CSC3810: Database Principles — Lab: Intro To JDBC

Implementation of a Three-Tier JDBC Application

Justin Featherstone

March 24th, 2025

1 Introduction

This pdf documents the implementation of a three-tier Java Database Connectivity (JDBC) application that interacts with multiple databases. The application follows the Data Access Layer (DAL) architecture pattern and demonstrates different types of JDBC statements for database interaction. The three databases used in this project are:

- MealPlanning
- ArcadeGames
- VideoGameSystems

2 Architecture Overview

The application follows a three-tier architecture:

1. **Presentation Layer**: Handles user interactions through a console-based menu system
2. **Business Logic Layer**: Defines data transfer objects (DTOs) for storing query results
3. **Data Access Layer**: Manages database connections and provides database operations

3 Technical Environment

- **JDBC Driver**: MySQL Connector/J
- **Version**: 8.0.30
- **Database Type**: MySQL
- **JDK Version**: 17

4 Implementation Details

4.1 Data Manager Implementation (Requirement 1)

The DataMgr class successfully implements the core requirements:

- Centralized connection management through singleton pattern
- Connections to multiple databases (MealPlanning, ArcadeGames)
- Proper resource management with connection closing
- Robust error handling and logging

Code quality is high, with proper exception handling and logging using java.util.logging.

4.2 Three-Layer Architecture (Requirement 2)

The codebase successfully implements the three-layer architecture:

4.2.1 Presentation Layer

- IntroToPresentationLayer.java handles user interaction
- Clean separation from business logic
- User input handling for database credentials

4.2.2 Business Logic Layer

Evidence of DTOs (Data Transfer Objects):

- Recipe class
- Ingredient class
- ArcadeGame, Player, and Score classes

4.2.3 Data Access Layer

Well-structured DAL implementation:

- DataMgr for connection management
- Separate DAL classes for different databases
- Proper resource cleanup

4.3 Multiple DAL Implementation (Requirement 3)

The ArcadeGamesDAL class demonstrates:

- Statement usage for basic queries
- PreparedStatement for parameterized queries
- CallableStatement for stored procedures
- Consistent error handling and logging

5 Technical Implementation Details

5.1 JDBC Statement Types

The codebase demonstrates all three JDBC statement types:

1. Basic Statement:

- Used in getAllRecipes() method
- Suitable for static queries

2. PreparedStatement:

- Used in getIngredientsForRecipe()
- Prevents SQL injection
- Better performance for repeated execution

3. CallableStatement:

- Implemented in getRecipesFromStoredProcedure()
- Proper parameter handling
- Stored procedure execution

6 Areas for Improvement

- Connection pooling could be implemented for better performance
- Transaction management could be added
- More comprehensive error recovery mechanisms
- Unit tests could be added

7 Learning Outcomes

Through this lab, I gained practical experience with:

- JDBC database connectivity
- Three-tier architecture implementation
- Different types of SQL statements
- Resource management in database applications
- Error handling and logging
- Software design patterns (Singleton, DAO)

8 Conclusion

The implementation successfully meets all core requirements while demonstrating good software engineering practices. The code is well-structured, maintainable, and follows proper separation of concerns. The experience provided valuable insights into real-world database application development and the importance of proper architectural design.