Established – 1961 Subject: Object Oriented Programming System

SEVA SADAN'S

R. K. TALREJA COLLEGE

OF

ARTS, SCIENCE & COMMERCE

ULHASNAGAR – 421 003

CERTIFICATE

This is to certify that Mr. MAYANK SINGH RAJPUT of F.Y. Information Technology (FYIT) Roll No. 2541051 has satisfactorily completed the Object Oriented Programming using C++ Mini Project entitled MOVIE RENTAL SYSTEM during the academic year 2025 – 2026, as a part of the practical requirement. The project work is found to be satisfactory and is approved for submission.

PROF. INCHARGE **HEAD OF DEPT**

Asst.Proff.Kumodh Kukreja Prof.Laxmi Jeshwani

INDEX

1. Introduction

The Movie Rental System using C++ is a console-based application designed to simulate the working of a real-life movie rental shop. This system helps users to view available movies, rent movies, return rented movies, and calculate rental charges in an organized manner.

The project is developed using ObjectOriented Programming (OOP) concepts which make the system modular, reusable, and easy to understand. Instead of handling movies manually, this system automates the rental process and keeps track of movie availability.

By implementing classes, objects, inheritance, encapsulation, and other OOP features, the Movie Rental System provides a practical understanding of how software solutions are developed for real-world problems. This project is suitable for beginners and helps strengthen core C++ and OOP fundamentals

- **Topic Description**

Topic Name: Movie Rental System Using C++

The Movie Rental System is a menu-driven application that allows users to manage movie rentals through a single interface. The system mainly focuses on the following features:

1. **Display available movies**
2. **Rent a movie**
3. **Return a movie**
4. **Show rental details**

Each movie has attributes such as movie name, genre, rental price per day, and availability status. When a user rents a movie, the availability is updated automatically to prevent double renting.

The system uses a base class to store common properties and derived classes to manage specific movie-related operations. Rental charges are calculated based on the number of days the movie is rented.

This system reduces manual work, avoids confusion, and provides a smooth rental experience using simple console interactions.

- **OOP Concepts Used**

a) Class and Object

- **Class acts as a blueprint for movies and rental operations.**
- **Objects represent real movie records.**

Example classes used:

- **Movie**
- **RentalSystem**

b) Inheritance

Inheritance is used to reuse common properties such as availability status.
Example:

```
class Rental { public:
   bool available;
};

class Movie : public Rental {
};
```

c) Encapsulation

Data members are kept private and accessed using public functions, which improves data security.

Example:

```
class Movie { private:
   string name; public:
   void setData();
};
```

d) Polymorphism

Same function name is used with different behaviors using function overloading style display functions.

e) Static Data Members

Static variables are used to manage movie stock shared among all objects.

---

f) Abstraction

The user interacts only with menu options like rent or return movie, while internal logic remains hidden.

- **Implementation of Code**

```
#include <iostream> #include
<string>
using namespace std;

// -------- BASE CLASS --------
class Rental { public:     bool
available = true;
};


// -------- MOVIE CLASS -------- class
Movie : public Rental { private:
    string movieName;
string genre;     int
pricePerDay;
```

```
public:
   void setMovie(string name, string g,
int price) {
        movieName = name;
        genre = g;
        pricePerDay = price;
    }

void displayMovie() {
        cout << "\nMovie Name : "
<< movieName;
        cout << "\nGenre      : " << genre;
        cout << "\nPrice/Day  : Rs "
<< pricePerDay;
        cout << "\nStatus     : " <<
(available ? "Available" :
"Rented") << endl;
    }

int rentMovie(int days) {
available = false;
        return pricePerDay * days;
```

```
    }
    void returnMovie() {
        available = true;
    }

bool isAvailable() {
        return available;
    }
};
```

```
// -------- MAIN SYSTEM --------
int main() {     Movie m1;
m1.setMovie("Inception",
"Sci-Fi", 150);

    int choice, days;

do {
        cout << "\n\n--- Movie Rental
System ---";         cout << "\n1. View
Movie";         cout << "\n2. Rent
Movie";
        cout << "\n3. Return Movie";
        cout << "\n4. Exit";         cout
<< "\nEnter choice: ";
        cin >> choice;

switch (choice) {         case
1:
            m1.displayMovie();
break;

        case 2:
```

```
            if (m1.isAvailable()) {
cout << "Enter number of days: ";
                cin >> days;                cout
<< "Total Rent: Rs
" << m1.rentMovie(days) <<
endl;
            } else {
                cout << "Movie already
rented!\n";
            }
            break;

   case 3:
            m1.returnMovie();
cout << "Movie returned
successfully.\n";
            break;
```

```
case 4:
            cout << "Thank you!\n";
break;

        default:            cout
<< "Invalid
choice!\n";
        }

  } while (choice != 4);

    return 0;
}
```

. Output

```
===== Movie Rental System =====
1. View Movies
2. Rent Movie
3. Return Movie
4. Exit
Enter choice: 1

--- Available Movies ---
1. Inception | Status: Available
2. Interstellar | Status: Available
3. Avengers | Status: Available
4. Joker | Status: Available
5. Titanic | Status: Available

===== Movie Rental System =====
1. View Movies
2. Rent Movie
3. Return Movie
4. Exit
Enter choice:
```

```
Enter choice: 2
Enter Movie ID to rent: 4
Enter rental days: 7
Movie rented successfully for 7 days.
Total Rent: Rs. 770
```

```
===== Movie Rental System =====
1. View Movies
2. Rent Movie
3. Return Movie
4. Exit
Enter choice: 3
Enter Movie ID to return: 4
Movie returned successfully.
```

```
===== Movie Rental System =====
1. View Movies
2. Rent Movie
3. Return Movie
4. Exit
Enter choice: 4
Thank you for using Movie Rental System.


=== Code Execution Successful ===
```

OUTPUT key points

- **Displays available movie details**
- **Allows user to rent a movie for selected days**
- **Calculates total rental charges**
- **Prevents renting an already rented movie**
- **Successfully returns the movie and updates availability**

. Conclusion

The Movie Rental System successfully demonstrates the use of Object-Oriented Programming concepts using C++. The system automates movie rental operations such as viewing, renting, and returning movies.

Key OOP features like inheritance, encapsulation, abstraction, and polymorphism improve code organization and reusability. The menu-driven interface makes the system easy to use and beginnerfriendly.

This project provides a solid foundation for developing more advanced rental or inventory management systems in the future.