Synopsis: Movie Ticket Booking System (Console Application)

DHIRAJ PAL

SID: AF05001363

Abstract

This document outlines the **Movie Ticket Booking System**, a foundational Java console application developed to demonstrate core programming proficiency. The project utilizes **Object-Oriented Programming (OOP)** principles to structure data models and leverages the **Java Collections Framework** (specifically ArrayList and HashMap) to manage data per- sistence and execute real-time transactional logic, including the full suite of **CRUD (Create, Read, Update, Delete)** operations. The system's primary function is to manage movie show- times and seat availability through console interaction.

1 Project Objectives

The primary objectives of this project are twofold:

1. To implement a functional, transactional system using standard Java practices.
2. To practically apply complex data structures, specifically using HashMap for efficient keyed lookups and ArrayList for dynamic inventory management.

2 System Architecture and Models

The application follows a clear separation of concerns, dividing responsibilities among distinct classes. The business logic is isolated within the BookingManager class, which interacts directly with the data models.

2.1 Data Models (Entities)

Table 1: Project Data Models

Class	Role and Key Data
Movie	Represents a single screening; tracks movie details and manages the availableSeats ArrayList CustomerStores the customer's personal information (name, phone).
Ticket	The transactional object; links a specific Movie and Customer to a set of bookedSeats.

2.2 Service Layer

The BookingManager acts as the persistent data store and transaction processor.

- **Data Storage:** Movies are stored in a HashMap<Integer, Movie> for quick retrieval using the Movie ID. Tickets are stored in an ArrayList<Ticket>.
- **CRUD Functions:** It encapsulates methods like bookTicket (Create/Update) and cancelTicket (Delete/Update) to ensure data integrity.

3 Implementation of CRUD Operations

The core functionality of the system is realized through collection operations corresponding to the CRUD pattern:

- **C (Create):** A new Ticket object is instantiated and added to the global bookings ArrayList.
- **R (Read):** Movie details are fetched instantly using the HashMap's get() method. All movies and tickets are retrieved via iteration or by accessing collection values.
- **U (Update):** Seat availability is updated by utilizing the remove() and add() methods on the Movie's internal ArrayList of seats during booking and cancellation, respectively.
- **D (Delete):** A specific Ticket is located by its ID and removed from the bookings ArrayList using the remove() method.