Software Requirements Specification

for

Mentora

Version 1.0 approved

Prepared by Mah Rukh

Group 8

March 13, 2025

Table of Contents

Table of Contents ii

Revision History iii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 2

1.5 References 3

2. Overall Description 3

2.1 Product Perspective 3

2.2 Product Functions 3

2.3 User Classes and Characteristics 3

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

3. External Interface Requirements 4

3.1 User Interfaces 4

3.2 Hardware Interfaces 5

3.3 Software Interfaces 5

3.4 Communications Interfaces 6

4. System Features 6

4.1 User Registration & Authentication 6

4.2 Course Browsing 7

4.3 Course Enrollment 8

4.4 Complete Module 9

4.5 Take Quiz 10

4.6 Submit Work 11

4.7 View Progress 12

4.8 Get Recommendation 13

4.9 Forum Access 14

5. Other Nonfunctional Requirements 15

5.1 Performance Requirements 15

5.2 Safety Requirements 15

5.3 Security Requirements 15

5.4 Software Quality Attributes 15

5.5 Business Rules 16

6. Other Requirements 16

Appendix A: Glossary 17

Appendix B: Analysis Models 18

Class Diagram: 18

DFD Level 0: 18

DFD Level 1: 19

Use Case Diagram: 20

Sequence Diagram: 21

Adaptive Assessment: 21

Assessment Submission: 22

Automated Learning: 23

Course Enrollment: 23

Difficulty Adjustment: 24

Grading: 25

Learning Recommendation: 26

Tracking: 27

State Diagrams: 28

User Session: 28

User Auth: 29

Student Progress: 30

Course Module: 31

Assessment: 32

Adaptive Engine: 32

Appendix C: To Be Determined List 33

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this Software Requirements Specification (SRS) document is to provide a comprehensive and detailed description of the "Mentora" mobile application, focusing on its functionalities, performance, and interactions with users. This document serves as a foundational guide for all stakeholders, including developers, designers, testers, and project managers, ensuring a clear understanding of the system's objectives and requirements.

"Mentora" is an adaptive learning platform designed to enhance the educational experience by offering personalized learning paths, adaptive assessments, and real-time performance analytics. This SRS encompasses the complete scope of the application, detailing both functional and non-functional requirements, user interactions, system features, and constraints. By outlining these elements, the SRS aims to facilitate a cohesive development process, minimize ambiguities, and align the project with the intended educational outcomes.

## Document Conventions

This SRS document adheres to the following conventions:

**Priority Levels:** Each requirement is assigned a priority level - High, Medium, or Low to indicate its significance to system functionality.

**Formatting:**

* **Section Headings:** Presented in bold to delineate major sections.
* **Requirement Titles:** Italicized to emphasize specific requirements.

**Document Structure:** The SRS is organized hierarchically, providing a systematic overview of software requirements.

**Change Management:** Updates or revisions are documented in a dedicated change log section to maintain version control.

These conventions ensure clarity, consistency, and effective communication among stakeholders.

## Intended Audience and Reading Suggestions

This Software Requirements Specification (SRS) document is intended for:

* **Developers:** To understand system functionalities and design constraints.
* **Project Managers:** To oversee project scope, timelines, and resource allocation.
* **Marketing Staff:** To grasp application features for effective promotion.
* **End Users (Students and Instructors):** To comprehend application benefits and functionalities.
* **Testers:** To design test cases ensuring quality and performance.

**Document Structure:**

1. **Introduction:** Overview of the project's purpose and scope.
2. **Overall Description:** User needs and system context.
3. **System Features and Requirements:** Detailed functional and non-functional requirements.
4. **Appendices:** Glossary and reference materials.

**Reading Recommendations:**

* **General Overview:** Start with the Introduction and Overall Description.
* **Implementation Details:** Focus on System Features and Requirements.
* **Clarifications:** Refer to Appendices.

This organization ensures each stakeholder can efficiently access relevant information.

## Product Scope

Mentora is a mobile-first learning platform designed to provide an adaptive and personalized educational experience. It enables students to enroll in courses, complete modules, take assessments, and receive AI-driven recommendations based on their performance. The system adjusts learning paths dynamically, ensuring that students receive tailored content suited to their pace and proficiency. Key objectives include enhancing learning efficiency, improving knowledge retention through adaptive techniques, and providing actionable insights via progress tracking. Mentora aligns with modern educational strategies by offering a scalable, future-proof platform that integrates seamlessly with emerging AI capabilities to optimize student learning.

## References

# Overall Description

## Product Perspective

Mentora is a **self-contained mobile learning platform** that offers **adaptive and personalized learning**. It is designed from scratch, integrating **user management, course modules, assessments, an adaptive learning engine, and analytics**. The system connects with **external learning databases, assessment tools, and analytics dashboards**, ensuring scalability and future AI integration. A high-level system diagram in the architecture documentation illustrates component interactions.

## Product Functions

Mentora provides the following key functionalities:

* **User Management:** Secure registration, login, and profile management.
* **Course Enrollment:** Browse, enroll, and access structured course modules.
* **Adaptive Learning:** Dynamic difficulty adjustment based on performance.
* **Assessments & Grading**: Quizzes, assignments, and projects with automated or instructor-based grading.
* **Progress Tracking:** Visual dashboards for monitoring learning progress.
* **Course Recommendations:** Personalized course/module suggestions based on pace and performance.
* **Mobile Optimization:** Offline access and UI tailored for mobile learning.

## User Classes and Characteristics

Mentora is designed for the following user classes:

* **Students** – Primary users who enroll in courses, complete modules, take assessments, and receive adaptive recommendations.
* **Instructors** – Create and manage courses, upload materials, grade assignments, and monitor student progress.
* **Administrators** – Oversee user management, maintain the system, and generate reports.

Each class has distinct access levels, with **students** focused on learning, **instructors** on teaching and evaluation, and **administrators** on system operations.

## Operating Environment

Mentora will operate as a **mobile application** compatible with **Android (8.0+) and iOS (12.0+)**. It will run on **smartphones and tablets** with minimum **2GB RAM and 16GB storage**. The system will require **internet access** for online features but support **offline content caching**. It integrates with **cloud storage** for data synchronization and follows **standard security protocols** to ensure data integrity and privacy.

## Design and Implementation Constraints

Mentora's development is constrained by **mobile platform limitations**, ensuring compatibility with **Android**. It must support **low-latency performance** and **efficient memory usage** for mobile devices. Security measures must comply with **data protection standards**. The system will integrate with **cloud storage**, requiring stable **network connectivity** for real-time synchronization. Development will follow **industry best practices**, leveraging **Kotlin/Java** for android development, **Firebase for backend services**, and **REST APIs** for external communication.

## User Documentation

Mentora will provide **a user manual, in-app tutorials, and online help guides** to assist users. Documentation will be available in **PDF and web-based formats**, covering **registration, course enrollment, adaptive learning features, and progress tracking**. Interactive tooltips and **FAQs** will be integrated within the app for real-time guidance.

## Assumptions and Dependencies

* The system assumes **stable internet connectivity** for real-time adaptive learning and content updates.
* It depends on **third-party APIs for authentication, plagiarism detection, and analytics**.
* The mobile app requires **Android 8.0+ and iOS 12+** for compatibility.
* The adaptive engine relies on **user performance data** for personalized recommendations.
* Future enhancements assume **cloud storage integration** for scalable content management.

# External Interface Requirements

## User Interfaces

The Mentora mobile app will feature a **modern, intuitive UI** following **Material Design guidelines** to ensure a seamless user experience.

* **Navigation:** Bottom navigation bar for quick access to Home, Courses, Progress, and Profile.
* **Screen Layout:** Consistent **XML-based UI** with adaptive layouts for various screen sizes.
* **Standard Elements:** Common UI elements like buttons, text fields, dropdowns, and sliders.
* **Error Handling:** Clear, user-friendly error messages with retry options.
* **Dark Mode Support:** Available for accessibility and user preference.
* **Keyboard Shortcuts:** Quick actions for key functionalities like search and submission.

The UI will be implemented using **Java, Kotlin, and XML** for native Android performance.

## Hardware Interfaces

The Mentora mobile app will run on **Android smartphones and tablets**, ensuring compatibility with a wide range of devices.

* **Supported Devices:** Android phones and tablets with **Android 8.0 (Oreo) or higher**.
* **Processor & RAM:** Optimized for devices with **at least 2GB RAM and a quad-core processor**.
* **Storage Requirements:** Requires **at least 100MB of free space** for installation and additional storage for offline content.
* **Sensors & Peripherals:** May utilize **microphone and camera** for interactive learning features.
* **Communication Protocols:** Uses **Wi-Fi and mobile data** for content updates, and **Bluetooth** (if needed for external accessories).

## Software Interfaces

The Mentora mobile app will integrate with various software components to ensure seamless functionality and communication.

* **Operating System:** Android **8.0 (Oreo) or higher**.
* **Programming Languages:** Developed using **Java, Kotlin, and XML** for UI design.
* **Database:** Uses **Firebase Firestore** for real-time data storage and **SQLite** for local offline storage.
* **APIs & Services:**
  + **Google Firebase Authentication** – For user login and authentication.
  + **Cloud Firestore** – To manage user progress, course content, and recommendations.
  + **Watsonx.AI API (if used later)** – For adaptive learning recommendations.
  + **RESTful APIs** – For secure communication between the mobile app and backend services.
* **Data Flow:**
  + **Incoming:** User inputs (login credentials, course selections, quiz responses).
  + **Outgoing:** Progress updates, recommendations, and learning analytics.

The app will ensure secure data transactions through **HTTPS protocols and encryption techniques** to maintain integrity and confidentiality.

## Communications Interfaces

The Mentora mobile app will support secure and efficient communication through various protocols and technologies:

* **Network Communication:**
  + Uses **HTTPS** for secure data transfer between the app and backend services.
  + RESTful API calls for fetching and updating course progress, recommendations, and user interactions.
* **Real-Time Communication:**
  + **Firebase Cloud Messaging (FCM)** for push notifications regarding course updates, reminders, and announcements.
* **Data Synchronization:**
  + **Firestore Sync** for real-time updates across devices.
  + **Local Storage (SQLite)** to enable offline mode, syncing with Firestore when the internet is available.
* **Security Measures:**
  + **OAuth 2.0 & Firebase Authentication** for user login and access control.
  + **AES Encryption** for sensitive data storage and transmission.

# System Features

## User Registration & Authentication

### Description and Priority:

The **User Registration & Authentication** feature allows students to create an account or log in to an existing one, ensuring secure access to the Mentora app. This is a fundamental feature required for accessing course content and other functionalities.

### Stimulus / Response Sequences:

* **Student Action:** Opens the app.
* **System Response:** Displays the login and registration options.
* **Student Action:** Selects ‘Register’ or ‘Login.’
* **System Response:** Shows the respective form for input.
* **Student Action:** Enters credentials (email, password, etc.).
* **System Response:** Validates credentials against the database.
* **Student Action:** Clicks ‘Submit.’
* **System Response:**
  + If valid, access is granted, and the student is redirected to the dashboard.
  + If invalid, an error message appears, prompting a retry.
* **Student Action (if credentials are forgotten):** Clicks ‘Forgot Password.’
* **System Response:** Sends a password recovery link to the registered email.

### Functional Requirements

* REQ-01: The system shall allow students to register using an email and password.
* REQ-02: The system shall authenticate users with valid credentials.
* REQ-03: The system shall display an error message if login credentials are incorrect.
* REQ-04: The system shall provide a ‘Forgot Password’ option for account recovery.
* REQ-05: The system shall redirect authenticated users to their dashboard.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC1** |
| **Use Case Name** | User Registration & Authentication |
| **Actor(s)** | Student (S) |
| **Description** | Allows a student to create a new account or access an existing one. |
| **Preconditions** | The student has the Mentora app installed and internet connectivity. |
| **Postconditions** | The student is authenticated and redirected to their dashboard. |
| **Main Flow** | 1. Student opens the app. 2. Selects 'Register' or 'Login'. 3. Enters required credentials. 4. System verifies credentials. 5. Access granted to the dashboard. |
| **Alternate Flow** | 3a. If credentials are incorrect, an error message is displayed, and the student is prompted to retry. |
| **Exceptions** | 3b. If the student forgets the password, they can initiate a password recovery process. |

## Course Browsing

### Description and Priority:

The **Course Browsing** feature allows students to explore available courses within the Mentora app. They can view categorized courses, search for specific topics, and apply filters to find relevant options efficiently.

### Stimulus / Response Sequences:

* **Student Action:** Navigates to the course catalog.
* **System Response:** Displays available courses categorized by subject.
* **Student Action:** Browses the listed courses.
* **Alternate Flow:** Student uses search or filter options to refine the course listings.

 **Exception Handling:** No exceptions anticipated.

### Functional Requirements

* REQ-01: The system shall display a categorized list of available courses when a student accesses the course catalog.
* REQ-02: The system shall provide search functionality to allow students to find specific courses by keywords.
* REQ-03: The system shall enable filtering by subject, difficulty level, instructor, or other relevant attributes.
* REQ-04: The system shall update the course catalog dynamically to reflect new and removed courses.
* REQ-05: The system shall ensure a responsive and user-friendly interface for smooth navigation.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC2** |
| **Use Case Name** | Course Browsing |
| **Actor(s)** | Student (S) |
| **Description** | Enables students to view available courses. |
| **Preconditions** | The student is logged in. |
| **Postconditions** | The student has a list of courses to consider for enrollment. |
| **Main Flow** | 1. Student navigates to the course catalog. 2. System displays courses categorized by subject. |
| **Alternate Flow** | 2a. Student uses search or filter options to refine course listings. |
| **Exceptions** | None. |

## Course Enrollment

### Description and Priority:

The **Course Enrollment** feature enables students to enroll in courses of their choice, granting them access to relevant course materials and activities. This is a core feature of the Mentora app.

### Stimulus / Response Sequences:

* **Student Action:** Selects a course from the course catalog.
* **Student Action:** Clicks on the ‘Enroll’ button.
* **System Response:** Confirms enrollment and updates the student’s course list.
* **Exception Handling**: If enrollment fails (e.g., course is full), the system displays an error message and prompts the student to try another course.

### Functional Requirements

* REQ-01: The system shall allow students to enroll in a course by clicking the ‘Enroll’ button.
* REQ-02: The system shall update the student's enrolled course list upon successful enrollment.
* REQ-03: The system shall notify the student of successful enrollment via an on-screen confirmation message.
* REQ-04: If enrollment fails due to course capacity limits or system errors, the system shall display an appropriate error message.
* REQ-05: The system shall restrict students from enrolling in the same course multiple times.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC3** |
| **Use Case Name** | Course Enrollment |
| **Actor(s)** | Student (S) |
| **Description** | Allows students to enroll in selected courses. |
| **Preconditions** | The student is logged in and has selected a course. |
| **Postconditions** | The student is enrolled and can access course materials. |
| **Main Flow** | 1. Student selects a course. 2. Clicks 'Enroll'. 3. System confirms enrollment and updates the student's course list. |
| **Alternate Flow** | None. |
| **Exceptions** | 3a. If enrollment fails (e.g., course is full), an error message is displayed. |

## Complete Module

### Description and Priority:

The **Complete Module** feature enables students to engage with course materials, complete assessments, and track their progress. This is essential for structured learning and adaptive recommendations.

### Stimulus / Response Sequences:

* **Student Action:** Accesses the module.
  + **System Response:** Displays module content (videos, exercises, and assessments).
* **Student Action:** Engages with module content.
  + **System Response:** Allows the student to pause and resume as needed.
* **Student Action:** Completes the module assessment.
* **System Response:** Evaluates the assessment.
* **System Response:** If the assessment is passed, marks the module as completed and updates progress.
* **Exception Handling:** If the assessment is not passed, the system prompts the student to retry.

### Functional Requirements

* REQ-01: The system shall allow students to access course modules and interact with content (videos, exercises).
* REQ-02: The system shall enable students to pause and resume module content as needed.
* REQ-03: The system shall provide an assessment at the end of each module.
* REQ-04: The system shall evaluate the assessment and determine if the module is completed.
* REQ-05: If the student passes the assessment, the system shall mark the module as completed and update progress.
* REQ-06: If the student fails the assessment, the system shall prompt them to retry.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC4** |
| **Use Case Name** | Complete Module |
| **Actor(s)** | Student (S) |
| **Description** | Facilitates students in completing course modules. |
| **Preconditions** | The student is enrolled in the course. |
| **Postconditions** | The module is marked as completed, and progress is updated. |
| **Main Flow** | 1. Student accesses the module. 2. Engages with content (videos, exercises). 3. Completes module assessment. 4. System records completion and updates progress. |
| **Alternate Flow** | 2a. Student can pause and resume module content. |
| **Exceptions** | 3a. If assessment is not passed, the student is prompted to retry. |

## Take Quiz

### Description and Priority:

The **Take Quiz** feature allows students to attempt quizzes as part of their course assessment. It ensures that students can answer questions within a set time limit, receive feedback, and have their performance recorded.

### Stimulus / Response Sequences:

* **Student Action:** Starts the quiz.
  + **System Response:** Loads quiz questions and displays a timer.
* **Student Action:** Answers each question within the time limit.
  + **System Response:** Saves responses and updates progress.
* **Student Action:** Clicks ‘Submit’ after completing the quiz.
  + **System Response:** Processes answers and records results.
* **System Response:** Provides immediate feedback on performance.
  + **Alternate Flow:** If time expires before submission, the quiz auto-submits.
  + **Exception Handling:** If a technical issue occurs, the system allows resuming from the last saved point..

### Functional Requirements

* REQ-01: The system shall allow students to start a quiz from an enrolled module.
* REQ-02: The system shall display a countdown timer for time-limited quizzes.
* REQ-03: The system shall save students' answers in real-time.
* REQ-04: The system shall automatically submit the quiz if the time limit expires.
* REQ-05: The system shall evaluate quiz responses and provide immediate feedback.
* REQ-06: The system shall allow students to resume the quiz from the last saved point in case of a technical issue.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC5** |
| **Use Case Name** | Take Quiz |
| **Actor(s)** | Student (S) |
| **Description** | Allows students to participate in quizzes. |
| **Preconditions** | The student is enrolled and has accessed the relevant module. |
| **Postconditions** | Quiz results are recorded, and feedback is provided. |
| **Main Flow** | 1. Student starts the quiz. 2. Answers questions within the time limit. 3. Submits quiz. 4. System evaluates and provides immediate feedback. |
| **Alternate Flow** | 2a. If time expires, the quiz auto-submits. |
| **Exceptions** | 4a. If technical issues occur, the student can resume the quiz from the last saved point. |

## Submit Work

### Description and Priority:

The **Submit Work** feature allows students to upload and submit assignments or projects within the app. This ensures that submissions are recorded, and instructors are notified for grading.

### Stimulus / Response Sequences:

* **Student Action:** Uploads the completed assignment or project.
  + **System Response:** Stores the uploaded file and displays a preview.
* **Student Action:** Adds optional comments or notes.
  + **System Response:** Saves the comments along with the submission.
* **Student Action:** Clicks ‘Submit’.
  + **System Response:** Confirms submission and sends a notification to the instructor.
  + **Alternate Flow:** Student can save a draft before final submission.
  + **Exception Handling:** If submission fails due to connectivity or file issues, an error message is displayed, and the student is prompted to retry.

### Functional Requirements

* REQ-01: The system shall allow students to upload assignments in various file formats (e.g., PDF, DOCX, JPG).
* REQ-02: The system shall provide an option to add comments before submission.
* REQ-03: The system shall allow students to save a draft of their work before final submission.
* REQ-04: The system shall confirm submission and notify the instructor.
* REQ-05: The system shall display an error message if submission fails and provide a retry option.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC6** |
| **Use Case Name** | Submit Work |
| **Actor(s)** | Student (S) |
| **Description** | Enables students to submit assignments or projects. |
| **Preconditions** | The student has completed the assignment or project. |
| **Postconditions** | Submission is recorded, and the instructor is notified for grading. |
| **Main Flow** | 1. Student uploads the completed work. 2. Adds any necessary comments. 3. Clicks 'Submit'. 4. System confirms submission and notifies the instructor. |
| **Alternate Flow** | 1a. Student can save a draft before final submission. |
| **Exceptions** | 4a. If submission fails, an error message is displayed, and the student is prompted to retry. |

## View Progress

### Description and Priority:

The **View Progress** feature enables students to track their learning journey, including completed modules, grades, and personalized recommendations. This helps students understand their performance and identify areas for improvement.

### Stimulus / Response Sequences:

* **Student Action:** Navigates to the progress dashboard.
* **System Response:** Retrieves and displays progress metrics.
* **System Action:** Shows data such as completed modules, quiz scores, and recommendations.
* **Alternate Flow:** Student applies filters or sorts metrics based on preferences.
* Exception **Handling:** No exceptions anticipated.

### Functional Requirements

* REQ-01: The system shall display a progress dashboard with completed modules, grades, and learning recommendations.
* REQ-02: The system shall allow students to filter or sort progress data based on parameters like course, completion percentage, or grades.
* REQ-03: The system shall update progress in real-time as students complete activities.
* REQ-04: The system shall provide visual indicators (graphs, charts, or bars) for progress representation.
* REQ-05: The system shall ensure smooth data retrieval with minimal loading time**.**.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC7** |
| **Use Case Name** | View Progress |
| **Actor(s)** | Student (S) |
| **Description** | Allows students to monitor their learning progress. |
| **Preconditions** | The student is enrolled in at least one course. |
| **Postconditions** | The student gains insights into their performance and areas needing improvement. |
| **Main Flow** | 1. Student navigates to the progress dashboard. 2. System displays metrics like completed modules, grades, and recommendations. |
| **Alternate Flow** | 2a. Student can filter or sort metrics based on preferences. |
| **Exceptions** | None. |

## Get Recommendation

### Description and Priority:

The **Get Recommendations** feature provides students with personalized course or module suggestions based on their past performance and learning preferences. This helps optimize their learning path and improve engagement.

### Stimulus / Response Sequences:

* **Student Action:** Accesses the recommendations section.
  + **System Response:** Retrieves learning history and preferences.
  + **System Action:** Analyzes past performance and learning patterns.
  + **System Action:** Displays personalized course or module suggestions.
  + **Alternate Flow:** Student provides feedback to refine future recommendations.
  + **Exception Handling:** No exceptions anticipated.

### Functional Requirements

* REQ-01: The system shall analyze a student’s completed modules, grades, and engagement to generate personalized learning recommendations.
* REQ-02: The system shall display suggested courses or modules based on adaptive learning techniques.
* REQ-03: The system shall allow students to provide feedback on recommendations to refine future suggestions.
* REQ-04: The system shall update recommendations dynamically as students progress.
* REQ-05: The system shall ensure that recommendation generation is efficient and minimally impacts system performance.

### Use-Case Description Table:

|  |  |
| --- | --- |
| **Use Case ID** | **UC8** |
| **Use Case Name** | Get Recommendations |
| **Actor(s)** | Student (S) |
| **Description** | Provides personalized course or module suggestions. |
| **Preconditions** | The student has completed some modules or courses. |
| **Postconditions** | The student receives tailored recommendations to enhance learning. |
| **Main Flow** | 1. Student accesses the recommendations section. 2. System analyzes performance and preferences. 3. Displays suggested courses or modules. |
| **Alternate Flow** | 2a. Student can provide feedback to refine recommendations. |
| **Exceptions** | None. |

## Forum Access

### Description and Priority:

The **Forum Access** feature allows students to participate in discussions related to their enrolled courses. It provides an interactive space where students can ask questions, share knowledge, and collaborate with peers.

### Stimulus / Response Sequences:

* **Student Action:** A student navigates to the course forum.
* **Response:** Displays all active discussions related to the course.
* **Student Action:** The student posts a new question or responds to an existing thread.
* **System Response:** Saves the post and updates the discussion thread.
* **Student Action:** The student upvotes or reports a post.
* **System Response:** Updates the post rating or flags it for review by moderators.

### Functional Requirements

* + **REQ-1:** The system shall allow students to view and participate in course-specific forums.
  + **REQ-2:** The system shall support posting, replying, and editing messages in discussion threads.
  + **REQ-3**: The system shall allow students to upvote or report posts.
  + **REQ-4:** The system shall notify users when someone replies to their post.
  + **REQ-5**: The system shall allow instructors or moderators to remove inappropriate content.

### Use-Case Description Tables:

|  |  |
| --- | --- |
| **Use Case ID** | **UC9** |
| **Use Case Name** | Forum Access |
| **Actor(s)** | Student (S) |
| **Description** | Enables students to participate in course-related discussions. |
| **Preconditions** | The student is enrolled in the course. |
| **Postconditions** | The student engages in community |

# Other Nonfunctional Requirements

## Performance Requirements

* The system should **respond within 2 seconds** for standard interactions like course browsing and progress tracking.
* Quiz evaluation and recommendation generation should be completed **within 5 seconds** of submission.
* The platform should support **at least 10,000 concurrent users** without performance degradation.
* Offline content should be accessible **instantly**, with sync operations completing **within 10 seconds** upon reconnection.

## Safety Requirements

* Data loss prevention mechanisms should be implemented, ensuring **automatic backups** every 24 hours.
* Users should be warned about **session timeouts** and unsaved work to prevent accidental data loss.
* Compliance with **digital accessibility standards** should be maintained to ensure safe usage for all users, including those with disabilities.

## Security Requirements

* User authentication must support OAuth, email/password, and two-factor authentication (2FA).
* Encryption (AES-256) should be used for storing sensitive user data and SSL/TLS for secure data transmission.
* Role-based access control (RBAC) should ensure that only authorized users (students, instructors, admins) can access specific functionalities.
* The platform should be GDPR-compliant, ensuring user privacy and data protection

## Software Quality Attributes

* **Availability:** The system should have **99.9% uptime**, ensuring continuous access to learning materials.
* **Usability:** A clean and intuitive UI with easy navigation should be prioritized.
* **Scalability:** The platform should allow seamless expansion as the user base grows.
* **Interoperability:** The system should support **integration with third-party APIs** for video hosting, payment gateways, and learning analytics.
* **Maintainability:** Code should follow **modular and well-documented** best practices to ensure easy updates and debugging.

## Business Rules

* Only **enrolled students** can access course materials and submit work.
* **Instructors** can create, update, and grade courses but **cannot modify user accounts**.
* **Admins** have full control over **user management, system configurations, and content moderation**.
* Course completion certificates will only be issued upon meeting the **minimum completion and assessment criteria.**

# Other Requirements

### ****Database Requirements****

* The system should use a **relational database (e.g., PostgreSQL, MySQL)** to store user profiles, course data, quiz results, and progress tracking.
* All data should support **ACID (Atomicity, Consistency, Isolation, Durability)** compliance to maintain integrity.
* A **NoSQL database (e.g., MongoDB, Firebase)** can be used for handling dynamic content like discussion forums and notifications.
* The database should support **sharding and replication** to ensure scalability.

### ****Internationalization Requirements****

* The platform should support **multiple languages**, with English as the default. Additional language support can be added via **language packs**.
* Courses should support **multiple subtitle options** for video content.
* Date and time formats should be adjustable based on the **user’s region**.

### ****Legal Requirements****

* The platform must comply with **GDPR** (General Data Protection Regulation) and **COPPA** (Children’s Online Privacy Protection Act) to protect user data.
* **Terms of Use and Privacy Policy** must be explicitly accepted by users before registration.
* **Intellectual property rights** must be respected, ensuring that **content uploaded by instructors** is either original or has proper licensing.

### ****Reuse Objectives****

* The platform should use **modular and reusable components** for future scalability.
* The authentication and payment processing systems should be built as **independent microservices** to allow reuse in other projects.
* **APIs should be RESTful** and documented for potential integration with third-party services.

### ****Accessibility Requirements****

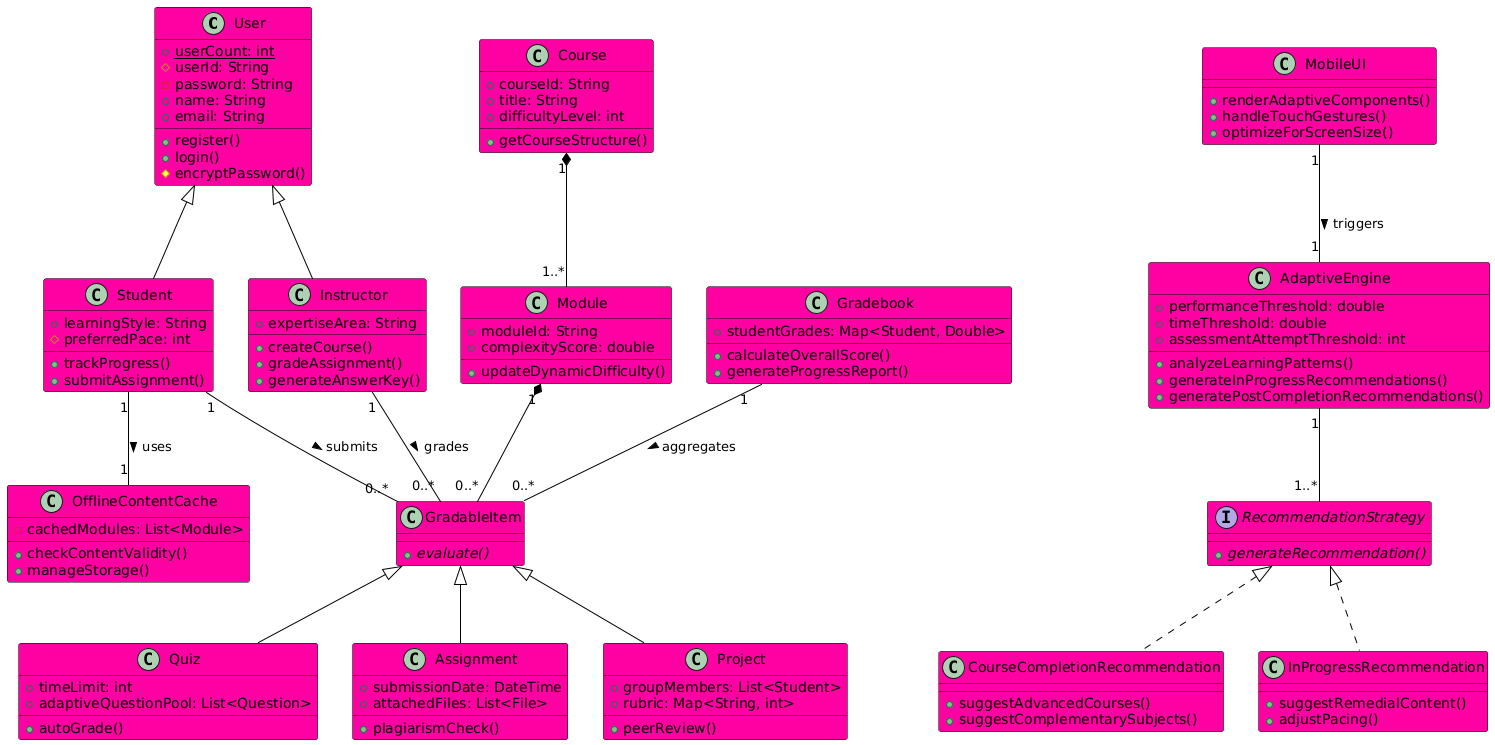
* The platform should be **WCAG 2.1 compliant** to ensure usability for users with disabilities.
* Keyboard navigation and screen reader support should be provided.
* Video lectures should include **closed captions** for hearing-impaired users

Appendix A: Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **SRS** | Software Requirements Specification – A document outlining the system’s requirements. |
| **ACID** | Atomicity, Consistency, Isolation, Durability – Principles ensuring reliable database transactions. |
| **OAuth** | Open Authorization – A protocol allowing secure authentication without exposing passwords. |
| **RBAC** | Role-Based Access Control – A security model that assigns permissions based on user roles. |
| **GDPR** | General Data Protection Regulation – A legal framework ensuring data privacy and protection. |
| **WCAG** | Web Content Accessibility Guidelines – A set of standards ensuring web accessibility for all users. |
| **REST API** | A web service architecture that allows interaction with the system using HTTP requests. |
| **Microservices** | A software architecture where different functionalities are built as independent services. |
| **Sharding** | A database architecture pattern that splits large databases into smaller, faster, and more manageable parts. |
| **NoSQL** | A type of database designed for scalability and flexibility, often used for handling unstructured data. |

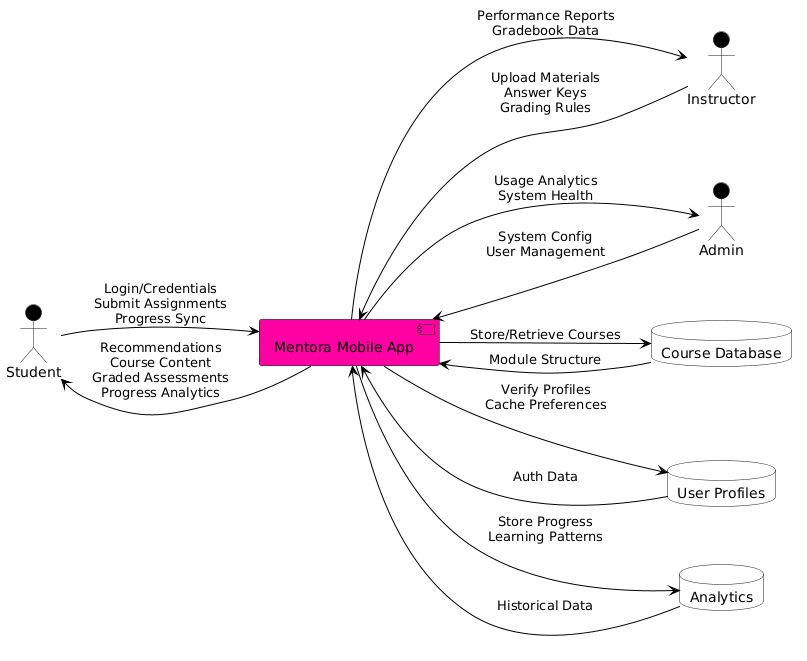
Appendix B: Analysis Models

## Class Diagram:

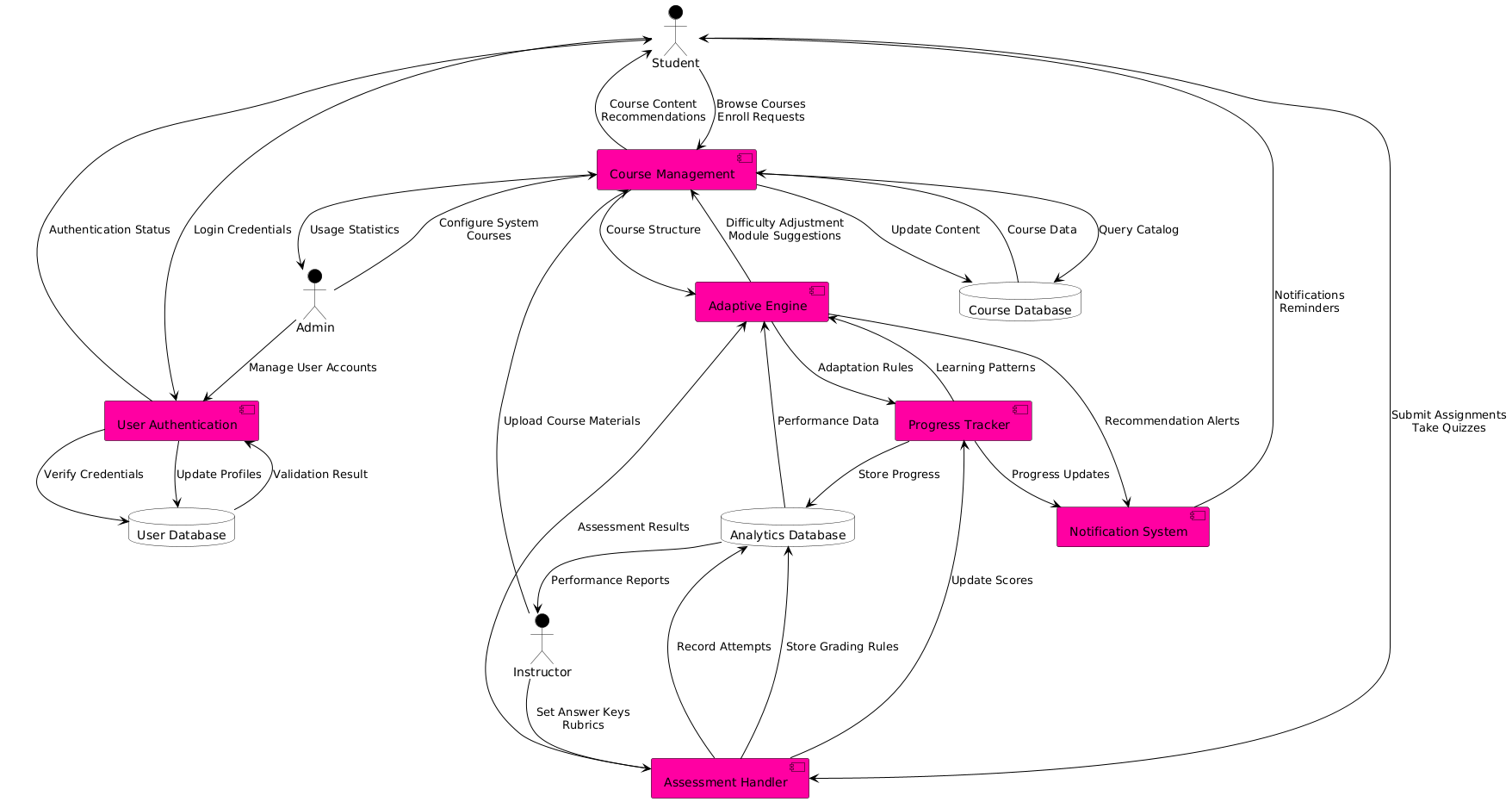


## Data Flow Diagram:

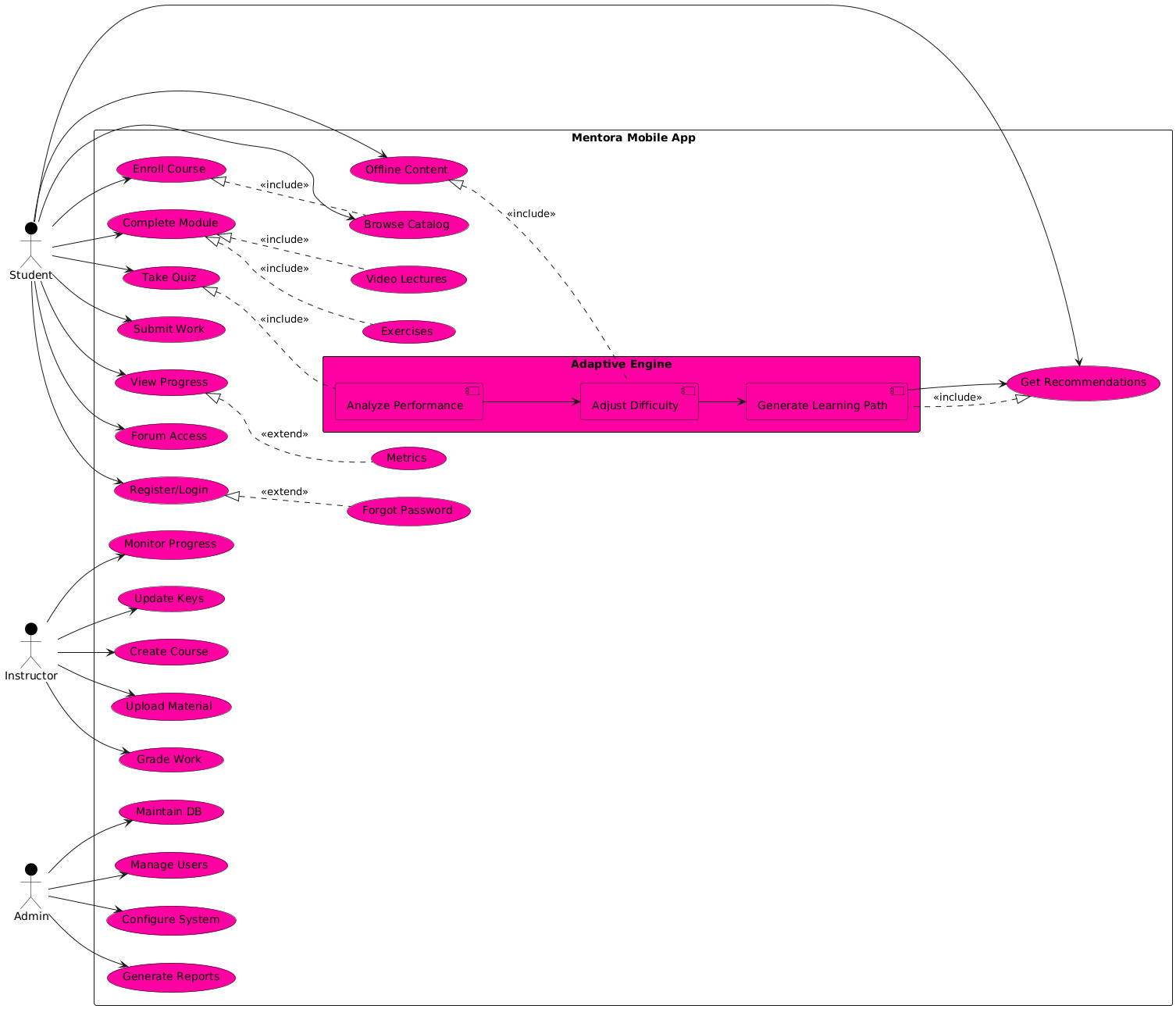
### DFD Level 0:



### DFD Level 1:

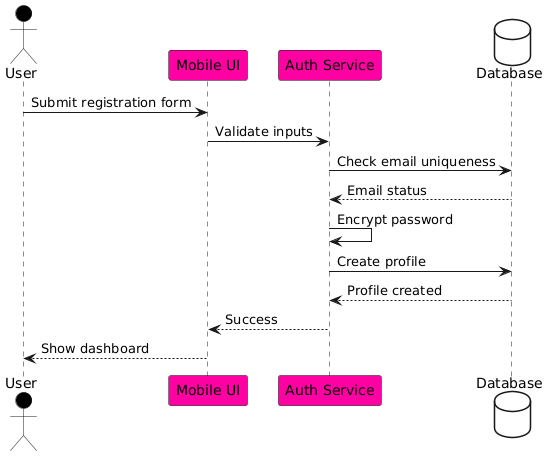


## Use Case Diagram:

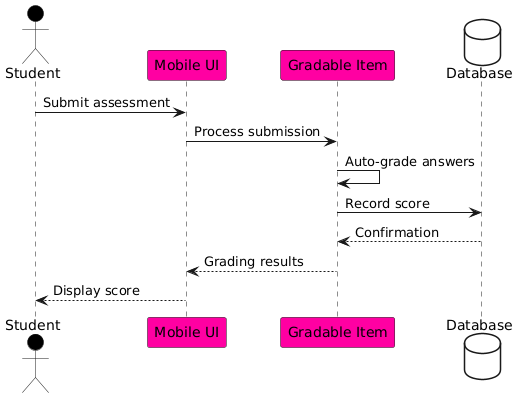


## Sequence Diagrams:

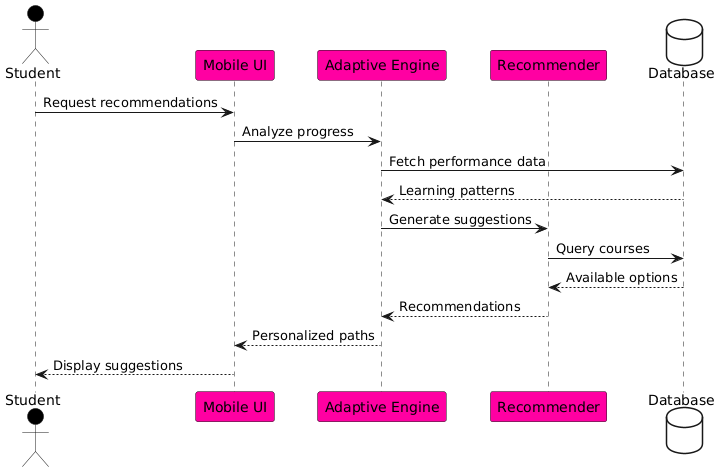
### User Registration & Profile Management:



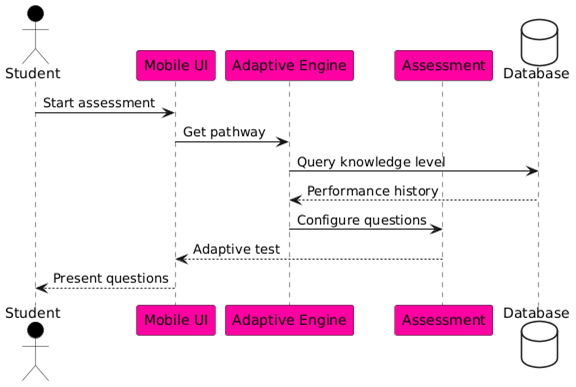
### Assessment Submission & Auto grading:



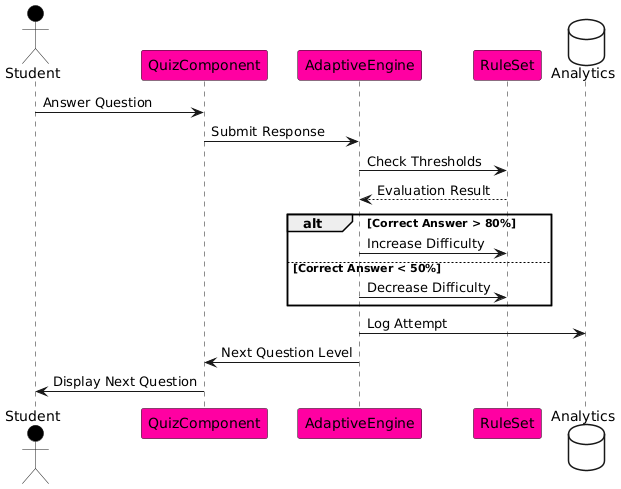
### Adaptive Learning Path Recommendations:



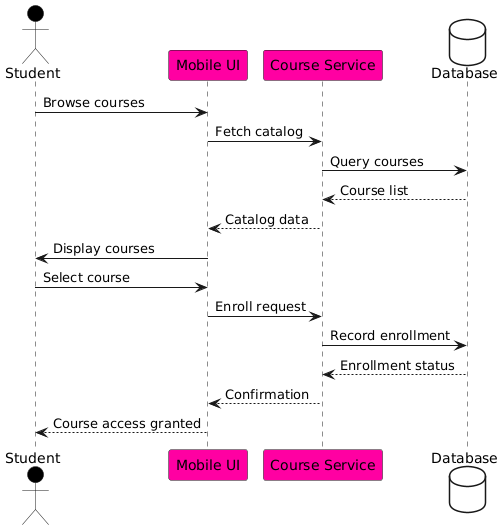
### Adaptive Assessment Pathway:



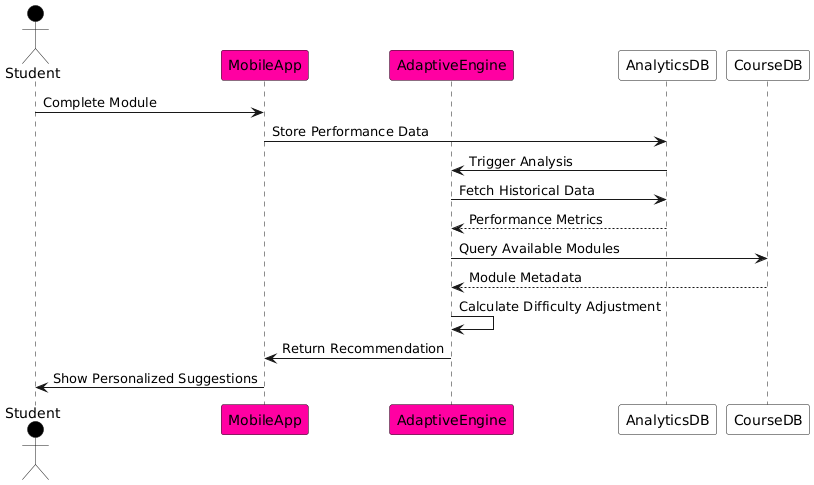
### Automated Learning:



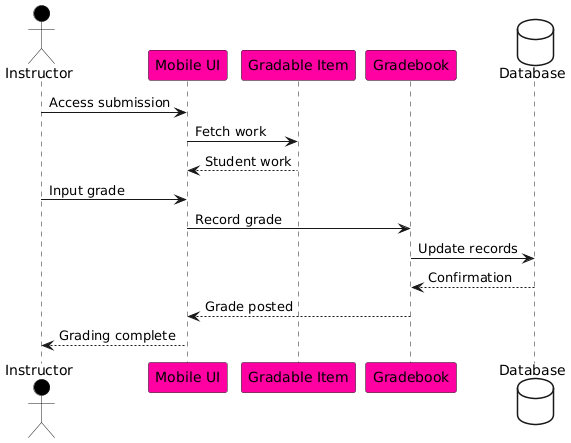
### Course Browsing & Enrollment:



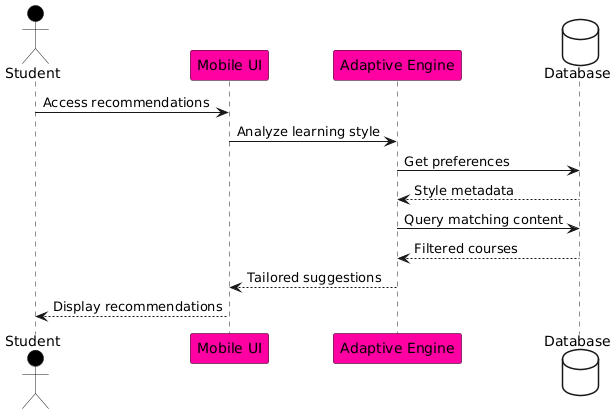
### Module Access & Difficulty Level:



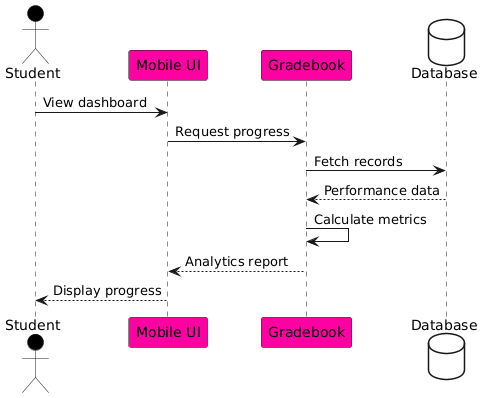
### Instructor Led Grading:



### Learning Style Recommendation:

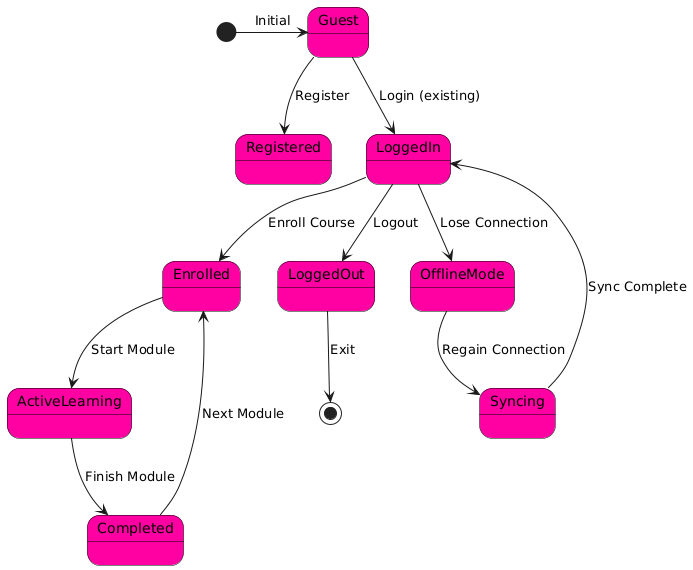


### Progress Tracking:

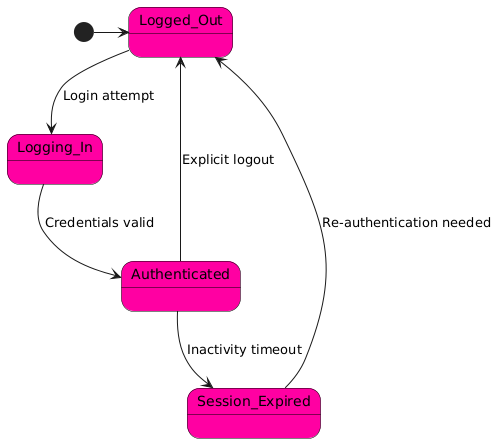


## State Diagrams:

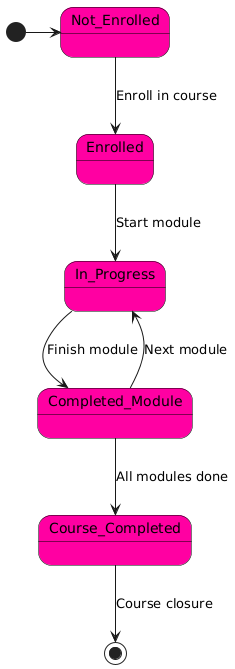
### User Session:



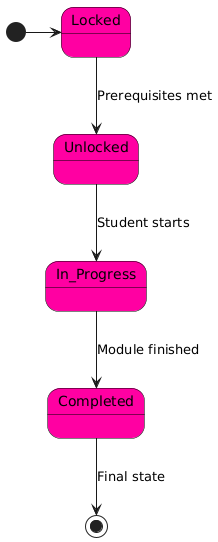
### User Authentication:



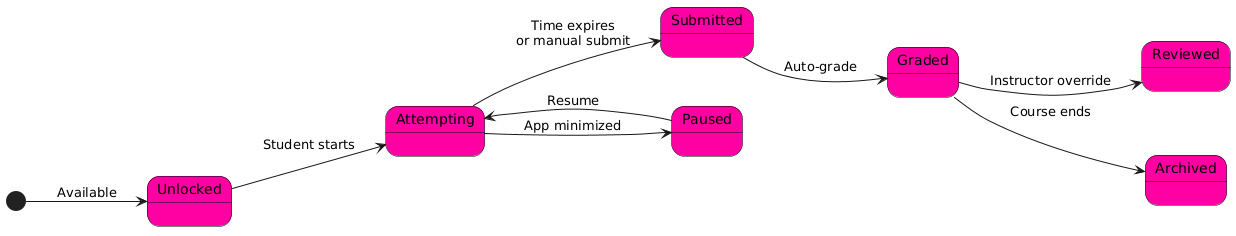
### Student Progress:



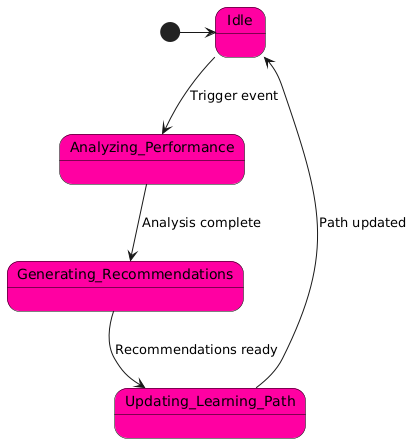
### Course Module:



### Adaptive Assessment:



### Adaptive Engine:



Appendix C: To Be Determined List

To be determined