E-Commerce Product Catalog

Developed by:

Aadvik Krishna (24scse1410243)

Raj Chauhan (24scse1410288)

Pandit Krishna (24scse1410289)

Aman Gill (24scse1410275)

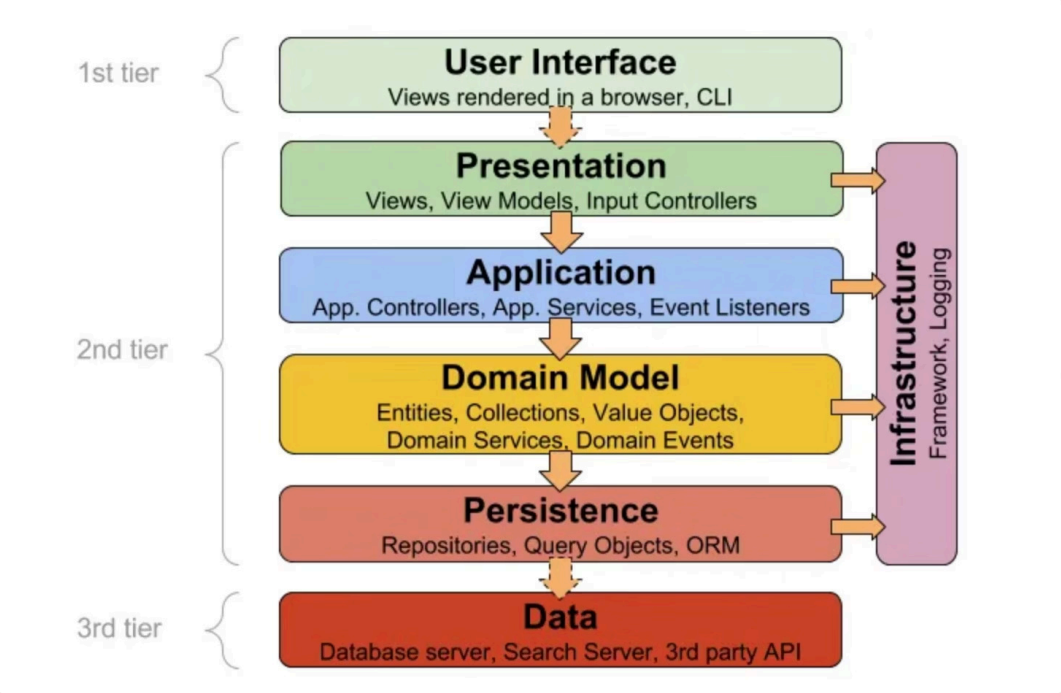
Project Overview

- An interactive Java-based desktop application to manage product listings in an online store.
- Enables users to perform CRUD (Create, Read, Update, Delete) operations on product data.
- Incorporates end-to-end flow from UI to database ensuring full-stack understanding.

Technologies Used

- Java (JDK 11+): Core programming language for backend and GUI.
- MySQL: Relational database to store persistent product data.
- JDBC (Java Database Connectivity): Java API to connect and interact with MySQL database.
- Swing: Java's GUI toolkit for building a responsive and interactive user interface.
- Maven: Used for project and dependency management to ensure clean builds.
- Eclipse/VS Code: IDEs utilized for development, debugging, and testing.
- GitHub: Source code repository for version control and collaboration.
- MySQL Connector/J: JDBC driver required to connect Java application with MySQL.

Project Architecture & Structure



- Model Layer: Contains Java POJOs (Plain Old Java Objects) representing data entities.
- DAO Layer: Responsible for data access and encapsulates all interactions with the DB.
- UI Layer: Java Swing forms for interacting with end users through buttons, tables, forms.
- Main Application: Entry point of the system triggering UI and logic integration.
- Resources: Contains properties/configuration files like DB credentials.

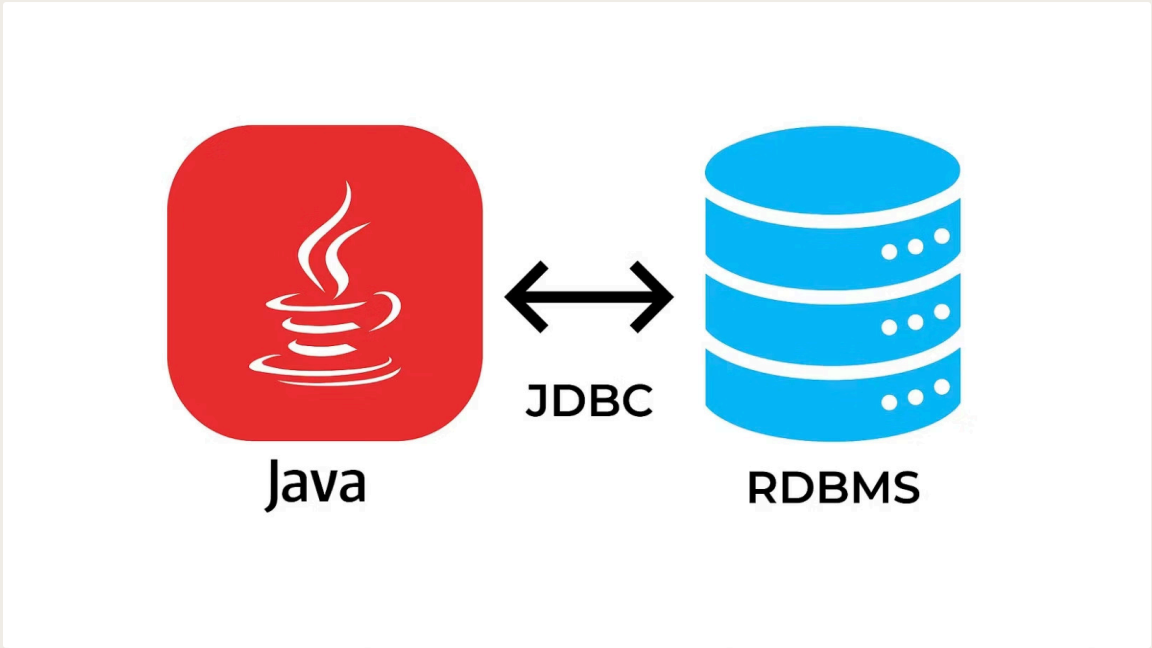
Database Design & Schema

- Designed a normalized database schema to ensure data integrity and easy scalability.
- products Table:
- - product_id (INT, Primary Key, Auto-incremented)
- - name (VARCHAR), description (TEXT)
- - price (DECIMAL), quantity (INT)
- Supports indexing and optimized queries for scalable product search.

Creating MySQL Table - SQL Script

```
CREATE TABLE products (
  product_id INT PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(100) NOT NULL,
  description TEXT,
  price DECIMAL(10, 2) NOT NULL,
  quantity INT NOT NULL
);
```

JDBC Integration



- Database connection is managed via JDBC using MySQL Connector.
- Steps followed:
  - Load JDBC Driver Class.
  - Use DriverManager to establish a connection.
  - Execute Prepared Statements to perform DB operations.
- Benefits: Reusability, maintainability, and prevention of SQL injection attacks.

Model and DAO Implementation

- Product.java: Java class encapsulating attributes of a product.
- ProductDAO.java: Data Access Object class performing DB operations like insert, update, delete, fetch.
- Methods follow single responsibility principle and handle SQLExceptions gracefully.

User Interface (UI) Design

- UI built using Java Swing, providing a desktop application experience.
- Includes labeled text fields, action buttons, and a JTable for listing product records.
- Main actions are Add Product, Update Product, Delete Product, and View All Products.
- User-friendly layout that follows UI/UX best practices.

UI Aesthetics & Responsiveness

- Minimalist and professional visual design using standard Swing components.
- Strategic use of layout managers like GridBagLayout for structured alignment.
- JTable with scroll pane for handling large datasets.
- Field validations and error prompts enhance user experience.

Code Quality & Best Practices

- Clean, modular code adhering to Java coding conventions.
- Functions are logically grouped in classes following OOP principles.
- Comprehensive inline comments for maintainability and clarity.
- Uses try-with-resources for auto-closing DB connections to avoid memory leaks.

Setup & Running Instructions

- Install JDK and MySQL on your machine.
- Create a database named 'ecommerce' and run the provided SQL script.
- Clone the GitHub repository and open the project in Eclipse or VS Code as a Maven project.
- Edit db.properties to add correct username and password.
- Build the project and run the Main class to launch the UI application.

GitHub Repository

- Contains entire project source code, organized by MVC principles.
- Includes SQL script, sample data, and README with setup instructions.
- Publicly available for reviewers to test and run.

Summary & Learnings

- Hands-on experience with Java desktop development and database integration.
- Improved understanding of software architecture and JDBC APIs.
- Learned how to structure Java projects for scalability and readability.
- Acquired knowledge of UI/UX principles applied to Java Swing.