Experiment – 1.2

Student Name: Lokesh Yadav
UID: 22BCS16604
Branch: BE-CSE
Section/Group:IOT-631/A
Semester: 6th
Date of Performance: - -25
Subject Name: Java
Subject Code: 22CSH-359

1. Aim: Design and implement a simple inventory control system for a small video rental store.

2. Objective: The objective is to design and implement a **simple inventory control system** for a **small video rental store**, enabling efficient management of **movie stock, rentals, and returns**. The system should provide **real-time inventory tracking**, minimize rental conflicts, and enhance **customer service** by ensuring availability and seamless transactions.

3. Algorithm :

- **Define Classes:**
  - **Video:** To represent each video, with attributes such as video ID, title, genre, and availability status.
  - **Inventory:** To manage the list of videos, including adding and removing videos from the inventory.
  - **Customer**: To represent customers, with attributes such as customer ID, name, and rented videos.
  - **RentalSystem:** To control the process of renting and returning videos.
- **Video Class:**
  - Define the video with attributes such as `videoID`, `title`, `genre`, and `isAvailable`.
  - Define methods to mark the video as rented and returned.
- **Inventory Class:**
  - Maintain a list of videos (ArrayList).
  - Implement methods to add new videos, display available videos, and check if a video is available.
- **Customer Class:**
  - Define a list to store rented videos.
  - Implement methods to rent a video (if available) and return it.

- **RentalSystem Class:**
    - **Handle the main functionality:** list available videos, allow customers to rent and return videos, and display the inventory status

4. Code/Implementation :

```
import java.util.ArrayList;
import java.util.Scanner;

// Class representing a Video
class Video {
    private String title;
    private boolean isAvailable;

    public Video(String title) {
        this.title = title;
        this.isAvailable = true;
    }

    public String getTitle() {
        return title;
    }

    public boolean isAvailable() {
        return isAvailable;
    }

    public void rent() {
        if (isAvailable) {
            isAvailable = false;
        } else {
            System.out.println("Error: Video is already rented out.");
        }
    }

    public void returnVideo() {
        if (!isAvailable) {
            isAvailable = true;
        } else {
            System.out.println("Error: Video was not rented.");
        }
    }

    @Override
    public String toString() {
        return "Title: " + title + " | Available: " + (isAvailable ? "Yes" : "No");
    }
}
// Class representing the Video Store
class VideoStore {
    private ArrayList<Video> inventory;
```

```
public VideoStore() {
    inventory = new ArrayList<>();
}


// Add a new video to the inventory
public void addVideo(String title) {
    for (Video video : inventory) {
        if (video.getTitle().equalsIgnoreCase(title)) {
            System.out.println("Error: Video already exists in the inventory.");
            return;
        }
    }
    inventory.add(new Video(title));
    System.out.println("Video added successfully: " + title);
}

// List all videos in the inventory
public void listInventory() {
    if (inventory.isEmpty()) {
        System.out.println("No videos in inventory.");
    } else {
        System.out.println("Inventory:");
        for (int i = 0; i < inventory.size(); i++) {
            System.out.println((i + 1) + ". " + inventory.get(i));
        }
    }
}

// Rent a video
public void rentVideo(String title) {
    for (Video video : inventory) {
        if (video.getTitle().equalsIgnoreCase(title)) {
            if (video.isAvailable()) {
                video.rent();
                System.out.println("You rented: " + title);
            } else {
                System.out.println("Video is currently unavailable.");
            }
            return;
        }
    }
    System.out.println("Error: Video not found in inventory.");
}

// Return a video
public void returnVideo(String title) {
    for (Video video : inventory) {
        if (video.getTitle().equalsIgnoreCase(title)) {
            if (!video.isAvailable()) {
                video.returnVideo();
```

```
                System.out.println("You returned: " + title);
            } else {
                System.out.println("Error: Video was not rented.");
            }
            return;
        }
    }
    System.out.println("Error: Video not found in inventory.");
  }
}

// Main class to run the Video Rental System
public class VideoRentalSystem {
  public static void main(String[] args) {
    VideoStore store = new VideoStore();
    Scanner scanner = new Scanner(System.in);

    while (true) {
      System.out.println("\n---  Video Rental Store ---");
      System.out.println("1. Add Video");
      System.out.println("2. List Inventory");
      System.out.println("3. Rent Video");
      System.out.println("4. Return Video");
      System.out.println("5. Exit");
      System.out.print("Enter your choice: ");

      int choice = -1;
      if (scanner.hasNextInt()) {
        choice = scanner.nextInt();
      } else {
        System.out.println("Invalid choice. Please enter a number.");
        scanner.next(); // Consume invalid input
        continue;
      }
      scanner.nextLine();
      switch (choice) {
        case 1:
          System.out.print("Enter video title to add: ");
          String titleToAdd = scanner.nextLine().trim();
          store.addVideo(titleToAdd);
          break;
        case 2:
          store.listInventory();
          break;
        case 3:
          System.out.print("Enter video title to rent: ");
          String titleToRent = scanner.nextLine().trim();
          store.rentVideo(titleToRent);
          break;
        case 4:
          System.out.print("Enter video title to return: ");
          String titleToReturn = scanner.nextLine().trim();
```

```
                store.returnVideo(titleToReturn);
                break;
            case 5:
                System.out.println("Exiting the system. Goodbye!");
                scanner.close();
                return;
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
}
```

5.Output:

```
--- Video Rental Store ---
1. Add Video
2. List Inventory
3. Rent Video
4. Return Video
5. Exit
Enter your choice: 1
Enter video title to add: Lokesh Yadav 16604
Video added successfully: Lokesh Yadav 16604

--- Video Rental Store ---
1. Add Video
2. List Inventory
3. Rent Video
4. Return Video
5. Exit
Enter your choice: 2
Inventory:
1. Title: Lokesh Yadav 16604 | Available: Yes
```

6. Learning Outcomes:

- **Object-Oriented Programming (OOP) Concepts** – Gained hands-on experience in **class design**, **encapsulation**, and **object interactions** using Java.
- **Data Structures & Collections** – Learned to manage a dynamic inventory using **ArrayList**, ensuring efficient storage and retrieval of video records.
- **User Interaction & Input Handling** – Implemented **interactive console-based input handling** using `Scanner`, with validation for **error-free user input**.
- **Business Logic Implementation** – Developed a real-world **rental system** with **inventory management, renting, and returning functionalities**, reinforcing problem-solving skills.
- **Code Structuring & Modularity** – Designed a **well-structured program** by separating concerns into **Video, VideoStore, and VideoRentalSystem classes**, ensuring **maintainability and scalability**.