Object Oriented Programming

Lab Project

Jan 12, 2023

Submitted by:

Muhammad Hozefa Rauf

FA21-BCS-057

Submitted to:

Ms Saneeha Amir

Car Rental Management System

Introduction:

A car rental system is a software application that allows users to rent vehicles from a car rental company. The system can be implemented using an object-oriented programming language such as Java. The system will typically have classes for representing the different types of vehicles, classes for representing customers and rental transactions, and classes for performing various tasks such as calculating rental fees, checking vehicle availability, and processing payments. Additionally, the system may include interfaces for interacting with users, such as a web-based interface or a command-line interface, and may also interact with other systems such as a reservation system or a fleet management system.

Modules:

1. **Admin:**
   Admin is a person that can access all the information , add a new member, manipulate their information and search by different aspects.
2. **Costumer:**
   Costumer will be one who will sign in and can rent a car from this system

3. **Vehicle Management:**
   This module will allow the system to keep track of the different types of vehicles available for rent, as well as their current status (e.g. available, rented, under maintenance).
4. **Customer Management:**
   This module will allow the system to store information about customers, such as their contact details and rental history.
5. **Reservation and Booking:**
   This module will allow customers to search for and reserve vehicles, and will also keep track of the current reservations and booking.
6. **Rental and Billing:**
   This module will handle the rental transactions, including calculating rental fees, processing payments and generating invoices.
7. **User Management:** This module will manage the user authentication, authorizations and user management tasks.

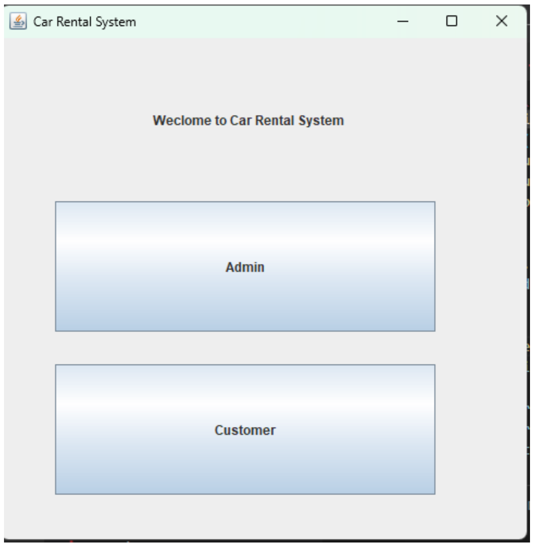
Object Oriented Approach:

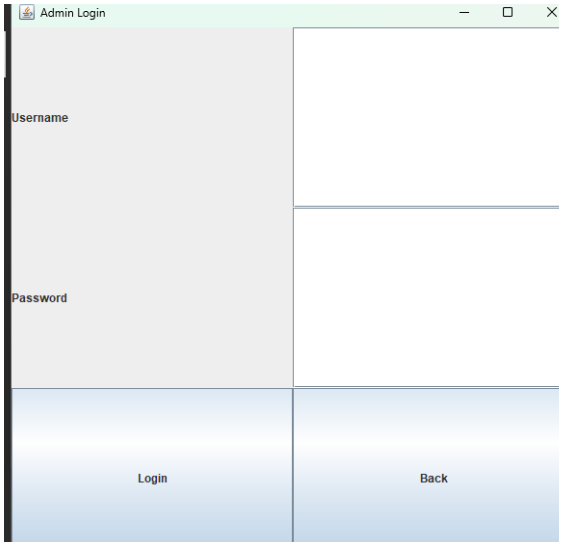
It is necessary to understand some of the concepts used extensively in object-oriented programming. These include:

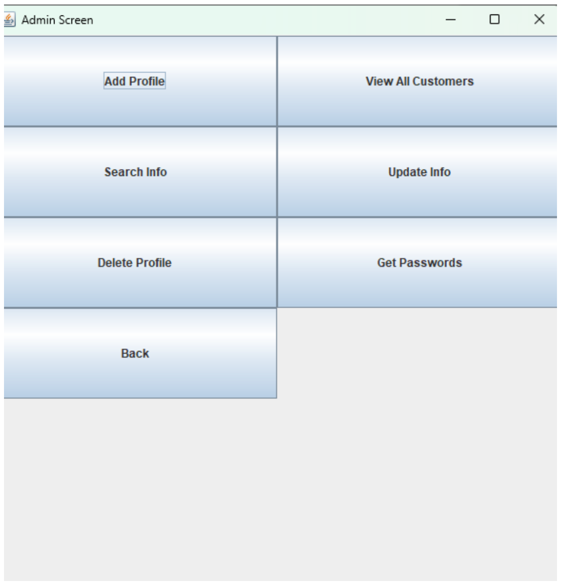
- Objects
- Classes
- Data abstraction and encapsulation

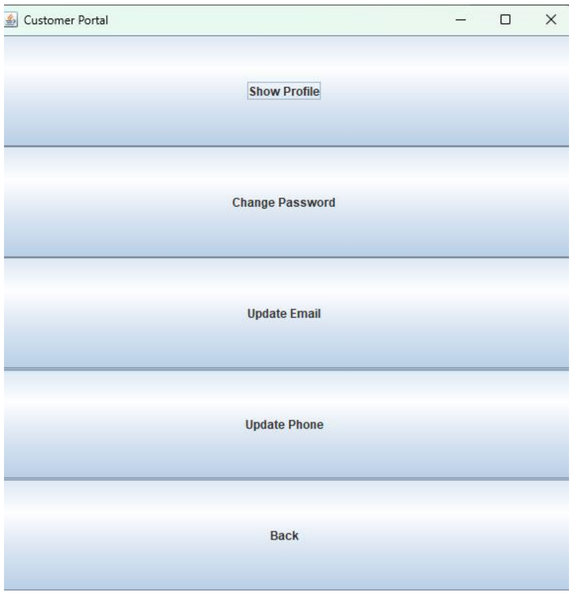
- Polymorphism
- Inheritance

GUI:









Add Customer:

```java
public void addCustomer(Customer s) {
       ObjectOutputStream oos = null;
       // write to file
       try {
           if (!f.exists()) {
               f = new File("Customers.ser");
           }
           if (f.exists()) {

               oos = new MyObjectOutputStream(new FileOutputStream(f, true));
               oos.writeObject(s); // write object to file
           } else {
               oos = new ObjectOutputStream(new FileOutputStream(f, true));
               oos.writeObject(s); // it will write the object to the file.
           }

       } catch (IOException e) {
           e.printStackTrace();
       }

       catch (Exception e) {
           System.err.println("Cannot Write Object");
       }

       // For closing File

       if (oos != null) {
           try {
               oos.close();
           } catch (IOException e) {
               e.printStackTrace();
           }
       }
   }
```

Delete:

```java
public boolean removeCustomer(String ID) {

       boolean found = false;
       ObjectInputStream oo = null;
```

```
        try {
            oo = new ObjectInputStream(new FileInputStream("Customers.ser"));

            try {
                while (true) {

                    Customer s = (Customer) oo.readObject();
                    collectionCustomers.add(s);

                }
            } catch (EOFException e) {
                // Move to the next line broda
            }
            // now we will move sequentially..

            oo.close();

            // removing the specified object from the arraylist
            for (int i = 0; i < collectionCustomers.size(); i++) {
                if (collectionCustomers.get(i).getID().equals(ID) ) {
                    found = true;
                    collectionCustomers.remove(i);
                }
            }


            // now again writing the Arraylist objects in the file first time we
will create a new file and then we will append
            // Object for writing class (ObjectOutputStream)
            ObjectOutputStream oos = null;
            // write to file
            int counter = 0;

            if (collectionCustomers.size() > 0) {
                for (int i = 0; i < collectionCustomers.size(); i++) {

                    System.out.println("Writing again to the file");

                    if (counter > 0) {
                        // when you are running it for the second and afterwards
iterations you will append the file
                        oos = new MyObjectOutputStream(new FileOutputStream(f,
true));
                        oos.writeObject(collectionCustomers.get(i));
```

```java
                    } else {
                        // for the first time you will create a new file
                        oos = new ObjectOutputStream(new FileOutputStream(f));
                        oos.writeObject(collectionCustomers.get(i));
                        counter++;
                    }

                }

                // For closing File

                if (oos != null) {
                    oos.close();
                }

            }
            else if (collectionCustomers.size() == 0) {
                // System.out.println("File deleting");
                f.delete();
                // System.out.println("File deleted");
            }
        }

        catch (Exception e) {
            e.printStackTrace();
        }


        return found;

    }
```

Update:

```java
public boolean updateName(String ID, String firstName, String lastName) {

        boolean found = false;

        ObjectInputStream oo = null;
        try {

            oo = new ObjectInputStream(new FileInputStream("Customers.ser"));
            try {
                while (true) {

                    Customer k = (Customer) oo.readObject();
                    collectionCustomers.add(k);
                }
            } catch (EOFException e) {

            }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }

        finally {
            try {
                oo.close();
            } catch (IOException e) {

            }
        }



        for (int i = 0; i < collectionCustomers.size(); i++) {
            if (collectionCustomers.get(i).getID().equalsIgnoreCase(ID)) {

                found = true;
                collectionCustomers.get(i).setFirstName(firstName);
                collectionCustomers.get(i).setLastName(lastName);
            }
        }

        //* now again writing the Arraylist Objects to the file. first time it
will create the file again and only then it will append!
        // file object
        // f = new File("Students.ser");
        ObjectOutputStream oos = null;
        int counter = 0;

        try {
            for (int i = 0; i < collectionCustomers.size(); i++) {
                if (counter > 0) {
                    oos = new MyObjectOutputStream(new FileOutputStream(f,
true));
                    oos.writeObject(collectionCustomers.get(i));

                } else {
                    oos = new ObjectOutputStream(new FileOutputStream(f));
                    oos.writeObject(collectionCustomers.get(i));
                    counter++;
                }
            }

            // For closing File

            if (oos != null) {
                oos.close();
            }
        }

        catch (IOException e) {
            e.printStackTrace();
        }

        return found;

    }
```

Search:

```
 public ArrayList<Customer> searchByCNIC(String CNIC) {
        ArrayList<Customer> a = new ArrayList<>();
        ObjectInputStream oo = null;

        if (!f.exists()) {
            return a;
        }
        try {
            oo = new ObjectInputStream(new FileInputStream("Customers.ser"));

            while (true) {

                // Reading object is below
                Customer s = (Customer) oo.readObject();
                if (s.getCNIC().equalsIgnoreCase(CNIC)) {
                    a.add(s);
                }
            }

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }

        catch (EOFException e) {

        }

        catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        catch (IOException e) {
            e.printStackTrace();
        }

        finally {
            try {
                oo.close();

            } catch (IOException e) {
            }

        }

        return a;
    }

    public ArrayList<Customer> searchByPhone(String phone) {
        ArrayList<Customer> a = new ArrayList<>();
        ObjectInputStream oo = null;


        if (!f.exists()) {
            return a;
        }
        try {
            oo = new ObjectInputStream(new FileInputStream("Customers.ser"));

            while (true) {
```

```java
                // Reading object is below
                Customer s = (Customer) oo.readObject();
                if (s.getPhoneNo().equalsIgnoreCase(phone)) {
                    a.add(s);
                }
            }

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }

        catch (EOFException e) {

        }

        catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        catch (IOException e) {
            e.printStackTrace();
        }

        finally {
            try {
                oo.close();

            } catch (IOException e) {
            }

        }


        return a;
    }
```