WALTON NJIRU FUNDI

BSCIT-01-0026/2025

**Car Rental System**

**Introduction:**

A car rental system is a software application designed to facilitate vehicle rentals from a car rental company. This system can be developed using an object-oriented programming language like Java. It typically includes classes to represent different vehicle types, customer details, and rental transactions. Additionally, it features components for key functionalities such as calculating rental fees, checking vehicle availability, and processing payments. The system may also provide user interfaces, such as a web-based or command-line interface, and may integrate with external systems like reservation or fleet management platforms.

Modules:

1. Admin:

The admin has full access to system information, enabling them to add new members, modify user details, and conduct searches based on various criteria.

2. Customer:

Customers can sign in to the system and rent a vehicle based on availability.

3. Vehicle Management:

This module tracks all available vehicles, their categories, and their status (e.g., available, rented, or under maintenance).

4. Customer Management:

This module stores and manages customer details, including contact information and rental history.

5. Reservation and Booking:

This module enables customers to search for available vehicles, make reservations, and manage existing bookings.

6. Rental and Billing:

This module handles rental transactions, rental fee calculations, payment processing, and invoice generation.

7. User Management:

This module manages user authentication, authorization, and overall user account management.

Object-Oriented Approach:

The system leverages key object-oriented programming principles, including:

Objects

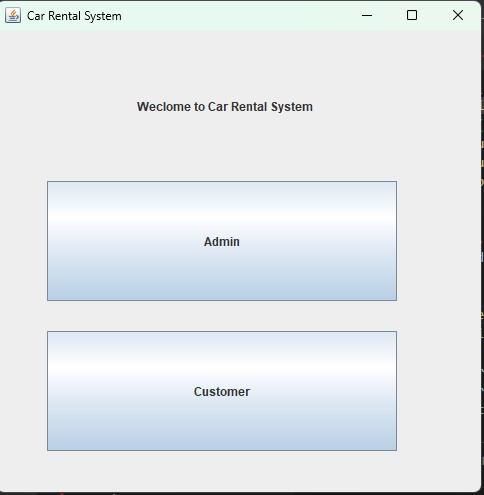
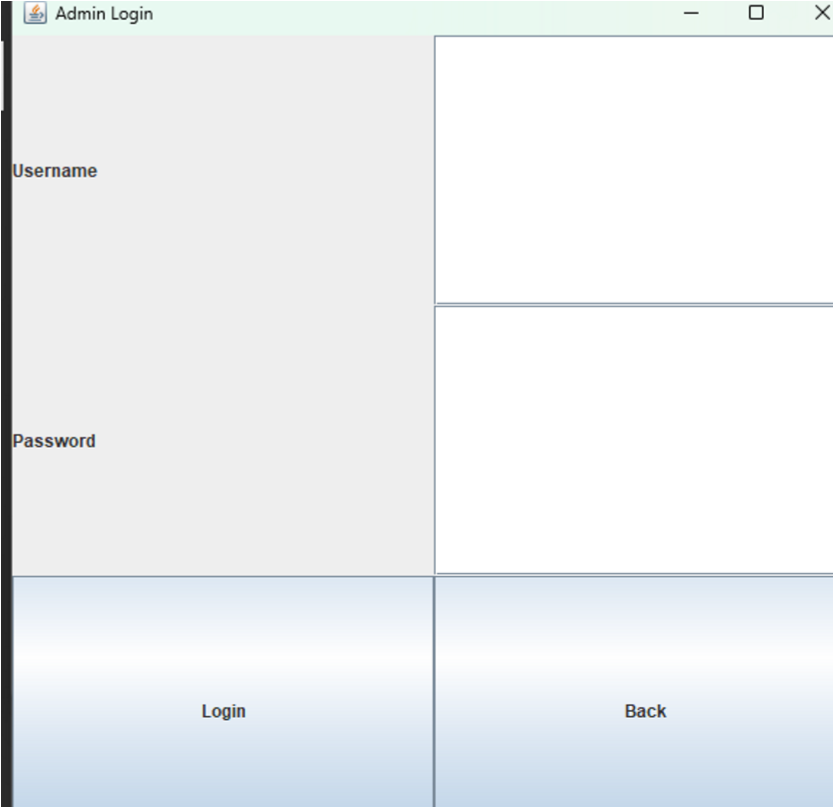
Classes

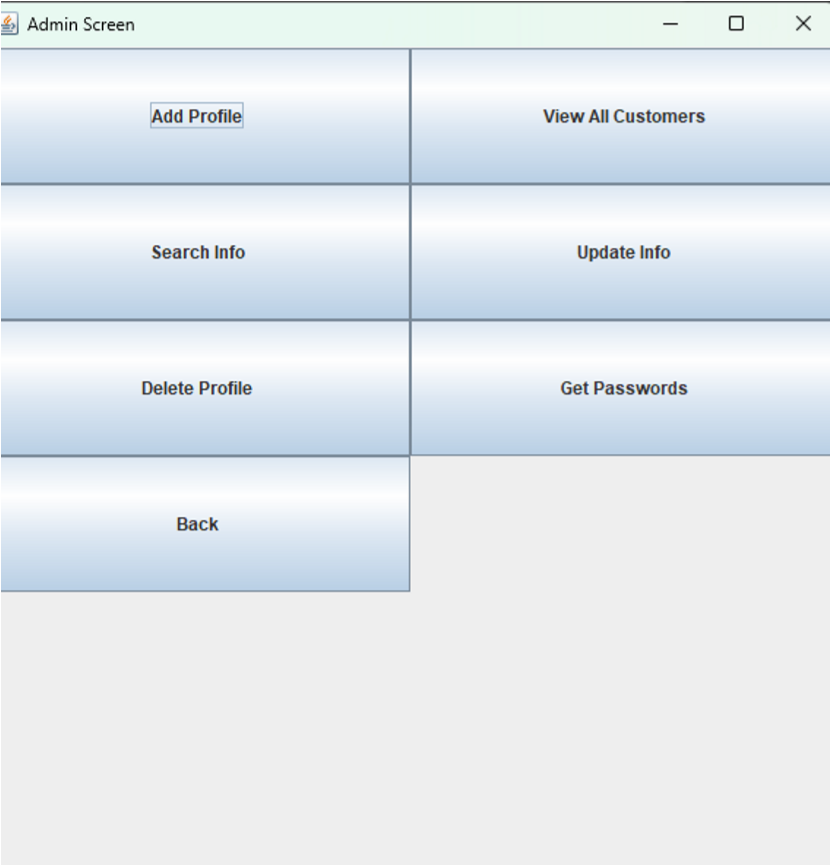
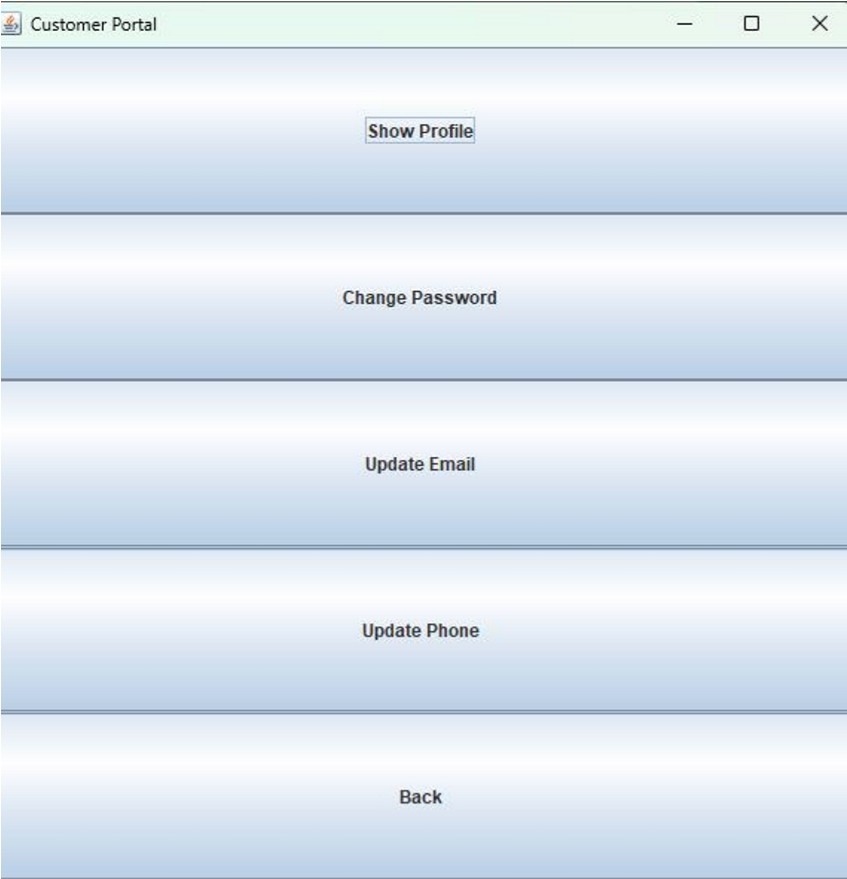
Data Abstraction & Encapsulation

Polymorphism

Inheritance

GUI:

Add Customer:

|  |
| --- |
| 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:

|  |
| --- |
| 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)); |

} 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:

|  |
| --- |
| 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) { |

// 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;  } |