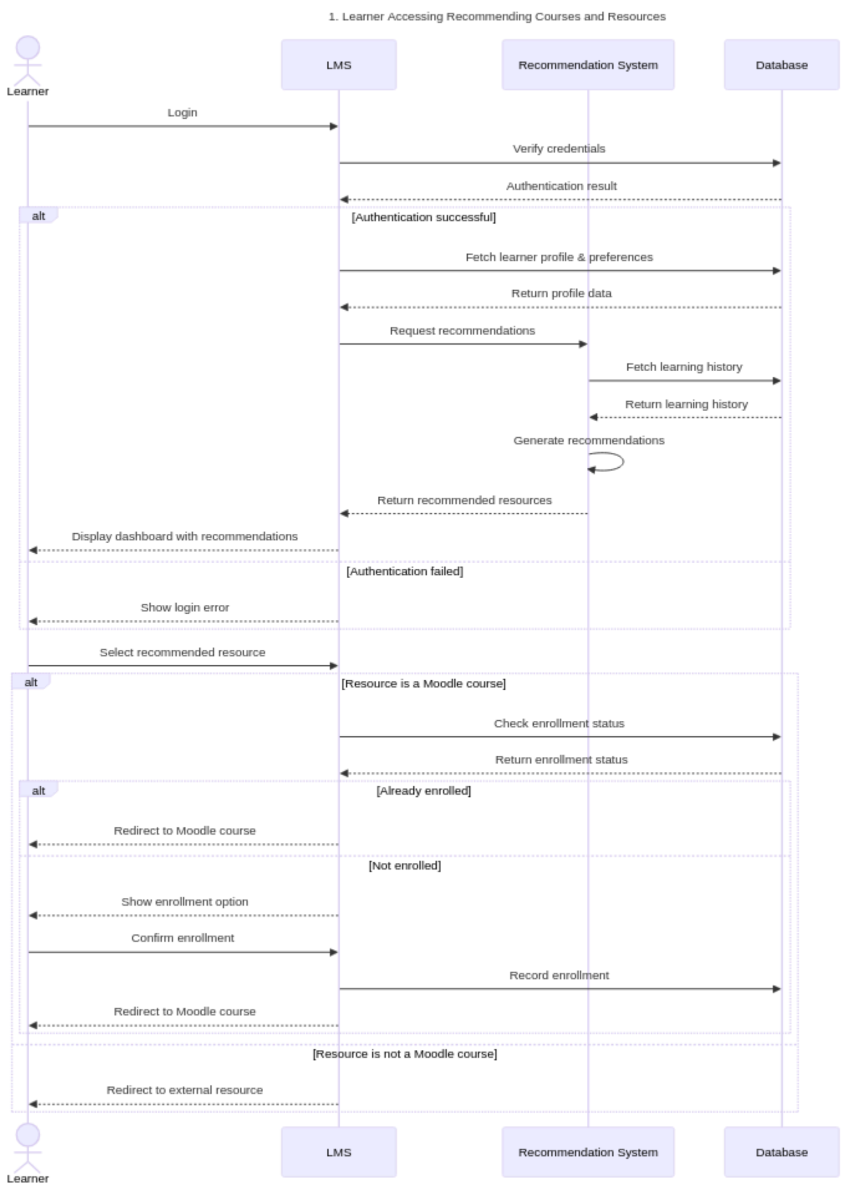
9	Submission	`submissionId`
		`assignmentId`
		`userId`
		`content`
		`submittedAt`
		`grade`
			`-` (No specific methods for this class)

Class Diagram:

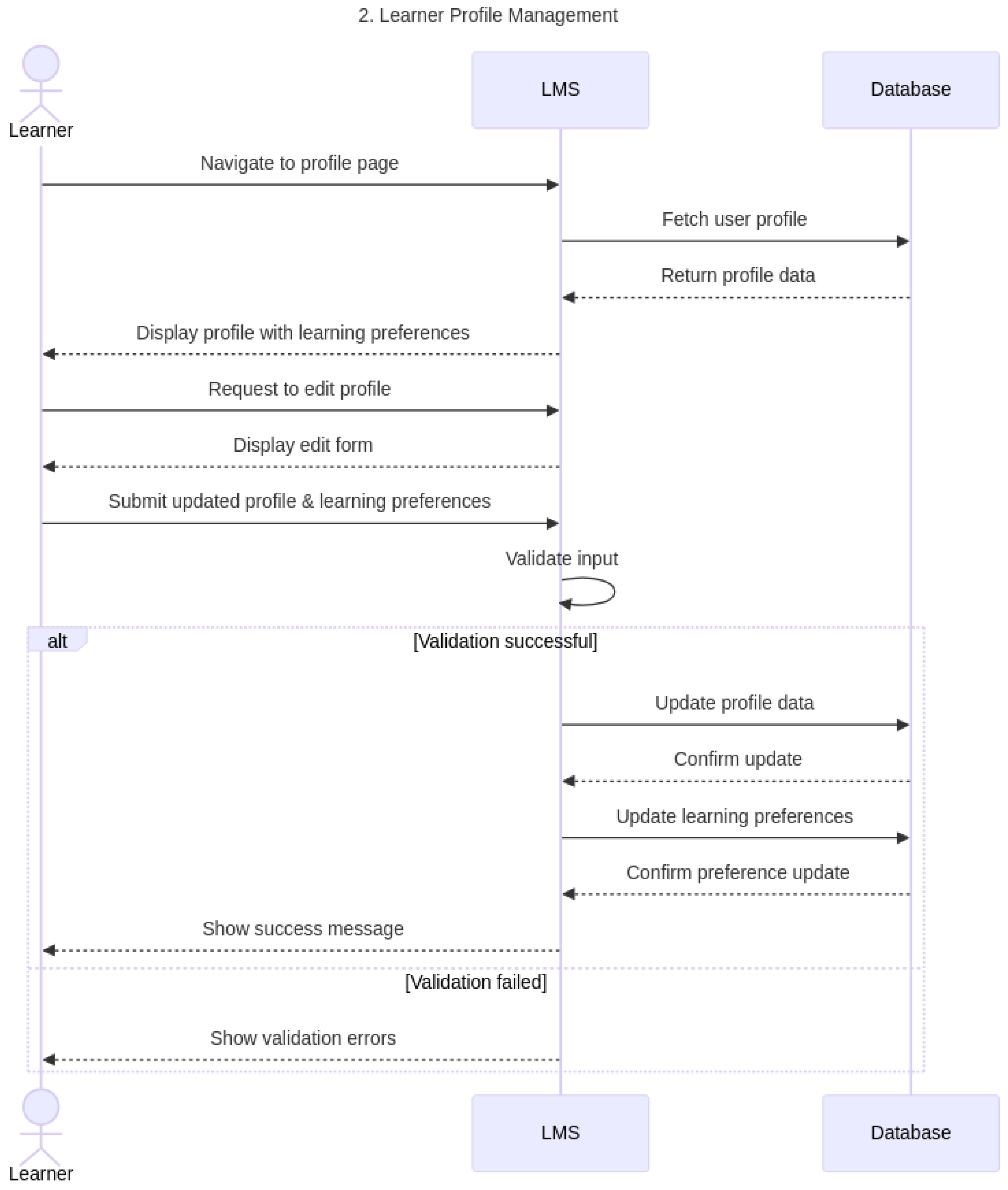


Sequence Diagrams

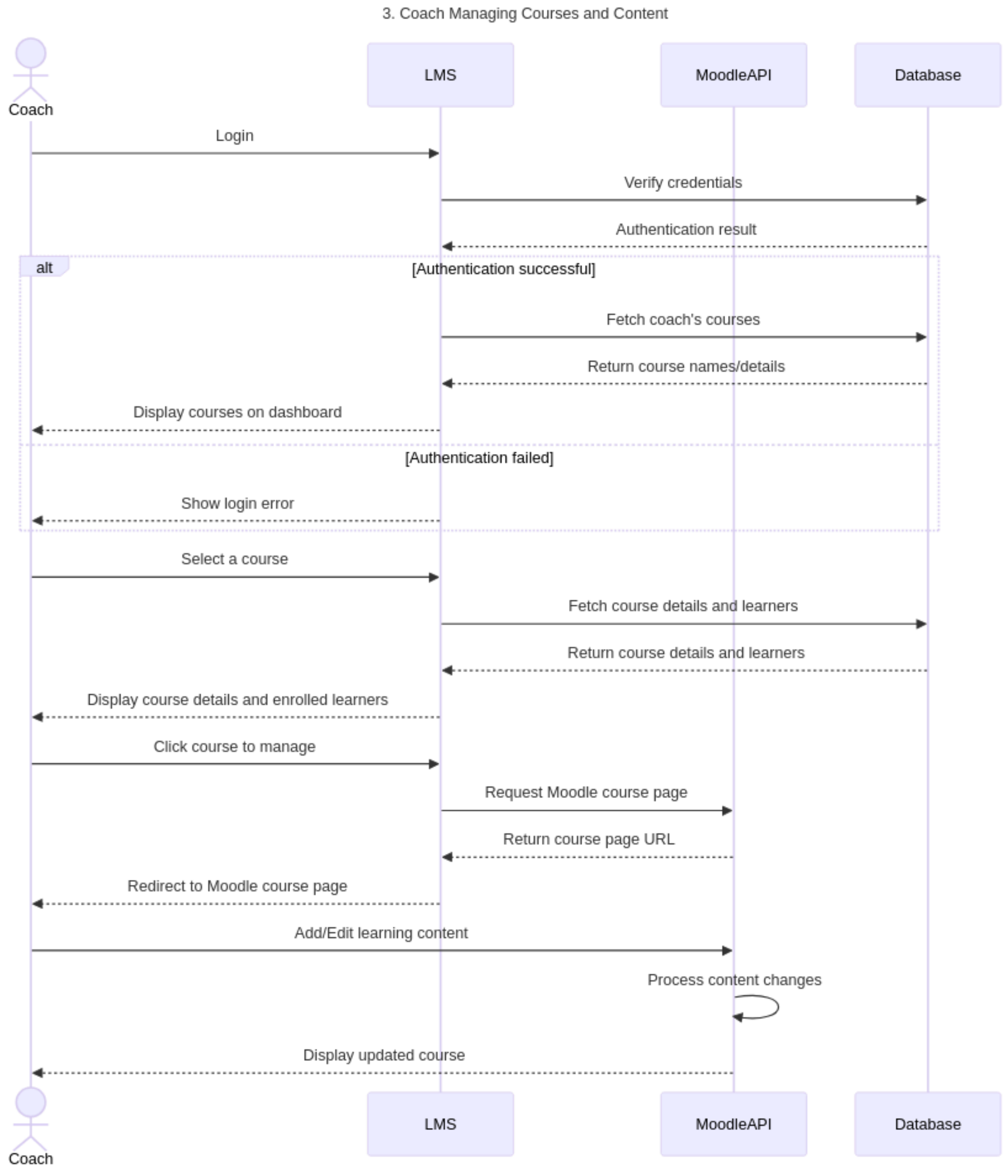
1. Learner Accessing Recommending Courses and Resources



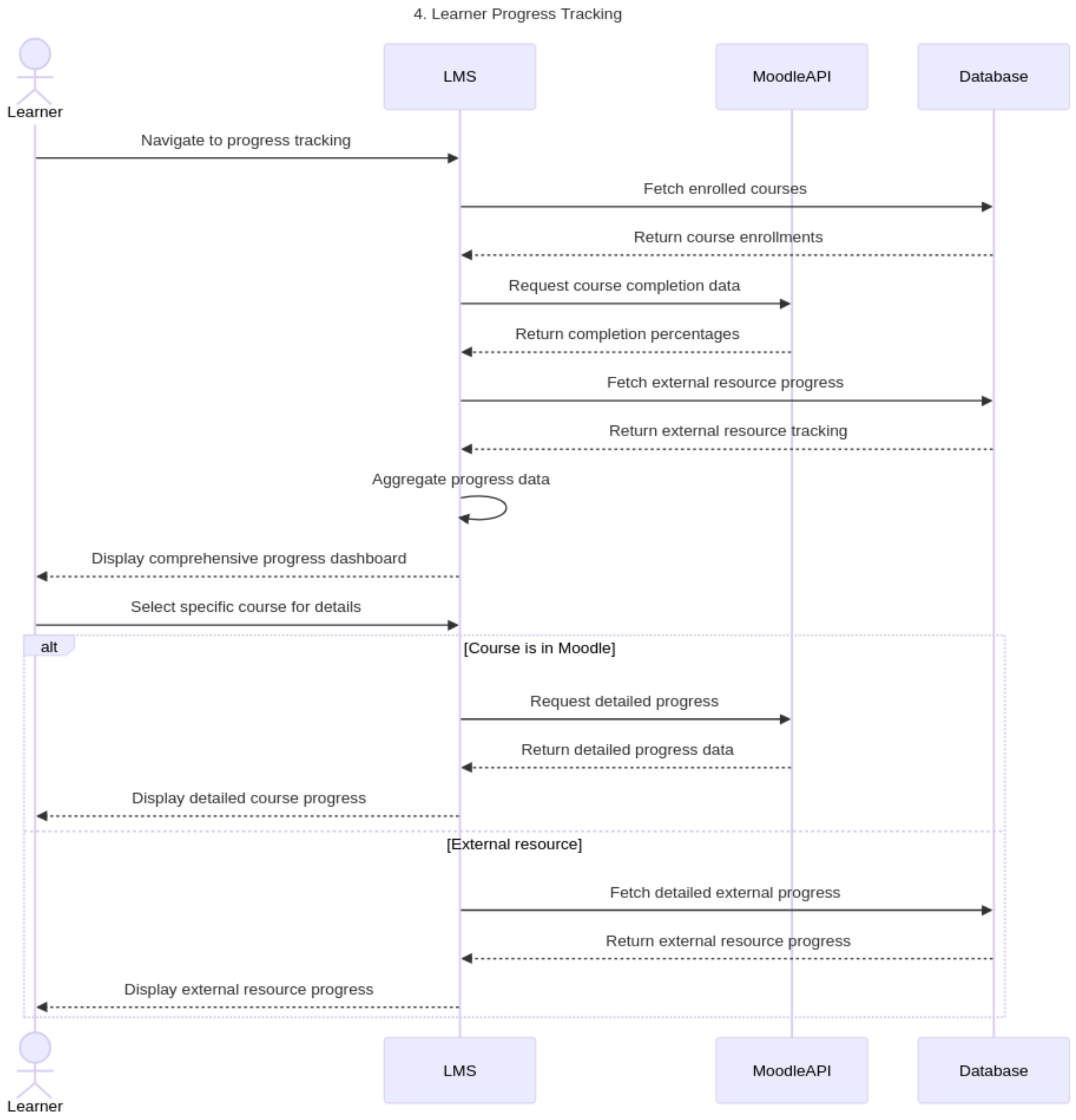
2. Learner Profile Management



3. Coach Managing Courses and Content


3. Coach Managing Courses and Content
Coach
LMS
MoodleAPI
Database
Login
Verify credentials
Authentication result
alt
[Authentication successful]
Fetch coach's courses
Return course names/details
Display courses on dashboard
[Authentication failed]
Show login error
Select a course
Fetch course details and learners
Return course details and learners
Display course details and enrolled learners
Click course to manage
Request Moodle course page
Return course page URL
Redirect to Moodle course page
Add/Edit learning content
Process content changes
Display updated course
Coach
LMS
MoodleAPI
Database

4. Learner Progress Tracking


4. Learner Progress Tracking
Learner
LMS
MoodleAPI
Database
Navigate to progress tracking
Fetch enrolled courses
Return course enrollments
Request course completion data
Return completion percentages
Fetch external resource progress
Return external resource tracking
Aggregate progress data
Display comprehensive progress dashboard
Select specific course for details
alt
[Course is in Moodle]
Request detailed progress
Return detailed progress data
Display detailed course progress
[External resource]
Fetch detailed external progress
Return external resource progress
Display external resource progress
Learner
LMS
MoodleAPI
Database

Design Rationale

1. Decision on Custom Build vs Building on top of Moodle

Alternatives Considered:

- Build a custom course management system from scratch.
- Use Moodle LMS.
- Use other LMS like Canvas.

Final Choice:

- **Moodle LMS** was chosen due to its open-source nature, flexibility, and support for structured course management.

Rationale:

- Moodle enables efficient course management without the need for custom development. While integration and customization require effort, its flexibility outweighs the costs. Custom systems demand significant resources, and alternatives like Canvas have licensing costs and limited customization.

2. Server Allocation and Scalability

Alternatives Considered:

- Single server for both backend and frontend.
- Two nodes one for Moodle & frontend, one for backend & recommendation engine.

Final Choice:

- Two VM nodes: One for Moodle & frontend, one for backend & recommendation engine.

Rationale:

- The two VM setup ensures better performance scaling and fault isolation by keeping the frontend, backend, and recommendation engine separate. The single server approach was not considered due to potential performance bottlenecks and higher risk of failure, which could negatively impact user experience and scalability.

3. User Interface Design

Alternatives Considered:

- Design with all features on one page.

- Focused and simplified UI with a streamlined profile, prioritized learning activities, and additional details in tabs.

Final Choice:

- Focused and simplified UI with a streamlined profile and prioritized learning activities.

Rationale:

- The chosen UI design prioritizes essential learning activities, offering a streamlined user experience. The alternative approach with all features on one page was not preferred, as it could lead to a cluttered experience, making navigation more complex for users.

4. Login and Redirection to Courses in Moodle

Alternatives Considered:

- Manual login for Moodle.
- Auto-login using Single Sign-On (SSO).

Final Choice:

- Auto-login through token-based authentication.

Rationale:

- Token-based authentication improves user experience by automating the login process. Although backend integration with Moodle authentication is required, this approach ensures a smooth user journey without manual login steps.

5. Video Content Strategy

Alternatives Considered:

- Store file paths rather than video or file objects in Moodle.
- Direct Moodle video or file uploads.

Final Choice:

- Store file paths rather than video or file objects in Moodle.

Rationale:

- Storing file paths keeps video management more efficient and gives better control over user experience while minimizing storage constraints.