RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM-602 105

CS23333 OOPS Using Java

Laboratory Record Note Book

Name :

Year / Branch / Section :

University Register No. : ..

College Roll No. : .

Semester : .

Academic Year : .

RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution

BONAFIDE CERTIFICATE

Name: ……

Academic Year: …… **Semester:** …… **Branch:** ……

Register No.

Certified that this is the bonafide record of work done by the above student in the.....Laboratory during the academic year 2025- 2026

Signature of Faculty in-charge

Submitted for the Practical Examination held on……

Internal Examiner **External Examiner**

INDEX

EX.NO	DATE	NAME OF THE EXPERIMENT	GITHUB QR
1		I/O, Data Types, Operators	
2		Control Structures	
3		Arrays	
4		Strings	
5		Classes & Objects	
6		Inheritance	
7		Interface	
8		Exceptions	
9		Collections	
10		Collections	
11		Project	
12		Lambda	

INVESTOR-STARTUP CONNECT

A MINI-PROJECT REPORT

Submitted by

MUKESH V 240701339

MOHAMED HUMAID 2407013322

in partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"INVESTOR-STARTUP CONNECT"** is the Bonafide work of **"MUKESH V, MOHAMED HUMAID ZAHIR HUSSAIN,** who carried out the project work under my supervision.

SIGNATURE

B. Deepa

Assistant Professor (SG)

Dept. of Computer Science and Engineering,
Rajalakshmi Engineering College
Chennai

This mini project report is submitted for the viva voce examination to be held on

_____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

This project presents a Java desktop application named *Investor & Startup Platform* designed to connect startups seeking funding with potential investors. Utilizing Java Swing for the user interface and MySQL for database management, the platform enables secure user registration and login with role-based access for startups and investors. Startups can post detailed business ideas and funding requirements, while investors can browse these opportunities and express their interest. All user actions and data are stored persistently, facilitating effective tracking of investor interest. This project demonstrates the integration of database operations, user authentication, and interactive GUI design to support early-stage investment collaboration.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson DR. **M. THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head of the Department **Dr. E.M. MALATHY** and our Deputy Head of the Department **Dr. J. MANORANJINI for** being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Dr. G. DHARANI DEVI,** for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

1. MUKESH V

2. MOHAMED HUMAID

TABLE OF CONTENTS

CHAPTER NO. TITLE

CHAPTER 1

INTRODUCTION

1.1) INTRODUCTION

In today's business environment, connecting startups with potential investors is crucial for innovation and economic growth. This project develops a desktop application using Java Swing and MySQL that facilitates this interaction efficiently. It overcomes traditional hurdles of networking by providing a digital platform for startups to register, post ideas with funding needs, and for investors to explore and express interest in these startups.

SCOPE OF THE WORK

The system includes:

- User registration with role-based access (Startup / Investor)
- Startup idea posting and management
- Investor browsing and interest expression
- Persistent storage with MySQL for user and startup data
- User-friendly GUI for seamless interaction

Future enhancements could include admin panels, online payments, and notification systems.

1.2) PROBLEM STATEMENT

Manual networking and funding processes are inefficient and error-prone. There is a need for an accessible digital platform to streamline matchmaking between startups and investors, ensuring data consistency and security.

1.4) AIM AND OBJECTIVES OF THE PROJECT

Aim:
To build an interactive desktop application enabling startups to showcase ideas and investors to find funding opportunities.

Objectives:

- Develop a Java Swing GUI application
- Integrate a MySQL database backend
- Implement role-based access control
- Provide real-time updates of startups and investor interests

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 Hardware Specifications

- **Processor: AMD Ryzen 5**
- **Memory: 8 GB**
- **Storage: 256 GB HDD**

2.2 Software Specifications

- **OS: Windows 11**
- **Front-End: Java Swing**
- **Back-End: MySQL**
- **Programming Languages: Java, SQL**

CHAPTER 3

MODULE DESCRIPTION

3.1 User Module

Handles registration, login, and role assignment for startups and investors.

3.2 Startup Module

Allows startups to post and manage startup ideas and funding requests.

3.3 Investor Module

Enables investors to browse startups, filter and search, and express interest.

3.4 Interest Tracking Module

Tracks investor interests and notifies startups accordingly.

CHAPTER 4

SAMPLE CODING

```
import javax.swing.*;

import javax.swing.border.EmptyBorder;
```

```
import javax.swing.table.DefaultTableModel;

import java.awt.*;

import java.awt.event.*;

import java.sql.*;



public class InvestorStartupApp extends JFrame {

    private Connection conn;

    private JTextField emailField, passwordField;

    private JButton loginBtn, registerBtn;



    // Investor dashboard components

    private JTable startupTable;

    private DefaultTableModel startupTableModel;

    private JButton expressInterestBtn, investorLogoutBtn;



    // Startup dashboard components

    private JTextField startupNameField, fundingField;

    private JTextArea descriptionArea;
```

```
private JButton postIdeaBtn, startupLogoutBtn;

private DefaultTableModel myStartupsTableModel;

private JTable myStartupsTable;



private int currentUserId;

private String currentUserName;

private String currentUserRole;



public InvestorStartupApp() {

   setTitle("Investor & Startup Platform");

   setSize(800, 600);

   setDefaultCloseOperation(EXIT_ON_CLOSE);

   setLocationRelativeTo(null);

   connectDatabase();

   showLoginScreen();

   setVisible(true);

}
```

```
    private void connectDatabase() {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(
                "jdbc:mysql://localhost:3306/investor_connect", "root",
"your_password");
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, "Database connection failed: " +
e.getMessage());
            System.exit(1);
        }
    }

    private void showLoginScreen() {
        getContentPane().removeAll();
        JPanel loginPanel = new JPanel(null);
        loginPanel.setBorder(new EmptyBorder(20, 20, 20, 20));

        JLabel title = new JLabel("Login");
```

```
title.setFont(new Font("Segoe UI", Font.BOLD, 24));

title.setBounds(350, 40, 100, 30);

loginPanel.add(title);


JLabel emailLabel = new JLabel("Email:");

emailLabel.setBounds(280, 100, 100, 25);

loginPanel.add(emailLabel);


emailField = new JTextField();

emailField.setBounds(360, 100, 200, 30);

loginPanel.add(emailField);


JLabel passwordLabel = new JLabel("Password:");

passwordLabel.setBounds(280, 140, 100, 25);

loginPanel.add(passwordLabel);


passwordField = new JPasswordField();

passwordField.setBounds(360, 140, 200, 30);
```

```
    loginPanel.add(passwordField);


    loginBtn = new JButton("Login");
    loginBtn.setBounds(360, 190, 200, 30);
    loginPanel.add(loginBtn);


    registerBtn = new JButton("Register");
    registerBtn.setBounds(360, 230, 200, 30);
    loginPanel.add(registerBtn);


    loginBtn.addActionListener(e -> loginUser());
    registerBtn.addActionListener(e -> showRegisterScreen());


    setContentPane(loginPanel);
    revalidate();
    repaint();
}
```

```
private void showRegisterScreen() {

   getContentPane().removeAll();

   JPanel registerPanel = new JPanel(null);

   registerPanel.setBorder(new EmptyBorder(20, 20, 20, 20));



   JLabel title = new JLabel("Register");

   title.setFont(new Font("Segoe UI", Font.BOLD, 24));

   title.setBounds(350, 40, 150, 30);

   registerPanel.add(title);



   JLabel nameLabel = new JLabel("Name:");

   nameLabel.setBounds(280, 100, 100, 25);

   registerPanel.add(nameLabel);



   JTextField nameField = new JTextField();

   nameField.setBounds(360, 100, 200, 30);

   registerPanel.add(nameField);
```

```
JLabel emailLabel = new JLabel("Email:");

emailLabel.setBounds(280, 140, 100, 25);

registerPanel.add(emailLabel);


JTextField emailRegField = new JTextField();

emailRegField.setBounds(360, 140, 200, 30);

registerPanel.add(emailRegField);


JLabel passwordLabel = new JLabel("Password:");

passwordLabel.setBounds(280, 180, 100, 25);

registerPanel.add(passwordLabel);


JPasswordField passwordRegField = new JPasswordField();

passwordRegField.setBounds(360, 180, 200, 30);

registerPanel.add(passwordRegField);


JLabel roleLabel = new JLabel("Role:");

roleLabel.setBounds(280, 220, 100, 25);
```

```
registerPanel.add(roleLabel);


JComboBox<String> roleCombo = new JComboBox<>(new String[]{"startup",
"investor"});
roleCombo.setBounds(360, 220, 200, 30);
registerPanel.add(roleCombo);


JButton submitRegBtn = new JButton("Submit");
submitRegBtn.setBounds(360, 270, 200, 30);
registerPanel.add(submitRegBtn);


JButton backBtn = new JButton("Back");
backBtn.setBounds(360, 310, 200, 30);
registerPanel.add(backBtn);


submitRegBtn.addActionListener(e -> {
  try {
    String name = nameField.getText().trim();
```

```
            String email = emailRegField.getText().trim();
            String password = new String(passwordRegField.getPassword()).trim();
            String role = (String) roleCombo.getSelectedItem();

            if (name.isEmpty() || email.isEmpty() || password.isEmpty()) {
               JOptionPane.showMessageDialog(this, "All fields are required.");
               return;
            }

            PreparedStatement pst = conn.prepareStatement(
                  "INSERT INTO users(name, email, password, user_type) VALUES (?, ?,
?, ?)");
            pst.setString(1, name);
            pst.setString(2, email);
            pst.setString(3, password);
            pst.setString(4, role);
            pst.executeUpdate();
```

```
            JOptionPane.showMessageDialog(this, "Registration successful. Please
login.");
            showLoginScreen();
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(this, "Error: " + ex.getMessage());
        }
    });

    backBtn.addActionListener(e -> showLoginScreen());

    setContentPane(registerPanel);
    revalidate();
    repaint();
  }

  private void loginUser() {
    try {
      String email = emailField.getText().trim();
```

```
String password = new String(((JPasswordField) passwordField).getPassword()).trim();


if (email.isEmpty() || password.isEmpty()) {
   JOptionPane.showMessageDialog(this, "Please enter both email and password.");
   return;
}


PreparedStatement pst = conn.prepareStatement(
    "SELECT * FROM users WHERE email = ? AND password = ?");
pst.setString(1, email);
pst.setString(2, password);


ResultSet rs = pst.executeQuery();


if (rs.next()) {
   currentUserId = rs.getInt("id");
   currentUserName = rs.getString("name");
```

```
            currentUserRole = rs.getString("user_type");



            if ("startup".equals(currentUserRole)) {

                showStartupDashboard();

            } else {

                showInvestorDashboard();

            }

        } else {

            JOptionPane.showMessageDialog(this, "Invalid credentials.");

        }



    } catch (SQLException ex) {

        JOptionPane.showMessageDialog(this, "Login failed: " + ex.getMessage());

    }

}



private void showStartupDashboard() {

    getContentPane().removeAll();
```

```
setTitle("Startup Dashboard - " + currentUserName);



JPanel panel = new JPanel(null);



JLabel label = new JLabel("Post Startup Idea");

label.setFont(new Font("Segoe UI", Font.BOLD, 20));

label.setBounds(300, 10, 250,30);

panel.add(label);



JLabel nameLabel = new JLabel("Startup Name:");

nameLabel.setBounds(50, 60, 120, 25);

panel.add(nameLabel);



startupNameField = new JTextField();

startupNameField.setBounds(180, 60, 500, 25);

panel.add(startupNameField);



JLabel descLabel = new JLabel("Idea Description:");
```

```
descLabel.setBounds(50, 100, 120, 25);

panel.add(descLabel);


descriptionArea = new JTextArea();

JScrollPane descriptionScroll = new JScrollPane(descriptionArea);

descriptionScroll.setBounds(180, 100, 500, 100);

panel.add(descriptionScroll);


JLabel fundingLabel = new JLabel("Funding Needed:");

fundingLabel.setBounds(50, 220, 120, 25);

panel.add(fundingLabel);


fundingField = new JTextField();

fundingField.setBounds(180, 220, 500, 25);

panel.add(fundingField);


postIdeaBtn = new JButton("Post Idea");

postIdeaBtn.setBounds(350, 270, 150, 30);
```

```
    panel.add(postIdeaBtn);


    startupLogoutBtn = new JButton("Logout");
    startupLogoutBtn.setBounds(650, 10, 100, 30);
    panel.add(startupLogoutBtn);


    postIdeaBtn.addActionListener(e -> postStartupIdea());
    startupLogoutBtn.addActionListener(e -> showLoginScreen());


    setContentPane(panel);
    revalidate();
    repaint();
}


private void postStartupIdea() {
    try {
        String startupName = startupNameField.getText().trim();
        String description = descriptionArea.getText().trim();
```

```
        double funding = Double.parseDouble(fundingField.getText().trim());


        if (startupName.isEmpty() || description.isEmpty()) {
           JOptionPane.showMessageDialog(this, "Please fill all fields.");
           return;
        }


        PreparedStatement pst = conn.prepareStatement("INSERT INTO
startups(startup_name, description, founder_email) VALUES (?, ?, ?)");
        pst.setString(1, startupName);
        pst.setString(2, description);
        pst.setString(3, getCurrentUserEmail());


        pst.executeUpdate();
        JOptionPane.showMessageDialog(this, "Startup idea posted successfully!");


        startupNameField.setText("");
        descriptionArea.setText("");
```

```
            fundingField.setText("");
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(this, "Posting failed: " +
ex.getMessage());
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this, "Invalid funding amount");
        }
    }

    private void showInvestorDashboard() {
        getContentPane().removeAll();
        setTitle("Investor Dashboard - " + currentUserName);

        JPanel panel = new JPanel(new BorderLayout());

        startupTableModel = new DefaultTableModel(new String[]{"ID", "Startup
Name", "Description", "Founder"}, 0);
        startupTable = new JTable(startupTableModel);
        JScrollPane scrollPane = new JScrollPane(startupTable);
```

```
JPanel buttonPanel = new JPanel();


expressInterestBtn = new JButton("Express Interest");

investorLogoutBtn = new JButton("Logout");


buttonPanel.add(expressInterestBtn);

buttonPanel.add(investorLogoutBtn);


panel.add(scrollPane, BorderLayout.CENTER);

panel.add(buttonPanel, BorderLayout.SOUTH);


expressInterestBtn.addActionListener(e -> expressInterest());

investorLogoutBtn.addActionListener(e -> showLoginScreen());


loadStartups();


setContentPane(panel);
```

```
        revalidate();
        repaint();
    }

    private void loadStartups() {
        startupTableModel.setRowCount(0);
        try {
            PreparedStatement pst = conn.prepareStatement("SELECT * FROM
startups");
            ResultSet rs = pst.executeQuery();
            while(rs.next()) {
                startupTableModel.addRow(new Object[]{
                        rs.getInt("id"),
                        rs.getString("startup_name"),
                        rs.getString("description"),
                        rs.getString("founder_email")
                });
            }
```

```
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(this, "Failed to load startups: " + ex.getMessage());
        }
    }

    private void expressInterest() {
        int selectedRow = startupTable.getSelectedRow();
        if(selectedRow == -1) {
            JOptionPane.showMessageDialog(this, "Please select a startup to express interest.");
            return;
        }

        int startupId = (int) startupTableModel.getValueAt(selectedRow, 0);

        try {
            PreparedStatement pst = conn.prepareStatement("INSERT INTO interests(investor_id, startup_id) VALUES (?, ?)");
```

```
            pst.setInt(1, currentUserId);
            pst.setInt(2, startupId);
            pst.executeUpdate();

            JOptionPane.showMessageDialog(this, "Interest expressed successfully!");
        } catch (SQLException ex) {
            JOptionPane.showMessageDialog(this, "Failed to express interest: " +
ex.getMessage());
        }
    }

    private String getCurrentUserEmail() {
        try {
            PreparedStatement pst = conn.prepareStatement("SELECT email FROM
users WHERE id = ?");
            pst.setInt(1, currentUserId);
            ResultSet rs = pst.executeQuery();
            if(rs.next()) {
                return rs.getString("email");
```

```
            }

        } catch(SQLException e) {

            JOptionPane.showMessageDialog(this, "Failed to get user email: " +
e.getMessage());

        }

        return "";

    }



    public static void main(String[] args) {

        SwingUtilities.invokeLater(InvestorStartupApp::new);

    }

}
```

TABLES USED

1)No.of Users:

id	name	email	password	user_type
1	Test User	test@gmail.com	test123	startup
2	mukesh	mukeshvelu005@gmail.com	123456	startup
3	humaid	humaid@gmail.com	humaid	investor

2) Startups:

id	startup_name	description	founder_email
1	FinLiberate	Loan optimization application for the users	mukeshvelu005@gmail.com
2	ArchLink	Architecture platform to help users to create their own designs at home	mukeshvelu005@gmail.com
3	workfix	we help people to connect with electricians	mukeshvelu005@gmail.com

3) List all startups with number of interested investors:

id	startup_name	description	interested_investors
1	FinLiberate	Loan optimization application for the users	1
2	ArchLink	Architecture platform to help users to create their own designs at home	1
3	workfix	we help people to connect with electricians	1

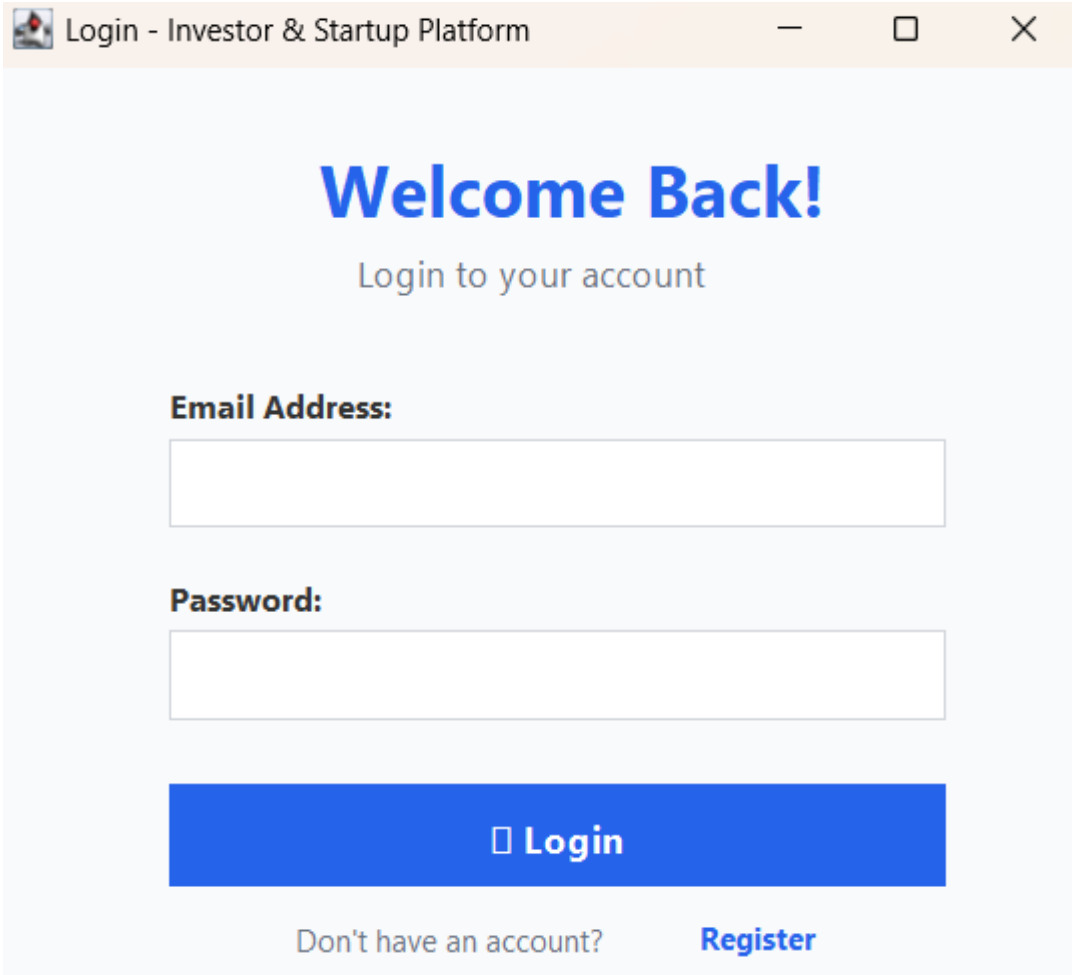
4) List startups posted by a startup founder:

id	startup_name	description
1	FinLiberate	Loan optimization application for the users
2	ArchLink	Architecture platform to help users to create their own designs at home
3	workfix	we help people to connect with electricians

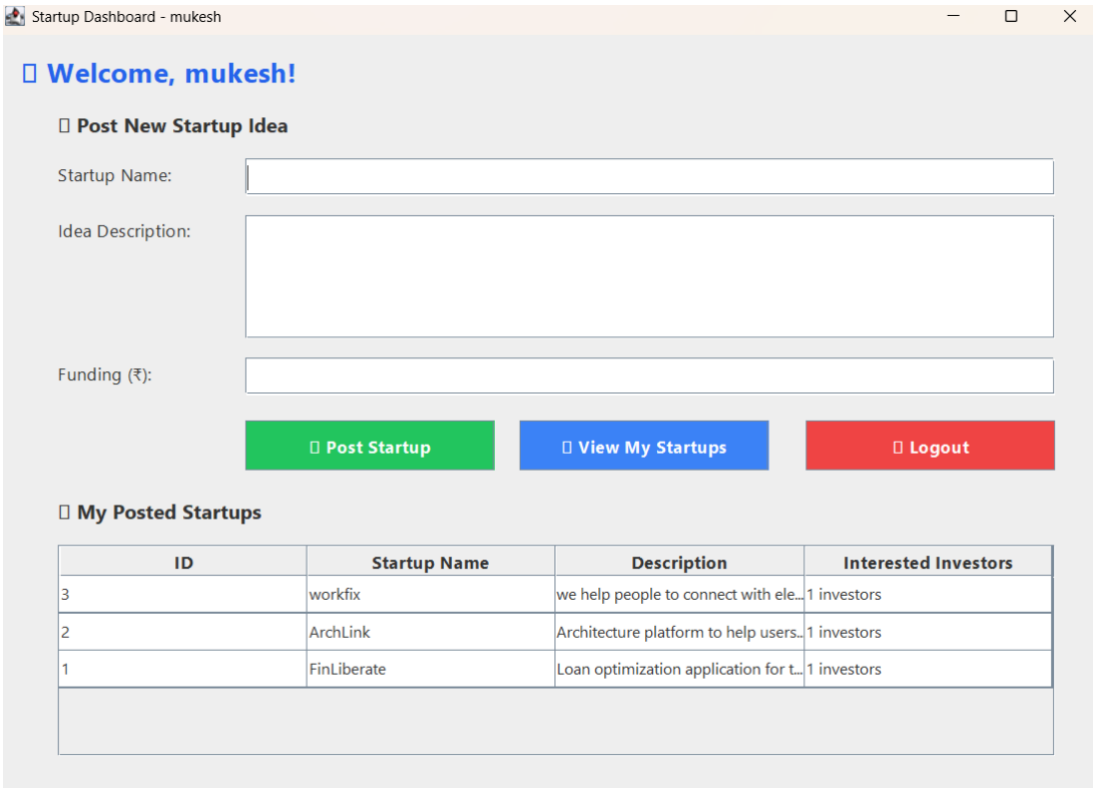
CHAPTER 5

SCREEN SHOTS

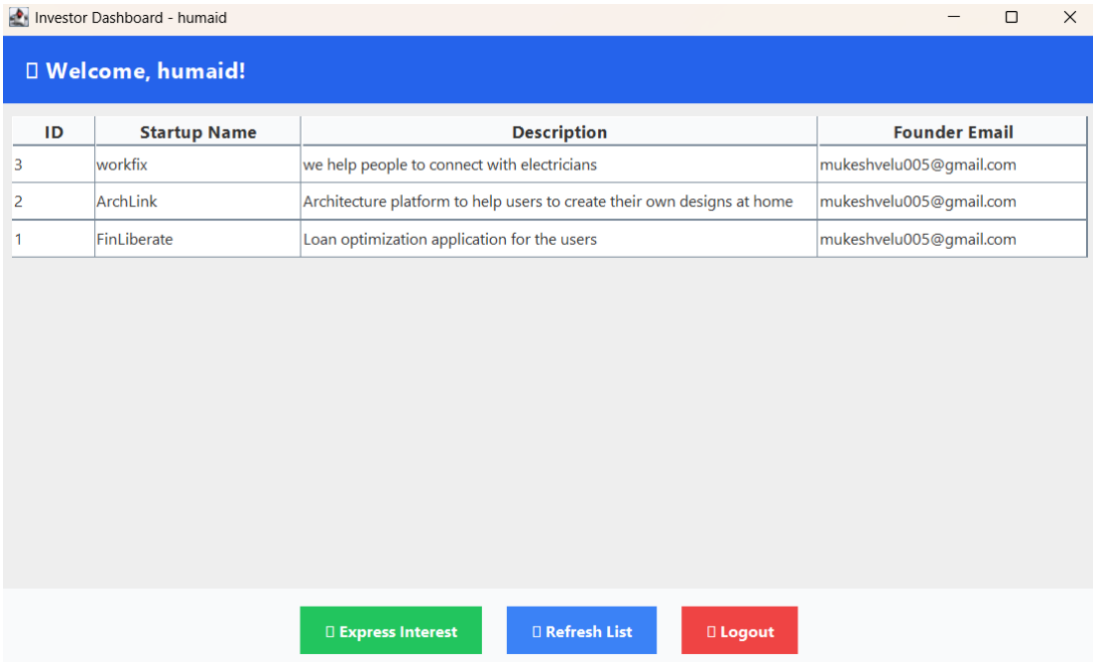
5.1) Login Page



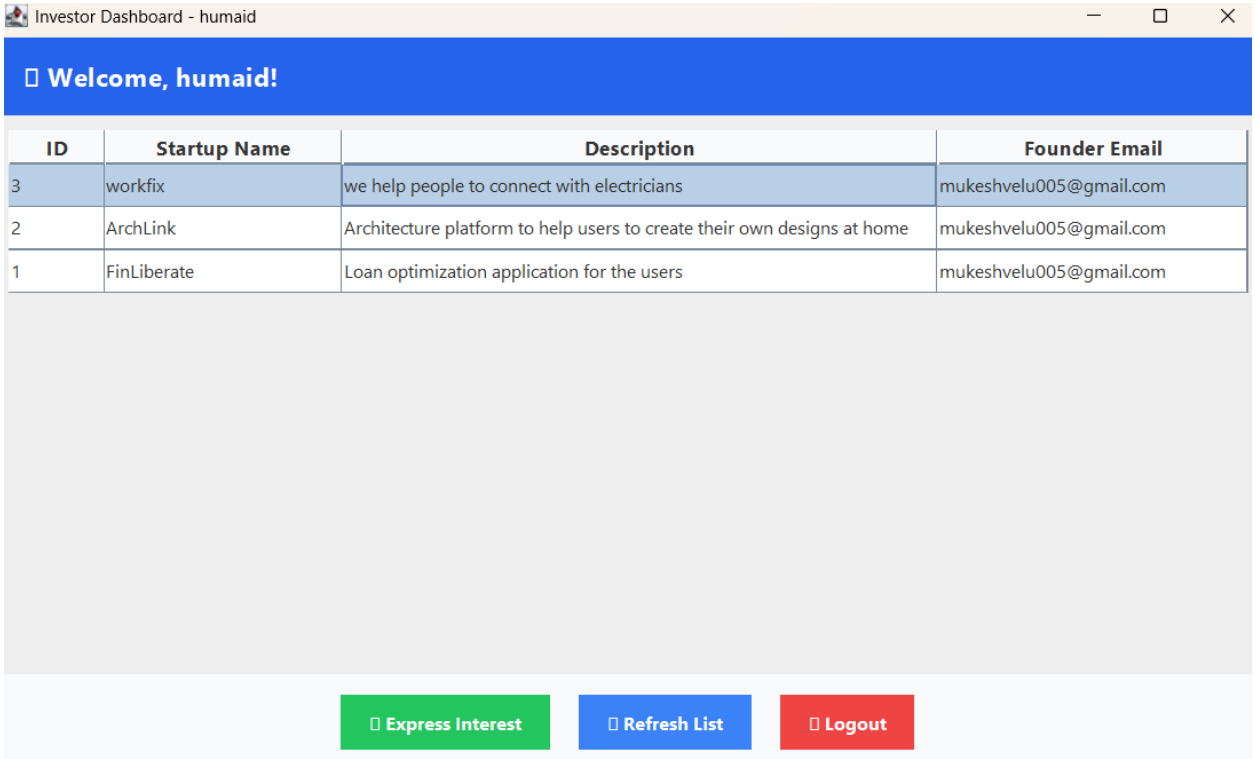
5.2) FOUNDER's PAGE



5.3) INVESTORS PAGE



5.4) INVESTORS RESPONSE



CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

This project successfully demonstrates the design and implementation of a desktop-based Investor & Startup Platform using Java Swing and MySQL. It effectively bridges the gap between startups seeking funding and investors looking for new opportunities by providing secure user registration, role-based access, and interactive

dashboards. Startups can post their ideas and funding needs, while investors can browse opportunities and express interest, all with real-time data persistence. The application enhances communication, organization, and transparency in early-stage investment, showcasing practical skills in software development, database management, and user interface design. Future enhancements could expand functionality and improve usability, further supporting the startup ecosystem.

CHAPTER 7 - REFERENCES

1. Kaur, Harleen, and Ravi Kumar. "A Study of Startup Funding Platforms and Their Impact on Early-Stage Investment." *International Journal of Management*, 2024.

2. Singh, Manpreet, and Ankit Sharma. "Design and Implementation of an Investor-Startup Matching System Using Java and MySQL." *International Journal of Computer Applications*, vol. 175, no. 12, 2025, pp. 12-18.

3. MySQL Documentation. "MySQL 8.0 Reference Manual." Oracle Corporation, 2024. https://dev.mysql.com/doc/

4. Oracle Documentation. "Java Platform, Standard Edition 17 API Specification." Oracle Corporation, 2024. https://docs.oracle.com/en/java/javase/17/