Cairo University  
Faculty of Computers and Artificial Intelligence

**CS251**

**Intro. to Software Engineering**

**Personal Investment Management Software**

Software Requirements Specifications

Version 1.0

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20231109 | Aly El-Deen Yasser Ali | ali.el.badry.747@gmail.com | 01286964627 |
| 20231189 | Nagham Wael Mohamed Elsayed | [naghamw63@gmail.com](mailto:naghamw63@gmail.com) | 01007600773 |
| 20230280 | Fatema El-Zhraa Ahmed Mohamed El-Fiky | [fatmaelfeky922@gmail.com](https://studcu-my.sharepoint.com/personal/11410120230280_stud_cu_edu_eg/Documents/fatmaelfeky922@gmail.com) | 01221990828 |

Mars of 2025

Contents

[Team 3](#_Toc128921814)

[Document Purpose and Audience 3](#_Toc128921815)

[Introduction 3](#_Toc128921816)

[Software Purpose 3](#_Toc128921817)

[Software Scope 3](#_Toc128921818)

[Definitions, acronyms, and abbreviations 4](#_Toc128921819)

[Requirements 4](#_Toc128921820)

[Functional Requirements 4](#_Toc128921821)

[Non Functional Requirements 4](#_Toc128921822)

[System Models 5](#_Toc128921823)

[Use Case Model 5](#_Toc128921824)

[Enriched User Stories 5](#_Toc128921825)

[System Navigation Map 7](#_Toc128921826)

[Tools 7](#_Toc128921827)

[Ownership Report 7](#_Toc128921828)

[Policy Regarding Plagiarism: [To be removed] اقرأ هذا الجزء ثم احذفه 8](#_Toc128921829)

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20231109 | Aly El-Deen Yasser Ali | ali.el.badry.747@gmail.com | 01286964627 |
| 20231189 | Nagham Wael Mohamed Elsayed | [naghamw63@gmail.com](mailto:naghamw63@gmail.com) | 01007600773 |
| 20230280 | Fatema El-Zhraa Ahmed Mohamed El-Fiky | [fatmaelfeky922@gmail.com](https://studcu-my.sharepoint.com/personal/11410120230280_stud_cu_edu_eg/Documents/fatmaelfeky922@gmail.com) | 01221990828 |

# Document Purpose and Audience Purpose This document outlines the functional, non-functional, and technical requirements for the Personal Investment Management Software. It defines the system's purpose, features, scope, and key technologies to guide the development process. Audience 1) Software Developers

# 2) Project Managers

# 3) UI/UX Designers

# 4) Testers & Quality Assurance Engineers

# 5) Stakeholders

# Introduction

## Software Purpose This app is designed to provide individuals with the ability to track and manage their financial investments across multiple asset types, including stocks, real estate, savings, and cryptocurrencies. Since one of our goals is to enhance financial literacy, we have added interactive learning modules to help users develop smarter money management habits. Additionally, the software automates budgeting and debt repayment to improve financial stability, offering personalized insights and recommendations based on real-time financial data. By integrating various financial sources and visualizing investment performance, this software enables users to make well-informed decisions and take control of their financial future.

## Software Scope

## Portfolio Tracking – Monitor different assets and analyze investment performance.

## Financial Education – Interactive modules, quizzes, and AI-driven insights for money management.

## Goal-Based Budgeting – Set and track income, savings, and retirement goals.

## Debt Management – Automate debt repayments using a predefined percentage of income.

## Risk & Asset Allocation – Help users diversify investments and manage risks.

## Real-Time Data Integration – Sync with banking systems, brokerage accounts, and stock exchanges.

## Visualization & Reports – Provide detailed charts and graphs for financial analysis.

## 

## Definitions, acronyms, and abbreviations

|  |  |
| --- | --- |
| Phrase | Definition |
| ROI (Return on Investment) | A financial metric that calculates the profitability of an investment. |
| Portfolio Tracking | Monitoring different assets and investments in real-time. |
| Asset Allocation | The strategy of diversifying investments across different asset classes. |
| Net Worth Computation | A feature that calculates a user’s total financial worth (assets - liabilities). |
| API (Application Programming Interface) | A set of protocols for integrating with banking systems, stock exchanges, and payment gateways. |
| Debt Management | Features that automate debt repayment using a predefined percentage of monthly income. |
| Unbanked & Underbanked | |  | | --- | |  |  |  | | --- | | Individuals without full access to traditional banking services, relying on cash or mobile wallets. | |
| Cloud-Based Deployment | Hosting the application on cloud services such as AWS, Google Cloud, or Azure for scalability. |

# Requirements

## Functional Requirements

|  |  |
| --- | --- |
| Requirement ID | Requirement Statement |
| FR01 | The First screen appears to the user is the login/sign-up screen |
| FR02 | The application will redirect to the screen showing to two input boxes for the username and password in case the user chooses to log in |
| FR03 | The application checks the registered credentials for security and to enable the user to log in. Also, he could reset his/her password through sending link to his/her email |
| FR04 | The application will show the signup screen if the user chooses to sign up showing some required fields to fill in such as name, username, password, email, phone number, country, city, National ID, Bank account number and Different investments. |
| FR05 | The application will redirect you to your dashboard which consists of:Some computations of net-worth and rate of investment  1. **Some graphs and visualizations** displaying how the investment flows and how much more do I need to reach the specified financial target. 2. **The financial obligations** that must be settled at specific times . 3. **Presenting and auto-tracking the last updates in the investments .** |
| FR06 | Also, the application provides some functionalities that could be done by viewing a panel to choose from as follows:  1. **Setting new targets**, as the investor has continuously renewable targets. 2. **Adding new investment sources and integrating them with the account**, as the growth of the investment the investor makes more investments through time so, the addition part is important for that reason. 3. **Identifying potential risks**, analyzing them and balances investments , which helps the users minimize risks and loss. 4. **Suggestions for enlarging your investments**. 5. **Providing some tutorials and educational activities** to learn how to invest efficiently and professionally and with minimizing risks . 6. **Connect with experts** to get some advice and tips. 7. **Offers retirement planning** , helping the users to plan for their retirement savings . 8. **Requesting assistance in managing the saving process**. |

## Non Functional Requirements

|  |  |
| --- | --- |
| Measures | Details |
| Performance | The login\sign-up page should take less than 1 second to appear.  * After the user enters his information, showing the dashboard should take less than 3 seconds. * The Dashboard should appear immediately after login in time end. * The system should respond to user actions within 2 seconds for 95% of requests * The application should handle at least 10,000+ concurrent users without performance degradation. * The app size should be low for all the low-end devices and old ones. * The app should put a limit for a transaction of money be less than or equal to 20,000 EGP |
| Scalability | The system should scale to 1milion+ user without downtime |
| Reliability and availability | If a server fails, requests should be redirected to a backup server within 5 seconds to prevent downtime. |
| Security | All user data and transactions should be fully encrypted.  * Users must authenticate using authentication apps for enhanced security * Use machine learning models to detect unusual transactions and block fraudulent activities. * Limit access based on user roles (e.g., admin, investor, customer support) to prevent unauthorized access. * Conduct monthly security tests to find and fix vulnerabilities before attackers do. |
| Usability | The platform should be easy to use with a clear dashboard and real-time graphs.  * The system must work seamlessly across mobile, tablets, and desktops, adapting to screen sizes dynamically. |
| Portability and compatibility | The application should work on Windows, macOS, Linux, iOS, and Android without performance issues.  * App should support for multiple payment methods * App should be multilingual, as it should be available in Arabic, English, and other regional languages. |

# System Models

## Use Case Model

* **Using UML, write the use case model expressing the system actors & operations**
* **Write a definition for each actor and what his role is.**

## Enriched User Stories

* **Using below table template, for each requirement write an enriched user story specifying the details of each use case and showing the interaction to implement this use case.** 
  + **If one requirement is so big, you could divide it to more than one user story.**
  + **If some requirements are not major, you could plugin them in other user stories.**
* **Flow of events should be very detailed**
* **User Story #1**

|  |  |
| --- | --- |
| **User Story ID** | US #1 |
| **User Story Name** |  |
| **Actors** |  |
| **Description** | **As** a …..  **I like** to be able to …..  **So** ………….. |
| **Per condition** |  |
| **Post condition** |  |
| **Acceptance Criteria** | **Describe when we can decide that this user story is correctly implemented and accepted. For example:**  **Given** I’m a logged-out system user and I’m on the Sign-In page  **When** I fill in the “Username” and “Password” fields with my authentication credentials and I click the Sign-In button  **Then** the system signs me in |

* **Scenarios**

**Normal Scenario**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- User Enter Card and Password.  2- Click **Submit** |  |
|  | 3- System Verify user data  4- System displays list of Mobile companies |
| 5- User Select Vodafone from the list |  |
|  | 6- System retrieves Vodafone bills |
| 7- And so on |  |
|  |  |
|  |  |
|  |  |

**Exceptional Scenario**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- User Enter Card and Password.  2- Click **Submit** |  |
|  | 3- Card is invalid  4- Systems rejects card and displays an error message |
|  |  |
|  |  |

* **Screen Design**

**Give a draft design of the screen(s) on which this user story will be implemented.**

**Do it as a as wireframe or a mockup. Use a tool to do that. Give each screen a number and name.**

* **Data Dictionary:**

| **Element Label** | **Type/Length** | **Data Validation / Business Rule** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

* **User Story #1**

|  |  |
| --- | --- |
| **User Story ID** | US #1 |
| **User Story Name** | Signup for user |
| **Actors** | User |
| **Description** | **As** a User  **I like** to be able to sign up to the application  **So** I can login in and use the application |
| **Per condition** |  |
| **Post condition** | National ID should not be duplicated with the data base |
| **Acceptance Criteria** | **Given** I’m a user and I’m on the sign-up  **When** I fill in the sign up information fields with my credentials and I click sign up  **Then** the system signs me up |

**00.**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- Click on sign up |  |
|  | 2- System Show data that is required |
| 3- User Fill the input fields with data |  |
|  | 4- System verify that this national id is not in system  6- Show the dashboard to the user |

**Exceptional Scenario**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- Click on sign up |  |
|  | 2- Show the data should be entered |
| 3- User fill the input fields with his data |  |
|  | 4- system verify that national id is not in system  5- Show the user that this national id signed up before |

* **Screen Design**
* **Data Dictionary**

| **Element Label** | **Type/Length** | **Data Validation / Business Rule** |
| --- | --- | --- |
| Name field | Text < 100 character | Structured text |
| Email | Text < 100 character | Structured text |
| Password | Alphanumeric> 8 and < 20 | Password confirmation and must  include an uppercase letter, numbers  and/or special characters |
| National ID | Numeric equal to 14 digits | Check it in the database of civil registry |
| Phone number field | Numeric < 15 digit | Confirmed by text message |

* **User Story #2**

|  |  |
| --- | --- |
| **User Story ID** | US #2 |
| **User Story Name** | Login for user |
| **Actors** | User |
| **Description** | **As** a User  **I like** to be able to login to the application  **So** I can reach dashboard and use the application |
| **Per condition** |  |
| **Post condition** | Email and password exist in data base |
| **Acceptance Criteria** | **Given** I’m a user and I’m on the login page**.**  **When** I fill in email and password and click login  **Then** the system logins me in |

**Normal Scenario**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- Click on login in |  |
|  | 2- System Show data that is required |
| 3- User Fill the input fields with data |  |
|  | 4- System verify that email and password are correct  6- Show the dashboard to the user |

**Exceptional Scenario**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- Click on sign up |  |
|  | 2- Show the data should be entered |
| 3- User fill the input fields with his data |  |
|  | 4- system verify that email and password are wrong.  5- Show the user that email or password is wrong |

## System Navigation Map

* **Draw a navigation map that show how the screens are related (See example at** <https://stuff.mit.edu/afs/sipb/project/android/docs/training/design-navigation/wireframing.html>)

# Tools

* **Write a list of all tools used to develop the design (e.g., ArgoUML, Visual-Paradigm, mocqus, etc.)**

# Ownership Report

* **Remove the following notes and any red notes**
* **For every item in this document, write the owners.**
* **Team leader must verify the table with the team members.**

|  |  |
| --- | --- |
| **Student** | **Items he created** |
| **Mohammad Ali Sayed** | **Part of Use Case Model, Non-Functional Requirements, and User Stories #1 and #2.** |
|  |  |
|  |  |

# Policy Regarding Plagiarism: [To be removed] اقرأ هذا الجزء ثم احذفه

* **Remove this part and all red instructions**
* **Students have collective ownership and responsibility of their project. Any violation of academic honesty will have severe consequences and punishment for ALL team members.**

1. **تشجع الكلية على مناقشة الأفكار و تبادل المعلومات و مناقشات الطلاب حيث يعتبر هذا جوهريا لعملية تعليمية سليمة**
2. **ساعد زملاءك على قدر ما تستطيع و حل لهم مشاكلهم فى الكود و لكن تبادل الحلول غير مقبول و يعتبر غشا.**
3. **أى حل يتشابه مع أى حل آخر بدرجة تقطع بأنهما منقولان من نفس المصدر سيعتبر أن صاحبيهما قد قاما بالغش.**
4. **قد توجد على النت برامج مشابهة لما نكتبه هنا أى نسخ من على النت يعتبر غشا يحاسب عليه صاحبه.**
5. **إذا لم تكن متأكدا أن فعلا ما يعد غشا فلتسأل المعيد أو أستاذ المادة.**
6. **فى حالة ثبوت الغش سيأخذ الطالب سالب درجة المسألة ، و فى حالة تكرار الغش سيرسب الطالب فى المقرر.**