EXPERIMENT 2

Student Name: Dhruv Sorout **UID: 22BET10062**

Branch: BE – IT **Section/Group: BET_KRG_IOT-3B**

Semester: 6th **Date: 18/01/2025**

Subject Name: PBLJ With Lab **Subject Code: 22ITH-359**

1. Aim:

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

2. Objective:

To design and implement a user-friendly inventory control system for a small video rental store, enabling efficient management of video inventory, including functionalities for adding, renting, and returning videos.

3. Algorithm:

Step 1: Define Classes

I. Item Class:
- Attributes: title (String), available (boolean).
- Methods:
  - rent(): Mark item as rented if available.
  - returnItem(): Mark item as available if rented.
  - toString(): Return item details.

II. Video Class:
- Extends Item.
- Adds genre (String).
- Overrides toString() to include genre.

III. Inventory Class:
- Manages a list of Item objects.
- Methods:
  - addItem(Item): Add a unique item to inventory.
  - displayItems(): Show all inventory items.
  - rentItem(String): Rent an item by title.
  - returnItem(String): Return an item by title.

Step 2: Implement Main Program:

I. Create an Inventory object.

II. Display a menu with options:
  - Add a Video: Input title and genre, then add to inventory.

- Display Inventory: List all items.
- Rent a Video: Input title, check availability, and mark as rented.
- Return a Video: Input title and mark as returned if rented.
- Exit: Close the program.

III. Loop until the user selects "Exit."

4. Code:

```
import java.util.ArrayList;
import java.util.Scanner;
public class VideoRentalApplication {
    public static void main(String[] args) {
        Inventory inventory = new Inventory();
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("\n--- Video Rental System ---");
            System.out.println("1. Add a Video");
            System.out.println("2. Display Inventory");
            System.out.println("3. Rent a Video");
            System.out.println("4. Return a Video");
            System.out.println("5. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();
            scanner.nextLine();
            switch (choice) {
                case 1:
                    System.out.print("Enter video title to add: ");
                    String title = scanner.nextLine().trim();
                    System.out.print("Enter video genre: ");
                    String genre = scanner.nextLine().trim();
                    inventory.addItem(new Video(title, genre));
```

```
                    break;
                case 2:
                    inventory.displayItems();
                    break;
                case 3:
                    System.out.print("Enter video title to rent: ");
                    String rentTitle = scanner.nextLine().trim();
                    inventory.rentItem(rentTitle);
                    break;
                case 4:
                    System.out.print("Enter video title to return: ");
                    String returnTitle = scanner.nextLine().trim();
                    inventory.returnItem(returnTitle);
                    break;
                case 5:
                    System.out.println("Exiting the system. Goodbye!");
                    scanner.close();
                    return;
                default:
                    System.out.println("Invalid option. Please try again.");
            }
        }
    }
}
// Base class representing an Item
class Item {
    private String title;
    private boolean available;
```

```
public Item(String title) {
    this.title = title;
    this.available = true;
}
public String getTitle() {
    return title;
}
public boolean isAvailable() {
    return available;
}
public void rent() {
    if (available) {
        available = false;
    } else {
        System.out.println("Error: This item is already rented out.");
    }
}
public void returnItem() {
    if (!available) {
        available = true;
    } else {
        System.out.println("Error: This item was not rented.");
    }
}
@Override
public String toString() {
    return "Title: " + title + " | Available: " + (available ? "Yes" : "No");
```

```
    }
}
// Derived class specifically for Videos
class Video extends Item {
    private String genre;
    public Video(String title, String genre) {
        super(title);
        this.genre = genre;
    }
    public String getGenre() {
        return genre;
    }
    @Override
    public String toString() {
        return super.toString() + " | Genre: " + genre;
    }
}
// Class to manage Inventory using encapsulation
class Inventory {
    private ArrayList<Item> items;
    public Inventory() {
        items = new ArrayList<>();
    }
    public void addItem(Item item) {
        for (Item i : items) {
            if (i.getTitle().equalsIgnoreCase(item.getTitle())) {
                System.out.println("Error: Item already exists in inventory.");
                return;
```

```
            }
        }
        items.add(item);
        System.out.println("Item added: " + item.getTitle());
    }
    public void displayItems() {
        if (items.isEmpty()) {
            System.out.println("No items in inventory.");
        } else {
            System.out.println("Available Inventory:");
            for (int i = 0; i < items.size(); i++) {
                System.out.println((i + 1) + ". " + items.get(i));
            }
        }
    }
    public boolean rentItem(String title) {
        for (Item item : items) {
            if (item.getTitle().equalsIgnoreCase(title)) {
                if (item.isAvailable()) {
                    item.rent();
                    System.out.println("You rented: " + title);
                    return true;
                } else {
                    System.out.println("This item is currently unavailable.");
                    return false;
                }
            }
        }
```

```
        System.out.println("Error: Item not found.");
        return false;
    }
    public boolean returnItem(String title) {
        for (Item item : items) {
            if (item.getTitle().equalsIgnoreCase(title)) {
                if (!item.isAvailable()) {
                    item.returnItem();
                    System.out.println("You returned: " + title);
                    return true;
                } else {
                    System.out.println("Error: This item was not rented.");
                    return false;
                }
            }
        }
        System.out.println("Error: Item not found.");
        return false;
    }
}
```

5. Output:

```
--- Video Rental System ---
1. Add a Video
2. Display Inventory
3. Rent a Video
4. Return a Video
5. Exit
Choose an option: 2
No items in inventory.

--- Video Rental System ---
1. Add a Video
2. Display Inventory
3. Rent a Video
4. Return a Video
5. Exit
Choose an option: 1
Enter video title to add: OOPS using Java.
Enter video genre: Study, Technology
Item added: OOPS using Java.

--- Video Rental System ---
1. Add a Video
2. Display Inventory
3. Rent a Video
4. Return a Video
5. Exit
Choose an option: 3
Enter video title to rent: OOPS using JAVA.
You rented: OOPS using JAVA.

--- Video Rental System ---
1. Add a Video
2. Display Inventory
3. Rent a Video
4. Return a Video
5. Exit
Choose an option: 5
Exiting the system. Goodbye!
```

6. Learning Outcomes:

- Design and build a simple inventory management application.
- Understand object-oriented design with classes and objects.
- Implement encapsulation for data security and access.
- Gain experience with Java's ArrayList for inventory management.
- Practice conditional logic and error handling.
- Learn class interaction and object management.