**CEBU INSTITUTE OF TECHNOLOGY**

**UNIVERSITY**

COLLEGE OF COMPUTER STUDIES

Software Requirements Specifications

for

MyAlkansya

Members:

Cartilla, Craig Matthew

Paras, Christian Hans Israel E.

Tigley, Aeron Raye V.

Change History

Table of Contents

Change History 2

Table of Contents 3

1. Introduction 4

1.1. Purpose 4

1.2. Scope 4

1.3. Definitions, Acronyms and Abbreviations 4

1.4. References 4

2. Overall Description 5

2.1. Product perspective 5

2.2. User characteristics 5

2.4. Constraints 5

2.5. Assumptions and dependencies 6

3. Specific Requirements 7

3.1. External interface requirements 7

3.1.1. Hardware interfaces 7

3.1.2. Software interfaces 7

3.1.3. Communications interfaces 7

3.2. Functional requirements 7

Module 1 7

Module 2 8

3.4 Non-functional requirements 8

Performance 8

Security 8

Reliability 8

# Introduction

## Purpose

The purpose of this Software Requirements Specification (SRS) is to outline the functional and non-functional requirements for MyAlkansya, a financial management application. This document serves as a guide for stakeholders, including developers, project managers, and end-users, to understand the capabilities and limitations of the software. 

## Scope

MyAlkansya is designed to assist users in managing their finances through features such as income and expense tracking, budgeting, and data visualization. The application will be available on both web and mobile platforms.

It will not provide investment advice or tax preparation services. The primary objective is to enhance users' financial health by providing tools for effective money management.

## Definitions, Acronyms and Abbreviations

* SRS: Software Requirements Specification
* API: Application Programming Interface
* DBMS: Database Management System

## References

* *Provide a complete list of all documents referenced elsewhere in the SRS;*
* *Identify each document by title, report number (if applicable), date, and publishing organization;*
* *Specify the sources from which the references can be obtained.*

# Overall Description

**System:** MyAlkansya

**MyAlkansya** is an easy-to-use app to help you manage your money. It lets you track your income and expenses, set budgets, and see your financial health with helpful charts. The app offers features such as different login options, currency conversion, and the ability to export your data. Whether it's on the web or mobile. MyAlkansya makes managing your finances simple and efficient.

## Product perspective

* *Put software product into perspective with other related products. If the product is independent and totally self-contained, it should be so stated here. If the SRS defines a product that is a component of a larger system, as frequently occurs, then this subsection should relate the requirements of that larger system to functionality of the software and should identify interfaces between that system and the software.*
* *A block diagram showing the major components of the larger system, interconnections, and external inter- faces can be helpful.*
* *Describe the modular decomposition of the components using the format below:*

*Module 1*

*Transaction 1.1*

*Transaction 1.2*

*Module 2*

*Transaction 2.1*

*Transaction 2.2*

*. . .*

## User characteristics

* *Describe all user types and their roles and privileges in the system*

## 2.4. Constraints

* *Provide a general description of any other items that will limit the developer’s options.*
* *Regulatory policies;*
* *Hardware limitations (e.g., signal timing requirements);*
* *Interfaces to other applications;*
* *Parallel operation;*
* *Audit functions;*
* *Control functions;*
* *Reliability requirements;*
* *Criticality of the application;*
* *Safety and security considerations.*

## 2.5. Assumptions and dependencies

*This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not avail- able, the SRS would then have to change accordingly.*

# Specific Requirements

## External interface requirements

### 3.1.1. Hardware interfaces

*This should specify the logical characteristics of each interface between the software product and the hard- ware components of the system. This includes configuration characteristics (number of ports, instruction sets, etc.). It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.*

### 3.1.2. Software interfaces

*This should specify the use of other required software products (e.g., a data management system, an operating system, or a mathematical package), and interfaces with other application systems (e.g., the linkage between an accounts receivable system and a general ledger system).*

### 3.1.3. Communications interfaces

*This should specify the various interfaces to communications such as local network protocols, etc.*

## Functional requirements

### Function 1

#### 1.1 **User Authentication (Web)**

* **Manual Login:** Users can register and log in using their email and password.
* **Social Media Login:** Options for users to register and log in using Google and Facebook accounts for ease of access.

#### 1.2 **User Authentication (Mobile)**

* **Manual Login:** Users can register and log in using their email and password.
* **Social Media Login:** Options for users to register and log in using Google and Facebook accounts for ease of access.
* **Biometrics Login:** Use fingerprint or face ID to login.

##### . . .

### Function 2

#### 2.1 **Dashboard (Web)**

##### A user-friendly dashboard that provides an overview of financial health, including total income, total expenses, and remaining budget.

#### 2.2 **Dashboard (Mobile)**

##### A user-friendly dashboard that provides an overview of financial health, including total income, total expenses, and remaining budget.

### Function 3

#### 3.1 **Expense Tracking (Web)**

* Users can add, edit, and delete expenses, categorizing them into predefined categories (e.g., Food, Transportation, Entertainment).

Each expense entry includes:

* + Amount
  + Category
  + Date
  + Notes

#### 3.2 **Expense Tracking (Mobile)**

* Users can add, edit, and delete expenses, categorizing them into predefined categories (e.g., Food, Transportation, Entertainment).

Each expense entry includes:

* + Amount
  + Category
  + Date
  + Notes

### Function 4

#### 4.1 **Income Tracking (Web)**

Users can add, edit, and delete income sources with details such as:

* Amount
* Source (e.g., Salary, Freelance)
* Date

#### 4.2 **Income Tracking (Mobile)**

Users can add, edit, and delete income sources with details such as:

* Amount
* Source (e.g., Salary, Freelance)
* Date

### Function 5

#### 5.1 **Budgeting (Web)**

* Users can set monthly budgets for different categories and receive alerts when they exceed their budget.

#### 5.2 **Budgeting (Mobile)**

* Users can set monthly budgets for different categories and receive alerts when they exceed their budget.

### Function 6

#### 6.1 **Currency Converter (Web)**

* Users can select their preferred currency, and all transactions will be recorded in that currency.
* Integrate **ExchangeRate-API** for real-time currency conversion to facilitate expense tracking in multiple currencies.

#### 6.2 **Currency Converter (Mobile)**

* Users can select their preferred currency, and all transactions will be recorded in that currency.
* Integrate **ExchangeRate-API** for real-time currency conversion to facilitate expense tracking in multiple currencies.

### Function 7

#### 7.1 **Data Analytics with Chart.js (Web)**

* Visualize spending patterns and income sources using interactive charts.
* Monthly reports summarizing income and expenses, allowing users to analyze their financial trends easily.

#### 7.2 **Data Analytics with Chart.js (Mobile)**

* Visualize spending patterns and income sources using interactive charts.
* Monthly reports summarizing income and expenses, allowing users to analyze their financial trends easily.

### Function 8

#### 8.1 **Savings Goals (Web)**

* Users can set savings goals (e.g., saving for a vacation) and track their progress visually.

#### 8.2 **Savings Goals (Mobile)**

* Users can set savings goals (e.g., saving for a vacation) and track their progress visually.

### Function 9

#### 9.1 **Export to Google Sheets (Web)**

* Users can export their monthly summary of expenses and income directly to Google Sheets for further analysis.
* Utilize the **Google Sheets API** to facilitate this feature.

#### 9.2 **Export to Google Sheets (Mobile)**

* Users can export their monthly summary of expenses and income directly to Google Sheets for further analysis.
* Utilize the **Google Sheets API** to facilitate this feature.

### Function 10

#### 10.1 **Data Insights and Tips (Web)**

* Provide personalized spending insights and tips based on user data to help users make informed financial decisions.

#### 10.2 **Data Insights and Tips (Mobile)**

* Provide personalized spending insights and tips based on user data to help users make informed financial decisions.

#### Non-functional requirements

### Performance

##### Details

### Security

##### Details

### Reliability

##### Details

SRS Suggestions Cartilla (1/28/2025) check lang

**As per Project Requirements.**

### ****1. System Features****

Expand and refine the features to meet the requirement of at least six distinct functionalities:

1. **User Authentication** (if not already included):
   * Extend with multi-factor authentication for enhanced security.
2. **Dashboard**:
   * Add customizable widgets to let users tailor their dashboard view.
3. **Expense and Income Tracking**:
   * Integrate OCR (Optical Character Recognition) to auto-scan receipts for expense input.
4. **Budgeting**:
   * Introduce recurring budget templates for common categories (e.g., rent, groceries).
5. **Data Visualization**:
   * Implement advanced filters (e.g., date range, category) for deeper insights.
6. **Mobile-Specific Feature**:
   * Add push notifications for budget alerts or expense reminders.

### ****2. External Integrations****

To demonstrate system integration:

* **ExchangeRate-API**: As specified, use for real-time currency conversion.
* **Google Sheets API**: Enable exporting data to Google Sheets. (need padaw ta more research ani)
* **Additional Integration**: Include a payment gateway (e.g., Stripe or PayPal) for premium features like advanced analytics or ad-free experience.

### ****3. Mobile Application Enhancements****

Build the mobile app with **Kotlin** to complement the web app:

* **Biometric Login**: Use Android’s Fingerprint and FaceID APIs. (suggestion ni sir)
* **Offline Mode**: Allow data entry when offline, syncing with the database when reconnected.
* **Custom Notifications**: Offer insights like "You’ve spent 80% of your dining budget this month."

### ****4. Database (pili lng sa ta which)****

For external database integration, **MySQL on AWS RDS** is ideal for scalability and reliability. Create:

* A **normalized schema** for income, expense, and user data.
* Optimized **queries and indexing** for faster analytics.

### ****Or if Firebase:****

* **Firestore**: Use Cloud Firestore for structured, scalable data storage.
* **Firebase Authentication**: Simplify user login with built-in support for email, Google, and Facebook authentication.
* **Firebase Cloud Messaging**: Enable push notifications for budget alerts or reminders.
* **Firebase Hosting**: Deploy your web app easily with global content delivery.

Note: The database should support both the web and mobile applications.

### ****5. Technology Stack****

* **Backend**: Java Spring Boot with Maven.
* **Frontend**: React.js for web, using TailwindCSS for styling and Redux for state management.
* **Mobile**: Kotlin with Jetpack Compose for modern UI.
* **Database**: MySQL on AWS RDS.

### ****6. Functional Requirements Adjustments****

Enhance the SRS document:

* Ato addan og **error handling** requirements, such as:
  + Handling invalid inputs in financial fields (e.g., negative numbers).
  + Managing conflicts during multi-user edits.
* Include **security requirements**:
  + Encrypt sensitive data (e.g., bcrypt for passwords).
  + Use HTTPS and secure tokens for API communication.