# Software Requirements Specification

# Expense T**racker Application**

**Members:**

**Muhammad Mubashir**

**Muhammad Azeem Sikandar**

**Saad Ikram**

**Submitted to:**

**Dr Nabeel Ahmad**

## **Introduction**

## 1.1 Purpose

This document outlines the software requirements specification for an Expense Tracker application, a desktop-based system designed to help users manage and track their personal expenses across multiple accounts.

1.2 Scope

The Expense Tracker application is a Java-based desktop application that allows users to:

- Manage multiple accounts

- Record and track expenses

- Categorize expenses

- View expense summaries

- Export expense data

### 1.3 Definitions and Acronyms

- GUI: Graphical User Interface

- CSV: Comma-Separated Values

- SRS: Software Requirements Specification

## Overall Description

## 2.1 Product Perspective

The Expense Tracker is a standalone desktop application built using Java Swing framework. It provides a user-friendly interface for personal expense management.

### 2.2 Product Features

- Account management

- Expense entry and tracking

- Expense categorization

- Expense summary viewing

- Data export capability

- Expense deletion

### 2.3 User Classes and Characteristics

- Primary Users: Individuals wanting to track personal expenses

- Secondary Users: Small business owners or professionals managing business expenses

### 2.4 Operating Environment

- Platform: Desktop (Windows, macOS, Linux)

- Java Runtime Environment (JRE) 8 or higher

- Minimum display resolution: 600x400 pixels

## 3. System Features

### 3.1 Account Management

#### 3.1.1 Description

Users can create and manage multiple accounts for tracking expenses.

#### 3.1.2 Functional Requirements

- REQ-1: Users can add new accounts

- REQ-2: Users can select existing accounts

- REQ-3: Account names must be unique

- REQ-4: Account names cannot be empty

## 3.2 Expense Management

### 3.2.1 Description

Users can add, view, and delete expense entries.

### 3.2.2 Functional Requirements

- REQ-5: Users can add new expenses with amount, description, and category

- REQ-6: System automatically records date for new expenses

- REQ-7: Users can delete existing expenses

- REQ-8: System displays expenses in a tabular format

- REQ-9: System calculates and displays total expenses

### 3.3 Expense Categorization

#### 3.3.1 Description

Users can categorize expenses using predefined categories.

#### 3.3.2 Functional Requirements

- REQ-10: System provides predefined categories (Food, Transport, Utilities, Other)

- REQ-11: Users must select a category for each expense

## 3.4 Data Export

### 3.4.1 Description

Users can export expense data to CSV format.

### 3.4.2 Functional Requirements

- REQ-12: System allows export of expense data to CSV format

- REQ-13: Exported data includes date, description, amount, and category

## 4. External Interface Requirements

### 4.1 User Interface

#### 4.1.1 Main Window

- Minimum size: 600x400 pixels

- Three main sections:

- Account Management Panel

- Expense Input Panel

- Expense Table Panel

#### 4.1.2 Interface Elements

- Account Management:

- Account selection dropdown

- New account input field

- Add account button

- Expense Input:

- Amount input field

- Description input field

- Category dropdown

- Add expense button

- Expense Table:

- Scrollable table with columns: Date, Description, Amount, Category

- Total expense display

- Export and Delete buttons

## 4.2 Software Interfaces

- Java Swing Framework

- Java AWT Framework

- System Look and Feel

## 5. Non-functional Requirements

### 5.1 Performance

- Response time for adding/deleting expenses: < 1 second

- Table refresh time: < 0.5 seconds

- Maximum supported expenses: 10,000 entries

### 5.2 Security

- Data validation for all input fields

- Read-only table cells to prevent direct editing

### 5.3 Usability

- Intuitive GUI layout

- Clear error messages

- Confirmation for delete operations

- Responsive design

### 5.4 Reliability

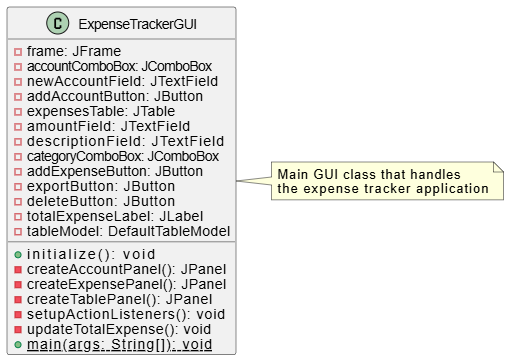
- Auto-calculation of total expenses

- Input validation for amount fields

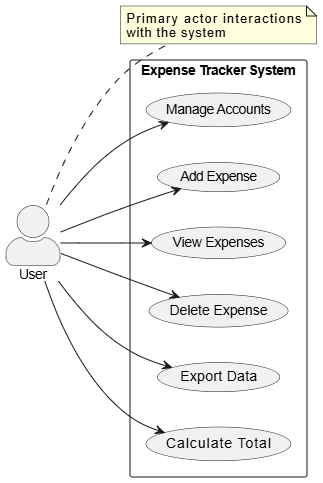
- Error handling for invalid inputs

# 6. UML Diagrams

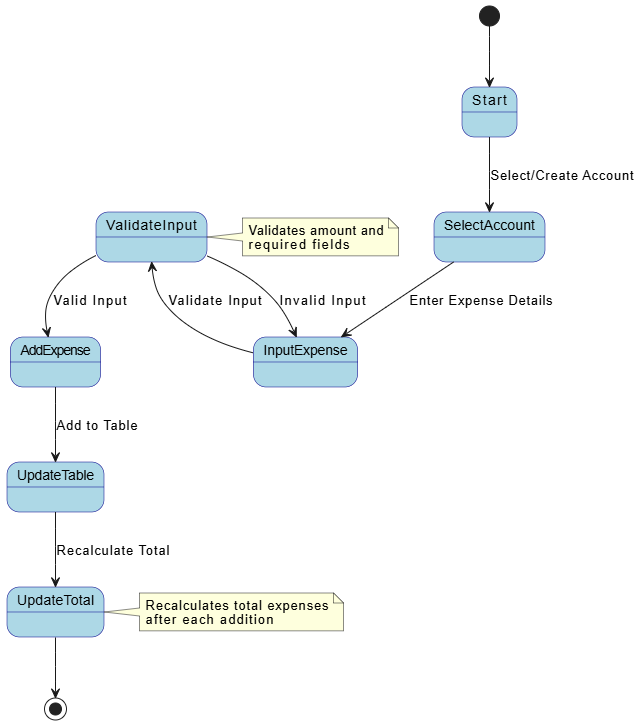
### 6.1 Class Diagram



## 6.3 Use Case Diagram



## 6.2 Activity Diagram



### 7. Appendices

### 7.1 Assumptions and Dependencies

- Java Runtime Environment availability

- System Look and Feel support

- Adequate system resources

## 7.2 Future Enhancements

- Data persistence using database

- Multiple currency support

- Budget tracking

- Expense analytics and reporting

- Custom categories

- Data backup and restore