**Investment Matching Platform**

**Connecting Dreams with Opportunities**

**Prepared by:**

Saba Siddiq

24L-0540  
FAST-University

**Course Instructor:**

Usman Anwar

Dated:  
13-5-25

**Abstract**

The Investment Matching Platform is a C++ application designed to connect investors with business owners based on shared interests and financial compatibility. Utilizing the SFML library for its graphical interface, the platform allows users to register as investors or business owners, input relevant financial and categorical data, and find matches based on investment capacity and business needs. The project leverages object-oriented programming (OOP) principles such as inheritance, polymorphism, and composition to ensure modular and maintainable code. This report details the project’s objectives, implementation, and testing, highlighting the application of OOP concepts to achieve a robust system architecture

## **Table of Contents**

1. **Introduction**
   * The Problem We’re Solving
   * Why This Matters
   * Our Goals
2. **How We Built It**
   * OOP Magic: The Tech Behind the Scenes
   * UML Diagram (Visual Breakdown)
3. **Testing & Results**
   * Did It Work? Here’s the Proof.
4. **What’s Next?**
   * Cool Ideas for the Future

**1. Introduction**

**1.1 Problem Statement**  
  
In the modern entrepreneurial ecosystem, connecting investors with suitable business opportunities is often challenging due to information asymmetry and lack of centralized platforms. Manual matchmaking processes are time-consuming and inefficient, leading to missed opportunities for both investors and business owners.

**1.2 Objectives**  
  
The primary objectives of the Investment Matching Platform are:

* To develop a user-friendly graphical interface for investor and business owner registration.
* To implement a matching algorithm that pairs investors with businesses based on category and financial capacity.
* To utilize OOP principles to create a scalable and maintainable codebase.
* To ensure data persistence through file storage.

**1.3 Motivations**  
  
The motivation behind this project stems from the need to streamline investment processes in a digital age. By automating the matchmaking process, the platform reduces barriers to investment, fosters economic growth, and provides a practical application of OOP concepts in a real-world scenario.

**2. OOP Concepts Used**

The project extensively employs OOP principles to enhance code quality and scalability:

* **Inheritance**: The User class serves as an abstract base class, with Investor and BusinessOwner derived classes inheriting common attributes like username, password, name, and email. This reduces code duplication and ensures consistent behavior across user types.
* **Polymorphism**: The display and serialize methods are declared as pure virtual in the User class, implemented differently in Investor and BusinessOwner. This allows the Platform class to call these methods polymorphically, enhancing flexibility.
* **Composition**: The Platform class composes the User array, sf::RenderWindow, and other SFML objects to manage the application’s state and UI. This modular design isolates responsibilities, making the system easier to extend.
* **Aggregation**: The MatchMaker class operates independently, aggregating Investor and BusinessOwner objects to perform matching without owning them. This loose coupling supports reusability.

These OOP features collectively contribute to a modular, extensible, and maintainable codebase, facilitating future enhancements.

**3. Class Diagrams**

Below is a textual representation of the UML class diagrams for the project’s architecture, detailing public, protected, and private members.

|  |
| --- |
| User |
| -Username:string  -password:string  -name:string  -email:string  -isinvestor:boolername: string |

|  |
| --- |
| Matchmaker |
| +ismatch():boolername: string |

|  |
| --- |
| Investor |
| -investementCapacity  -interestedCategory  -investementGoals  : string |

|  |
| --- |
| BusinessOwner |
| -businessName  -businessDescription  -marketSale  -InvestementNeeded  -category me: string |

**Key:**

* **-** Private | **+** Public | **#** Protected
* Arrows show **inheritance** (↑) and **dependency** (→).

**4. Test Cases**

The following test cases validate the platform’s core functionalities:

1. **Investor Registration**
   * **Input**: Username: "inv1", Password: "pass123", Name: "saba", Email: "saba@gmail.com", Investment Capacity: 50000, Category: Technology (1), Goals: "Support startups"
   * **Output**: "Registration successful!" notification, user added to users array, data saved to data.txt.
2. **Business Owner Registration**
   * **Input**: Username: "biz1", Password: "pass456", Name: "sara", Email: "sara @gmail.com", Business Name: "TechCorp", Description: "AI solutions", Market Sales: 100000, Investment Needed: 30000, Category: Technology (1)
   * **Output**: "Registration successful!" notification, user added, data saved.
3. **Login Authentication**
   * **Input**: Username: "inv1", Password: "pass123"
   * **Output**: "Login successful! Welcome, saba", transitions to Investor Dashboard.
4. **Matching (Investor Perspective)**
   * **Input**: Investor "inv1" presses "1" on Investor Dashboard
   * **Output**: Displays "Business: TechCorp, Description: AI solutions, Category: Technology, Investment Needed: $30000".
5. **Matching (Business Owner Perspective)**
   * **Input**: Business Owner "biz1" presses "1" on Business Dashboard
   * **Output**: Displays "Investor: saba(saba@gmail.com), Investment Capacity: $50000, Category Interest: Technology, Goals: Support startups".
6. **Invalid Username**
   * **Input**: Username: "inv2"
   * **Output**: "Username not found. Type 'new' to register."
7. **Invalid Password**
   * **Input**: Username: "inv1", Password: "wrongpass"
   * **Output**: "Incorrect password.", returns to login screen.
8. **Data Persistence**
   * **Input**: Press "2" on dashboard to save, restart application
   * **Output**: Users loaded from data.txt, "Loaded 2 users from file." notification.

**5. Future Directions**

To enhance the platform’s scope, the following features are proposed:

* **Advanced Matching Algorithm**: Incorporate machine learning to consider additional factors like risk tolerance and business growth potential.
* **User Profiles**: Allow users to upload detailed profiles with documents and multimedia.
* **Messaging System**: Enable direct communication between matched investors and business owners.
* **Multi-Platform Support**: Develop mobile and web versions for broader accessibility.
* **Security Enhancements**: Implement password encryption and secure data storage.
* **Analytics Dashboard**: Provide users with insights into market trends and investment success rates.