import java.awt.\*;

import java.awt.event.\*;

import java.util.ArrayList;

import javax.swing.JOptionPane;

public class BankApplication extends Frame implements ActionListener {

private static final long serialVersionUID = 1L;

ArrayList<String[]> accountList = new ArrayList<>();

Label label;

TextField textField;

Button submitButton;

Image backgroundImage;

public BankApplication() {

// Load the background image

backgroundImage = Toolkit.getDefaultToolkit().getImage("C:\\Users\\Admin\\Downloads\\background1.jpg");

accountList.add(new String[]{"123456", "1234567890", "John Doe", "1990-01-01"});

accountList.add(new String[]{"234567", "9876543210", "Jane Smith", "1985-05-10"});

accountList.add(new String[]{"345678", "9998887776", "Alice Johnson", "1995-12-25"});

accountList.add(new String[]{"456789", "1112223334", "Bob Brown", "1978-08-15"});

accountList.add(new String[]{"567890", "4445556667", "Emily Davis", "2000-04-30"});

Font labelFont = new Font("Andromeda", Font.PLAIN, 20);

Font textFont = new Font("Arial", Font.PLAIN, 20);

Font buttonFont = new Font("Broadway", Font.BOLD, 16);

setTitle("Bank Application");

setSize(700,500);

setLayout(new GridBagLayout());

label = new Label("Enter Account Number or Phone Number:");

label.setFont(labelFont);

textField = new TextField(20);

textField.setFont(textFont);

submitButton = new Button("Submit");

submitButton.setFont(buttonFont);

add(label);

add(textField);

add(submitButton);

GridBagConstraints gbc = new GridBagConstraints();

gbc.gridy = 0;

gbc.insets = new Insets(20, 20, 20, 20);

gbc.anchor = GridBagConstraints.CENTER;

add(label, gbc);

gbc.gridy = 1;

add(textField, gbc);

gbc.gridy = 2;

add(submitButton, gbc);

submitButton.addActionListener(this);

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent closewindow) {

dispose();

}

});

setVisible(true);

}

public void paint(Graphics g) {

super.paint(g);

g.drawImage(backgroundImage,1,1,getWidth(), getHeight(), this);

}

public void actionPerformed(ActionEvent e) {

if (e.getSource() == submitButton) {

String input = textField.getText();

boolean isValid = false;

String[] accountDetails = null;

for (String[] account : accountList) {

if (account[0].equals(input) || account[1].equals(input)) {

isValid = true;

accountDetails = account;

break;

}

}

if (isValid) {

BankOperations bankOperations = new BankOperations(accountDetails);

dispose();

} else {

JOptionPane.showMessageDialog(this, "Invalid Account Number or Phone Number!", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

public static void main(String[] args) {

new BankApplication();

}

}

class BankOperations extends Frame implements ActionListener {

/\*\*

\*

\*/

private static final long serialVersionUID = 1L;

String[] accountDetails;

double balance = 0;

Button depositButton, showBalanceButton, withdrawButton, interestButton, exitButton;

Image backgroundImage;

public BankOperations(String[] accountDetails) {

this.accountDetails = accountDetails;

backgroundImage = Toolkit.getDefaultToolkit().getImage("C:\\Users\\Admin\\Downloads\\background1.jpg");

setTitle("Bank Operations");

setSize(700,500);

setLayout(new FlowLayout());

depositButton = new Button("Deposit");

showBalanceButton = new Button("Show Bank Balance");

withdrawButton = new Button("Withdraw");

interestButton = new Button("Interest Comparison");

exitButton = new Button("Exit");

add(depositButton);

add(showBalanceButton);

add(withdrawButton);

add(interestButton);

add(exitButton);

depositButton.addActionListener(this);

showBalanceButton.addActionListener(this);

withdrawButton.addActionListener(this);

interestButton.addActionListener(this);

exitButton.addActionListener(this);

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

dispose();

}

});

setVisible(true);

}

public void paint(Graphics g) {

super.paint(g);

g.drawImage(backgroundImage, 0, 0,getWidth(), getHeight(), this);

}

public void actionPerformed(ActionEvent e) {

if (e.getSource() == depositButton) {

String amount = JOptionPane.showInputDialog(this, "Enter the amount to deposit:");

try {

double depositAmount = Double.parseDouble(amount);

if (depositAmount > 0) {

balance += depositAmount;

JOptionPane.showMessageDialog(this, "Amount deposited successfully!");

} else {

JOptionPane.showMessageDialog(this, "Invalid amount!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(this, "Invalid input!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} else if (e.getSource() == showBalanceButton) {

JOptionPane.showMessageDialog(this, "Your bank balance is: $" + balance);

} else if (e.getSource() == withdrawButton) {

if (balance > 0) {

String amount1 = JOptionPane.showInputDialog(this, "Enter the amount to withdraw:");

try {

double withdrawAmount = Double.parseDouble(amount1);

if (withdrawAmount > 0 && withdrawAmount <= balance) {

balance -= withdrawAmount;

JOptionPane.showMessageDialog(this, "Amount withdrawn successfully!");

} else {

JOptionPane.showMessageDialog(this, "Invalid amount!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(this, "Invalid input!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Insufficient balance!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} else if (e.getSource() == interestButton) {

String interestAmount = JOptionPane.showInputDialog(this,"Enter the rate of interest:") ;

String time = JOptionPane.showInputDialog(this,"Enter the time in years:");

try{

double interest = Double.parseDouble(interestAmount);

double years = Double.parseDouble(time);

if(interest > 0.0 && interest<=100.0 && years!=0){

JOptionPane.showMessageDialog(this,"The Interest Amount is"+(balance\*interest\*years)/100);

}

}

catch(NumberFormatException exp){

JOptionPane.showMessageDialog(this,"Invalid details","Error",JOptionPane.ERROR\_MESSAGE);

}

} else if (e.getSource() == exitButton) {

dispose();

}

}

}