Architecture description template
for use with ISO/IEC/IEEE 42010:2011

Architecture Description of Learning Management System Architecture

"Bare bones" edition version: 2.2

Template prepared by:
Rich Hilliard
r.hilliard@computer.org

Contents

Contents

1 Introduction

1.1 Identifying information

1.2 Supplementary information

1.3 Other information

1.3.1 Architecture evaluations

1.3.2 Rationale for key decisions

2 Stakeholders and concerns

2.1 Stakeholders

2.2 Concerns

2.3 Concern–Stakeholder Traceability

3 Viewpoints+

3.1 Conceptual View

3.2 Overview

3.3 Framed concerns and typical stakeholders

3.3.1 Concerns

3.3.2 Typical stakeholders

3.4 View: Conceptual View

3.4.1 Models+

3.4.2 Component Diagram

3.4.2 Layer Diagram

3.4.3 Known Issues with View

3.1 Execution View

3.2 Overview

3.3 Framed concerns and typical stakeholders

3.3.1 Concerns

3.3.2 Typical Stakeholders

3.4 View: Execution View

3.4.1 Models+

3.4.2 Sequence Diagram

3.4.3 Known Issues with View

4 Evaluation

A) Architecture decisions and rationale

A.1 Decisions

1 Introduction

This chapter describes introductory information items of the AD, including identifying and supplementary information.

1.1 Identifying information

Architecture name: Learning Management System Application.

System of Interest: a web-based application for managing and organizing online courses, assessments, and user interactions, designed for students, instructors, and admins. Built using Java Spring boot for RESTful API services and PostgreSQL for data storage.

1.2 Supplementary information

- **Date of Issue**: 27 Nov. 2024.
- **Version**: 1.0.
- **Authors**: Abanoub Essam, Marcelino Maximos, Mariam Ayman, Mohrael Raafat, Josiane Usama, and Maria Alfons.
- **Scope:** This document explains the LMS system's conceptual and execution viewpoints and focuses on its architecture styles, key design decisions, and evaluation.
- **Context:** The LMS system provides a backend service for managing online courses, the main users of the system are the students, instructors, and the admins who control and manage the settings of the system.
  The admins manage overall system settings by creating users, managing the added course, and viewing the list of enrolled students per course.
  The instructors can create courses, manage the course content, add assignments and quizzes, manage the students registered in this course, upload media files in courses, generate an OTP to maintain students' attendance, and track quiz scores, assignment submissions, and attendance.
  The students can enroll in courses, access course materials, take quizzes, hand in assignments, view assignments and quiz grades, and select the lesson to attend by entering the OTP received from the instructor.
  The system also has a notification system for enrollment confirmation, graded assignments, and course-related updates.
- **Glossary**
  - *LMS*: Learning Management System.
  - *RBAC*: Role-Based Access Control.
  - *REST*: Representational State Transfer.
  - *OTP*: One-Time Password

1.3 Other information

1.3.1 Architecture evaluations

We are applying the ATAM method to evaluate scenarios in the system (refer to Chapter 5).

1.3.2 Rationale for key decisions

- **Using layered modular architecture style:**
    - **Rationale:** dividing the system into layers like the application layer, business logic layer, and data access layer. In the application layer, it controls the role of users and gives permissions regarding the role of the user. In the Business logic layer, it executes the logic of authentication by implementing rules for course creation, enrollment, attendance, grading, and notifications, automating quiz grading, and validating OTPs for attendance. In the data access layer, it will operate on the entities like users, courses, lessons, and grades. It will secure stored data. This provides improvements like separation of concerns, offering flexibility and scalability to the system and some reusability for the components of the system.
- **Using PostgreSQL database:**
    - **Rationale:** implementation constraint, better performance for complex queries, and native JSON support.
- **Using Java Spring boot as a backend:**
    - **Rationale:** Implementation constraint provides easier customization and management and doesn't require XML configuration.
- **Using spring security for authentication and authorization:**
    - **Rationale:** it is a built-in security feature that is easily integrated with the Spring boot project and offers password encryption and hashing like bcrypt.
- **Using OTP for lesson enrollment without expiration:**
    - **Rationale:** improving user experience as students could preserve their OTPs and enroll in lessons whenever they are available to watch it, but this reduces the security.

2 Stakeholders and concerns

This chapter contains information items for stakeholders of the architecture, the stakeholders' concerns for that architecture, and the traceability of concerns to stakeholders.

2.1 Stakeholders

- **Admin**: The one responsible for user and system management.
- **Students**: Users of the system who enroll in courses, participate in assessments (quizzes and assignments), and generally monitor their learning processes.
- **Developers**: The team responsible for building and maintaining the system.
- **IT support teams**: Ensuring the system runs smoothly, handling technical issues, maintaining security, and implementing updates.
- **Business owner**: The business owner organizes and operates the business and takes financial risks.
- **Instructor**: creates and manages courses, assignments, quizzes and grades students

2.2 Concerns

- **Functionality**: the development team ensures that the system runs as required.
- **Scalability**: the development team designs the system to handle increasing users and data efficiently, and the business owner considers scaling the business or the project to achieve more success and outcomes.
- **Security**: the development team Protect user data and the system from unauthorized access, the business owner is concerned with the security of the data of sensitive business data, the IT support team encrypts the data transmission and storage, and Students' and instructors' data and grades must be safe from any form of unauthorized access, for admins the privacy of sensitive data is important.
- **Cost**: the business owner tries to achieve the best possible outcome with the minimum budget he can.
- **Availability**: for students and instructors the system should be available most of the time and never break down specifically during assessment or quiz times for students, for admins the system should be available to handle the management and settings of the system.
- **Performance**: the IT support team handles multiple concurrent users without performance degradation and quick response times for user actions and support requests. For students, the system should be responsive, and the content in any course, assignment, quiz, or anything else should load up quickly, for instructors. The system should handle many students and courses, Fast upload and retrieval of course material, and Minimal delays in loading courses, material, and student performance data, for admins the performance is a must having a smooth and fast interface.
- **Usability**: for students and instructors the system should be easy to use. Obviously, in terms of its functions, the number of clicks per operation should be minimal, and there should be a simple user interface.

- **Reliability**: The system must always be available and function correctly, especially during critical times like assignment submissions or exams. This is important for students and instructors, for admins the data consistency during operations like creating or updating or deleting a user or a course.

2.3 Concern–Stakeholder Traceability

	Admin	Instructor	Student	Developers	IT Support Team	Business owner
Performance	X	X	X	-	X	-
Maintainability	-	-	-	X	-	-
Functionality	-	-	-	X	-	-
Usability	X	X	X	-	-	-
Reliability	X	-	X	-	-	-
Scalability	-	-	-	X	-	X
Availability	X	X	X	-	-	X
Security	X	X	X	X	X	X
Observability	-	-	-	-	X	-
Cost	-	-	-	-	-	X

3 Viewpoints+

3.1 Conceptual View

Representing the Conceptual View of the System.

3.2 Overview

The conceptual view of the system represents how the system components interact and how interfaces are defined in the system.

3.3 Concerns and Stakeholders

3.3.1 Concerns

- Ensuring the design aligns with the architectural style is a concern to the system architects.
- The responsibilities of each component and the roles of the interfaces are a concern to the developers.
- The overview of the system and the interactions of the components and the high-level system functionalities is a concern to the project managers.

3.3.2 Typical stakeholders

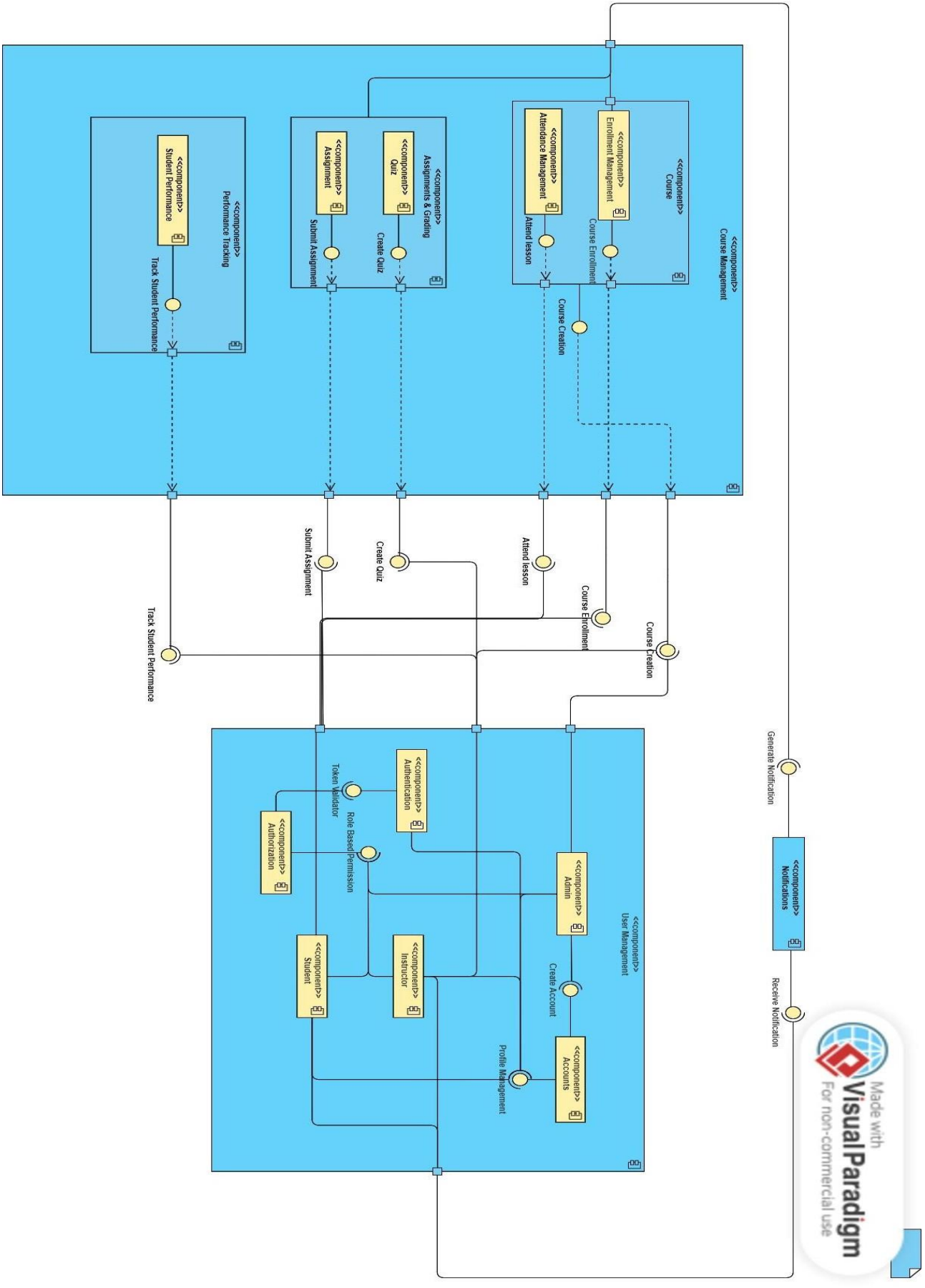
- System Architects
- Developers
- Project managers

3.4 View: Conceptual View

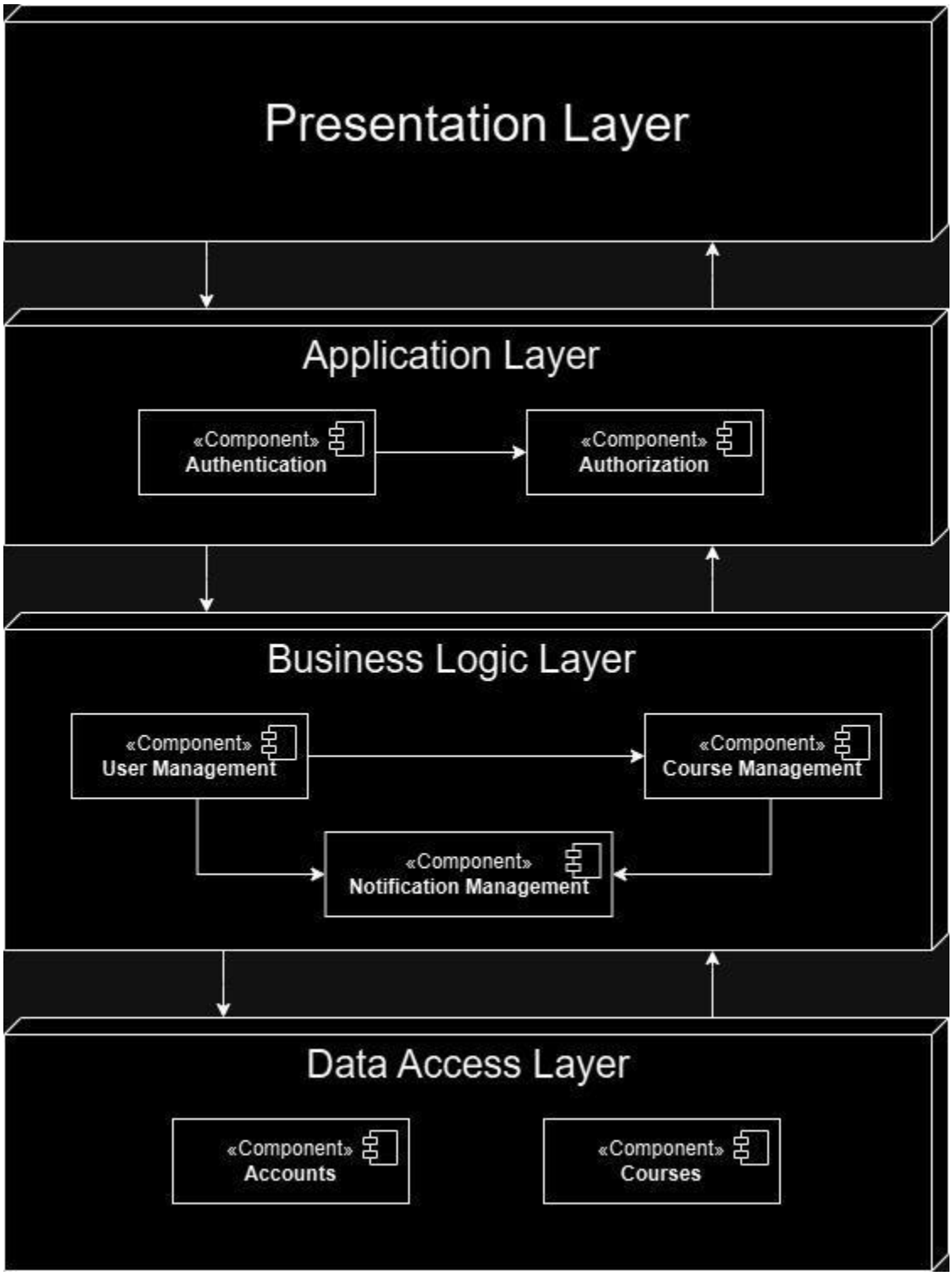
3.4.1 Models+

1. Component Diagram
2. Layer Diagram

3.4.2 Component Diagram



3.4.2 Component Diagram



3.4.3 Known Issues with View

- **Lack of detail:** the conceptual view doesn't provide enough details for stakeholders who need deeper information like the QA teams.
- **Ambiguity:** if the component diagram is not clear, stakeholders might misinterpret the architecture.

3.1 Execution View

Representing the Execution View of the System.

3.2 Overview

The conceptual view of the system represents how the system components interact and how interfaces are defined in the system.

3.3 Concerns and Stakeholders

3.3.1 Concerns

- Managing real-time updates for features like notifications is a concern to the developers and the system architects.
- Handling runtime errors gracefully (e.g., failed quiz submission or assignment uploads). This is a concern for the developers
- Clear logging for debugging and support this is a concern to the developers and system architects.

3.3.2 Typical stakeholders

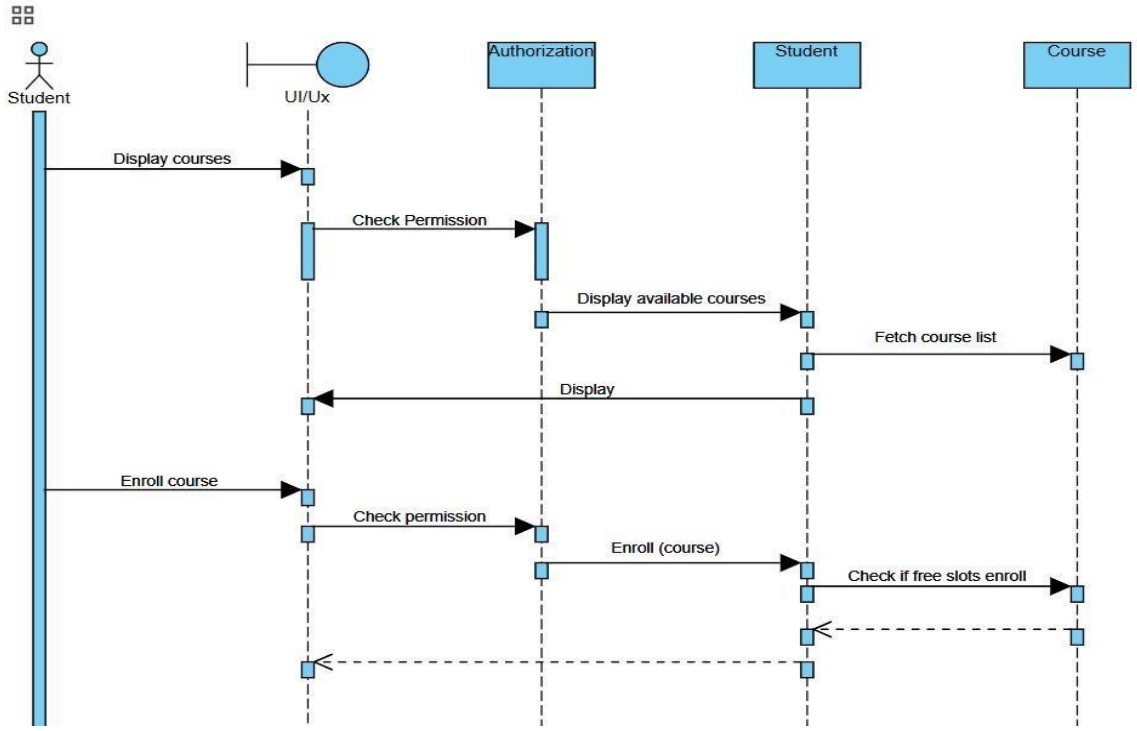
- System Architects
- Developers

3.4 View: Execution View

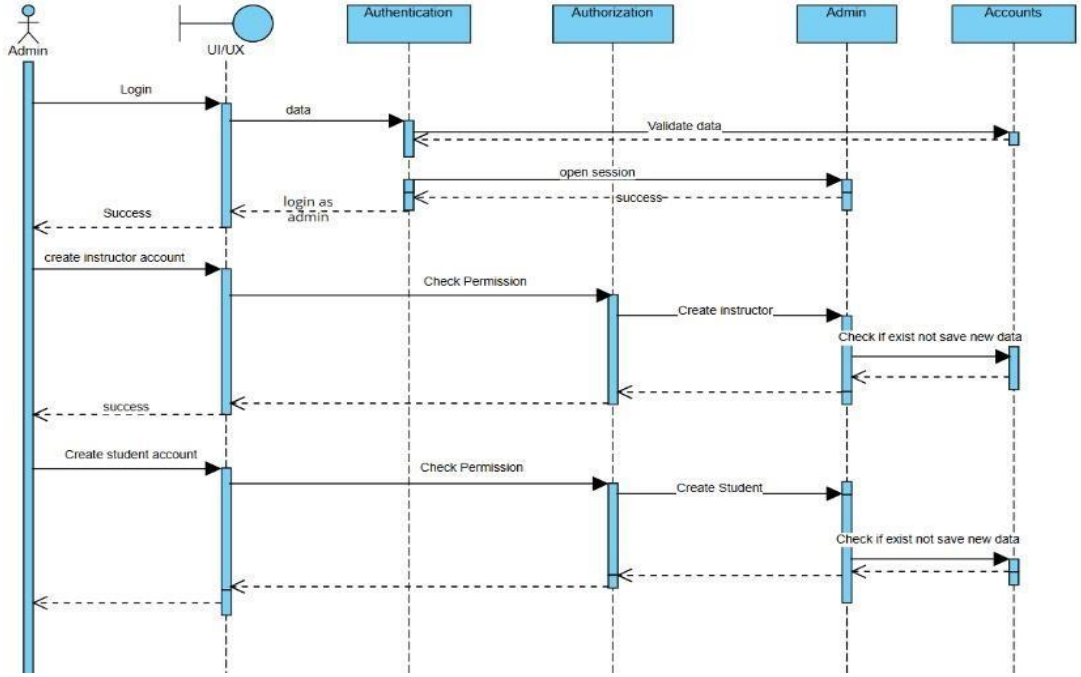
3.4.1 Models+

1. Sequence Diagram

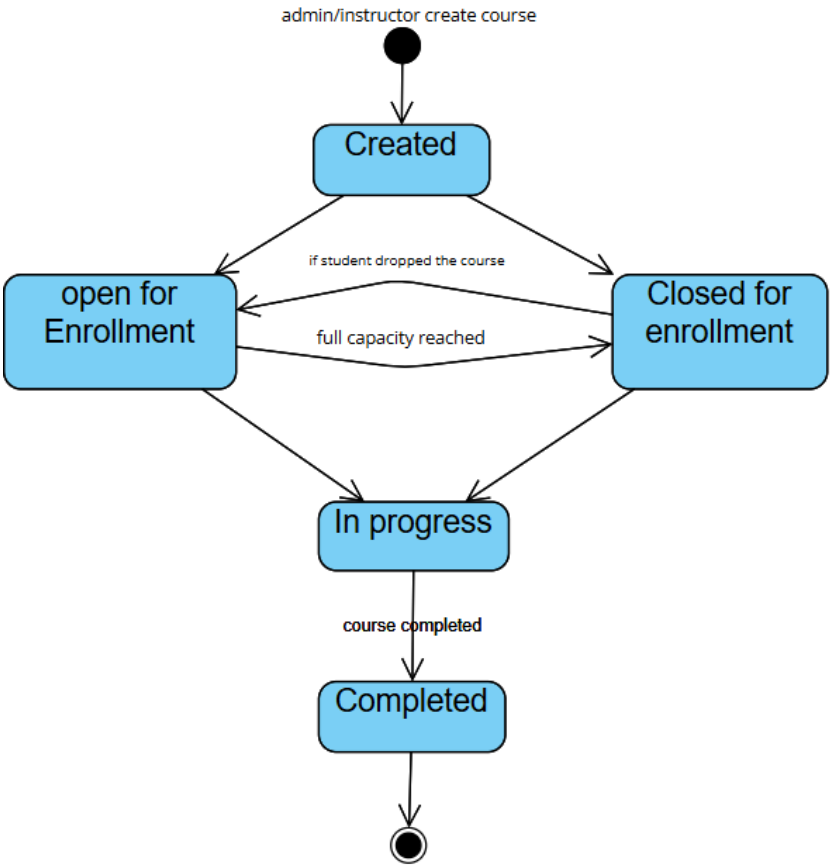
3.4.2 Sequence Diagram



3.4.2 Sequence Diagram



3.4.3 State Diagram



3.4.4 Known Issues with View

- **Performance Bottlenecks:** the runtime interactions between components may lead to delays.
- **Error Handling:** runtime errors are not handled gracefully which may cause crashes.

4 Evaluation

ATAM Scenarios Evaluation		
Availability	The server is crowded during quiz times by students	Caching static data like quiz questions using server-side caching
Availability	The network connection fails while submitting a quiz	Save the user's quiz progress to the database as the user proceeds, so that when the network restores the user's progress will be restored from where he left.
Scalability	Under heavy load, users may experience slow response times, or in extreme cases, the system may become unresponsive.	The application server will be scaled horizontally by adding more instances (as an example, the system will be deployed on a two-machine cluster or use a scaling service in the cloud). This will help the system load the user requests across several servers hence in the high traffic periods it will not slow down.
Modifiability	The LMS needs to incorporate a new file format (e.g., PDF) for assignment submissions, in addition to the current supported formats (e.g., DOCX, PPT).	The backend logic for file handling will need to be revised to support the new file format, which includes updating validation and storage processes.
Performance	Students generate many simultaneous requests to view grades.	Grades are pre-cached, and the system serves them from the cache instead of querying the database repeatedly.
Security	If a user remains inactive for an extended period, such as 14 days, their session could remain open, posing a security risk.	To mitigate this risk, the system will automatically log out users who have been inactive for 14 days.

A) Architecture decisions and rationale

A.1 Decision

- **Using layered architecture style:**
    - **Rationale:** dividing the system into layers like the application layer, business logic layer, and data access layer. In the application layer, it controls the role of users and gives permissions regarding the role of the user. In the Business logic layer, it executes the logic of authentication by implementing rules for course creation, enrollment, attendance, grading, and notifications, automating quiz grading, and validating OTPs for attendance. In the data access layer, it will operate on the entities like users, courses, lessons, and grades. It will secure stored data. This provides improvements like separation of concerns, offering flexibility and scalability to the system and some reusability for the components of the system.
- **Using PostgreSQL database:**
    - **Rationale:** implementation constraint, better performance for complex queries, and native JSON support.
- **Using Java Spring boot as a backend:**
    - **Rationale:** Implementation constraint provides easier customization and management and doesn't require XML configuration.
- **Using spring security for authentication and authorization:**
    - **Rationale:** it is a built-in security feature that is easily integrated with the Spring boot project and offers password encryption and hashing like bcrypt.
- **Using OTP for lesson enrollment without expiration:**
    - **Rationale:** improving user experience as students could preserve their OTPs and enroll in lessons whenever they are available to watch it, but this reduces the security.