Practical 6

Modeling UML Class Diagrams

Topic: Structural and Behavioral Aspects | Class Diagram | Elements in Class Diagram | Class | Relationships | Draw Class Diagram

🥅 Aim of the Experiment:

To graphically represent the structure of a Movie Recommendation System using a UML Class Diagram, showcasing relationships, attributes, and operations of various classes involved.

📖 Introduction:

A UML Class Diagram is used to describe the structure of a system by showing its classes, attributes, methods, and relationships. For a Movie Recommendation System, it helps in understanding how different components like Users, Movies, Ratings, and the Recommendation Engine interact with each other.

🎯 Objectives:

After completing this experiment, you will be able to:

Graphically represent the essential classes of the Movie Recommendation System.

Show associations between classes (like User–Movie, Movie–Rating).

Identify and model the logical sequence of system operations pictorially using UML class relationships.

📚 Theory:

Structural and Behavioral Aspects:

Structural: Involves static components like classes, their attributes, and relationships.

Behavioral: Defines the operations and interactions between classes.

🧱 Class Diagram for Movie Recommendation System:

🎬 Main Classes and Elements:

User

Attributes: userID, name, email, age

Methods: login(), rateMovie(), browseMovies()

Movie

Attributes: movieID, title, genre, director, releaseDate

Methods: getDetails(), getAverageRating()

Rating

Attributes: ratingID, userID, movieID, score, timestamp

Methods: submitRating()

RecommendationEngine

Attributes: algorithmType

Methods: generateRecommendations(userID), filterByGenre()

Admin

Attributes: adminID, username, password

Methods: addMovie(), deleteMovie(), updateMovie()

🔗 Relationships:

User ↔ Rating: One-to-many (A user can rate many movies)

Movie ↔ Rating: One-to-many (A movie can be rated by many users)

User ↔ Movie: Many-to-many (Linked through Rating)

Admin ↔ Movie: Admin manages movie data (One-to-many)

RecommendationEngine ↔ User/Movie: Uses data to generate suggestions

🧩 Composition:

A Rating cannot exist without a User and a Movie, indicating composition.

RecommendationEngine is dependent on existing User and Movie data.

📊 Case Study:

Scenario: A user logs in, browses movies, rates a few, and receives personalized recommendations.

How Class Diagram Helps: It clearly defines object responsibilities and associations which help during coding or system modeling phases.

📚 References:

UML Distilled by Martin Fowler

Online UML diagramming tools like Lucidchart, Draw.io

IEEE standards for modeling and design