Shailja Choudhary 24MIM10166

Virtual Classroom Platform: Project Report

Document Details	Value
Project Title	Virtual Classroom Platform
Project Type	Traditional 3-Tier Java Web Application
Core Technologies	Java, JSP, Servlets, MySQL
Current Status	Core Model and Service Logic Implemented
Date of Report	November 2025

1. Introduction and Project Goals

The Virtual Classroom Platform is designed to create an interactive and centralized digital environment for remote teaching and learning. The primary goal is to facilitate essential e-learning functionalities, enabling teachers to manage courses and content while providing students with easy access to educational materials and assignments. The architecture follows established enterprise patterns to ensure maintainability and scalability.

2. Key Features

The following functionalities are implemented or planned for the system, based on the project requirements:

- **User Authentication:** Secure and separate login interfaces for **Students** and **Teachers**.
- **Role-Based Access Control:** Strict enforcement of privileges, ensuring only Teachers can manage course content.
- **Course Management:** Teachers have full CRUD (Create, Read, Update, Delete) capabilities over courses and subjects.
- **Content Sharing:** The platform supports uploading and linking various lecture materials, including video tutorials, presentations, and documents.

- **Interactive Tools:** Future plans include integration for messaging and collaborative features to enhance communication.

3. Technical Architecture and Design

The Virtual Classroom Platform utilizes a **Traditional 3-Tier Java Web Application Architecture**, ensuring a clear separation of concerns between the presentation, business logic, and data layers.

3.1. Layered Architecture

The core Java components are structured into the following logical layers, as demonstrated by the provided files:

Layer	**Component**	**Description**
Presentation	JSP, HTML/CSS/JS (Referenced in README.md)	Handles the user interface and captures user input. Requests are forwarded to the Controller/Service layer.
Service (Business Logic)	ClassroomService.java	Contains all the core logic, such as authentication (login) and course administration (createCourse). It implements security and validation rules.
Model (Data Structures)	User.java, Course.java	Defines the central data entities and their behavior. Models are independent of the data storage mechanism.
Data Access/Persistence	JDBC / MySQL (Referenced in README.md)	Manages connections and transactions with the **MySQL** database.

3.2. Object-Oriented Design (OOP) Implementation

- **Encapsulation and Security:** The User.java model encapsulates user data, including a critical field for passwordHash. Access to user state is controlled via methods, and the class includes a foundational method (checkPassword) to simulate secure credential validation.

- **Simulated Persistence:** The ClassroomService.java utilizes native Java **Map and HashMap** data structures to simulate a running database, allowing for decoupled development and demonstrating proficiency in data structure management before integrating JDBC or JPA/Hibernate.

4. Technology Stack

The project relies on proven, industry-standard technologies for a robust deployment environment:

Category	**Technology**	**Purpose**
Backend Core	Java JDK 8+	Execution environment for server-side logic.
Web Server	Apache Tomcat (Servlet Container)	Deploys and runs Java Servlets and JSP files.
Database	MySQL	Stores all persistent user, course, and content data.
Data Access	JDBC (MySQL Connector)	Java API for connecting to the MySQL database.
Build Tool	Apache Maven	Manages project dependencies and produces the distributable WAR file.

5. Core Data Model Analysis

The following entities form the backbone of the application's data structure:

5.1. User Model (User.java)

Represents all individuals in the system, differentiated by role:

- **Attributes:** id, username, passwordHash, name, email, and role (STUDENT or TEACHER).
- **Functionality:** Provides getters/setters and the checkPassword method for access control.

5.2. Course Model (Course.java)

Represents the educational content available on the platform:

- **Attributes:** id, title, description.
- **Relationships:** Contains a reference to the User object who is the teacher of the course.
- **Content:** Manages a List<String> of contentUrls for lectures and materials.

6. Deployment and Future Work

6.1. Local Setup Instructions

The project is packaged using Maven and deployed as a WAR file on a servlet container (e.g., Tomcat). A working MySQL database connection is required. Detailed steps are provided in the README.md.

6.2. Future Enhancements

Potential future development includes:

1. Replacing simulated persistence with a fully functional JDBC or Spring Data JPA **Repository Layer**.
2. Implementing a dedicated **Assignment/Grading** module.
3. Enhancing user interaction with a dedicated **REST API** using Spring Boot, replacing the JSP/Servlet controllers.