

**Walletify**

Project specifications

Ahmad

Hajar

Lara

Ibraheem

07/10/2024

# Problem statement

Managing personal finances can be overwhelming and confusing, especially when it comes to tracking income, expenses, and savings goals. Many people struggle with keeping track of their spending habits and budgeting, leading to financial mismanagement and difficulty in achieving financial goals. A simple, user-friendly tool is needed to assist users in maintaining control over their money by providing clear visibility into their financial activities.

# Proposed Solution

Walletify is designed to help users manage their personal finances by providing an easy-to-use website for tracking income, expenses, and savings goals. The app will allow users to input their daily financial transactions, categorize them, and visualize their spending patterns over time. By offering this streamlined and intuitive platform, Walletify aims to empower users to take control of their finances, track their budget effectively, and achieve their financial goals.

#### Project Goals:

* Enable users to register and securely manage their accounts.
* Provide simple ways for users to track income and expenses.
* Allow users to categorize their transactions for better clarity.
* Offer visual summaries of spending habits.
* Help users set and track savings goals.

# Features

# #1: Authentication

1. The User Registration: The user can register by providing the following details:

**Fist Name**

**Last Name**

**User Name**

**Email Address**

**Password**

* + **Password must be at least 6 characters long.**
  + **Must contain at least one lowercase letter.**
  + **Must contain at least one uppercase letter.**
  + **Must contain at least one digit.**
  + **Must contain at least one special character (e.g., @, #, !, etc.).**

**Confirmation of Email**

1. **Login:** Users can log into their accounts using their email and password.
2. **Forgot Password Feature**: Allows users to reset their password via email.
3. **Profile Management:** Users can update their profile information, including password changes, financial information, and other preferences.

#### #2: Expense Tracking

1. **Initial Balance and Savings Goal**:

* Users can input **an initial** **amount of money (**balance**),** set **a monthly savings value,** and define **a saving target.**
* **Each month, the savings value will be automatically deducted from the balance until the goal is reached.**

1. **Income and Expense Entry:** Users can input financial transactions by categorizing them as either income or expenses.

* Options include**: Food & Drink, Personal Care, Investment, Salary, Bonus, Health Care, etc.**
* **Automatic Date Selection:** The app will automatically capture and assign the current date and time for each transaction. Users do not need to manually select the date.

1. **Transaction History:** Users can view **the top 10 most recent transactions** on their dashboard, providing a quick snapshot of their recent financial activity.
2. **Budget Limit and Tracking**: Users can view their **current budget, saved per mounth, total savings and target goal.** The app will display the ratio of target achievement (progress toward the goal) in a visual format, such as a progress bar.

# Wireframes

Should contain all the wireframes that the user interacts with during his usage of the system.

[**What Is A Wireframe? Your Best Guide (careerfoundry.com)**](https://careerfoundry.com/en/blog/ux-design/what-is-a-wireframe-guide/)

# Software Diagrams

Should contain all the diagrams and drawings that clarifies the entities that make up the project, and how they interact with each other.

# User Stories for Walletify

#### 1. User Registration and Authentication

* **As a new user**, I want to register for an account so that I can securely manage my financial transactions and access personalized features.
* **As a returning user**, I want to log in to my account using my email and password so that I can access my financial data.

#### 2. User Profile Management

* **As a user**, I want to update my profile information (e.g., first name, last name) so that my account details are accurate and up to date.
* **As a user**, I want to change my password so that I can keep my account secure.

#### 3. Expense Tracking

* **As a user**, I want to input my financial transactions by categorizing them as income or spending so that I can keep track of my finances effectively.
* **As a user**, I want to select categories for my transactions (e.g., Food & Drink, Salary, Health Care) so that I can analyze my spending habits.
* **As a user**, I want the application to automatically capture the date and time of each transaction so that I can easily track when my expenses occurred.
* **As a user**, I want to view a history of my last ten transactions on my dashboard so that I can quickly reference my recent activities.

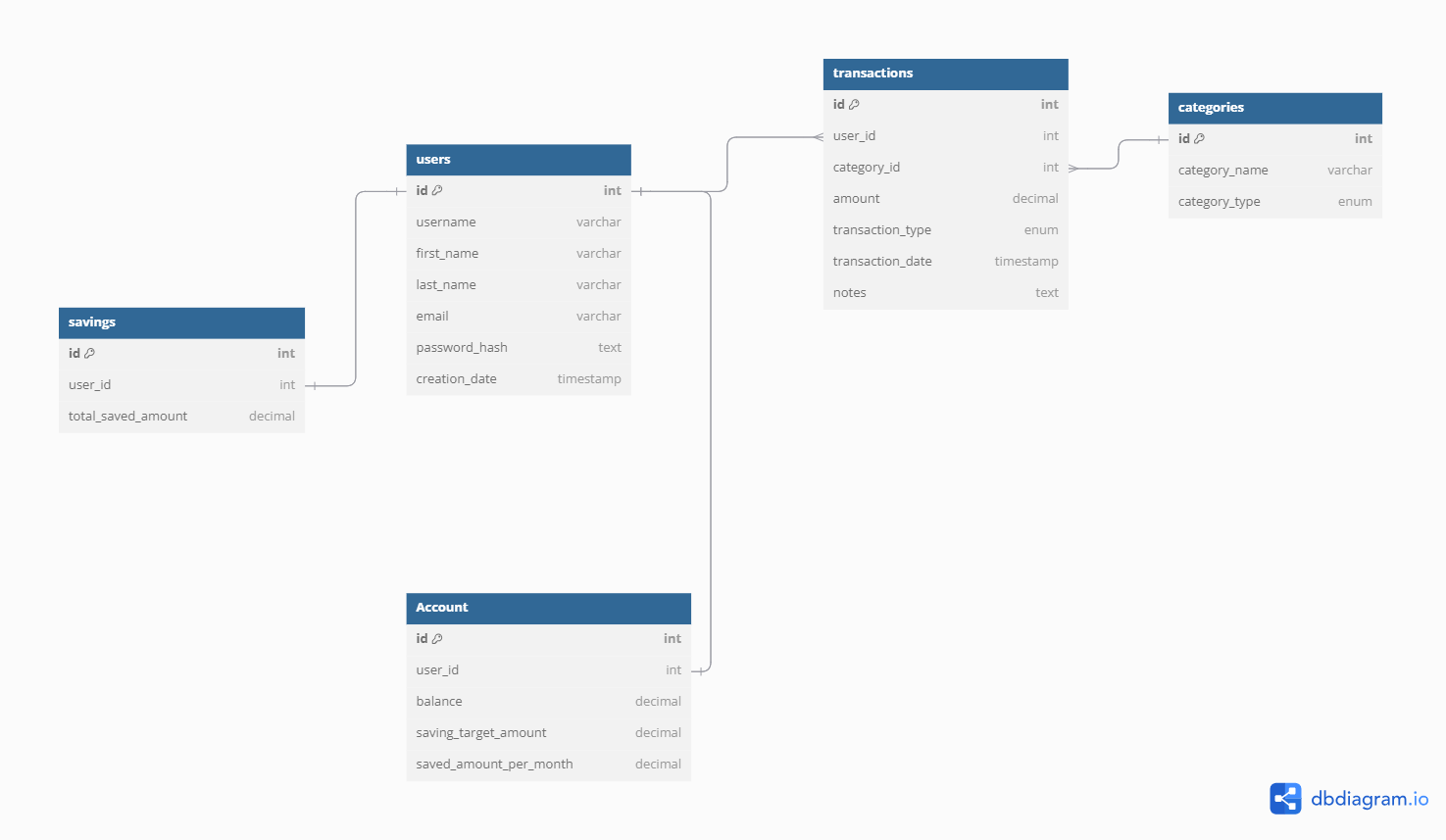
#### 4. Budget Management

* **As a user**, I want to see my savings, target budget, and balance on my dashboard so that I can monitor my financial progress.
* **As a user**, I want the application to show the percentage of my budget achieved so that I can gauge how well I am managing my finances.

#### 5. Goal Setting

* **As a user**, I want to enter my initial balance, saving amount per mounth and financial goals so that I can have a clear plan for my finances.
* **As a user**, I want to edit my saving amount per mounth, and financial goals so that my financial details are accurate and up to date.
* **As a user**, I want the application to deduct my savings monthly from my initial balance automatically until my goal is reached so that I can stay on track without manual effort.

# UML model diagram (ERD)



In this project, I am using a **Relational Database Management System (RDBMS)** with **MSSQL** as the database engine. I chose this system because the application has structured data that fits well into a relational schema, where each entity (such as users, categories, transactions, savings, and accounts) is represented as a table. MSSQL provides efficient query performance, data integrity through foreign keys, and supports ACID (Atomicity, Consistency, Isolation, Durability) transactions, which ensures reliable and consistent operations, particularly for financial applications.

**Tables and Relationships**

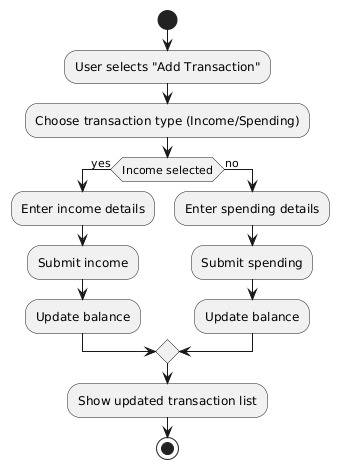
1. **Users**:
   * The **users** table stores the primary information about each user such as their name, email, and password hash.
   * **Primary Key**: id
   * **Relationships**:
     + One-to-One with **accounts** (each user has one unique account).
     + One-to-One with **savings** (each user has one savings record).
     + One-to-Many with **transactions** (each user can have multiple transactions).
2. **Categories**:
   * The **categories** table defines categories of transactions (e.g., 'IN' for income, 'OUT' for expenses).
   * **Primary Key**: id
   * **Relationship**:
     + Many-to-One with **transactions** (each transaction involves one category).
3. **Transactions**:
   * The **transactions** table records all financial transactions, including the amount, category, and type (income or expense).
   * **Primary Key**: id
   * **Relationships**:
     + Many-to-One with **users** (each transaction belongs to a user).
     + Many-to-One with **categories** (each transaction belongs to a category).
4. **Savings**:
   * The **savings** table stores the total saved amount for each user.
   * **Primary Key**: id
   * **Relationship**:
     + One-to-One with **users** (each user has one savings record).
5. **Account**:
   * The **account** table stores detailed financial data such as the current balance and saving target for each user.
   * **Primary Key**: id
   * **Relationships**:
     + One-to-One with **users** (each user has one unique account).

**ERD (Entity-Relationship Diagram)**

In the ERD, the following relationships are represented:

* **One-to-One**: Between users and accounts, and users and savings.
* **One-to-Many**: Between users and transactions, and categories and transactions.

# UML activity diagram



# Design patterns

***1. MVC Design Pattern***

*The web application is structured using the* ***Model-View-Controller (MVC)*** *design pattern, which ensures a separation of concerns and helps in organizing the codebase effectively.*

* ***Model:*** *This layer handles all the data-related tasks.*
* ***View:*** *The view is responsible for rendering the user interface and displaying data to the users.*
* ***Controller:*** *The controller handles user input, processes it, and interacts with the model to retrieve or manipulate data. It then updates the view accordingly.*

***Benefits of MVC:***

* ***Separation of Concerns.***
* ***Improved Maintainability.***
* ***Enhanced Testability.***
* ***Reusability of Components.***

***2. Factory Pattern***

*The* ***Factory Pattern*** *has been implemented to handle the creation of repository objects, which are responsible for data persistence and retrieval from the database. By using this pattern, object creation is abstracted and centralized, promoting code reusability and scalability.*

* ***Repository Factory:*** *Instead of manually instantiating repositories (such as TransactionRepository or UserRepository), the Factory Method is used to encapsulate the creation logic. This makes it easy to manage different data sources or repositories.*

***Benefits of Factory Method Pattern:***

* ***Encapsulation.***
* ***Flexibility.***
* ***Scalability.***

# Code Collaboration

This section should contain a link for your project on github. As well as a screenshot to your sonarqube results.

<https://github.com/ahmadhamarsheh/Walletify.git>

# Testing

* Write manual test cases for one main feature using: <https://app.testomat.io/>.
* Write unit tests for at least one main function.
* Links to your test cases should be presented in this section.

**Tasks schedule**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Feature | Task | Description | Priority | Estimated time | Due date | Assigned to |
| **Authentication** | Registration | … | Highest | 2 hours | 12/28/2022 | Your name |
| Login | … | Highest | 1 hour | 12/28/2022 | Your name |
| **Feature 2** | … | … | … | … | … | …. |

Priority could be one of these:

1. Highest
2. Medium
3. Lowest

Please create free accounts on trello.  
[Manage Your Team’s Projects From Anywhere | Trello](https://trello.com/)