```
import java.util.*;

class TicketBookingSystem {
    private final Set<Integer> bookedSeats = new HashSet<>();

    public synchronized boolean bookSeat(int seatNumber, String userType) {
        if (bookedSeats.contains(seatNumber)) {
            System.out.println("Error: Seat " + seatNumber + " already booked.");
            return false;
        }
        bookedSeats.add(seatNumber);
        System.out.println(userType + " Booking: Seat " + seatNumber + " confirmed.");
        return true;
    }
}

class BookingThread extends Thread {
    private final TicketBookingSystem system;
    private final int seatNumber;
    private final String userType;

    public BookingThread(TicketBookingSystem system, int seatNumber, String userType, int priority) {
        this.system = system;
        this.seatNumber = seatNumber;
        this.userType = userType;
        setPriority(priority);
    }

    @Override
    public void run() {
        system.bookSeat(seatNumber, userType);
    }
}

public class TicketBooking {
    public static void main(String[] args) {
        TicketBookingSystem system = new TicketBookingSystem();

        Thread vip1 = new BookingThread(system, 1, "VIP", Thread.MAX_PRIORITY);
        Thread vip2 = new BookingThread(system, 2, "VIP", Thread.MAX_PRIORITY);
        Thread regular1 = new BookingThread(system, 1, "Regular", Thread.NORM_PRIORITY);
        Thread regular2 = new BookingThread(system, 3, "Regular", Thread.NORM_PRIORITY);

        vip1.start();
        vip2.start();
        regular1.start();
        regular2.start();
    }
}
```

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

class Employee {
    int id;
    String name;
    double salary;

    Employee(int id, String name, double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }

    @Override
    public String toString() {
        return "ID: " + id + ", Name: " + name + ", Salary: " + salary;
    }
}

public class EmployeeManagement {
    static ArrayList<Employee> employees = new ArrayList<>();
    static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        while (true) {
            System.out.println("\n1. Add Employee");
            System.out.println("2. Update Employee");
            System.out.println("3. Remove Employee");
            System.out.println("4. Search Employee");
            System.out.println("5. Display All Employees");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    addEmployee();
                    break;
                case 2:
                    updateEmployee();
                    break;
                case 3:
                    removeEmployee();
                    break;
                case 4:
                    searchEmployee();
                    break;
                case 5:
                    displayEmployees();
                    break;
                case 6:
                    System.out.println("Exiting...");
                    return;
                default:
                    System.out.println("Invalid choice! Try again.");
            }
        }
    }
```

```
static void addEmployee() {
    System.out.print("Enter Employee ID: ");
    int id = scanner.nextInt();
    scanner.nextLine(); // Consume newline
    System.out.print("Enter Name: ");
    String name = scanner.nextLine();
    System.out.print("Enter Salary: ");
    double salary = scanner.nextDouble();

    employees.add(new Employee(id, name, salary));
    System.out.println("Employee added successfully!");
}

static void updateEmployee() {
    System.out.print("Enter Employee ID to update: ");
    int id = scanner.nextInt();
    for (Employee emp : employees) {
        if (emp.id == id) {
            scanner.nextLine(); // Consume newline
            System.out.print("Enter New Name: ");
            emp.name = scanner.nextLine();
            System.out.print("Enter New Salary: ");
            emp.salary = scanner.nextDouble();
            System.out.println("Employee updated successfully!");
            return;
        }
    }
    System.out.println("Employee not found!");
}

static void removeEmployee() {
    System.out.print("Enter Employee ID to remove: ");
    int id = scanner.nextInt();
    for (Employee emp : employees) {
        if (emp.id == id) {
            employees.remove(emp);
            System.out.println("Employee removed successfully!");
            return;
        }
    }
    System.out.println("Employee not found!");
}

static void searchEmployee() {
    System.out.print("Enter Employee ID to search: ");
    int id = scanner.nextInt();
    for (Employee emp : employees) {
        if (emp.id == id) {
            System.out.println("Employee Found: " + emp);
            return;
        }
    }
    System.out.println("Employee not found!");
}

static void displayEmployees() {
    if (employees.isEmpty()) {
        System.out.println("No employees found!");
        return;
    }
```

```
        System.out.println("Employee List:");
        for (Employee emp : employees) {
            System.out.println(emp);
        }
    }
}
```

```
import java.util.*;

class Card {
    String symbol;
    String rank;

    Card(String symbol, String rank) {
        this.symbol = symbol;
        this.rank = rank;
    }

    @Override
    public String toString() {
        return "Symbol: " + symbol + ", Rank: " + rank;
    }
}

public class CardCollection {
    static List<Card> cards = new ArrayList<>();
    static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        while (true) {
            System.out.println("\n1. Add Card");
            System.out.println("2. Search by Symbol");
            System.out.println("3. Display All Cards");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();
            scanner.nextLine();

            switch (choice) {
                case 1:
                    addCard();
                    break;
                case 2:
                    searchBySymbol();
                    break;
                case 3:
                    displayCards();
                    break;
                case 4:
                    System.out.println("Exiting...");
                    return;
                default:
                    System.out.println("Invalid choice! Try again.");
            }
        }
    }

    static void addCard() {
        System.out.print("Enter Symbol (Hearts, Diamonds, etc.): ");
        String symbol = scanner.nextLine();
        System.out.print("Enter Rank (Ace, 2, King, etc.): ");
        String rank = scanner.nextLine();

        cards.add(new Card(symbol, rank));
        System.out.println("Card added successfully!");
    }
```

```
    static void searchBySymbol() {
        System.out.print("Enter Symbol to search: ");
        String symbol = scanner.nextLine();
        boolean found = false;

        for (Card card : cards) {
            if (card.symbol.equalsIgnoreCase(symbol)) {
                System.out.println(card);
                found = true;
            }
        }

        if (!found) {
            System.out.println("No cards found with symbol: " + symbol);
        }
    }

    static void displayCards() {
        if (cards.isEmpty()) {
            System.out.println("No cards in the collection!");
            return;
        }
        System.out.println("Card Collection:");
        for (Card card : cards) {
            System.out.println(card);
        }
    }
}
```