MAPÚA MALAYAN COLLEGES MINDANAO

Java Application Development
OWN THE GROUND

A Modular Assessment submitted
in Partial Fulfillment of the Requirements for the Course
CPE141L Programming Logic and Design (Laboratory)

by:
Kent William P. Englisa
Kyle Miso
Le-Kim Ansao
Bachelor of Science in Computer Engineering

Course Professor
Marlon V. Maddara
College of Engineering and Architecture

May 2023

TABLE OF CONTENTS

OVERVIEW OF THE APPLICATION

Own the Ground Retail Management System is a desktop application built in Java using the NetBeans IDE and Apache Ant as the build tool. This system was designed as a student project to support the operations of *Own the Ground*, a pseudo-business specializing in the sale of classic footwear. The application provides a centralized platform for product management, customer interactions, and inventory tracking tailored to a boutique footwear retail environment.

The desktop application serves both staff and administrative users, offering an integrated system to manage core retail operations. Features include a visual product catalog, customer order processing, basic AI-driven style suggestions (using rule-based logic), and a sustainability information module that helps staff communicate product values to eco-conscious buyers. These functions are supported by a simple file-based database system, keeping the application lightweight and easily portable.

This system was developed to address three main challenges typically observed in small to mid-sized retail operations:

1. Limited product personalization tools
2. Lack of digital inventory control and customer tracking
3. Insufficient integration of sustainability metrics into the sales process

The project's direction was inspired by current trends in the retail industry, where businesses are increasingly expected to deliver personalized experiences and transparent product sourcing. Epsilon (2018) found that 80% of consumers prefer personalized interactions when shopping, and a joint report by First Insight & Baker Retailing Center (2021) emphasized that Gen Z consumers demand sustainability and ethical sourcing. Yet, many smaller retailers still rely on spreadsheets or manual processes, similar to the outdated methods observed in other industries like regional aviation.

This desktop-based solution is intentionally designed to be accessible and adaptable. While enterprise systems (e.g., Oracle Retail or Salesforce Commerce) offer deep integration, they are financially out of reach for many boutique retailers. In contrast, this application demonstrates how Java and basic development frameworks can be used to build a practical, cost-effective system that supports daily business functions without requiring a cloud-based infrastructure or ongoing subscription fees.

What makes this student project particularly valuable is its real-world relevance combined with the use of foundational programming techniques. Developed in Java using Swing for the graphical user interface, the system is highly customizable and designed for future enhancement. As skills improve, the project can evolve to include features like database integration (e.g., MySQL), mobile extensions, or cloud storage.

By solving immediate operational problems—such as inventory visibility, customer communication, and sales tracking—*Own the Ground Retail Management System* presents a scalable foundation for any niche retailer looking to embrace digital transformation in a practical and attainable way.

COMPLETE SOURCE CODE

Provide your complete source code here from your Java IDE. Utilize the following format:

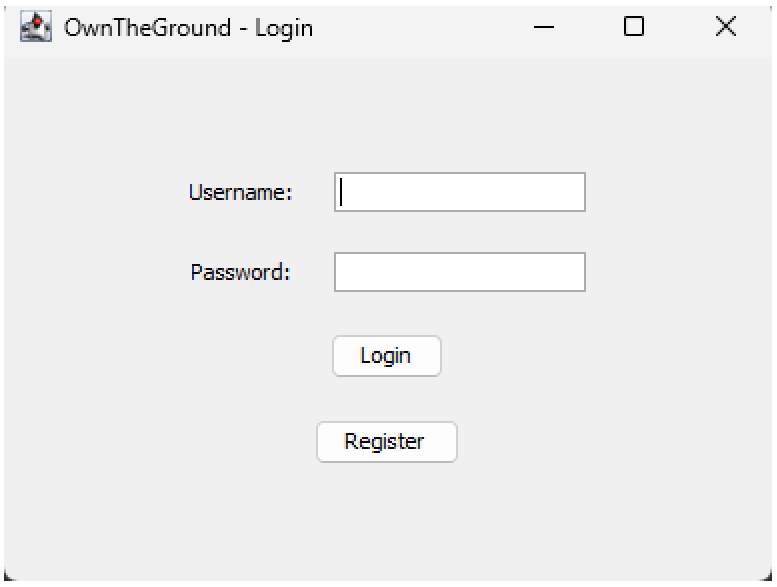
```
package main;

import main.views.LoginForm;
import javax.swing.UIManager;

public class Main {
    public static void main(String[] args) {
        try {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch(Exception e) {
            e.printStackTrace();
        }
        java.awt.EventQueue.invokeLater(() -> {
            new LoginForm().setVisible(true);
        });
    }
}

package main.services;

import main.models.*;

import java.io.*;
import java.util.*;
import javax.swing.*;

public class DatabaseService {
    private static DatabaseService instance;
    private final Map<String, User> users = new HashMap<>();
    private final List<Products> products = new ArrayList<>();
    private final List<Order> orders = new ArrayList<>();
    private final List<Transaction> transactions = new ArrayList<>();

    private final String dataDir = "data";
    private final String usersFile = dataDir + "/users.dat";
    private final String productsFile = dataDir + "/products.dat";
    private final String ordersFile = dataDir + "/orders.dat";
    private final String transactionsFile = dataDir + "/transactions.dat";

    private DatabaseService() {
        File dir = new File(dataDir);
        if(!dir.exists()) {
            dir.mkdirs();
        }
        loadData();

        if(users.isEmpty()) {
            users.put("admin",        new        User("admin",        "admin123",
"admin@example.com", "123 Admin St", true));
            saveData();
        }
        if(products.isEmpty()) {
            initProducts();
        }
    }

    public static DatabaseService getInstance() {
        if(instance == null) {
            instance = new DatabaseService();
        }
        return instance;
    }

    private void initProducts() {
        products.add(new Products(1, "Muscon", "Italian leather", 2199.00, new
String[]{"38", "39", "40", "41", "42"}));
        products.add(new  Products(2,  "Noir",  "Italian  leather",  2199.00,  new
String[]{"38", "39", "40", "41", "42"}));
```

```
        products.add(new Products(3, "Charcolux", "Italian leather", 2199.00, new
String[]{"36", "37", "38", "39", "40"}));
        products.add(new Products(4, "Gentleman", "Italian leather", 2199.00, new
String[]{"40", "41", "42", "43", "44"}));
        saveData();
    }


    public boolean addUser(User user) {
        if(users.containsKey(user.getUsername())) {
            return false;
        }
        users.put(user.getUsername(), user);
        saveData();
        return true;
    }

    public List<User> getAllUsers() {
        return new ArrayList<>(users.values());
    }

    public User getUser(String username) {
        return users.get(username);
    }

    public void updateUser(User user) {
        users.put(user.getUsername(), user);
        saveData();
    }

    public boolean deleteUser(String username) {
        if (users.containsKey(username)) {
            users.remove(username);
            saveData();
            return true;
        }
        return false;
    }

    public List<Products> getProducts() {
        return new ArrayList<>(products);
    }

    public Products getProductById(int id) {
        return products.stream()
                       .filter(p -> p.getId() == id)
                       .findFirst()
                       .orElse(null);
    }

    public boolean addProduct(Products product) {
        if (getProductById(product.getId()) != null) {
            return false;
        }
        products.add(product);
        saveData();
        return true;
    }

    public boolean updateProduct(Products updatedProduct) {
        for (int i = 0; i < products.size(); i++) {
            if (products.get(i).getId() == updatedProduct.getId()) {
                products.set(i, updatedProduct);
                saveData();
                return true;
            }
        }
        return false;
    }

    public boolean deleteProduct(int productId) {
        boolean removed = products.removeIf(p -> p.getId() == productId);
        if (removed) {
            saveData();
        }
        return removed;
    }
    public void addOrder(Order order) {
        orders.add(order);
        saveData();
    }

    public List<Order> getOrders(String username) {
        List<Order> userOrders = new ArrayList<>();
        for(Order o : orders) {
            if(o.getUsername().equals(username)) {
                userOrders.add(o);
            }
        }
        return userOrders;
    }

    public List<Order> getAllOrders() {
        return new ArrayList<>(orders);
    }

    public Order getOrderById(String orderId) {
        return orders.stream()
                     .filter(o -> o.getOrderId().equals(orderId))
                     .findFirst()
                     .orElse(null);
    }

    public void addTransaction(Transaction transaction) {
        transactions.add(transaction);
        saveData();
    }

    public List<Transaction> getUserTransactions(String username) {
        List<Transaction> userTransactions = new ArrayList<>();
        for(Transaction t : transactions) {
            if(t.getUsername().equals(username)) {
                userTransactions.add(t);
            }
        }
        return userTransactions;
    }

    private void loadData() {
        try {
            File dir = new File(dataDir);
            if(!dir.exists()) dir.mkdirs();

            if(new File(usersFile).exists()) {
                try   (ObjectInputStream   ois   =   new   ObjectInputStream(new
FileInputStream(usersFile))) {
                    Map<String,   User>   loadedUsers   =   (Map<String,   User>)
ois.readObject();
                    users.putAll(loadedUsers);
                }
            }

            if(new File(productsFile).exists()) {
                try   (ObjectInputStream   ois   =   new   ObjectInputStream(new
FileInputStream(productsFile))) {
                    List<Products>     loadedProducts     =     (List<Products>)
ois.readObject();
                    products.addAll(loadedProducts);
                }
            }

            if(new File(ordersFile).exists()) {
                try   (ObjectInputStream   ois   =   new   ObjectInputStream(new
FileInputStream(ordersFile))) {
                    List<Order> loadedOrders = (List<Order>) ois.readObject();
                    orders.addAll(loadedOrders);
                }
            }

            if(new File(transactionsFile).exists()) {
                try   (ObjectInputStream   ois   =   new   ObjectInputStream(new
FileInputStream(transactionsFile))) {
```

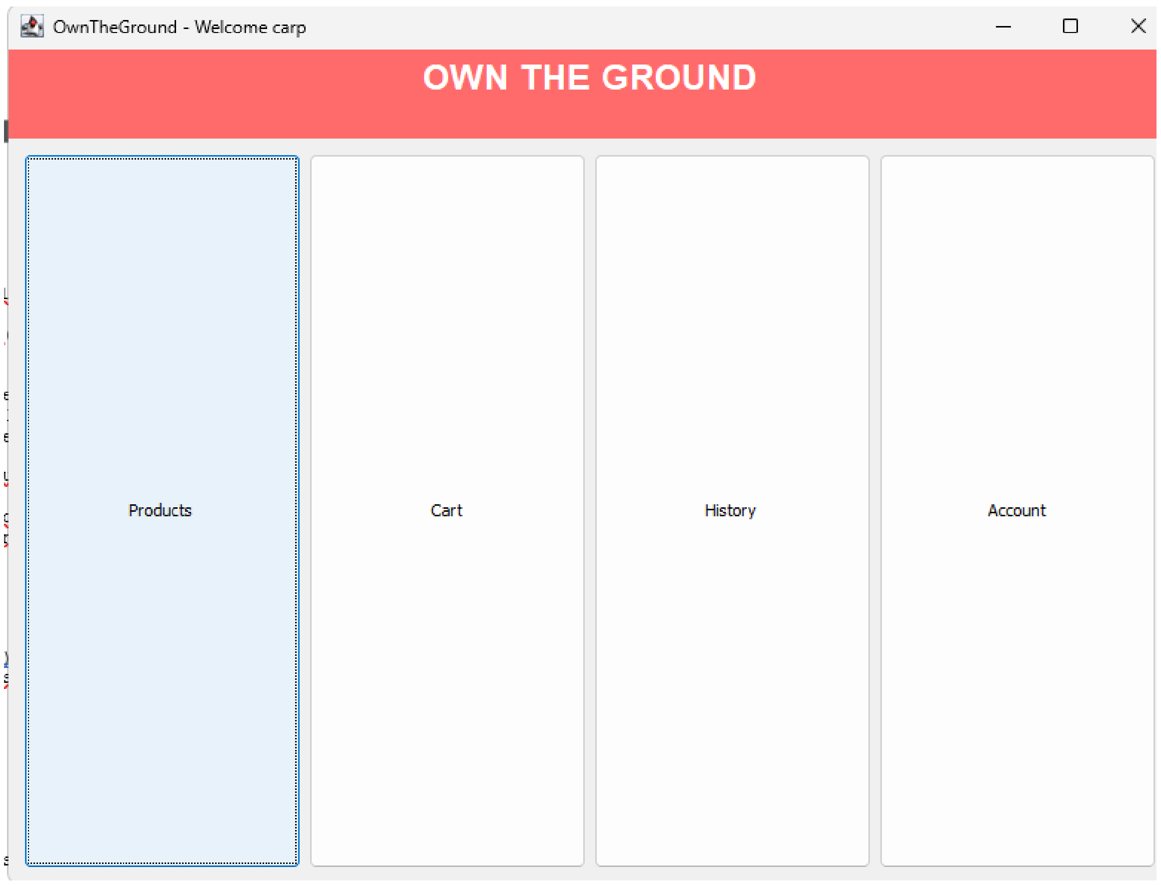
```
                    List<Transaction>  loadedTransactions  =  (List<Transaction>)
ois.readObject();
                    transactions.addAll(loadedTransactions);
                }
            }
        } catch (FileNotFoundException e) {
            System.out.println("Data files not found, starting with empty data
(this is normal for first run or if files were deleted).");
        }
        catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,  "Error  loading  data:  "  +
e.getMessage(), "Load Error", JOptionPane.ERROR_MESSAGE);
        }
    }

    private void saveData() {
        try {
            File dir = new File(dataDir);
            if(!dir.exists()) dir.mkdirs();

            try   (ObjectOutputStream   oos   =   new   ObjectOutputStream(new
FileOutputStream(usersFile))) {
                oos.writeObject(users);
            }

            try   (ObjectOutputStream   oos   =   new   ObjectOutputStream(new
FileOutputStream(productsFile))) {
                oos.writeObject(products);
            }

            try   (ObjectOutputStream   oos   =   new   ObjectOutputStream(new
FileOutputStream(ordersFile))) {
                oos.writeObject(orders);
            }

            try   (ObjectOutputStream   oos   =   new   ObjectOutputStream(new
FileOutputStream(transactionsFile))) {
                oos.writeObject(transactions);
            }
        } catch (IOException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,  "Error  saving  data:  "  +
e.getMessage(), "Save Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}
```

Highlight the key elements of your program, especially the two customized features that were designed. Use as many pages as possible.

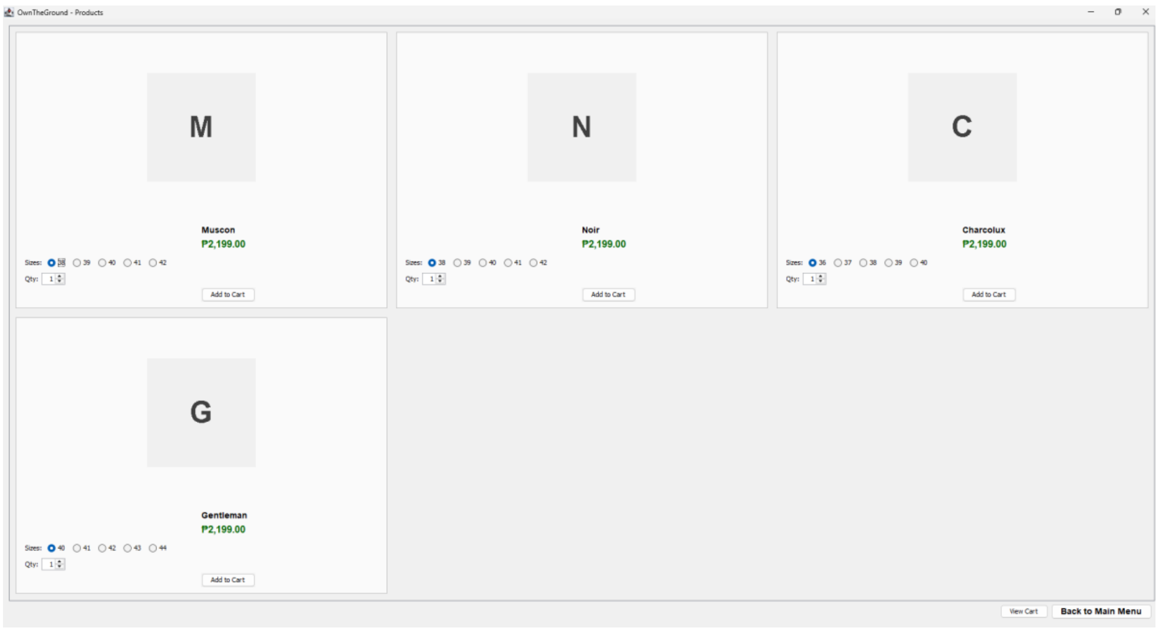
PROGRAM OUTPUT



Login interface



Main Interface


OwnTheGround - Products

M
Muscon
₱2,199.00
Sizes: [✓ 38] 39 40 41 42
Qty: 1
Add to Cart

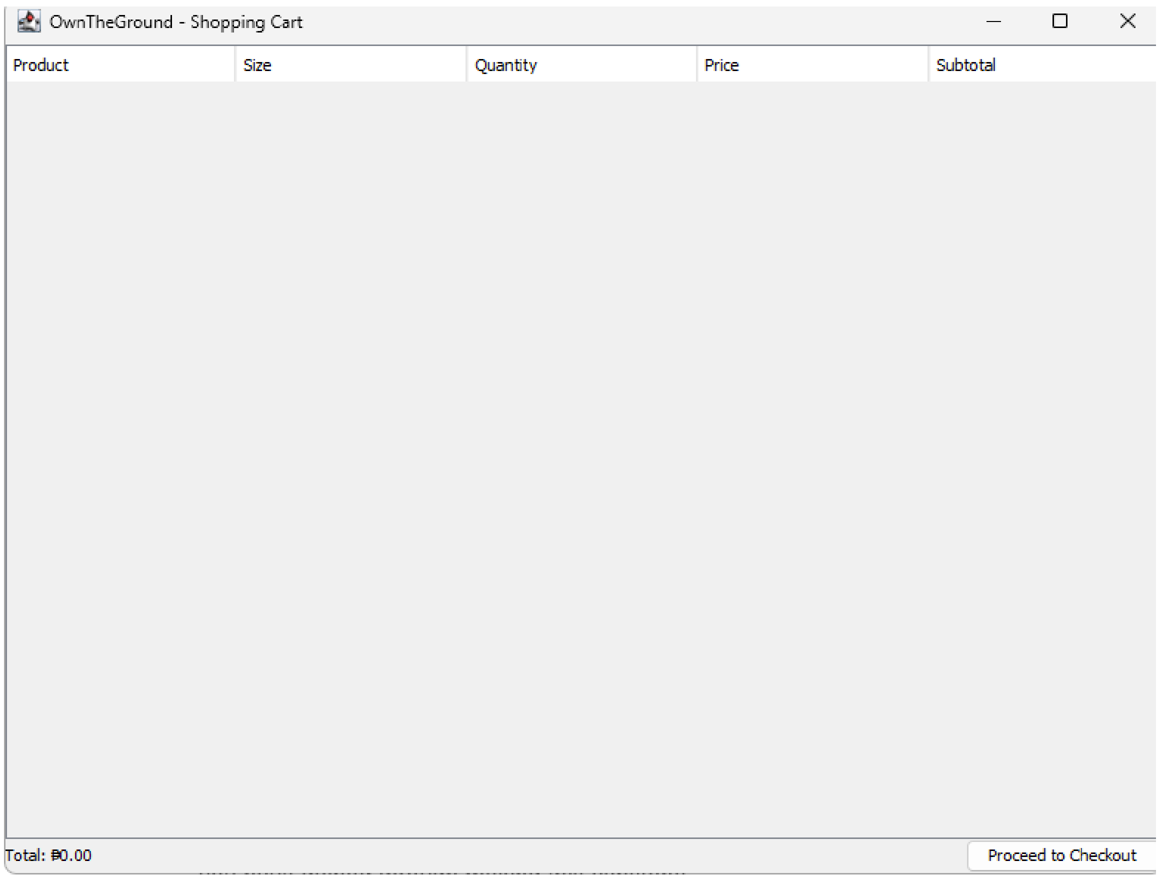
N
Noir
₱2,199.00
Sizes: [✓ 38] 39 40 41 42
Qty: 1
Add to Cart

C
Charcolux
₱2,199.00
Sizes: [✓ 36] 37 38 39 40
Qty: 1
Add to Cart

G
Gentleman
₱2,199.00
Sizes: [✓ 40] 41 42 43 44
Qty: 1
Add to Cart

View Cart
Back to Main Menu

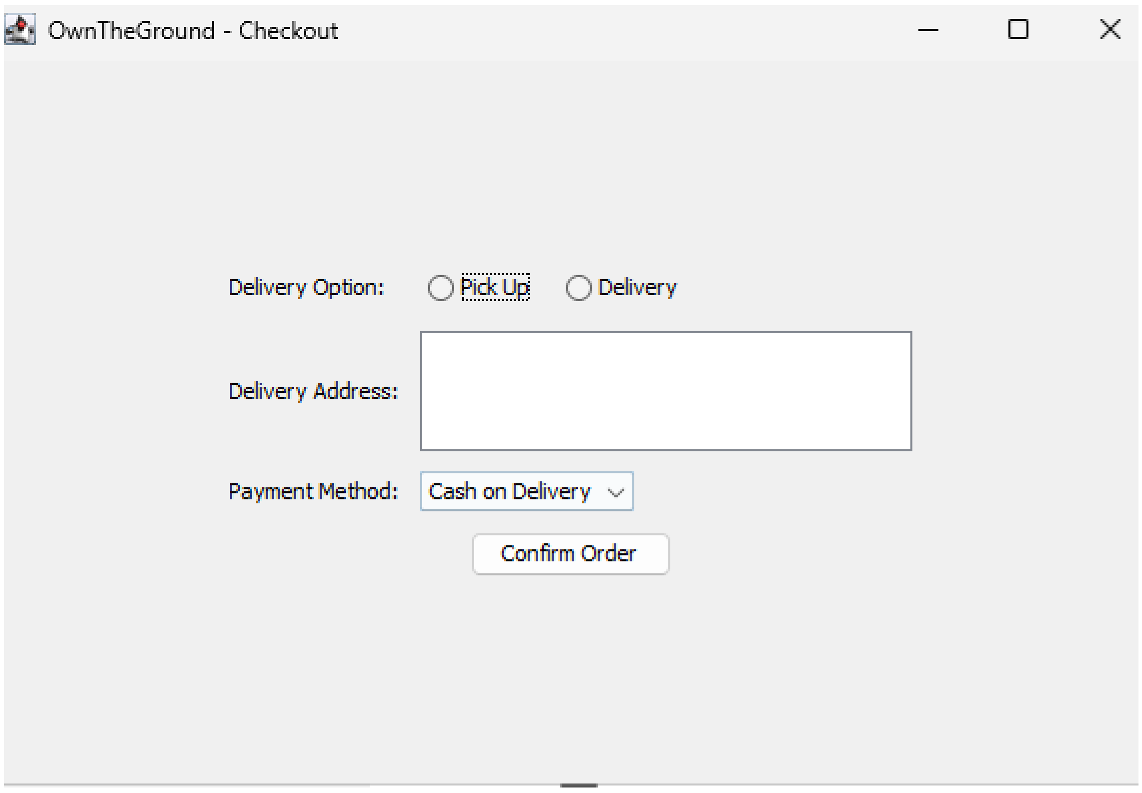
Products Interface


OwnTheGround - Shopping Cart

Product	Size	Quantity	Price	Subtotal

Total: ₱0.00
Proceed to Checkout

Cart Interface

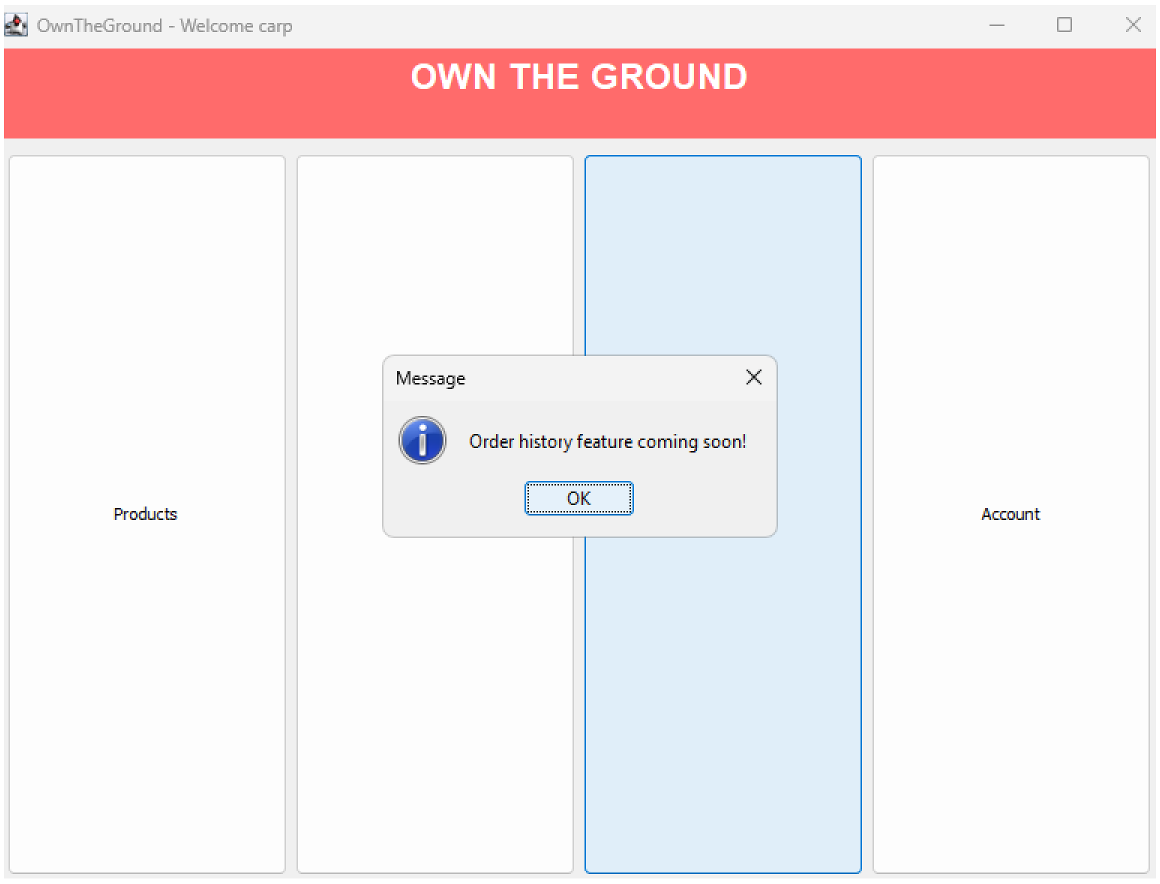


Delivery Option: Pick Up Delivery

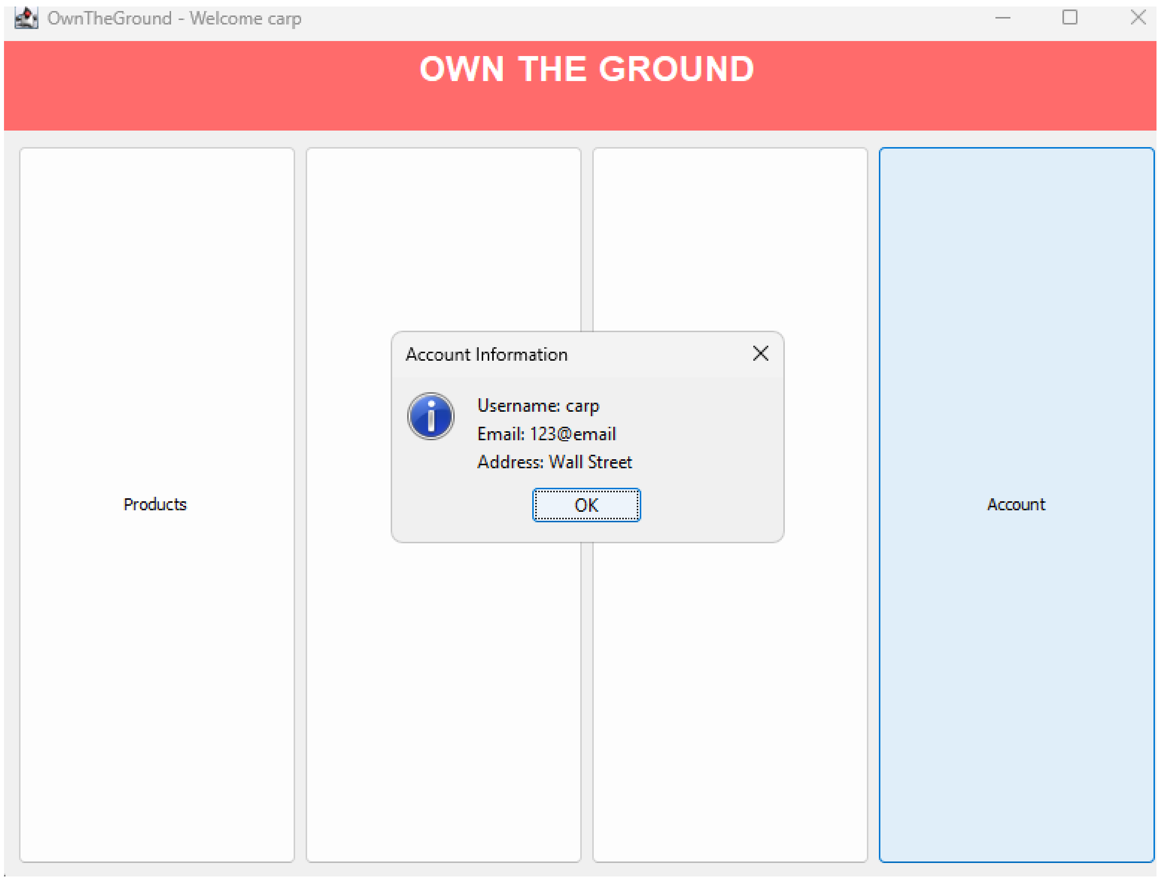
Delivery Address:

Payment Method: Cash on Delivery

Our Checkout

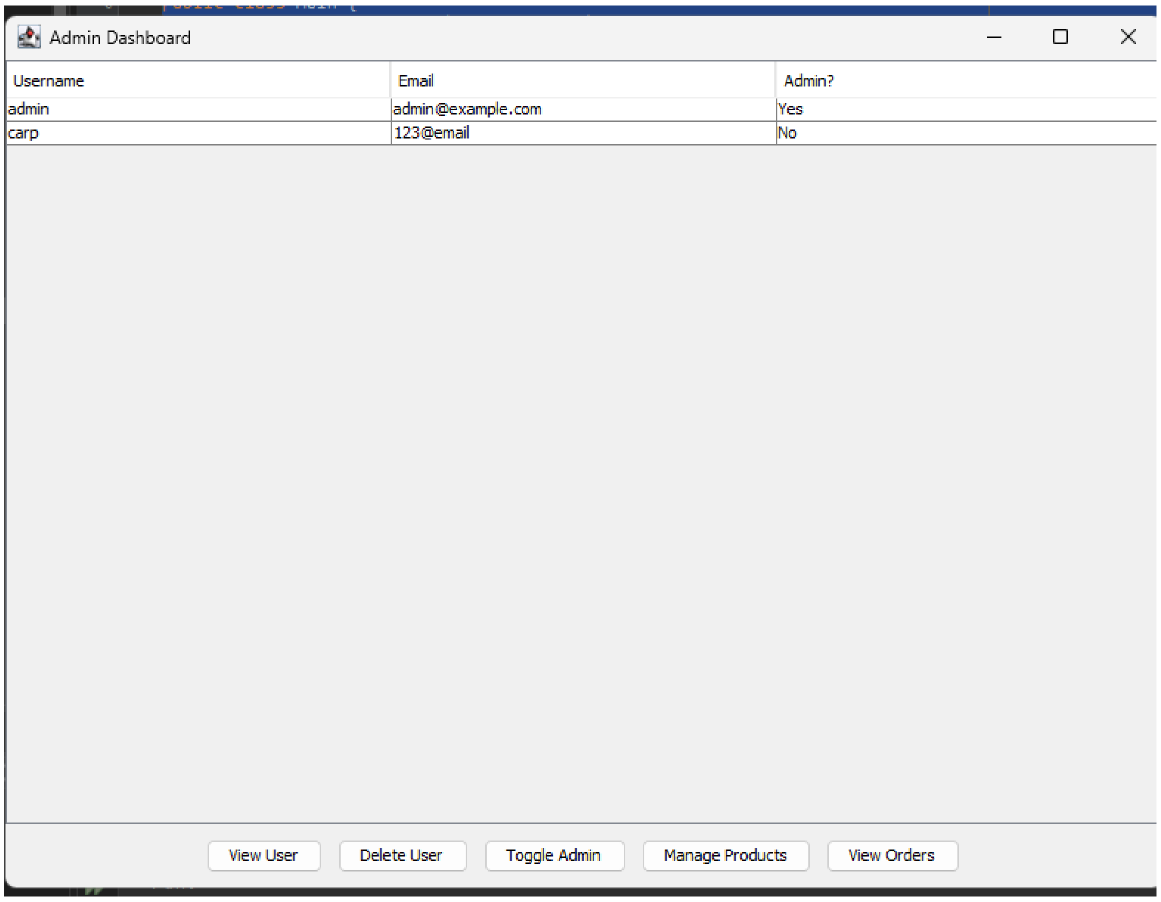


Order history feature coming soon!



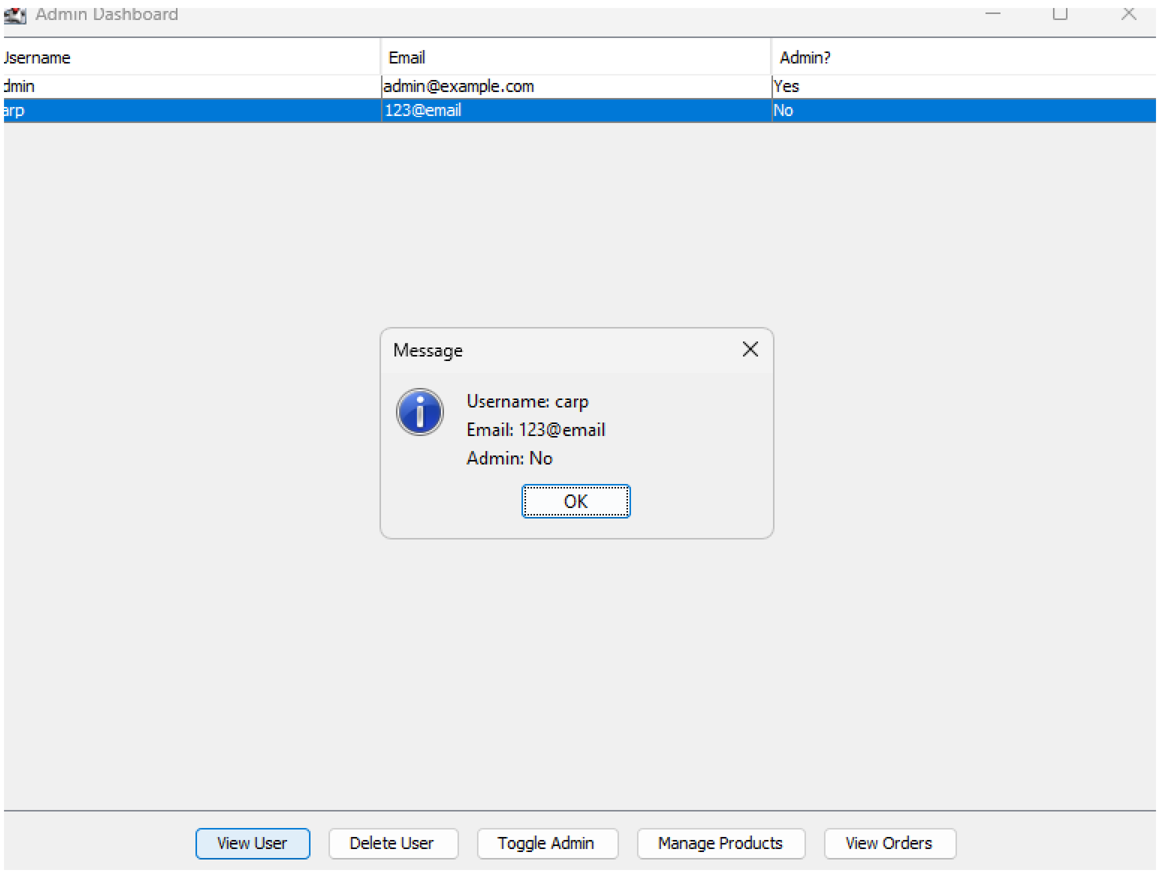
Username: carp
Email: 123@email
Address: Wall Street

Account Interface



Username	Email	Admin?
admin	admin@example.com	Yes
carp	123@email	No

Admin Dashboard



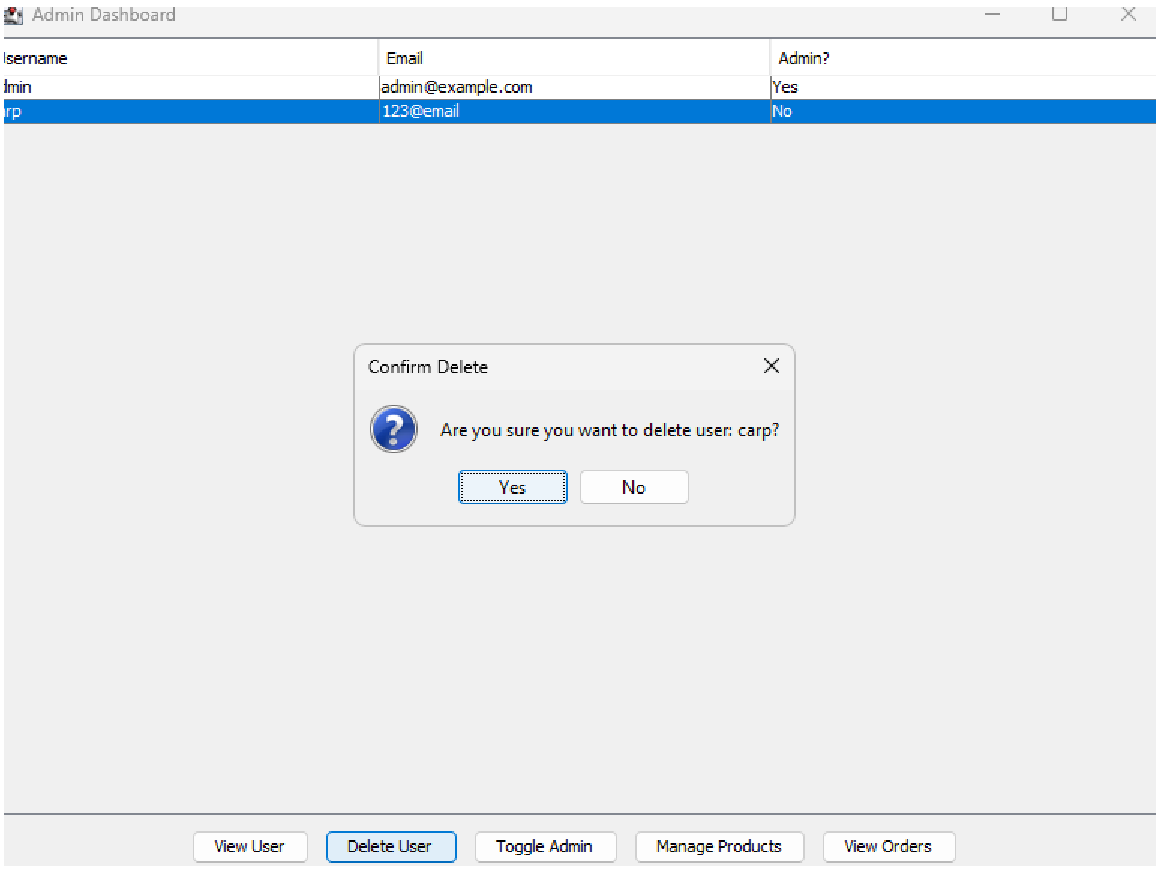
Jsername	Email	Admin?
dmin	admin@example.com	Yes
arp	123@email	No

Message

Username: carp
Email: 123@email
Admin: No

OK

View User



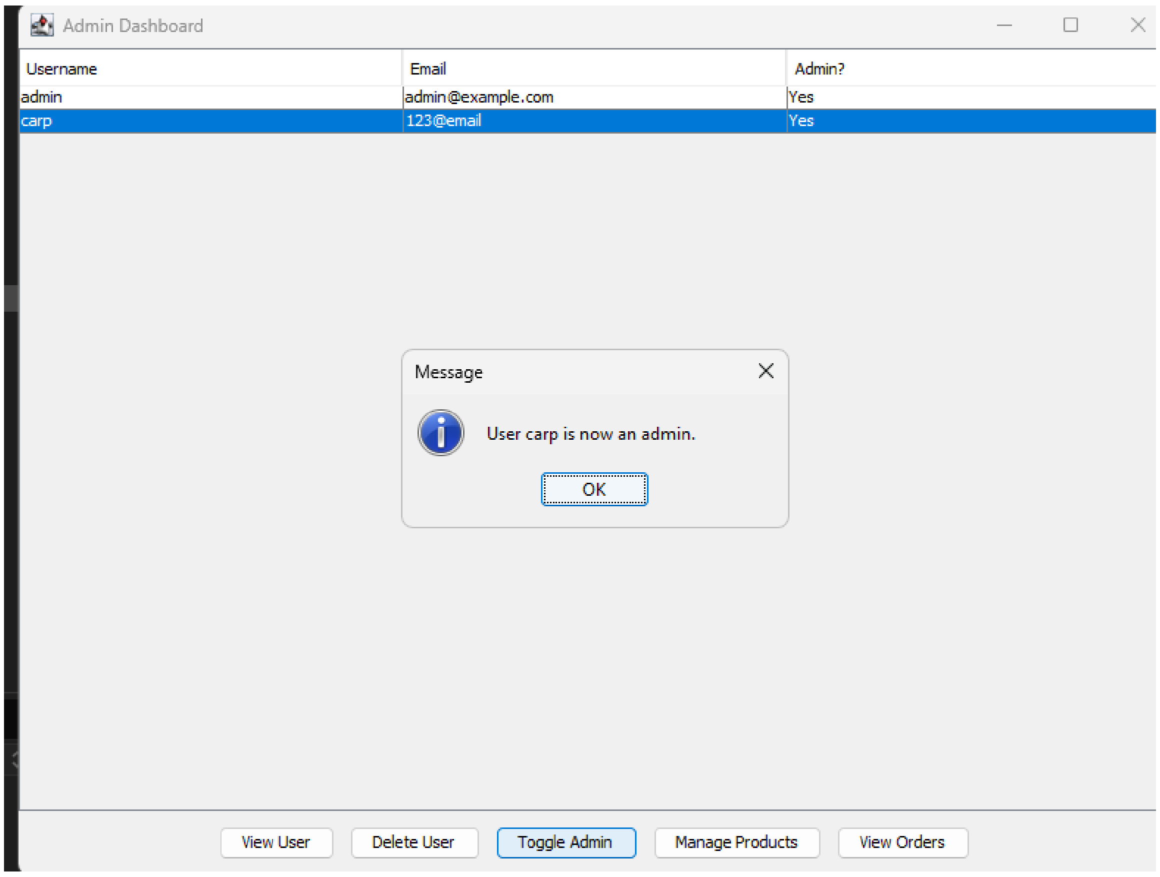
Isername	Email	Admin?
dmin	admin@example.com	Yes
rp	123@email	No

Confirm Delete

Are you sure you want to delete user: carp?

Yes No

Delete User

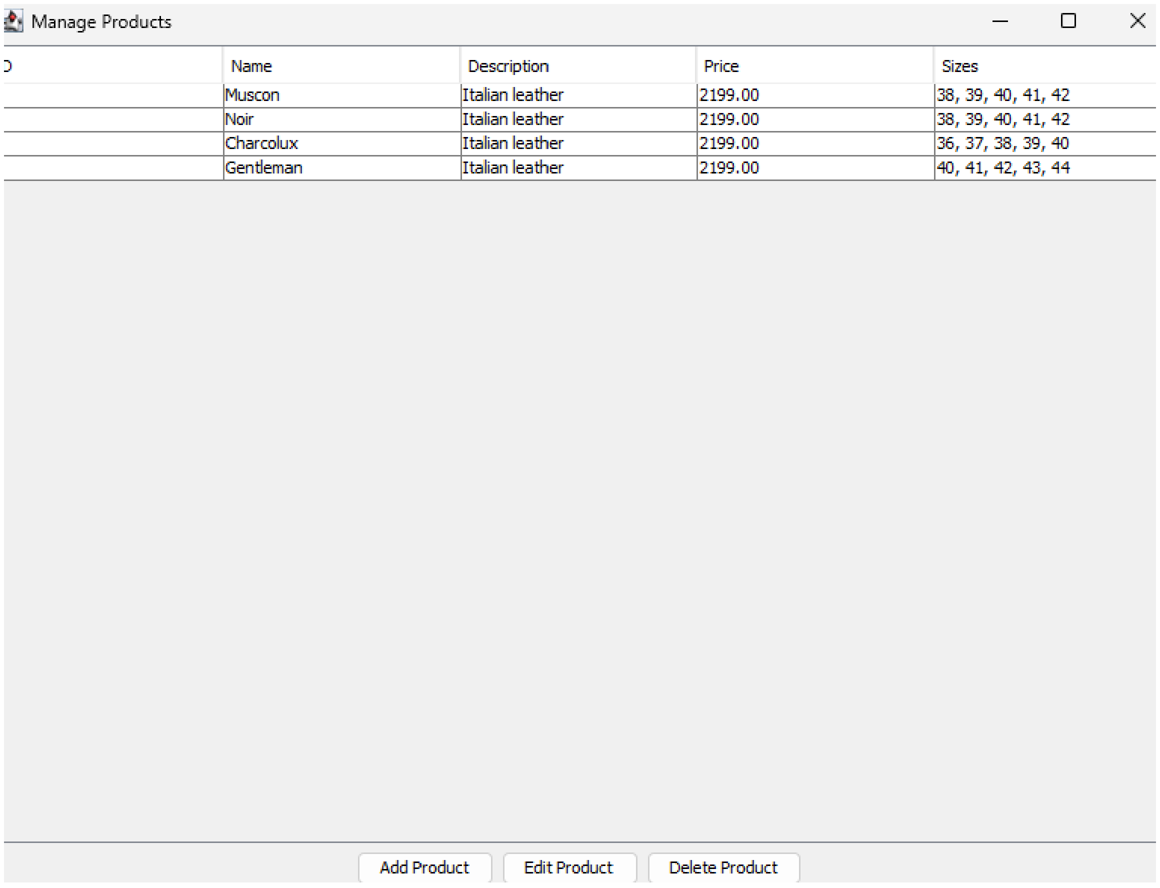


Username	Email	Admin?
admin	admin@example.com	Yes
carp	123@email	Yes

Message

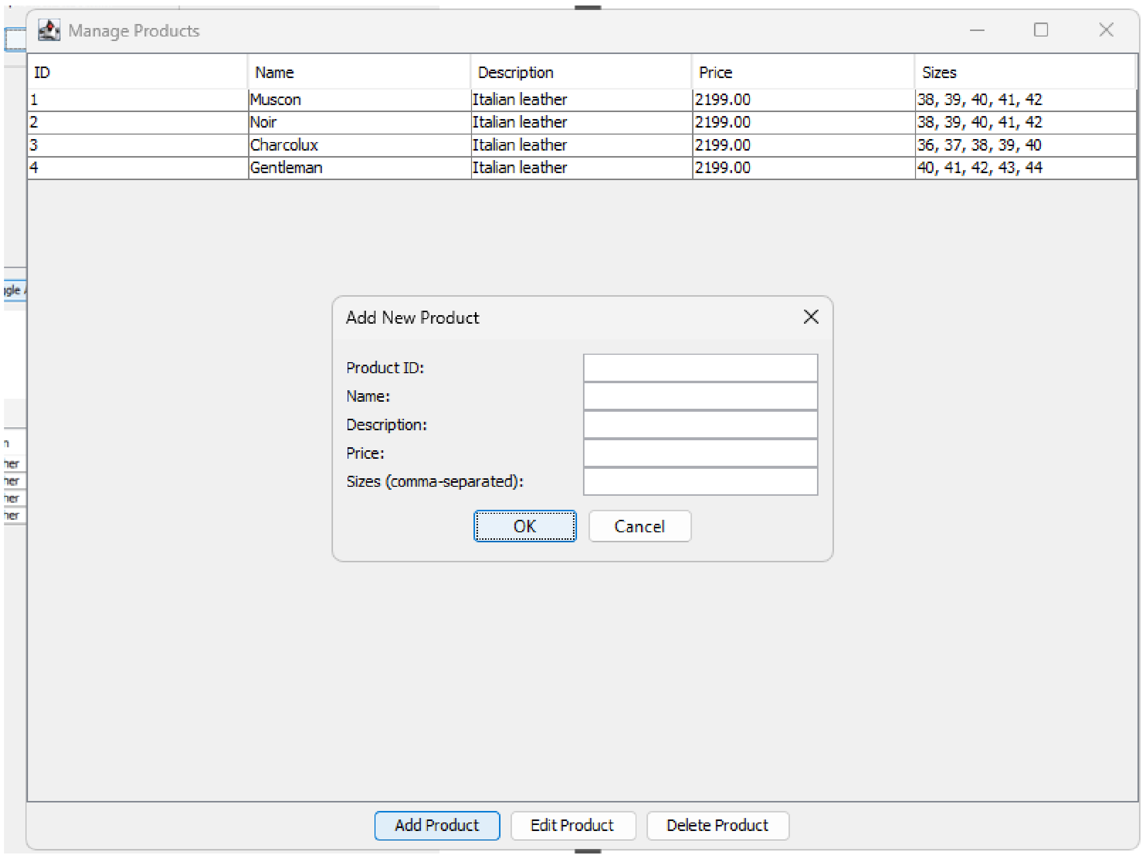
User carp is now an admin.

Admin Enabler



D	Name	Description	Price	Sizes
	Muscon	Italian leather	2199.00	38, 39, 40, 41, 42
	Noir	Italian leather	2199.00	38, 39, 40, 41, 42
	Charcolux	Italian leather	2199.00	36, 37, 38, 39, 40
	Gentleman	Italian leather	2199.00	40, 41, 42, 43, 44

Manage Products Interface

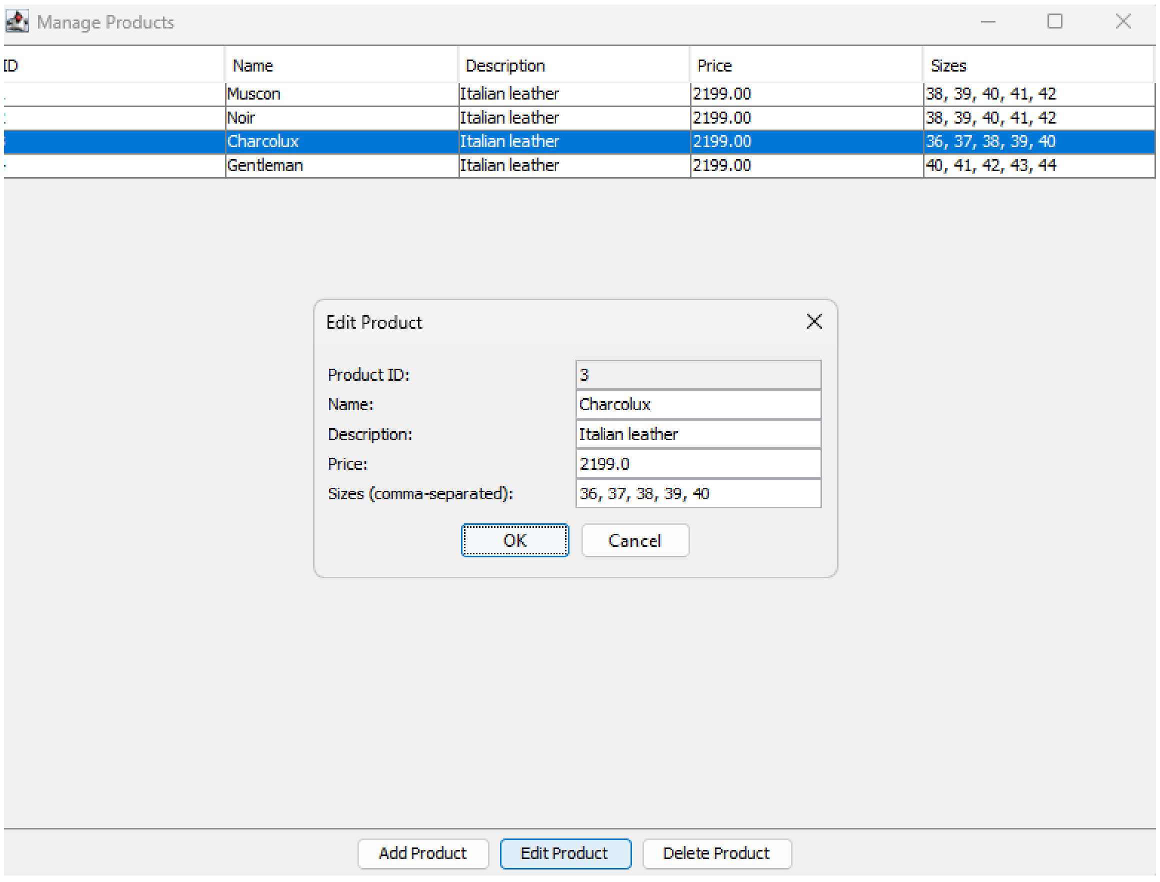


Manage Products

ID	Name	Description	Price	Sizes
1	Muscon	Italian leather	2199.00	38, 39, 40, 41, 42
2	Noir	Italian leather	2199.00	38, 39, 40, 41, 42
3	Charcolux	Italian leather	2199.00	36, 37, 38, 39, 40
4	Gentleman	Italian leather	2199.00	40, 41, 42, 43, 44

Add New Product

Product ID:	
Name:	
Description:	
Price:	
Sizes (comma-separated):	

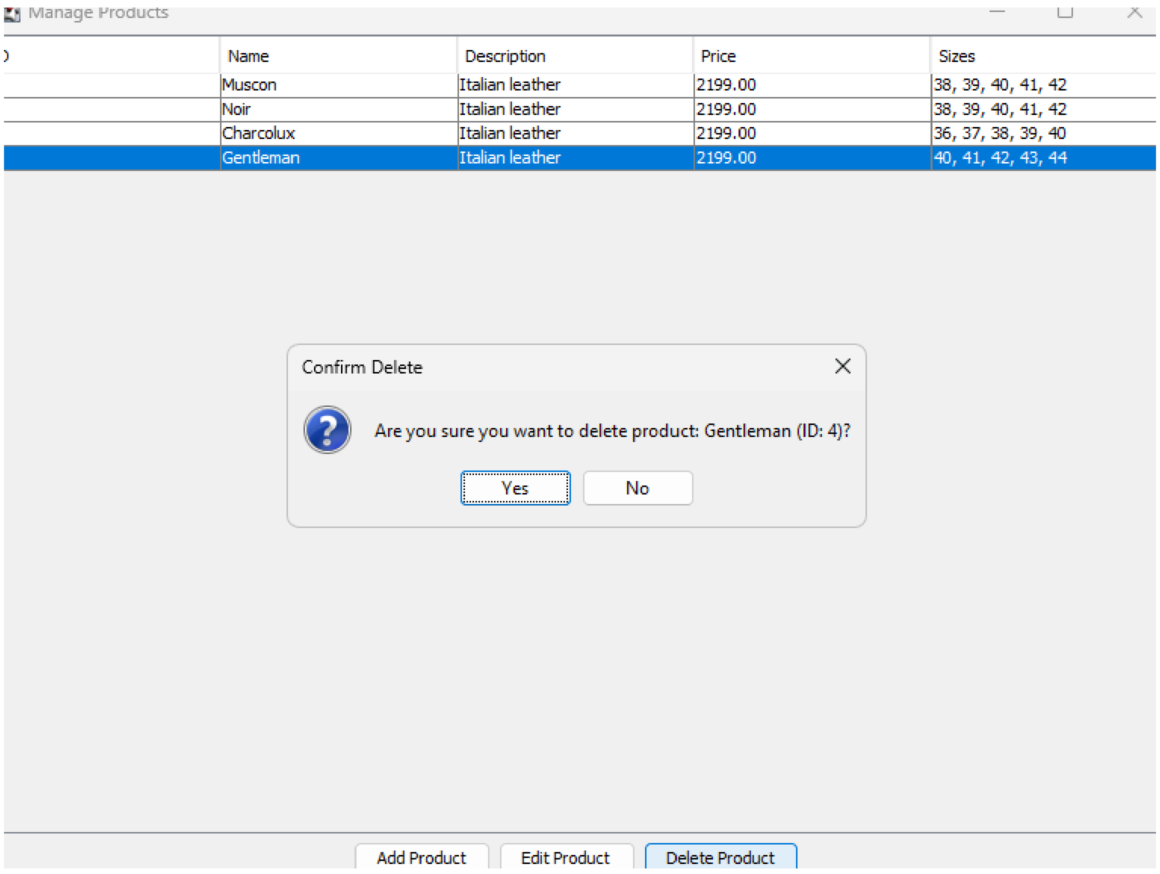


Manage Products

ID	Name	Description	Price	Sizes
	Muscon	Italian leather	2199.00	38, 39, 40, 41, 42
	Noir	Italian leather	2199.00	38, 39, 40, 41, 42
	Charcolux	Italian leather	2199.00	36, 37, 38, 39, 40
	Gentleman	Italian leather	2199.00	40, 41, 42, 43, 44

Edit Product

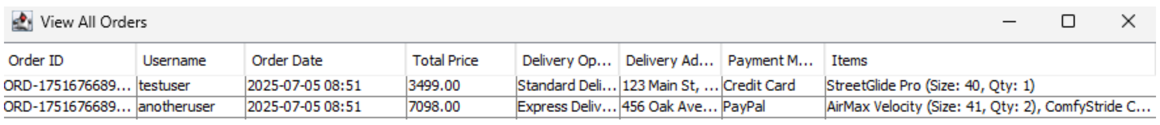
Product ID:	3
Name:	Charcolux
Description:	Italian leather
Price:	2199.0
Sizes (comma-separated):	36, 37, 38, 39, 40


Manage Products

)	Name	Description	Price	Sizes
	Muscon	Italian leather	2199.00	38, 39, 40, 41, 42
	Noir	Italian leather	2199.00	38, 39, 40, 41, 42
	Charcolux	Italian leather	2199.00	36, 37, 38, 39, 40
	Gentleman	Italian leather	2199.00	40, 41, 42, 43, 44

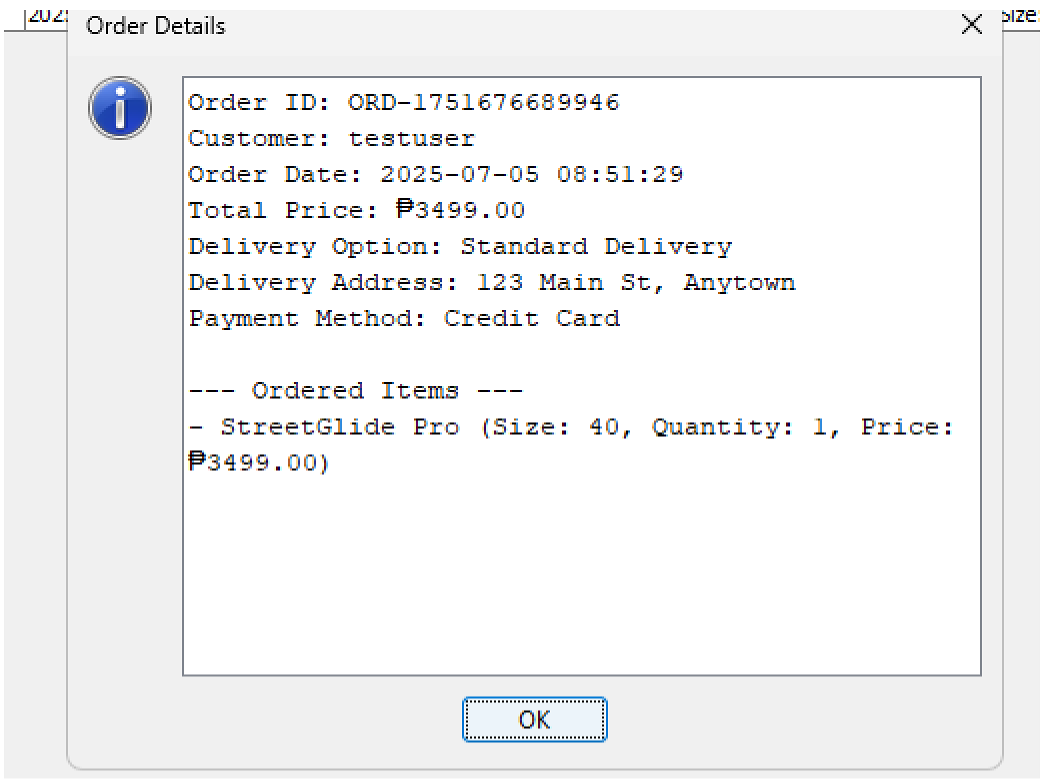
Confirm Delete

Are you sure you want to delete product: Gentleman (ID: 4)?


View All Orders

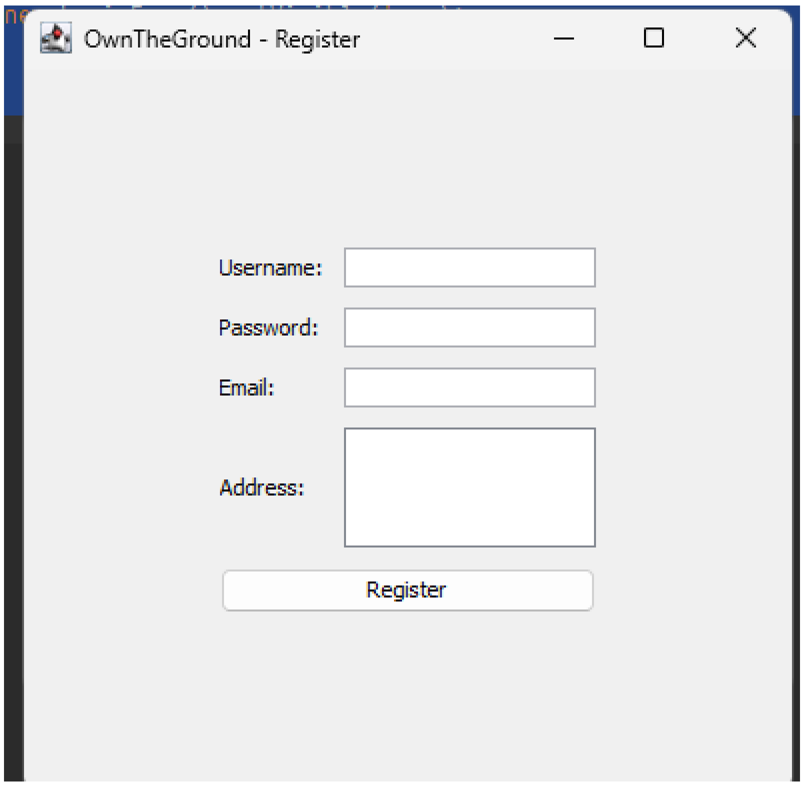
Order ID	Username	Order Date	Total Price	Delivery Op...	Delivery Ad...	Payment M...	Items
ORD-1751676689...	testuser	2025-07-05 08:51	3499.00	Standard Deli...	123 Main St, ...	Credit Card	StreetGlide Pro (Size: 40, Qty: 1)
ORD-1751676689...	anotheruser	2025-07-05 08:51	7098.00	Express Deliv...	456 Oak Ave...	PayPal	AirMax Velocity (Size: 41, Qty: 2), ComfyStride C...

List of orders

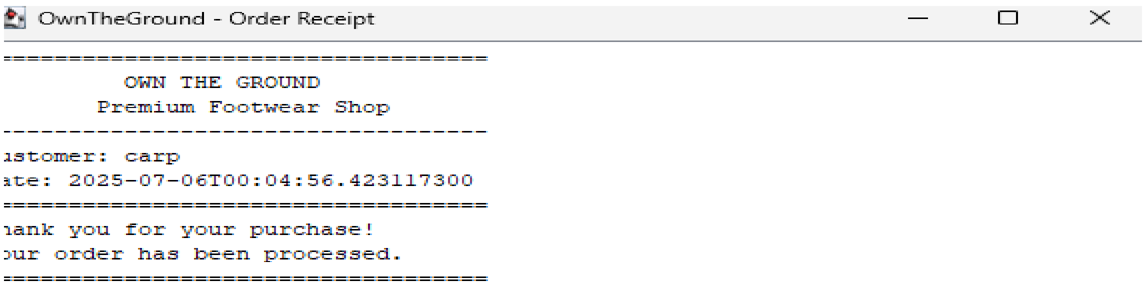

Order Details

```
Order ID: ORD-1751676689946
Customer: testuser
Order Date: 2025-07-05 08:51:29
Total Price: ₱3499.00
Delivery Option: Standard Delivery
Delivery Address: 123 Main St, Anytown
Payment Method: Credit Card

--- Ordered Items ---
- StreetGlide Pro (Size: 40, Quantity: 1, Price:
₱3499.00)
```

Order Details



Register Interface



```
===================================
         OWN THE GROUND
       Premium Footwear Shop
-----------------------------------
ıstomer: carp
ate: 2025-07-06T00:04:56.423117300
===================================
ıank you for your purchase!
our order has been processed.
===================================
```

Receipt Interface

COMPLETE APPLICATION GUIDE

1. **Login – give a username and password if you do not have then press register.**
2. **Registration – give Username, password, email, and address.**
3. **Products Dashboard – Choose a product and add to cart**
4. **Shopping Cart – Proceed to checkout**
5. **Checkout – Choose your payment method and if pick up or delivery and the delivery address if you chose delivery.**

FINDINGS, OBSERVATIONS, AND COMMENTS

During the development of this project, we learned how to use Java with the NetBeans IDE and Apache Ant to create a working desktop application. A big part of our learning came from trying out different ideas, testing code, and fixing bugs through trial and error. We ended up spending too much time on doing the functionality, leading us to having no design at all, and we still couldn’t add the history function. One of the most difficult parts was the admin dashboard, as it was having conflicts with the other files. This Gui project made us understand that we underestimated this Gui project as we still had many unused features, this was a valuable learning experience for building and debugging projects with java.