A computer and earth map

AI-generated content may be incorrect.Cairo University  
Faculty of Computers and Artificial Intelligence

**CS251**

**Introduction to Software Engineering**

Money Minds

Software Requirements Specifications

Version 1.0

Loai Hataba 20230553

Abdullah Mohamed 20230231

Hossam Abdelaziz 20230121

March 2025

Contents

[Team 2](#_Toc193097310)

[Document Purpose and Audience 2](#_Toc193097311)

[Introduction 3](#_Toc193097312)

[Software Purpose 3](#_Toc193097313)

[Software Scope 3](#_Toc193097315)

[Definitions, acronyms, and abbreviations 3](#_Toc193097316)

[Requirements 4](#_Toc193097317)

[Functional Requirements 4](#_Toc193097318)

[Non Functional Requirements 6](#_Toc193097319)

[System Models 8](#_Toc193097321)

[Use Case Model 8](#_Toc193097322)

[Enriched User Stories 8](#_Toc193097323)

[System Navigation Map 12](#_Toc193097324)

[Tools 12](#_Toc193097325)

[Ownership Report 13](#_Toc193097326)

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20230553 | Loai Hataba | LoaiHataba@gmail.com | 01275397858 |
| 20230231 | Abdullah Mohamed | abdallamohammmed649@gmail.com |  |
| 20230121 | Hossam Abdelaziz | hossamabdelaziz2295@gmail.com |  |

# Document Purpose and Audience

Purpose:

The purpose of this paper is to make the requirements for this software more understandable by outlining the features that the software must have in order to reduce development time and expenses.

Audience:

* Software Development Team
* Stakeholders

# Introduction

## Software Purpose

## The purpose of the Money Minds Software is to help users set and achieve financial goals while providing guidance for better financial planning and expense management.

## Software Scope

Money Minds Software implements a budget tracking system for individual users or even organizations, offering various functionalities for expense tracking, income management, setting financial goals, spending analysis, and budget planning. The app helps users monitor their financial activities, generate reports, and gain insights into their spending habits to make informed financial decisions.

## Definitions, acronyms, and abbreviations

|  |  |
| --- | --- |
| Phrase | Definition |
| **Budget** | A financial plan written down to decide how much money would be spent each month on what. |
| **Income Sources** | Something that provides a regular supply of money, such as employment, investments, or real estate |

# 

# Requirements

## Functional Requirements

|  |  |
| --- | --- |
| Requirement ID | Requirement State |
| FR01 | As soon as the user launches the application, a login/sign-up screen will appear. |
| FR02 | If the user chooses to log in, the application will provide a box for them to enter their username and password. |
| FR03 | If a user chooses to register, the app will provide a signup page where they can enter their personal information, including first name, last name, password, gender, country, and email address. And asking if the user would like to sync with their bank account. This registration process will enable personalized features and secure access to their financial data. |
| FR04 | The application shall have a panel for users to manage their budgets, consisting of:   1. **“**My Budget” tab where users can:    * Create a new budget by specifying income sources, expense categories, and budget limits.    * Edit or delete existing budgets.    * View a summary of their financial status, including total income, total expenses, and remaining budget. 2. **“**Transactions” tab where users can:    * Add new transactions by entering details such as category, amount, date, and payment method.    * Edit or delete recorded transactions.    * Filter transactions by date, category, or amount. 3. “Goals” tab where users can:  * View their financial goals * Set new financial goals by specifying target savings, timeframes, and desired spending limits. * Track progress toward their financial goals based on their recorded transactions. |
| FR05 | The application shall provide a Reports panel where users can:   1. View graphical representations (charts, graphs) of their spending and savings trends. 2. Generate summary reports of monthly or yearly income and expenses. 3. Export reports in PDF or Excel format for external use. |
| FR06 | The application shall provide notifications to help users stay on track with their financial goals:   1. Budget Alerts: Notify users when they are approaching or exceeding their budget. 2. Bill Reminders: Send reminders for upcoming bills and due payments. 3. Goal Progress Notifications: Notify users of their progress toward financial goals. |

## Non-Functional Requirements

|  |  |
| --- | --- |
| **Measure** | **Details** |
| **Performance** | * The **sign-in/sign-up screen** should load within **3 seconds** under normal network conditions. * Every panel in the system should take **less than 7 seconds** to load, even under peak usage. |
| **Scalability** | * The system must support **up to 3,000 users** without experiencing performance degradation. * It should be designed to scale dynamically as the user base grows. |
| **Maintainability** | * The system should have an **80% probability** of being maintainable within **72 hours** in case of critical failures. * Critical issues should have a **75% probability** of resolution within three days. |
| **Reliability & Availability** | * The system should maintain **at least 90% uptime** per month, ensuring minimal downtime. * The system should function without major issues at least **90% of the time** in each month. |
| **Portability & Compatibility** | * The system should be cross-platform, supporting Desktop, Android, and iOS devices. * The app should be optimized for both low-end and high-end mobile devices, ensuring smooth performance. * The system should be compatible with the latest and previous two major OS versions (e.g., Android 12, 13, 14 and iOS 16, 17, 18). * The app should work in both online and offline modes, allowing users to access essential features without an internet connection. |
| **Security** | * Users must only access their **own** payment details and private data. * **Unauthorized users must not** access restricted pages or perform unauthorized actions. * The system must follow **OWASP security standards** to prevent common web vulnerabilities. * Data must be **encrypted at rest and in transit** to protect user privacy. |
| **Usability** | * The interface should be **user-friendly**, intuitive, and require minimal training. * The system should follow **UX best practices** for accessibility and ease of navigation. |

# System Models

## Use Case Model

## A screenshot of a computer

## 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** | User Sign-up |
| **Actors** | User |
| **Description** | **As** a user  **I like** to be able to sign-up to the application  **So** that I can sign in and use it’s services |
| **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 user and I’m on the sign-up screen  **When** I fill in all the fields with my information and I click sign-up  **Then** the system signs me up |

* **Scenarios**

**Normal Scenario**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- User clicks on sign-up |  |
|  | 2- Redirect to sign-up page  3- System displays boxes to enter information |
| 3- User Fills in the boxes  4- User chooses whether they want to sync with their bank account |  |
|  | 5- System verifies the user information   * If user chooses to sync system adds more features to user package * Else system doesn’t add certain features to user’s account |
|  | 6-System saves user info in the database |

**Exceptional Scenario**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1- User clicks on sign-up |  |
|  | 2- Redirect to sign-up page  3- System displays boxes to enter information |
| 3- User Fills in the boxes |  |
|  | 6- System verifies the user information  7- System finds input information invalid  8- System displays “Please enter valid information.” 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** |
| --- | --- | --- |
| Name field | Text < 50 characters | Structured Text |
| Email field | Text < 50 characters | Structured Text |
| Password field | Alphanumeric < 50 Characters | Password confirmation and must include an uppercase letter, numbers and/or special characters |
| Country field | A-Z list |  |
| Sync with bank acc field | Check box |  |

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

* **User Story #2**

|  |  |
| --- | --- |
| **User Story ID** | US #2 |
| **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 |

## 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.** |
|  |  |
|  |  |