Movie Ticket Management System

Name: SHINDE PURVA PRASAD

Roll No.: 134

# Title

Movie Ticket Management System

This document outlines the implementation of an Movie Ticket Management System using Hibernate ORM in Java. It includes the setup, configuration, and execution of core airline functionalities like flight registration, passenger management, booking flights, canceling bookings, and viewing transaction history.

**Objective**

To develop an airline management system using Hibernate to demonstrate the use of object-relational mapping for database operations.

**Tools and Technologies Used**

1. Java
2. Hibernate ORM
3. MySQL Database
4. Maven
5. Eclipse/IntelliJ IDEA

**System Requirements**

1. JDK 8 or above
2. MySQL Server
3. Hibernate Core Library
4. Maven for dependency management

**System Design**

The system includes the following major components:

* Movie Entity
* Customer Entity
* TicketTransaction Entity
* Hibernate DAO Layer
* Service Layer
* Main Application Layer

**Hibernate Configuration**

xml

CopyEdit

<hibernate-configuration>

<session-factory>

<property name="hibernate.connection.driver\_class">com.mysql.cj.jdbc.Driver</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/airlinedb</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">password</property>

<property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

<property name="show\_sql">true</property>

<property name="hbm2ddl.auto">update</property>

<mapping class="model.Movie"/>

<mapping class="model.Customer"/>

<mapping class="model.TicketTransaction"/>

</session-factory>

</hibernate-configuration>

**Entity Classes**

**Movie.java**

java

CopyEdit

@Entity

public class Movie {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String flightNumber;

private String destination;

private int capacity;

}

**Customer.java**

java

CopyEdit

@Entity

public class Customer {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private String email;

}

**TicketTransaction.java**

java

CopyEdit

@Entity

public class TicketTransaction {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String status; // Booked / Canceled

private Date date;

@ManyToOne

private Movie flight;

@ManyToOne

private Customer passenger;

}

**DAO and Service Layer**

Same structure as the BankService and HibernateUtil classes, adjusted for Movie, Customer, and TicketTransaction handling.

**Main Application Class**

**Main.java**

import os

from zipfile import ZipFile

# Define the directory structure and create Eclipse project folder

project\_dir = "/mnt/data/MovieTicketManagementSystem\_Eclipse"

src\_dir = os.path.join(project\_dir, "src", "com", "movie", "management")

os.makedirs(src\_dir, exist\_ok=True)

# Sample Java files based on the assignment structure

main\_class = """

package com.movie.management;

public class Main {

public static void main(String[] args) {

MovieTicketService service = new MovieTicketService();

// Add a customer

Customer customer = new Customer();

customer.setName("John Doe");

customer.setEmail("john@example.com");

service.registerCustomer(customer);

// Add a movie

Movie movie = new Movie();

movie.setMovieName("Inception");

movie.setTheater("PVR");

movie.setCapacity(150);

service.addMovie(movie);

// Book and cancel ticket

service.bookTicket(movie.getId(), customer.getId());

service.cancelTicket(movie.getId(), customer.getId());

}

}

"""

movie\_class = """

package com.movie.management;

public class Movie {

private int id;

private String movieName;

private String theater;

private int capacity;

// Getters and Setters

}

"""

customer\_class = """

package com.movie.management;

public class Customer {

private int id;

private String name;

private String email;

// Getters and Setters

}

"""

ticket\_transaction\_class = """

package com.movie.management;

import java.util.Date;

public class TicketTransaction {

private int id;

private String status; // Booked / Cancelled

private Date date;

private Movie movie;

private Customer customer;

// Getters and Setters

}

"""

service\_class = """

package com.movie.management;

public class MovieTicketService {

public void registerCustomer(Customer customer) {

// Implementation logic

}

public void addMovie(Movie movie) {

// Implementation logic

}

public void bookTicket(int movieId, int customerId) {

// Booking logic

}

public void cancelTicket(int movieId, int customerId) {

// Cancel logic

}

}

"""

# Write sample files

with open(os.path.join(src\_dir, "Main.java"), "w") as f:

f.write(main\_class.strip())

with open(os.path.join(src\_dir, "Movie.java"), "w") as f:

f.write(movie\_class.strip())

with open(os.path.join(src\_dir, "Customer.java"), "w") as f:

f.write(customer\_class.strip())

with open(os.path.join(src\_dir, "TicketTransaction.java"), "w") as f:

f.write(ticket\_transaction\_class.strip())

with open(os.path.join(src\_dir, "MovieTicketService.java"), "w") as f:

f.write(service\_class.strip())

# Zip the project folder

zip\_path = "/mnt/data/MovieTicketManagementSystem\_Eclipse.zip"

with ZipFile(zip\_path, "w") as zipf:

for root, dirs, files in os.walk(project\_dir):

for file in files:

file\_path = os.path.join(root, file)

arcname = os.path.relpath(file\_path, project\_dir)

zipf.write(file\_path, arcname)

zip\_path