import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.HashMap;  
import java.util.Map;  
  
class BankAccount {  
 private String accountNumber;  
 private String accountHolder;  
 private double balance;  
  
 public BankAccount(String accountNumber, String accountHolder, double initialBalance) {  
 this.accountNumber = accountNumber;  
 this.accountHolder = accountHolder;  
 this.balance = initialBalance;  
 }  
  
 public String getAccountNumber() {  
 return accountNumber;  
 }  
  
 public String getAccountHolder() {  
 return accountHolder;  
 }  
  
 public double getBalance() {  
 return balance;  
 }  
  
 public void deposit(double amount) {  
 if (amount > 0) {  
 balance += amount;  
 }  
 }  
  
 public void withdraw(double amount) {  
 if (amount > 0 && amount <= balance) {  
 balance -= amount;  
 }  
 }  
}  
  
class BankingSystem {  
 private Map<String, BankAccount> accounts = new HashMap<>();  
  
 public boolean createAccount(String accountNumber, String accountHolder, double initialBalance) {  
 if (!accounts.containsKey(accountNumber)) {  
 accounts.put(accountNumber, new BankAccount(accountNumber, accountHolder, initialBalance));  
 return true;  
 }  
 return false;  
 }  
  
 public BankAccount getAccount(String accountNumber) {  
 return accounts.get(accountNumber);  
 }  
  
 public void deposit(String accountNumber, double amount) {  
 BankAccount account = accounts.get(accountNumber);  
 if (account != null) {  
 account.deposit(amount);  
 }  
 }  
  
 public void withdraw(String accountNumber, double amount) {  
 BankAccount account = accounts.get(accountNumber);  
 if (account != null) {  
 account.withdraw(amount);  
 }  
 }  
  
 public double getBalance(String accountNumber) {  
 BankAccount account = accounts.get(accountNumber);  
 return account != null ? account.getBalance() : -1;  
 }  
  
 public boolean deleteAccount(String accountNumber) {  
 if (accounts.containsKey(accountNumber)) {  
 accounts.remove(accountNumber);  
 return true;  
 }  
 return false;  
 }  
}  
  
class BankingApp {  
 private JFrame frame;  
 private JTextField accountNumberField;  
 private JTextField amountField;  
 private JTextArea textArea;  
 private BankingSystem bankingSystem;  
  
 public BankingApp() {  
 bankingSystem = new BankingSystem();  
 initializeUI();  
 }  
  
 private void initializeUI() {  
 frame = new JFrame("Banking System");  
 frame.setSize(500, 400);  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.setLayout(new BorderLayout(10, 10));  
 frame.setLocationRelativeTo(null);  
  
 JPanel topPanel = new JPanel();  
 topPanel.setLayout(new GridLayout(2, 2, 10, 10));  
  
 JLabel accountNumberLabel = new JLabel("Account Number:");  
 accountNumberField = new JTextField();  
 JLabel amountLabel = new JLabel("Amount:");  
 amountField = new JTextField();  
  
 topPanel.add(accountNumberLabel);  
 topPanel.add(accountNumberField);  
 topPanel.add(amountLabel);  
 topPanel.add(amountField);  
  
 JPanel buttonPanel = new JPanel();  
 buttonPanel.setLayout(new GridLayout(5, 1, 10, 10));  
  
 JButton createAccountButton = new JButton("Create Account");  
 JButton depositButton = new JButton("Deposit");  
 JButton withdrawButton = new JButton("Withdraw");  
 JButton checkBalanceButton = new JButton("Check Balance");  
 JButton deleteAccountButton = new JButton("Delete Account");  
  
 createAccountButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String accountNumber = accountNumberField.getText();  
 String accountHolder = JOptionPane.*showInputDialog*("Enter account holder name:");  
 double initialBalance = Double.*parseDouble*(amountField.getText());  
 if (bankingSystem.createAccount(accountNumber, accountHolder, initialBalance)) {  
 textArea.append("Account created for " + accountHolder + " with initial balance " + initialBalance + "\n");  
 } else {  
 textArea.append("Account already exists with this number.\n");  
 }  
 }  
 });  
  
 depositButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String accountNumber = accountNumberField.getText();  
 double amount = Double.*parseDouble*(amountField.getText());  
 bankingSystem.deposit(accountNumber, amount);  
 textArea.append("Deposited " + amount + " to account " + accountNumber + "\n");  
 }  
 });  
  
 withdrawButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String accountNumber = accountNumberField.getText();  
 double amount = Double.*parseDouble*(amountField.getText());  
 bankingSystem.withdraw(accountNumber, amount);  
 textArea.append("Withdrew " + amount + " from account " + accountNumber + "\n");  
 }  
 });  
  
 checkBalanceButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String accountNumber = accountNumberField.getText();  
 double balance = bankingSystem.getBalance(accountNumber);  
 if (balance >= 0) {  
 textArea.append("Balance of account " + accountNumber + ": " + balance + "\n");  
 } else {  
 textArea.append("Account not found.\n");  
 }  
 }  
 });  
  
 deleteAccountButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String accountNumber = accountNumberField.getText();  
 boolean isDeleted = bankingSystem.deleteAccount(accountNumber);  
 if (isDeleted) {  
 textArea.append("Account " + accountNumber + " has been deleted.\n");  
 } else {  
 textArea.append("Account " + accountNumber + " not found.\n");  
 }  
 }  
 });  
  
 buttonPanel.add(createAccountButton);  
 buttonPanel.add(depositButton);  
 buttonPanel.add(withdrawButton);  
 buttonPanel.add(checkBalanceButton);  
 buttonPanel.add(deleteAccountButton);  
  
 JPanel logPanel = new JPanel();  
 logPanel.setLayout(new BorderLayout());  
 textArea = new JTextArea(10, 40);  
 textArea.setEditable(false);  
 textArea.setBackground(new Color(240, 240, 240));  
 textArea.setFont(new Font("Arial", Font.*PLAIN*, 12));  
 JScrollPane scrollPane = new JScrollPane(textArea);  
 logPanel.add(scrollPane, BorderLayout.*CENTER*);  
  
 frame.add(topPanel, BorderLayout.*NORTH*);  
 frame.add(buttonPanel, BorderLayout.*WEST*);  
 frame.add(logPanel, BorderLayout.*CENTER*);  
 frame.setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 new BankingApp();  
 }  
}