package bank\_management\_system;

import java.util.ArrayList;

import java.util.List;

import java.util.regex.Pattern;

public class Customer {

private String customerId;

private String name;

private String address;

private String phoneNumber;

private List<Account> accounts;

private static int nextCustomerId = 1;

private static final Pattern PHONE\_NUMBER\_PATTERN = Pattern.compile("^\\+?[0-9]+$");

private static final Pattern NAME\_PATTERN = Pattern.compile("^[a-zA-Z\\s]+$");

public Customer(String name, String address, String phoneNumber) {

this.customerId = "CUST-" + (nextCustomerId++);

if (isValidName(name)) {

this.name = name;

} else {

System.err.println("Error: Invalid name format for customer. Name must contain only letters and spaces.");

this.name = "INVALID\_NAME";

}

this.address = address;

if (isValidPhoneNumber(phoneNumber)) {

this.phoneNumber = phoneNumber;

} else {

System.err.println("Error: Invalid phone number format for customer. Phone number must contain only digits and an optional leading '+' sign.");

this.phoneNumber = "INVALID\_PHONE";

}

this.accounts = new ArrayList<>();

}

public String getCustomerId() {

return customerId;

}

public String getName() {

return name;

}

public String getAddress() {

return address;

}

public String getPhoneNumber() {

return phoneNumber;

}

public List<Account> getAccounts() {

return new ArrayList<>(accounts); // Return a copy to prevent external modification

}

public boolean hasAccount() {

return !accounts.isEmpty();

}

public void setName(String name) {

if (isValidName(name)) {

this.name = name;

} else {

System.err.println("Error: Invalid name format. Name must contain only letters and spaces. Name not updated.");

}

}

public void setAddress(String address) {

this.address = address;

}

public void setPhoneNumber(String phoneNumber) {

if (isValidPhoneNumber(phoneNumber)) {

this.phoneNumber = phoneNumber;

} else {

System.err.println("Error: Invalid phone number format. Phone number must contain only digits and an optional leading '+' sign. Phone number not updated.");

}

}

private boolean isValidName(String name) {

return name != null && NAME\_PATTERN.matcher(name).matches();

}

private boolean isValidPhoneNumber(String phoneNumber) {

return phoneNumber != null && PHONE\_NUMBER\_PATTERN.matcher(phoneNumber).matches();

}

public void addAccount(Account account) {

if (account != null) {

// While the list allows multiple, the Bank's logic will prevent adding more than one.

this.accounts.add(account);

// System.out.println("Account " + account.getAccountNumber() + " added to customer " + this.name); // This will be handled by Bank

} else {

System.out.println("Account could not be added (null).");

}

}

public void removeAccount(Account account) {

if (accounts.remove(account)) {

System.out.println("Account " + account.getAccountNumber() + " removed from customer " + this.name);

} else {

System.out.println("Account not found for customer " + this.name);

}

}

public void displayCustomerInfo() {

System.out.println("\n--- Customer Details ---");

System.out.println("Customer ID: " + customerId);

System.out.println("Name: " + name);

System.out.println("Address: " + address);

System.out.println("Phone: " + phoneNumber);

System.out.println("Associated Accounts:");

if (accounts.isEmpty()) {

System.out.println(" No accounts linked.");

} else {

for (Account acc : accounts) {

System.out.println(" - Account Number: " + acc.getAccountNumber() + " (Type: " + acc.getClass().getSimpleName() + ", Balance: $" + String.format("%.2f", acc.getBalance()) + ")");

}

}

System.out.println("------------------------");

}

}