



Imam Abdulrahman Bin Faisal University  
College of Computer Science & Information Technology  
Department of Computer Science

**CS 411 – Software Engineering**  
**Term 1 – 2024/2025**

# **Final Report**

**For**

# **Expense Tracker**

**Version [1.0]**

**2024-12-7**

This final report was prepared and provided as a deliverable for Software Engineering, CS 411, Term 1, and it will be used by the Expense Tracker team.

Table of Content

Revision History ..... 7

Chapter 1: Introduction..... 12

    1.1 Purpose..... 13

    1.2 Project Overview..... 13

    1.3 Report Structure..... 13

2. .... 14

Chapter 2: SPMP ..... 14

2.1 Project Overview..... 15

    2.1.1 Purpose, Scope and Objectives..... 15

    2.1.2 Assumptions, Constraints and Risks..... 16

    2.1.3 Project Deliverables ..... 16

    2.1.4 Schedule and Budget Summary ..... 18

    2.1.5 Evolution of the Plan ..... 18

    2.1.6 References..... 18

    2.1.7 Definitions and Acronyms..... 19

    2.1.8 Document Structure..... 19

2.2 Project Organization ..... 20

    2.2.1 External Interfaces..... 20

    2.2.2 Internal Structure ..... 20

    2.2.3 Roles and Responsibilities ..... 22

2.3 Managerial Process Plans..... 23

    2.3.1 Start-up Plan ..... 23

    2.3.2 Work Plan ..... 24

    2.3.3 Project Tracking Plan ..... 28

    2.3.4 Risk Management Plan ..... 29

    2.3.5 Project Closeout Plan..... 31

2.4 Technical Process..... 31

    2.4.1 Process Model..... 31

    2.4.2 Methods, Tools, and Techniques..... 32

    2.4.3 Infrastructure ..... 33

    2.4.4 Product Acceptance ..... 33

2.5 Supporting Process Plans ..... 33

    2.5.1 Documentation..... 33

2.6	Additional Plans .....	34
3.	.....	35
	Chapter 3: SRS .....	35
3.1	Introduction.....	36
3.1.1	Purpose .....	36
3.1.2	Scope.....	36
3.1.3	Definitions, Acronyms, and Abbreviations .....	36
3.1.4	References.....	37
3.2	Overall description .....	38
3.2.1	Product perspective.....	38
3.2.2	Product functions.....	38
3.3	Specific Requirements .....	50
3.3.1	External interface requirements .....	50
4.	.....	58
	Chapter 4: SDS .....	59
4.1	Introduction.....	59
4.1.1	Purpose .....	59
4.1.2	Scope.....	60
4.1.3	Definitions, Acronyms, and Abbreviations .....	60
4.1.4	References.....	61
4.2	System overview .....	62
4.2.1	Administrator Skills .....	62
4.2.2	User Skills.....	62
4.2.3	System Functions.....	62
4.3	Design Considerations .....	63
4.3.1	Assumptions and Dependencies .....	63
4.3.1.2	Operating System.....	63
4.3.2	General Constraints .....	63
4.3.2.1	Hardware/Software Environment .....	63
4.3.2.3	Repository and Distribution Needs.....	64
4.4	User Interface Design.....	65
4.4.1	Overview of User Interface .....	65
4.4.2	Interface Design Rules.....	65
4.4.3	Screen Images .....	66
4.4.4	Screen Objects and Actions.....	71

4.5	System Architecture.....	78
4.5.1	Architectural Design Approach .....	78
4.5.2	Architectural Design .....	79
4.5.3	Subsystem Architecture .....	80
4.6	Data Design.....	87
4.6.1	Data Description .....	87
4.6.2	Data Dictionary.....	89
4.6.3	Database Description .....	91
4.7	Component Design.....	94
4.7.1	Common Functions.....	94
4.7.2	Admin Functions .....	95
4.7.3	User Functions .....	96
4.8	Detailed System Design.....	100
4.8.1	Classification .....	100
4.8.2	Definition.....	101
4.8.3	Responsibilities .....	102
4.8.4	Constraints .....	103
4.8.5	Composition.....	104
4.8.6	Uses/Interactions.....	105
4.8.7	Resources .....	117
4.8.8	Processing .....	118
4.8.9	Interface/Exports.....	123
4.8.10	Detailed Subsystem Design .....	123
4.9	Other Design Features.....	133
4.10	Requirements Traceability Matrix .....	133
5.	.....	135
Chapter 5:	STS.....	135
5.1	Introduction.....	136
5.1.1	Objectives .....	136
5.1.2	Testing Strategy .....	136
5.1.3	Scope.....	136
5.1.4	Reference Material.....	136
5.1.5	Definitions and Acronyms.....	137
5.2	Test Items.....	138

5.2.1	Program Modules.....	138
5.2.2	Job Control Procedures.....	138
5.2.3	User Procedures .....	138
5.2.4	Operator Procedures .....	138
5.3	Features To Be Tested.....	139
5.4	Features Not To Be Tested.....	139
5.5	Approach.....	139
5.5.1	Component Testing.....	139
5.5.2	Integration Testing.....	146
5.5.3	Conversion Testing .....	148
5.5.4	Job Stream Testing .....	148
5.5.5	Interface Testing .....	148
5.5.6	Security Testing .....	149
5.5.7	Recovery Testing .....	149
5.5.8	Performance Testing .....	150
5.5.9	Regression Testing.....	150
5.5.10	Acceptance Testing.....	150
5.5.11	Beta Testing .....	150
5.6	Pass / Fail Criteria.....	151
5.6.1	Suspension Criteria .....	151
5.6.2	Resumption Criteria .....	151
5.6.3	Approval Criteria .....	151
5.7	Testing Process .....	151
5.7.1	Test Deliverables .....	152
5.7.2	Testing Tasks .....	152
5.7.3	Responsibilities .....	152
5.7.4	Resources .....	152
5.7.5	Schedule.....	153
5.8	Environmental Requirements.....	154
5.8.1	Hardware.....	154
5.8.2	Software.....	154
5.8.3	Security .....	154
5.8.4	Tools .....	154

5.8.5 Publications..... 154

5.8.6 Risks and Assumptions..... 155

5.9 Change Management Procedures..... 155

5.10 Plan Approvals ..... 156

Appendix A: Status Report 1..... 157

Appendix B: Status Report 2..... 160

Appendix C: Comments And Changes..... 163

Appendix D: Samples of Code ..... 165

Revision History

Name	Date	Reason For Changes	Version
All members	Nov 28, 2024	Prepared initial version	0.1
All members	Nov 30, 2024	Added Introduction	0.2
All members	Dec 1, 2024	Added SPMP	0.3
All members	Dec 2, 2024	Added SRS	0.4
All members	Dec 3, 2024	Added SDS	0.6
All members	Dec 5, 2024	Added STS	0.7
All members	Dec 6, 2024	Added Appendices	0.8
All members	Dec 7, 2024	Reviewed and cleaned up file	1.0

Table 1 Revision History

Table of Tables

Table 1 Team Members ..... **Error! Bookmark not defined.**

Table 2 Revision History ..... 7

Table 3 Assumptions, Constraints and Risks ..... 16

Table 4 Project Deliverables..... 17

Table 5 Schedule and Budget Summary ..... 18

Table 6 List of Terminologies. .... 19

Table 7 Roles and Responsibilities ..... 23

Table 8 Project Phases ..... 24

Table 9 Staff Responsibilities. .... 24

Table 10 : Work Breakdown Structure..... 25

Table 11 Resource Allocation. .... 27

Table 12 Project Metrics.....	29
Table 13 Risk Management Plan. ....	30
Table 14 Project Deliverables.....	31
Table 15 Methods, Tools, and techniques. ....	32
Table 16 Infrastructure. ....	33
Table 17 Documentation.....	34
Table 18 Terminologies .....	37
Table 19 Acronyms.....	37
Table 20 Use Case Description Summary .....	43
Table 21 Use Case: Register .....	46
Table 22 Use Case: Login.....	47
Table 23 Use Case: Recover Password .....	47
Table 24 Use Case: Log Expenses.....	47
Table 25 Use Case: Set Budgets.....	47
Table 26 Use Case: Export Data.....	48
Table 27 Use Case: View Notifications.....	48
Table 28 Use Case: View Reports .....	48
Table 29 Use Case: Link Bank Account.....	48
Table 30 Use Case: Backup and Restore Data .....	49
Table 31 Analyze Expense Trends .....	49
Table 32 Manage User Accounts.....	49
Table 33 View/Edit/Delete Transactions .....	49
Table 34 Generate Financial Summaries .....	50
Table 35 Notify Users.....	50
Table 36 Start-up selection Components. ....	50
Table 37 Registration Interface Components. ....	51
Table 38 Login interface components. ....	52
Table 39 Expense/Budget Selection interface components.....	52
Table 40 Adding Expenses interface Components.....	53
Table 41 User's Add Funds interface Components.....	53
Table 42 Admin Financial management interface Components.....	55
Table 43 : Analysis expense interface components. ....	56
Table 44 Account Linking Interface Components.....	57
Table 45 Password Recovery Interface Components.....	57
Table 46 Terminology .....	60
Table 47 Acronyms.....	61
Table 48: Data Base Description Table. ....	89
Table 49 Data Dictionary Table. ....	90
Table 50: Components Classification.....	100
Table 51: Components Definitions. ....	101
Table 52: Components Responsibilities. ....	102
Table 53: Components Constraints. ....	103



<i>Table 54: Components Composition.</i>	105
<i>Table 55: External Resources</i>	117
<i>Table 56: Deadlock Situations</i>	117
<i>Table 57: Login Processing Table.</i>	118
<i>Table 58: Forgot Password Processing Table.</i>	118
<i>Table 59: Change Password Processing Table.</i>	119
<i>Table 60: Delete Transaction Processing Table.</i>	119
<i>Table 61: View User Transactions Processing Table.</i>	120
<i>Table 62: View Reports Processing Table.</i>	120
<i>Table 63: Manage User Accounts Processing Table.</i>	120
<i>Table 64: Register Processing Table.</i>	121
<i>Table 65: Add Expense Processing Table.</i>	121
<i>Table 66: Add Funds Processing Table.</i>	122
<i>Table 67: View Expense Analysis Processing Table.</i>	122
<i>Table 68: Link Bank Account Processing Table.</i>	123
<i>Table 69 Requirements Traceability Matrix</i>	134
<i>Table 70 Terminology Table</i>	137
<i>Table 71 Acronyms Table</i>	137
<i>Table 72 "Login" Table</i>	140
<i>Table 73 "Forgot Password" Table</i>	140
<i>Table 74 "Change Password" Table</i>	141
<i>Table 75 "Delete Transaction" Table</i>	141
<i>Table 76 "View User Transactions" Table</i>	142
<i>Table 77 "View Reports" Table</i>	142
<i>Table 78 "Manage User Accounts" Table</i>	143
<i>Table 79 "Register" Table</i>	144
<i>Table 80 "Add Expense" Table</i>	144
<i>Table 81 "Add Funds" Table</i>	145
<i>Table 82 "View Expense Analysis" Table</i>	145
<i>Table 83 "Link Bank Account" Table</i>	146
<i>Table 84 Admin Table</i>	147
<i>Table 85 User Table</i>	148
<i>Table 86 Interface Testing Table</i>	149
<i>Table 87 Resources Table</i>	152
<i>Table 88 Schedule Table</i>	153
<i>Table 89 Plan Approvals</i>	156
<i>Table 90: Changes Table.</i>	164

## Table of Figures

Figure 1 External Interface .....	20
Figure 2 Internal Structure.....	21
Figure 3 Project's Organizational Structure. ....	21
Figure 4 Schedule Allocation. ....	25
Figure 5 The five principles of Waterfall Model.....	32
Figure 6 Use Case Diagram: Admin System .....	40
Figure 7 Use Case Diagram: User System .....	41
Figure 8 Start-up Selection Interface.....	66
Figure 9 Registration interface .....	66
Figure 10 Login interface .....	67
Figure 11 Password recovery interface.....	67
Figure 12 Selection interface .....	68
Figure 13 Adding expense interface .....	69
Figure 14 Add funds interface.....	70
Figure 15 bank account linking interface .....	70
Figure 16 Analysis expenses interface .....	71
Figure 17 System Diagram .....	79
Figure 18 User Management Subsystem DFD .....	80
Figure 19 Expense Management Subsystem DFD .....	81
Figure 20 Budget Management Subsystem DFD .....	82
Figure 21 Reporting and Analysis Subsystem DFD .....	83
Figure 22 Notifications Subsystem DFD.....	84
Figure 23 Data Backup & Restore Subsystem DFD .....	85
Figure 24 Export Data Subsystem DFD .....	86
Figure 25: Class Diagram .....	91
Figure 26: Entity Relationship Diagram. ....	91
Figure 27: Relational Schema diagram. ....	93
Figure 28: Login Sequence Diagram .....	105
Figure 29: Forgot Password Sequence Diagram .....	106
Figure 30: Change Password Sequence Diagram .....	107
Figure 31: Delete Transaction Sequence Diagram .....	108
Figure 32: View User Transaction Sequence Diagram .....	109
Figure 33: View Reports Sequence Diagram .....	110
Figure 34: Manage User Accounts Sequence Diagram .....	111
Figure 35: Register Sequence Diagram .....	112
Figure 36: Add Expense Sequence Diagram .....	113

<i>Figure 37: Add Funds Sequence Diagram</i> .....	114
<i>Figure 38: View Expense Analysis Sequence Diagram</i> .....	115
<i>Figure 39: Link Bank Account Sequence Diagram</i> .....	116
Figure 40: Login Activity Diagram. ....	124
Figure 41: Forgot Password Activity Diagram.....	124
Figure 42: Change Password Activity Diagram .....	125
Figure 43: Delete transaction Activity Diagram. ....	126
Figure 44: View User Transactions Activity Diagram. ....	126
<i>Figure 45: View Reports Activity Diagram.</i> .....	127
Figure 46: Manage User Accounts Activity Diagram. ....	128
Figure 47: Register Activity Diagram. ....	129
Figure 48: Add Expense Activity Diagram .....	130
Figure 49 Add Funds Activity Diagram .....	131
Figure 50:View Expense Analysis Activity Diagram. ....	131
Figure 51: Link Bank Account Activity Diagram. ....	132
Figure 52: Password Recovery Algorithm. ....	166
Figure 53: Linking To banks. ....	167
Figure 54: Expense Analysis. ....	168
Figure 55:Managing User Accounts. ....	169

# Chapter 1: Introduction

## 1.1 Purpose

The purpose of this report is to document the development process and outcomes of the Expense Tracker project. This report outlines all stages of the project, from planning and requirements gathering to design, and testing. It serves as a comprehensive record of the project's progress and deliverables, highlighting the methodologies and strategies used to meet the specified goals.

## 1.2 Project Overview

The Expense Tracker application was designed to address the growing need for personal financial management tools. It enables users to efficiently track their expenses, set and monitor budgets, and analyze their spending patterns. The application incorporates essential features such as data visualization, notifications, and secure backup and restoration options.

Objectives:

- To develop a simple and intuitive tool for managing personal finances.
- To ensure the system provides reliable and accurate data management.
- To incorporate advanced features such as data export and reporting for better financial insights.

Problem Statement: Managing personal finances effectively can be challenging, especially without the right tools. Many existing solutions are either too complex or lack essential features tailored for individual users. The Expense Tracker bridges this gap by offering an accessible and user-focused solution for managing expenses and budgets.

## 1.3 Report Structure

This report is organized into the following chapters:

1. **Introduction:** Provides an overview of the project, its objectives, and the structure of the report.
2. **Software Project Management Plan (SPMP):** Details the planning, resource allocation, and risk management strategies used throughout the project.
3. **Software Requirements Specification (SRS):** Defines the functional and non-functional requirements of the system, including the features and constraints.
4. **Software Design Specification (SDS):** Describes the architectural design and subsystem components of the application.
5. **Software Test Specification (STS):** Outlines the testing methodology, pass/fail criteria, and results of the testing phase.
6. **Appendices:** Includes supplementary materials.

2.

## Chapter 2: SPMP

## 2.1 Project Overview

This portion of the paper gives a summary of the goals, assumptions, and deliverables of the project as well as the project's schedule and budget and provides an overview of the scope and objectives of Expense Tracker for which has been planned.

### 2.1.1 Purpose, Scope and Objectives

An Expense Tracker is software tool or a mobile application that keeps track of all your income and costs so you can fully understand your spending plan. by assisting consumers in automatically tracking, classifying, and evaluating their purchasing patterns. This solution provides a clear understanding of financial statistics and contains a plan that outlines a person's short- and long-term financial objectives along with a method for reaching them.

The purpose is to keep track of your spending, you may set spending priorities, monitor your progress, and determine what adjustments are necessary to reach the goals you want and build the financial future you desire. It also allows you to save emergency funds, which serve as a safety net. You'll become more cautious because of tracking your spending, which will show you just how much money you're spending. From there, you may pinpoint the issues and eliminate wasteful spending. You won't be as concerned about paying your bills and other costs if you feel confident about your financial situation. This makes you feel less stressed and anxious and improves your attitude on life.

The main objectives are:

- Designing and Implementing a User-Friendly Interface
- Make it intuitive and easy-to-navigate to easily allow the users to input, view, and manage their expenses.
- Expense Tracking and Categorization
- Features to allow users to categorize their expenses such as food, transport, entertainment etc...
- Budget Management
- Allow users to set and manage their budgets for different categories.
- Data Security and Privacy
- Make sure that the user's data is securely stored and protected in a database.
- Reporting and Analytics
- Add features that allows data visualization by generating details graphs and charts.
- Integration with Financial Institutions
- Integrate the app with banks and financial institutions APIs to automatically import and categorize transactions.

### 2.1.2 Assumptions, Constraints and Risks

Type	Description
Assumptions	<ul style="list-style-type: none"> <li>- Every team member contributes the same amount of effort.</li> <li>- Regular and frequent contact across team members.</li> <li>- Complete the project on schedule.</li> <li>- Under our project leader's supervision, the project will be presented without any errors.</li> </ul>
Constraints	<ul style="list-style-type: none"> <li>- Not getting to see the team members in person.</li> <li>- Online meetings with members have a time restriction.</li> <li>- The meeting may be disrupted by the internet connection.</li> <li>- There are not enough tools.</li> </ul>
Risks	<ul style="list-style-type: none"> <li>- Ineffective communication amongst team members may cause tasks to overlap.</li> <li>-A reduction in quality.</li> <li>-Integrated software tools might not cooperate -Different preferences for task completion.</li> </ul>

Table 2 Assumptions, Constraints and Risks

### 2.1.3 Project Deliverables

Deliverables	Date	Delivery method
Defining Project	(25 Aug. – 29 Aug.)	Softcopy
Project proposal	(8 Sep. – 12 Sep.)	Softcopy
Project Management Plan (SPMP)	(06 Oct. – 10 Oct.)	Softcopy



Project Status Report	(13 Oct. – 17 Oct.)	Softcopy
Project Requirements (SRS)	(27 Oct. – 31 Oct.)	Softcopy
Project Design (SDS)	(03 Nov. – 07 Nov.)	Softcopy
Submit Project Test Plan (STS)	(24 Nov. – 28 Nov.)	Softcopy
Project Presentation	(15 Dec. – 19 Dec.)	Softcopy

*Table 3 Project Deliverables*

### 2.1.4 Schedule and Budget Summary

December 2024 will mark the submission of the system and presentation. An approximate of 6500 SAR will be needed for the system's development. The project phases and their respective durations are included in the timetable that is summarized in the following table:

Tasks	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Duration	Dependencies
T1	✓	✓															2 weeks	
T2				✓													2 weeks	T1, T2 (M1)
T3					✓	✓	✓										3 weeks	T3(M2)
T4								✓									1 week	
T5									✓	✓							2 weeks	T4, T5(M3)
T6											✓						1 week	T6, (M4)
T7												✓	✓				1 week	T7, (M5)
T8														✓	✓	✓	3 weeks	T8, (M6)

Table 4 Schedule and Budget Summary

### 2.1.5 Evolution of the Plan

This is the initial version of the Expense Tracker Software Project Management Plan (SPMP). Our instructor, Nehad M. Ibrahim must first authorize any changes before updating draft and post it to the Team Cloud (Shared OneDrive folder).

### 2.1.6 References

- [1] IEEE Standard for Software Project Management Plan. [Online] 1998.IEEE Std 1058-1998.
- [2] Somerville, Ian, "Software Engineering", Addison Wesley (10th edition): 2016, ISBN:10:1- 292-09613-6, ISBN-13:978-1-292-09613-1.

### 2.1.7 Definitions and Acronyms

Terminology	Definition
IEEE Standard	A universal standard template that multiple individuals and organizations of different technical origins use to document.
SPMP	A detailed plan that describes the procedures, standards and project management tools that must be achieved by the team members.
SRS	A document that describes the software system to be developed and defining functional and non- functional requirements.
SDS	A document that specifies all data, architectural, interface and component-level design for the software.
STS	A document that explains the scope, approach, resources, and the intended test activities. It used to verify the system meets its design specifications and requirements.

*Table 5 List of Terminologies.*

### 2.1.8 Document Structure

There are six primary sections in this paper, which adheres to IEEE standards. The project overview is covered in the first section. It includes an overview of the project's goals, assumptions, and deliverables, as well as a summary of the budget and schedule, the development of the plan, and any definitions used throughout. The project organization portion is the second one. It includes the roles and responsibilities of each team member as well as internal and external interfaces. The managerial process plan is covered in the third part. It features a start-up strategy that covers staff members, project staff training, and estimating.

Furthermore, included in the third element is the work plan, which includes a budget, timeline, resource, and job breakdown structure allocation. It additionally includes a project tracking plan that includes risk management and project closeout plans in addition to requirement management, schedule control, quality control, reporting, and project metrics. The technical process plan is covered in section four. It includes infrastructure, procedures, tools, methodologies, and product acceptability in addition to the modal process. The supporting process plan, which contains the documentation, is the fifth section. Additional plans make up the sixth and final portion.

## 2.2 Project Organization

### 2.2.1 External Interfaces

The course instructor acts as the parent organization, providing guidance and evaluation for the project. External clients, including individuals, businesses, and organizations, will use the Expense Tracker to manage their finances. While no subcontracted organizations are involved, feedback from clients and the instructor will help improve the application to meet both academic and practical goals.

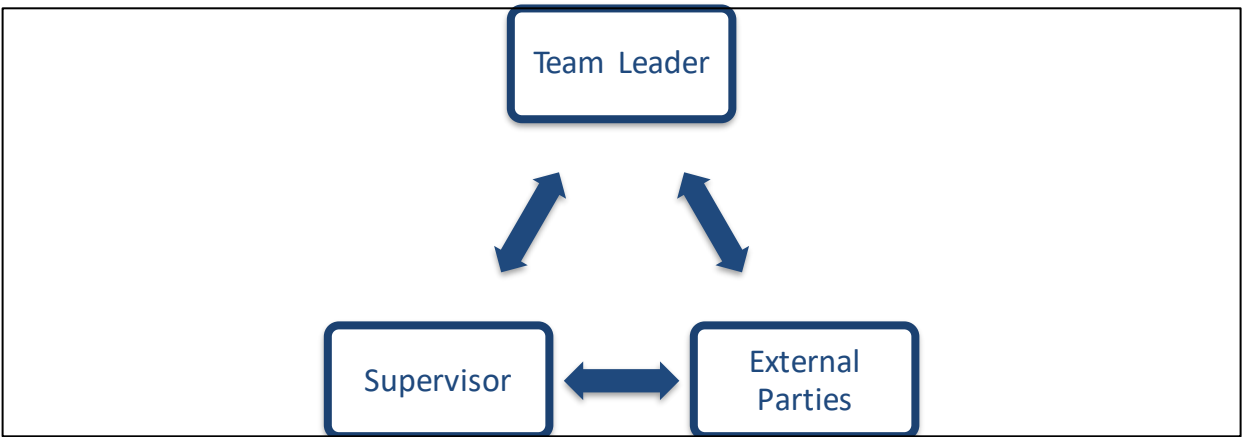


Figure 1 External Interface

### 2.2.2 Internal Structure

The Expense Tracker project will be supervised by Dr. Nehad Ibrahim, with the implementation carried out by the team members: Fawaz Altahini (Leader), Ali Al Hushayyish, Sultan Almutairi, Osama Almutairi, and Abdulrahman Alqahtani. Fawaz Altahini will direct the project, ensuring that all deliverables are completed successfully. The project follows a hierarchical structure, as illustrated in the accompanying organizational chart. Team members will adhere to established processes and strategies to ensure the project is completed accurately and within the scheduled timeframe. Regular meetings will be held weekly, either online or in person, to discuss the project's phases, address any challenges, and exchange ideas. The diversity of skills within the team is a key factor in achieving success, with each member assigned tasks based on their strengths and expertise. This balanced distribution of responsibilities will enhance productivity and contribute to the overall success of the Expense Tracker project.

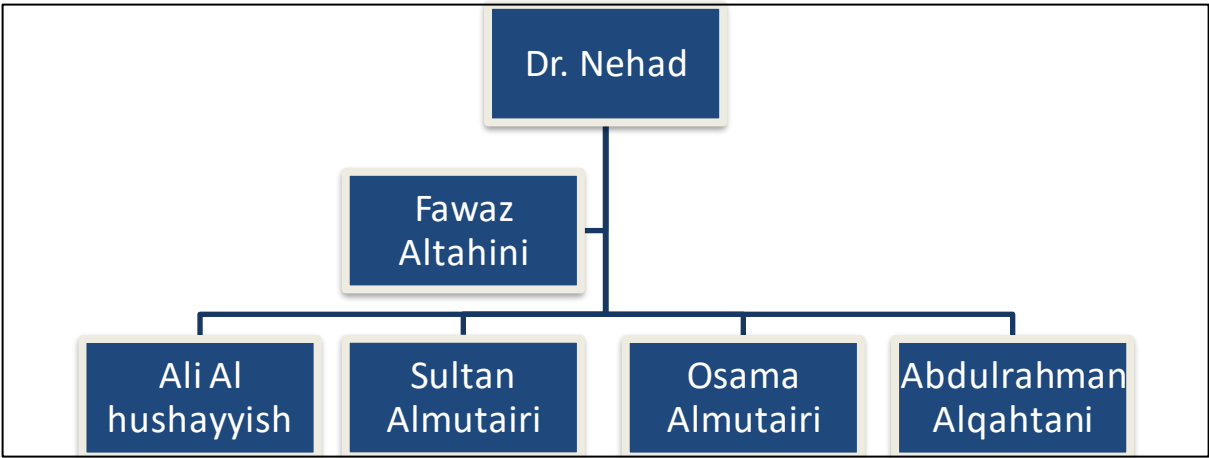


Figure 2 Internal Structure

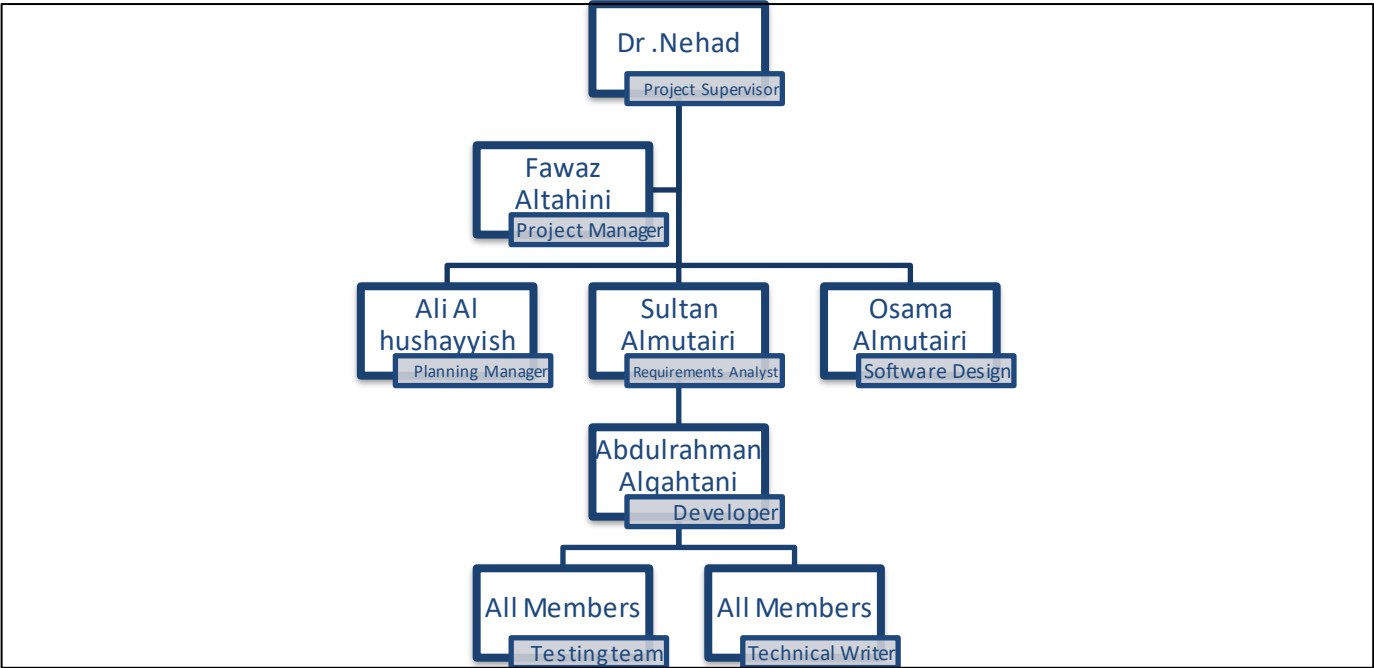


Figure 3 Project's Organizational Structure.

### 2.2.3 Roles and Responsibilities

ROLE	RESPONSIBILITIES	MEMBER
Project Supervisor	<ul style="list-style-type: none"> <li>Establish project deadlines</li> <li>Evaluate team performance</li> <li>Provide feedback and resolve issues</li> </ul>	Dr. Nehad Ibrahim
Project Manager	<ul style="list-style-type: none"> <li>Lead and manage the team</li> <li>Ensure timely delivery of tasks               <ul style="list-style-type: none"> <li>Distribute work among members</li> <li>Support team members</li> </ul> </li> </ul>	Fawaz Altahini
PLANNING MANAGER	<ul style="list-style-type: none"> <li>Define project objectives, purpose, and scope</li> <li>Create and update the project schedule</li> <li>Identify assumptions and risks</li> </ul>	Ali Al Hushayyish
Requirements Analyst	<ul style="list-style-type: none"> <li>Gather and analyze client requirements</li> <li>Categorize and prioritize requirements</li> <li>Document requirements</li> </ul>	Sultan Almutairi
Software Design	<ul style="list-style-type: none"> <li>Design user interfaces</li> <li>Develop the application architecture</li> <li>Collaborate on database design</li> </ul>	Osama Almutairi
DEVELOPMENT	<ul style="list-style-type: none"> <li>Write and review code</li> <li>Implement development methodologies               <ul style="list-style-type: none"> <li>Research and select programming tools</li> </ul> </li> </ul>	Abdulrahman Alqahtani
TESTING	<ul style="list-style-type: none"> <li>Define testing features</li> <li>Conduct software testing               <ul style="list-style-type: none"> <li>Report and fix bugs</li> </ul> </li> </ul>	All members

Technical Writer	<ul style="list-style-type: none"><li>• Create technical documentation</li><li>• Write user manuals and support materials</li></ul>	All members
------------------	---	-------------

*Table 6 Roles and Responsibilities*

## 2.3 Managerial Process Plans

This section of the Project Management Plan specifies the project management processes for the Expense Tracker. This section defines the plans for project start up, risk management, project work, project tracking and project close out.

### 2.3.1 Start-up Plan

#### 2.3.1.1 Estimates

The Expense Tracker is a small-scale academic effort, it is estimated to cost 4500 – 8500 SAR, with 120-150 hours of work per person required for development. Because it is an academic project, real costs are minimal, the main expense being any infrastructure needed - for example, optional cloud hosting – which typically cost around 1000 SAR. The development tools Java (NetBeans) and MySQL Workbench are available for free, limiting software costs. The confidence level for this cost estimate is 85%, allowing for minor unforeseen expenses.

The project is expected to be finished in sixteen weeks with a 90% confidence in the schedule. Development will follow the Waterfall model that breaks the project into different sequential phases. The project team is made up of 5 members: 1 project leader & 4 developers with all tools in place. Any additional resource needs, such as server infrastructure for testing, are expected to cost approximately 200-400 SAR for short-term usage.

Estimates of cost and schedule were derived using an analogy-based approach on historical data from small-scale university projects. Techniques like PERT (Program Evaluation and Review Technique) were used for time estimates. At key milestones the project will be re-estimated: at end of design phase, mid-development & after testing. Other re-estimations will be made if major changes occur. This keeps the project on track with low resource usage during development.

#### 2.3.1.2 Staffing

The Expense Tracker will need 5 Computer Science Students in various roles and skill levels across the project phases. They are made up of a Project Leader and 4 developers with intermediate skills in Java programming, database administration and front-end development.

Staffing will be allocated based on project milestones to ensure each phase has the resources needed to meet deadlines and deliverables. The Project Leader will be involved in all phases of the Project to ensure consistent oversight, milestone achievement and task coordination.

The following table showcases an overview of the staffing plan which will ensure that the team will be sufficiently resourced throughout the phases of the project.

Milestone	Human Resources	Duration
Project definition	Project Leader 4 Developers	1 Week
Submit Project proposal		1 Week
Submit Project Management Plan		2 Weeks
Submit Project Requirements		2 Weeks
Submit a status report		1 Week
Submit Project Design		3 Weeks
Submit Project Test Plan		1 Weeks
Code Delivery and Presentation		1 Week

*Table 7 Project Phases*

### 2.3.1.3 Project Staff Training

All team members will undergo necessary training and skill development to ensure a successful development of the Expense Tracker project. As the project is built on Java and MySQL team members will be trained in these technologies - best practices for front-and back-end development. The project Leader will coordinate tasks, manage timelines and ensure quality of work. Developers will be trained on all aspects of the project, UI design to database connections - everyone will contribute equally at every stage of development.

Roles	Responsibilities
Project Leader	Manage the project, make sure that all milestones are met, coordinate tasks, and manage the team's workflow
Lead Developer	Design and Implement core back-end functionality in Java, including DBMS using MySQL
Front-End Developer	Develop User-Interface components, ensure responsive design and fluid user experience
Full-Stack Developer	Work on both front-end and back-end integration, assist in DB queries, and ensure system connectivity
Testing and QA Specialist	Test different project components, identify bugs, and ensure code quality and functionality

*Table 8 Staff Responsibilities.*

## 2.3.2 Work Plan

### 2.3.2.1 Work Breakdown Structure

The Expense Tracker (WBS) divides the work into different phases each containing activities and deliverables that lead to the end of the phase. The WBS guarantees that all work required for project is clearly outlined and assigned to the appropriate team members. The project follows a sequential flow, and activities are grouped by project milestones.



Activity	Resources	Duration	Deliverables	Acceptance Criteria
Project Initiation	Project Leader, Team	1 week	Project goals, scope, team roles, and timeline	Goals, scope, roles clearly defined, and timeline approved
SPMP Creation	Project Leader, Team	2 weeks	Software Project Management Plan (SPMP)	Resources allocated, risks analyzed, costs estimated. SPMP document reviewed and approved by all members
SRS Preparation	Project Leader, Team	2 weeks	Software Requirements Specification (SRS)	Complete, detailed SRS document reviewed and approved
Project Design (SDS)	Lead Developer, Team	3 weeks	System Design Specification (SDS)	System design approved, UI design, and database schema ready
Development	Developers, Team	1 week	Semi-functional system (front-end & back-end)	Functionality matches requirements, security features integrated
Testing	QA Specialist, Team	2 weeks	Test plan, tested system	All test cases pass, no critical bugs, system operates as expected
Final Delivery, Presentation	Project Leader, Team	1 week	Final project code, documentation, presentation	Code deployment and demo showcased

Table 9 : Work Breakdown Structure

2.3.2.2 Schedule Allocation

The chart below demonstrates the timeline for completing each activity, starting from week 3 and ending in week 15 when the project should be finalized. The schedule is to be reviewed weekly to ensure on time delivery.

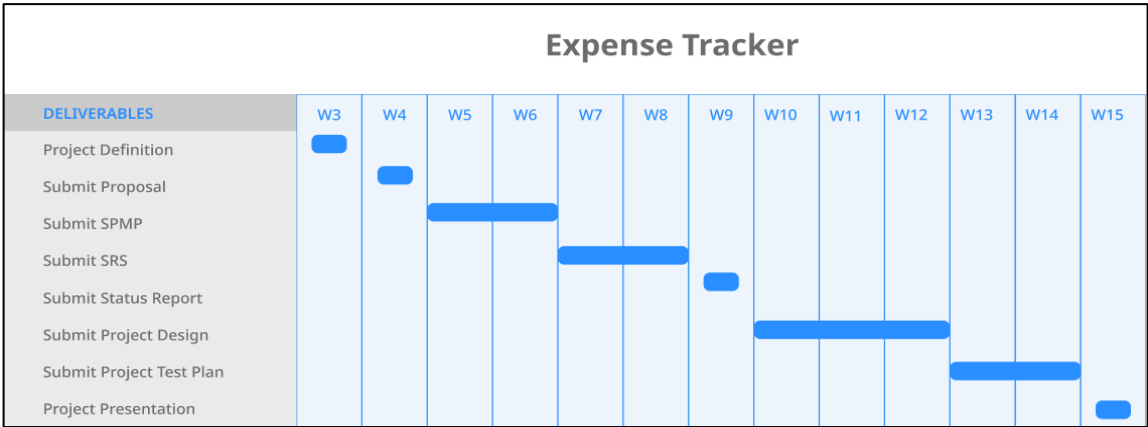


Figure 4 Schedule Allocation.

### 2.3.2.3 Resource Allocation

The table below showcases all resources allocated for the Expense Tracker:

Work Activity	Resources	Skill Level/ Requirements/ Description
Project Definition	Project Leader	Senior-Level, Project Planning
	Developers	Intermediate-Level, General Planning
	Administrative Support	Coordination for team meetings and documentation
	Computing Resources	Personal Computers
Project Proposal	Project Leader	Senior-Level, Proposal Writing
	Developers	Intermediate-Level, Proposal Writing
	Administrative Support	Document formatting and review
	Computing Resources	Personal laptops
	Software Tools	MS Word, Google Docs
SPMP	Project Leader	Senior Level, Project Planning
	Developers	Intermediate-Level, Cost Estimation, Risk Management
	Administrative Support	Documentation Support
	Computing Resources	Personal Computers
	Software Tools	MS Word, Google Docs
Project Requirements	Project Leader	Intermediate-Level, Requirements Gathering
	Developers	Intermediate-Level, Requirements Analysis
	Administrative Support	Documentation Support
	Computing Resources	Personal Computers
	Software Tools	MS Word, Google Docs
Status Report	Project Leader	Senior-Level, Report Creation
	Developers	Intermediate-Level, Progress Reporting
	Administrative Support	Formatting and submission assistance
	Computing Resources	Personal Computers
	Software Tools	MS Word, Google Docs
Project Design	Lead Developer	Senior Level, System Architecture Design
	Developers	Intermediate-Level, UI/UX and Database Design
	Administrative Support	Design documentation support
	Computing Resources	Personal Computers

	Software Tools	Figma, MySQL Workbench, MS Word, Google Docs
Test Plan	Testers	Intermediate-Level, Testing and Bug Reporting
	Administrative Support	Coordination of testing reports
	Computing Resources	Personal Computers
	Software Tools	JUnit, MS Word, Google Docs
Delivery & Presentation	Project Leader	Senior-Level, Final Submission and Presentation
	Developers	Intermediate-Level, Code-Delivery and Presentation
	Administrative Support	Report and presentation support
	Computing Resources	Personal Computers
	Software Tools	MS PowerPoint, Google Slides

Table 10 Resource Allocation.

#### 2.3.2.4 Budget Allocation

The Expense Tracker project's budget allocation is broken down by major work activities, covering personnel, computing resources, software tools, and other necessary costs. Personnel costs include training costs for team members which are allocated 2000 SAR. Computing resources are budgeted at 1500 SAR. Software tools, such as MS Word, and Figma are estimated at 1000 SAR in total. Additional costs include testing facilities (test servers) at 1000 SAR for the Test Plan milestone. Other costs, including meetings and travel, are not anticipated due to the academic nature of the project. The total estimated budget is 6500 SAR.

### 2.3.3 Project Tracking Plan

#### 2.3.3.1 Requirements Management

Managing requirements rely on defining any changes requested by the customer and controlling these changes via proper task scheduling. In general, the project will proceed as planned until the next iteration, where the changes requested will be implemented and tested, to minimize (or fully remove) any risk and/or disruptions that could occur during the software development.

#### 2.3.3.2 Schedule Control

The project manager regularly receives progress reports of each task distributed among team members to ensure that everything is going according to plan without risk. Every week or two, the project manager holds either a face-to-face or virtual meeting with the team members to discuss overall progress and address any risks associated with some tasks if any exist

#### 2.3.3.3 Quality Control

During the development of the project, quality evaluations will be done prior to preset milestones to ensure each deliverable meets the expectations of the customer at minimum. Each component of the software should function appropriately with minimal-to-zero errors occurring to achieve the best quality possible and overall satisfaction of the customer

#### 2.3.3.4 Reporting

The purpose of reporting is mainly focused on two perspectives: internal and external. Internal reports contain information regarding the inner workings of the project, such as code functionalities, and are communicated through face-to-face meetings or Zoom calls if circumstances forbid physical contact. External reports are mainly status reports regarding the progress of the overall project, which is given by each team member working on the software. Overall, reports are done weekly or bi-weekly depending on the progress of the project itself.

#### 2.3.3.5 Project Metrics

The project metrics are evaluated based on performance, progress done, and overall effectiveness of the plan. For this project, the measurements are done depending on four specific criteria, as shown in the table below:

Metrics	Description	Frequency of Collection
Cost	Overall budget used in developing the software	Within internal reports
Productivity	The amount of progress done compared to project plan expectations	Within external reports

Quality	Proper testing of software following developments and revisions	Within internal & external reports
Risk	Overall frequency of risks and how it's managed throughout the project	Within internal reports

Table 11 Project Metrics.

### 2.3.4 Risk Management Plan

Risk management is essential for the successful execution of the Expense Tracker project, involving the continuous identification, analysis, and prioritization of potential risks. The following table outlines key risks, their types, probabilities, potential effects, actions to address them, and preventive measures to minimize their impact. To ensure ongoing risk assessment, the team will conduct regular meetings to review risks and adapt mitigation strategies as necessary. This proactive approach will help safeguard project deliverables and enhance communication among team members, stakeholders, and clients.

No.	Risk	Risk Type	Probability	Effects	Actions	Prevention
1	Changing Requirements	Requirements	Low	Tolerable	Discuss with the customer about difficulties and impacts of changes.	Write clear and detailed requirements; conduct regular reviews with the customer to confirm requirements are still aligned.
2	Personnel Availability and Stability	People Risks	Moderate	Serious	Foster team cooperation to achieve goals; implement a backup plan for critical roles.	Ensure proper communication within the team; develop contingency plans for key roles; have a strategy for handling sudden absences, layoffs, or terminations.

3	Exceeding the Deadline	Estimation Risks	Moderate	Serious	Set early deadlines to ensure timely completion; monitor progress regularly.	Make realistic timelines for deliverables; create buffer periods for critical tasks and regularly adjust deadlines based on project status.
4	Lack of Knowledge	People Risks	Moderate	Serious	Identify knowledge gaps and provide training solutions; assess team skill sets.	Prepare external resources for project needs; schedule training sessions and encourage knowledge sharing within the team.
5	Inaccurate Time and Budget Estimation	Estimation Risks	Moderate	Tolerable	Develop a detailed budget and timeline; conduct reviews with stakeholders.	Regularly review estimations with the team; adjust budgets based on performance reviews and stakeholder feedback.
6	Defective Code	Technology Risks	Moderate	Serious	Conduct thorough testing and code reviews; implement continuous integration practices.	Regular monitoring of coding tools; establish a rollback plan for defective code; encourage pair programming to improve code quality.
7	Resistance to New Technology	Technology Risks	Moderate	Serious	Provide training and support for new tools; gather feedback on technology use.	Involve team members in the technology selection process; ensure adequate resources are available for training and support.

Table 12 Risk Management Plan.

### 2.3.5 Project Closeout Plan

The project closeout plan addresses how all phases and deliverables, which will be completed in identified key processes to effectively close out the project, will be carried out. It also includes planning the reassignment of staff wherein team members are assigned to other projects or specific roles per their expertise to ensure that resources are well utilized after the project. The project materials, which include source code, design documents, and reports, will be archived into the project repository for any future reference and for organizational documentation. Project metrics with regard to timelines, resource usage, budget data, and quality are captured and stored in business project databases for better insights in the future. Similarly, a post-mortem debrief will be conducted with the team to review lessons learned, challenges faced, and areas for further improvement. A full final report will also be prepared that summarizes lessons learnt, analysis of achieved objectives, and an overall review of performance against the project's objectives.

No.	Deliverable	Delivery method	Status
1	Project Proposal	Softcopy	Completed
2	Software Project Management Plan (SPMP)	Softcopy	Ongoing
3	Project status report	Softcopy	Ongoing
4	Software Requirements Specifications (SRS)	Softcopy	Ongoing
5	Software Design Specifications (SDS)	Softcopy	Ongoing
6	Software Test Plan (STP)	Softcopy	Ongoing
7	Final Project	Softcopy	Ongoing

*Table 13 Project Deliverables*

## 2.4 Technical Process

### 2.4.1 Process Model

For our Expense Tracker project, we chose the waterfall model because of its well-defined structure and systematic approach, which improves project management and quality control. This model works especially well for projects with clear requirements because it enables us to document each stage accurately before going on to the next. Each stage is integrated with a milestone that we will document each one with a date that we plan in advance and also each document will be revised by our instructor. The figure below illustrates the stages of the waterfall model:

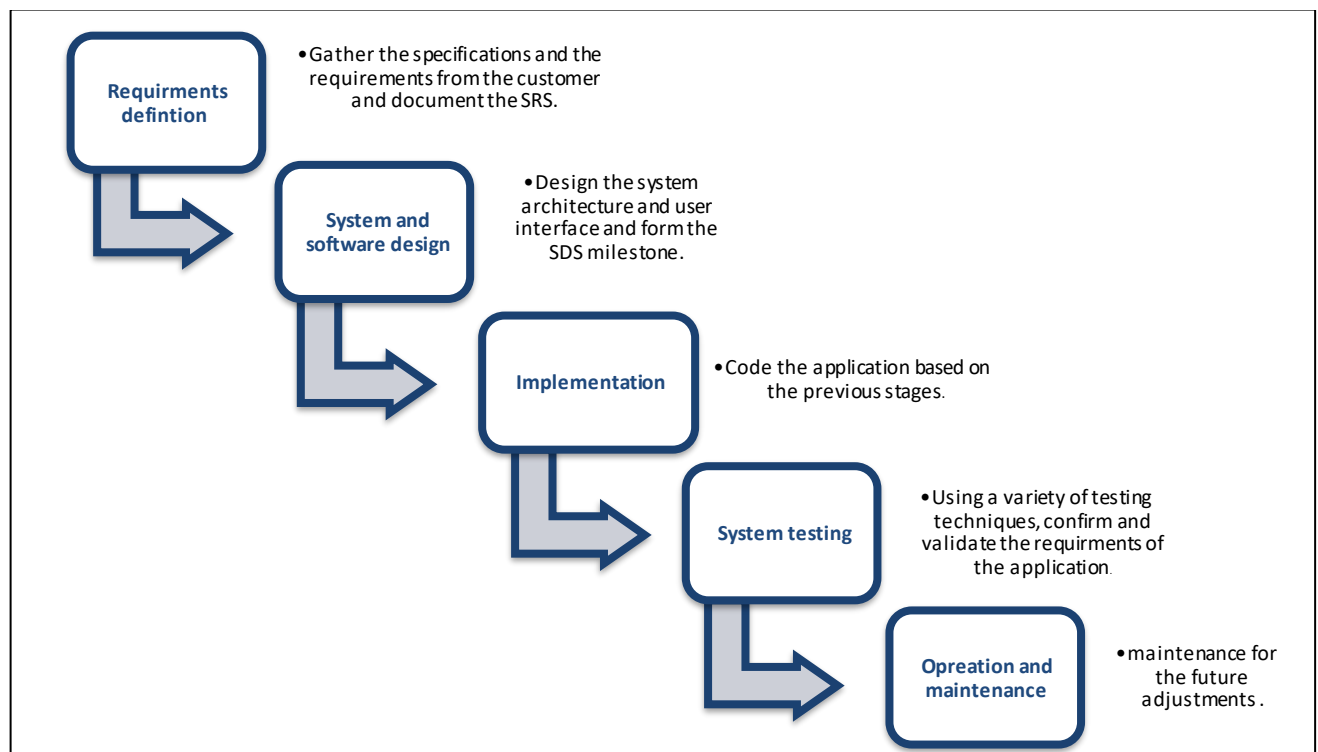


Figure 5 The five principles of Waterfall Model.

### 2.4.2 Methods, Tools, and Techniques

To meet our project's successful delivery, we will utilize a variety of tools and techniques some of which are outlined in the table below:

Phase	Tools	Methods	Technical Standards
Planning	<ul style="list-style-type: none"> <li>- Microsoft word</li> <li>- WhatsApp</li> </ul>	None	IEEE Std 1058 - 1998
Requirements Analysis	<ul style="list-style-type: none"> <li>- Microsoft word</li> </ul>	<ul style="list-style-type: none"> <li>- Activity Diagram</li> </ul>	IEEE Std 830 - 1998
System Design	<ul style="list-style-type: none"> <li>- Draw.io</li> </ul>	<ul style="list-style-type: none"> <li>- UML</li> <li>- Entity Relation Diagram</li> </ul>	IEEE Std 1016 - 1998
Implementation	<ul style="list-style-type: none"> <li>- NetBeans</li> <li>- MySQL Workbench</li> </ul>	None	None

Table 14 Methods, Tools, and techniques.



### 2.4.3 Infrastructure

The table below illustrates the infrastructure that will be conducted in the project:

Infrastructure	
Hardware	Each team member has his PC for building and administration.
Software	Development Tools such as NetBeans for Java, MySQL Workbench for database, and Microsoft Word for documenting.
Operating System	Each member's pc can operate with any system whether Windows or Linux or others.
Network	High-speed and secure LAN, WAN.
Policies, Procedures, Standards	Work ethics and rules must be obeyed by the team members.
Facilities	Every member needs to have a desk or office with devices to help with work, like a printer.

*Table 15 Infrastructure.*

### 2.4.4 Product Acceptance

In order to make sure that all deliverables satisfy project objectives and customer expectations, the Expense Tracker project will follow to a structured product acceptance plan. After the project is finished, the deliverables will go through a formal review process that entails analysis, testing, and demonstration. The project requirements will specify the objective criteria for acceptance, which will include data security, system performance, usability, and accuracy of financial calculations. To ensure mutual understanding and alignment, these requirements will be formalized in a signed agreement by the customer and the IM/IT organization. Before final acceptance is granted, technical processes like user acceptance testing (UAT), system demonstrations, and inspection of functionalities (budgeting, categorization, analytics) will be carried out to ensure that the product meets the agreed-upon standards.

## 2.5 Supporting Process Plans

### 2.5.1 Documentation

Document	Format Standard	Prepared by	Review by
Project Proposal	Template provided by the instructor	All the team members	To the instructor Dr. Nehad M. Ibrahim
Project management plan (SPMP)	IEEE Std 1058 - 1998		
Status report	Template provided by the instructor		
Project requirements (SRS)	IEEE Std 830 - 1998		

Project Design (SDS)	IEEE Std 1016 - 1998		
Project test plan (STS)	IEEE Std 829-1998		

Table 16 Documentation

2.6 Additional Plans

A variety of extra plans will be put into place to satisfy the Expense Tracker's contractual obligations and product requirements. User data protection will be provided by a security and privacy plan that complies with industry standards and uses encryption techniques. Plans for product installation and integration will specify how the program is to be configured and integrated with financial institutions using secure APIs. To help users learn the system, user training plans will include comprehensive manuals, tutorials, and support documentation. Plans for maintenance and support will be developed for long-term usage, guaranteeing regular system updates, bug fixes, and customer assistance. Furthermore, in the event that it becomes necessary, a plan for system transition will manage data conversion and migration from earlier systems. All these strategies will work together to guarantee the expense tracker's efficient, safe, and secure operation.

3.

## Chapter 3: SRS

### 3.1 Introduction

This section introduces the topics that are related to Expense Tracker Software Requirements Specifications (SRS) document. It explicit the following sections: The purpose of the SRS, intended users, scope, list of acronyms and abbreviations, list of references used in this document.

#### 3.1.1 Purpose

The purpose of this document is to provide a complete and comprehensive description of the requirements for the proposed Expense Tracker application. The SRS will clarify the functional and non-functional requirements. Also, it will cover all the interfaces, constrains and performance requirements of the application. In addition, the aimed audience for this project are:

- Client: The clients will check the comprehensiveness and reliability of the document.
- Project supervisor: The project supervisor will revision the document and provide commentary and feedback, and suggestions. In Addition, this document will assist the project supervisor follow up on the team's work progress.
- Team: The team members will use the document as a reference and instruction for all needed information to develop the application.

#### 3.1.2 Scope

This document explains the criteria for the proposed Expense Tracker Application, as derived from user feedback. The program will provide particular features and tools to assist individuals in properly monitoring and managing their finances. The Expense Tracker Application contains essential features including tracking income and expenses, creating budgets, and graphs, analyzing financial reports, and receiving important financial notifications.

Four main sections:

- User Management: Profile settings, login, and registration for the management of financial objectives and personal information.
- Tracking Revenue and Expenses: Record revenue and expenses in detail, establish regular transactions, and accommodate various currencies.
- Budgeting and Categories: Establish unique categories, set spending limits, and receive notifications when spending is approaching or exceeding the budget.
- Visualization and Reporting: To illustrate spending patterns, a dashboard overview featuring graphs, charts, and exportable reports (in PDF and CSV format) is provided.

#### 3.1.3 Definitions, Acronyms, and Abbreviations

Terminology	Definition

Software	The programs, routines, and symbolic languages that control the functioning of the hardware and direct its operation.
Database	A database is an organized collection of structured information, or data, typically stored electronically in a computer system.
Use case diagram	Used to show the relation between a set of actions (cases) that might occur in the system with the user (actor).
Budget	A financial plan that sets limits on spending within various categories (e.g., groceries, rent) based on income and financial goals.
Financial Management	The practice of overseeing and controlling income, expenses, and budgeting to achieve personal or financial goals.

Table 17 Terminologies

Acronym	Definition
SRS	Software Requirements Specifications
IEEE	Institute of Electrical and Electronics Engineers
ETA	Expense Tracker Application
UI	User Interface
PDF	Portable Document Format
APP	Mobile Application

Table 18 Acronyms

3.1.4 References

[1] Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education Limited.

[2] IEEE Computer Society. (1998). IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE.

[3] Pressman, R. S. (2014). Software Engineering: A Practitioner's Approach (8th ed.). McGraw-Hill Education.

## 3.2 Overall description

This section covers the Expense Tracker's software and will cover two subsections. The first subsection is the product perspective and, followingly, the product function, which explains the functions discussed later in the coming sections.

### 3.2.1 Product perspective

The Expense Tracker is an independent application designed for individuals to efficiently manage and track personal expenses. It provides core functionalities for expense logging, categorization, and budget monitoring in a single, standalone system. Unlike enterprise financial software, this application operates independently without dependencies on external systems or third-party integrations, making it a straightforward solution for everyday financial management. The application will be implemented as a mobile application, supporting secure storage of financial data locally, thus ensuring user privacy. This product is targeted toward individuals seeking a simple yet effective tool to oversee their daily finances and spending habits.

### 3.2.2 Product functions

The Expense Tracker application provides essential features for users to effectively track and manage their personal finances. The main functions are as follows:

1. User Registration and Login

- Enables new users to create an account and existing users to log in securely.
- Provides password recovery options and two-factor authentication for enhanced security.

2. Expense Entry and Management

- Allows users to add, edit, and delete expenses, categorizing them under headings such as "Food," "Transportation," or "Utilities."
- Each expense entry includes date, amount, category, payment method, and optional notes.

3. Budget Setup and Monitoring

- Enables users to set weekly, monthly, or custom budgets across multiple categories.

- Provides real-time updates on budget usage and visually highlights categories approaching their limits.

#### 4. Expense Reports and Analysis

- Generates customizable reports, summarizing expenses over selected time frames (e.g., weekly, monthly, or annually).
- Includes visual charts, such as pie charts and bar graphs, to help users analyze spending patterns across different categories.

#### 5. Notifications and Alerts

- Sends notifications if users approach or exceed set budget limits.
- Provides reminders for users to log daily expenses and periodic notifications to review spending.

#### 6. Data Export

- Users can export their expense data in formats like CSV or PDF for offline analysis.

#### 7. Data Backup and Restore

- Allows users to back up their data locally or to a secure cloud server.
- Provides a restore function for recovering data if needed.

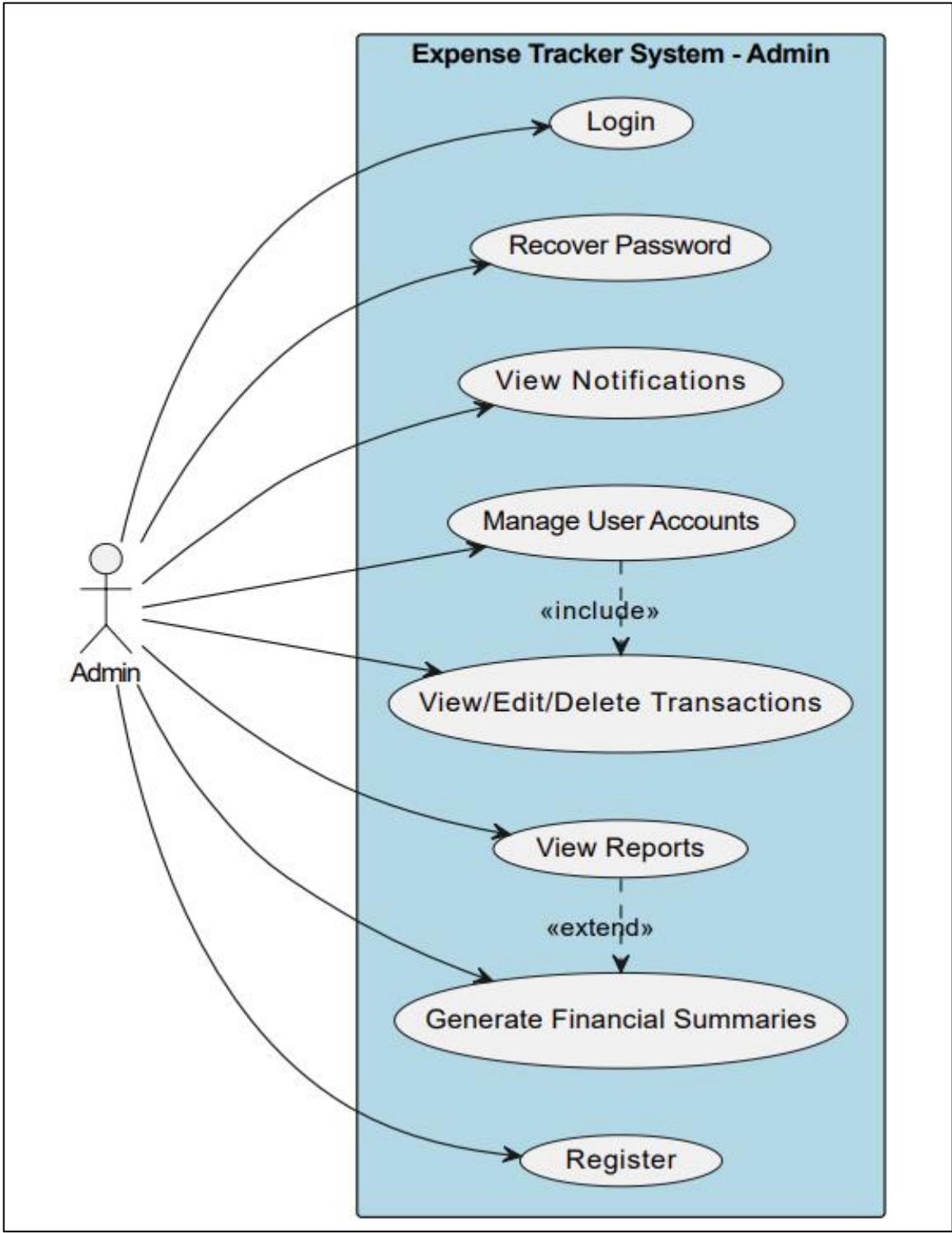


Figure 6 Use Case Diagram: Admin System



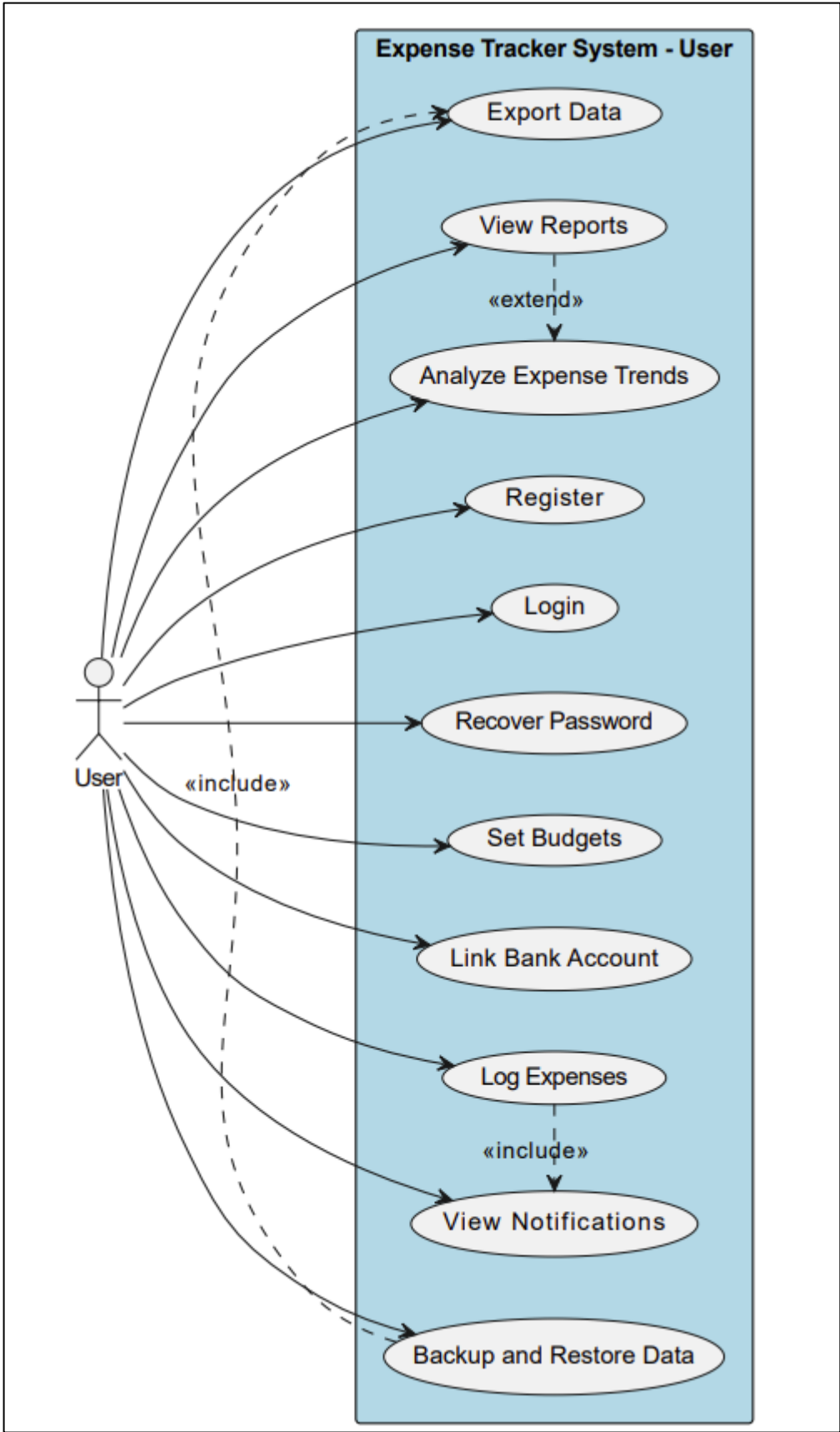


Figure 7 Use Case Diagram: User System

Use Case	Actors	Description	Preconditions	Postconditions
<b>Register</b>	User, Admin	Allows new users or admins to create an account.	User provides valid registration details.	Account is created and stored in the system.
<b>Login</b>	User, Admin	Authenticates a user or admin using credentials.	User or admin has registered an account.	User or admin is logged into the system.
<b>Recover Password</b>	User, Admin	Enables users or admins to reset their password if forgotten.	User or admin provides a valid email.	Password reset instructions are sent to the email.
<b>View Reports</b>	User	Generates and displays financial reports for the user.	User has logged expenses or set budgets.	Reports are generated and displayed.
<b>Log Expenses</b>	User	Allows users to add expenses, including details like amount, category, and date.	User is logged in.	Expense is added and saved in the system.
<b>View Notifications</b>	User	Allows users to view system-generated notifications, such as budget alerts or expense reminders.	Notifications exist in the system.	Notifications are marked as read or acknowledged.
<b>Set Budgets</b>	User	Enables users to set budget limits for various categories.	User has valid login credentials.	Budget is saved and tracked in the system.
<b>Link Bank Account</b>	User	Allows users to link their bank accounts for automatic expense tracking.	User provides valid bank credentials and authorization.	Bank account is linked, and transactions are synced.
<b>Backup and Restore Data</b>	User	Enables users to back up their data or restore it from a previous backup.	User provides backup location or selects a restore file.	Data is successfully backed up or restored.
<b>Export Data</b>	User	Allows users to export financial data in various formats like CSV or PDF.	User selects data to export.	Data is exported in the selected format.
<b>Analyze Expense Trends</b>	User	Analyzes spending patterns and provides insights using charts or summaries.	User has logged expenses over a period.	Insights are displayed to the user.
<b>Manage User Accounts</b>	Admin	Enables admins to add, update, or remove user accounts.	Admin is logged in with appropriate permissions.	Changes to user accounts are saved in the system.
<b>View/Edit/Delete Transactions</b>	Admin	Allows admins to view, edit, or delete user transaction records.	Admin is logged in and selects a user transaction.	Changes to transactions are applied in the system.

Generate Financial Summaries	Admin	Generates summaries of financial data for system analysis.	Financial data exists in the system.	Summaries are generated and available for analysis.
------------------------------	-------	--	--------------------------------------	---

Table 19 Use Case Description Summary

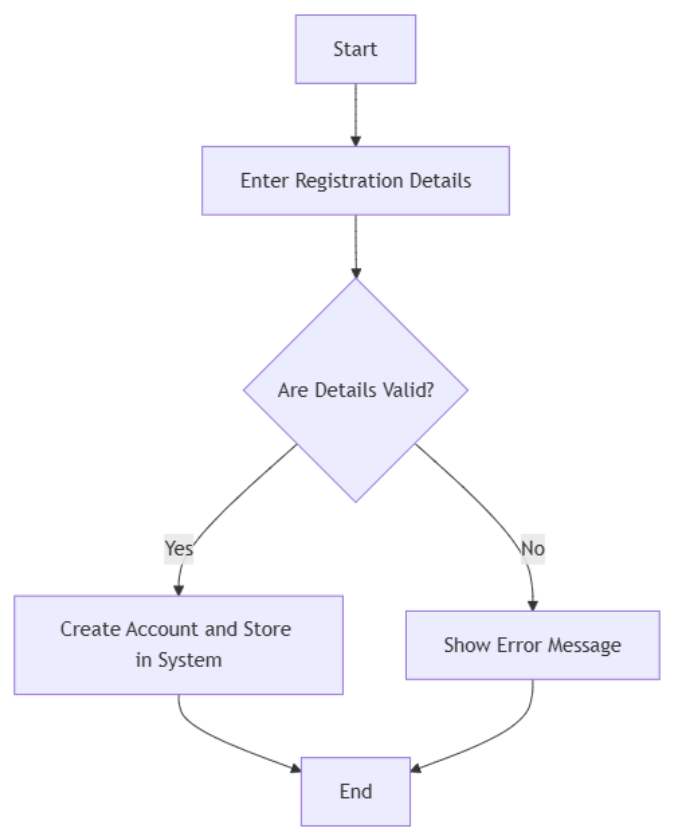


Figure 8 Register

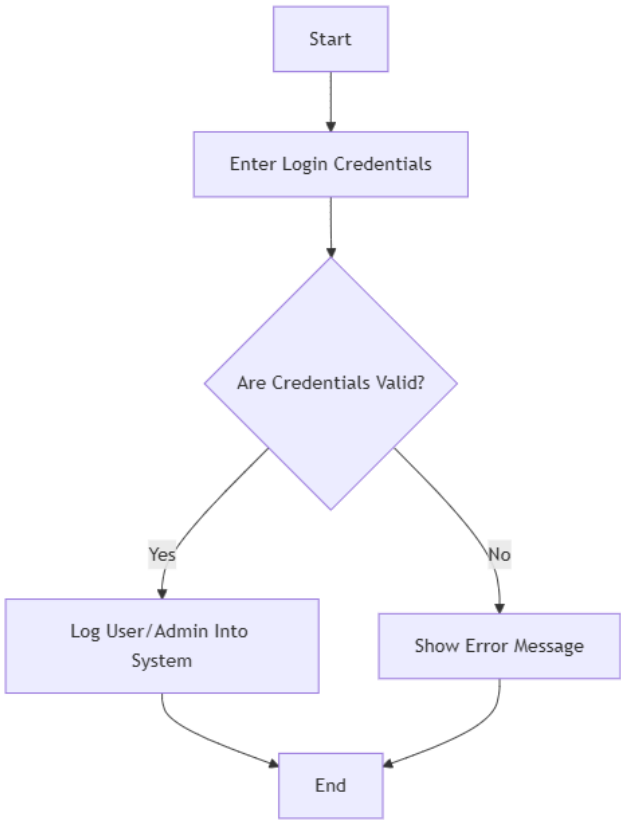


Figure 9. Login

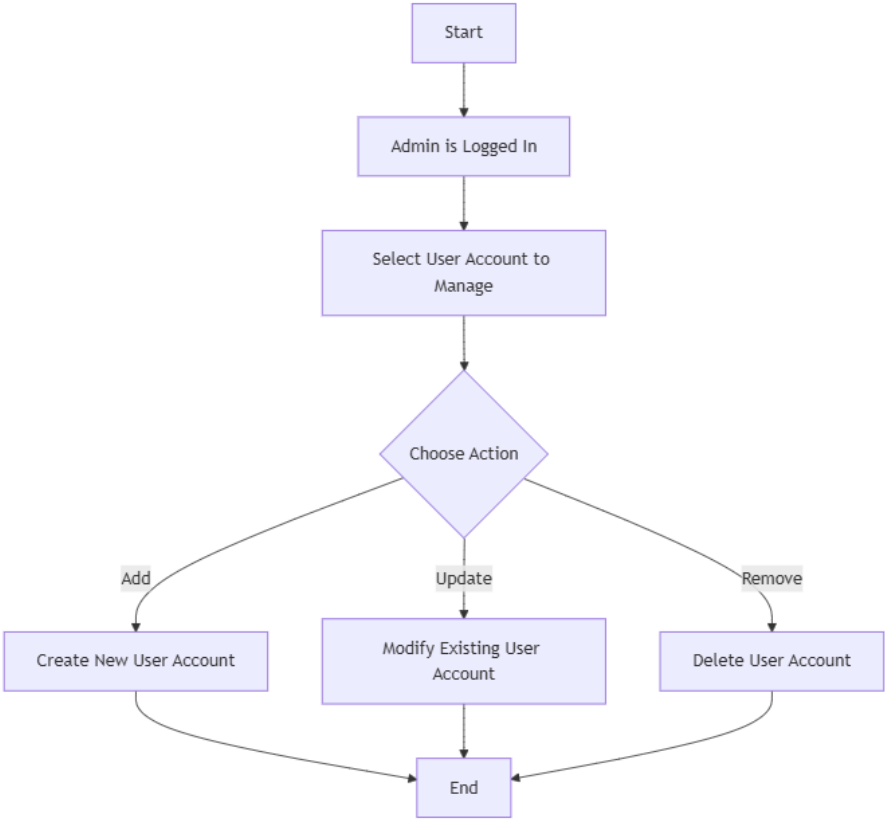


Figure 10. Manage User Accounts

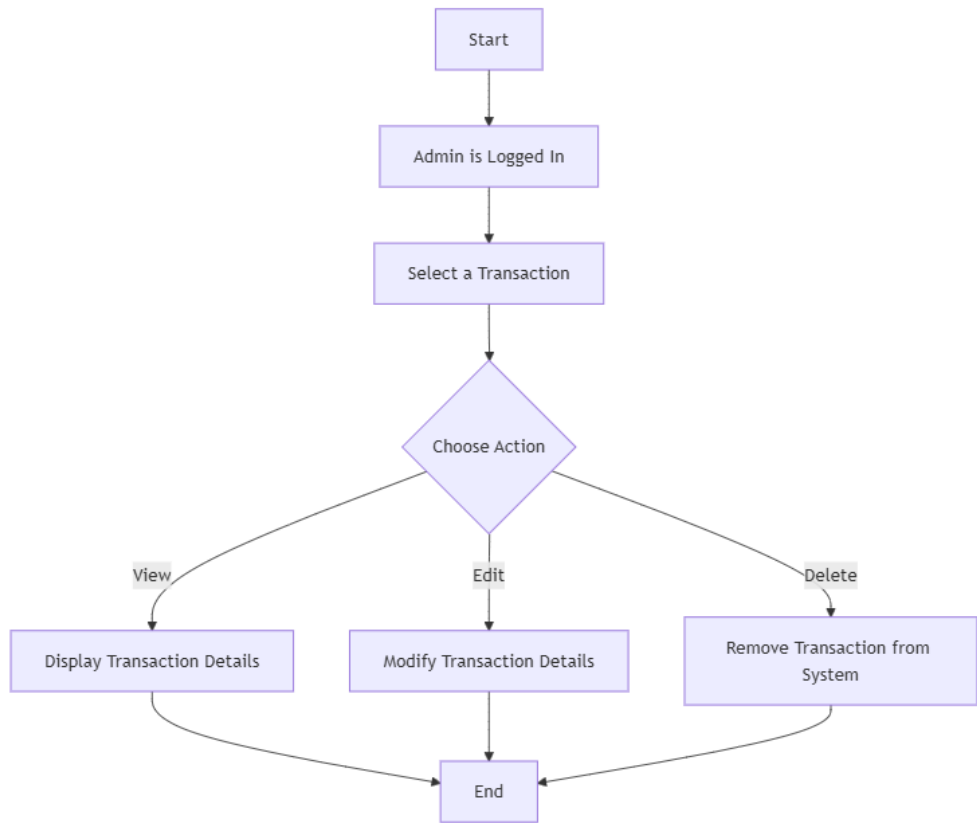


Figure 11. View/Edit/Delete Transactions

Attribute	Description
Actors	User, Admin
Description	Allows new users or admins to create an account.
Priority	High
Data	Registration details (e.g., name, email, password).
Stimuli	User submits registration form.
Response	System validates the data and creates an account.
Abnormal Condition	Duplicate email, invalid data, or system unavailability.

Table 20 Use Case: Register

Attribute	Description
Actors	User, Admin
Description	Authenticates a user or admin using valid credentials.
Priority	High
Data	Username and password.
Stimuli	User enters credentials and submits the login request.
Response	System verifies credentials and grants access.
Abnormal Condition	Incorrect credentials, locked account, or system downtime.

--	--

*Table 21 Use Case: Login*

Attribute	Description
Actors	User, Admin
Description	Enables users or admins to reset their password if forgotten.
Priority	Medium
Data	Registered email address.
Stimuli	User submits a password recovery request.
Response	System sends password reset instructions via email.
Abnormal Condition	Invalid or unregistered email, or email service unavailability.

*Table 22 Use Case: Recover Password*

Attribute	Description
Actors	User
Description	Allows users to log expenses with details like amount, date, and category.
Priority	High
Data	Expense details (e.g., amount, date, category).
Stimuli	User enters expense data and submits it.
Response	System validates and stores the expense.
Abnormal Condition	Invalid data format, or database errors during save operation.

*Table 23 Use Case: Log Expenses*

Attribute	Description
Actors	User
Description	Enables users to set budget limits for various categories.
Priority	Medium
Data	Budget category, limit, and time period.
Stimuli	User submits budget details.
Response	System saves and monitors the budget.
Abnormal Condition	Invalid budget values or failure to save data.

*Table 24 Use Case: Set Budgets*

Attribute	Description
Actors	User

Description	Allows users to export data in CSV or PDF format.
Priority	Medium
Data	Selected financial data.
Stimuli	User selects data and chooses export format.
Response	System generates and provides the export file.
Abnormal Condition	Unsupported export format or file system errors.

*Table 25 Use Case: Export Data*

Attribute	Description
Actors	User
Description	Allows users to view notifications such as budget alerts and reminders.
Priority	Medium
Data	Notification details (e.g., message, timestamp).
Stimuli	User clicks on the notifications tab.
Response	System retrieves and displays notifications.
Abnormal Condition	No notifications available or data retrieval error.

*Table 26 Use Case: View Notifications*

Attribute	Description
Actors	User
Description	Allows users to generate and view financial reports.
Priority	High
Data	Selected time period and report type.
Stimuli	User selects a report type and time range.
Response	System generates and displays the report.
Abnormal Condition	Incomplete data for the selected time range.

*Table 27 Use Case: View Reports*

Attribute	Description
Actors	User
Description	Allows users to link their bank accounts for automatic expense tracking.
Priority	Medium
Data	Bank credentials and authorization token.
Stimuli	User enters bank details and authorizes connection.
Response	System links the bank account and begins syncing transactions.
Abnormal Condition	Incorrect credentials or unsupported bank integration.

*Table 28 Use Case: Link Bank Account*

Attribute	Description
-----------	-------------



Actors	User
Description	Allows users to back up their data or restore it from a previous backup.
Priority	High
Data	Backup file or location.
Stimuli	User selects backup or restore options.
Response	System performs the backup or restores the data.
Abnormal Condition	Corrupt backup file or insufficient storage space.

Table 29 Use Case: Backup and Restore Data

Attribute	Description
Actors	User
Description	Analyzes spending patterns and provides insights through charts or summaries.
Priority	Medium
Data	Historical expense data.
Stimuli	User selects a time range for analysis.
Response	System generates visual trends and summaries.
Abnormal Condition	Insufficient data for meaningful analysis.

Table 30 Analyze Expense Trends

Attribute	Description
Actors	Admin
Description	Enables admins to add, update, or remove user accounts.
Priority	High
Data	User account details (e.g., name, email, role).
Stimuli	Admin selects a user and modifies details.
Response	System saves the changes to the account.
Abnormal Condition	Attempt to modify a non-existent account or invalid data entry.

Table 31 Manage User Accounts

Attribute	Description
Actors	Admin
Description	Allows admins to view, edit, or delete user transaction records.
Priority	Medium
Data	Transaction details (e.g., amount, date, category).
Stimuli	Admin selects a transaction to view or modify.
Response	System updates or removes the transaction.
Abnormal Condition	Data inconsistency or database update failure.

Table 32 View/Edit/Delete Transactions

Attribute	Description
Actors	Admin
Description	Generates financial summaries for system analysis.

Priority	Medium
Data	Aggregated expense and budget data.
Stimuli	Admin requests a financial summary.
Response	System generates and provides the summary.
Abnormal Condition	Incomplete data or unsupported summary type.

Table 33 Generate Financial Summaries

Attribute	Description
Actors	System
Description	Automatically notifies users about budget thresholds and expense reminders.
Priority	High
Data	Notification details (e.g., message, timing).
Stimuli	Triggered by system conditions (e.g., budget exceeded).
Response	Notification is sent to the user.
Abnormal Condition	Notification delivery failure due to server issues.

Table 34 Notify Users

3.3 Specific Requirements

This section describes the external interface's inputs and outputs, as well as the functional requirements for each intended user. The developer provided detailed descriptions of all requirements specified by the client.

3.3.1 External interface requirements

This section will provide a complete description of the system's input and output.

3.3.1.1 User interfaces

The Expense Tracker application will offer an easy-to-use interface where expenses can be easily inputted, viewed, and managed. Users can split spending into categories, budget, and keep data secure with information stored securely. The application will deliver reporting and analytics through graphs and charts while enabling integration with bank APIs for automatic imports of transactions and their categorization to further enhance the experience of financial management.

3.3.1.1.1 Start-up Selection Interface

This initial screen allows users to select whether they want to log in or register as a new user.

Field Name	Format	Requirements	I/O	Constraints	Comments
Login Button	Button	Required	Input	Enabled	Takes User to Login Interface
Register Button	Button	Required	Input	Enabled	Takes User to Register Interface

Table 35 Start-up selection Components.

### 3.3.1.1.2 Registration Interface

This screen allows new users to create an account by providing necessary information. Once registration is complete users taken to login interface.

Field Name	Format	Requirements	I/O	Constraints	Comments
First Name	Text	Required	Input	Must contain only letters	Takes User's First Name
Last Name	Text	Required	Input	Must contain only letters	Takes User's Last Name
Email	Text	Required	Input	Must match email formats and must be unique	Takes User's Email, Used as identifier for each user
Phone Number	Numeric	Required	Input	Must be Only numbers	Takes User's Phone number for contact
Password	Encrypted Text	Required	Input	Min 8 Characters	Must use in logins for security
Confirm Password	Encrypted Text	Required	Input	Must match Password Entered	Confirms User's choice of password
Register Button	Button	Required	Input	Enabled only if all Fields are valid	Submits User's Data for registry
Error Message	Text	Conditional	Output	Shown for invalid entries or mismatched passwords	Displays errors for corrections, like "Password mismatch."

Table 36 Registration Interface Components.

### 3.3.1.1.3 Login Interface

This interface handles login for both Admin and User's, routing them to the appropriate mode based on account type. If credentials belong to an Admin, then user is directed to Admin Mode, with additional admin-specific options and functionalities. If credentials belong to a user, then user is directed to User Mode, displaying standard user functionalities.

Field Name	Format	Requirements	I/O	Constraints	Comments
Email	Text	Required	Input	Must match registered email	Used to authenticate users
Password	Encrypted Text	Required	Input	Must match password policy and matches user's password	Ensure security
Login Button	Button	Required	Input	Enabled if both email and password fields are valid	Takes users into appropriated mode based on credentials
Forgot my Password	Link/Button	Optional	Input	Enabled	Directs users to the password recovery interface
Error Message	Text	Conditional	Output	Displayed for incorrect credentials	Shows error message if login fails (e.g.,

					"Invalid login credentials")
--	--	--	--	--	------------------------------

Table 37 Login interface components.

#### 3.3.1.1.4 Expense/Budget Selection interface

This interface enables users to choose whether to add funds or expenses after logging in, redirecting them to the appropriate entry interface.

Field Name	Format	Requirements	I/O	Constraints	Comments
Add Funds Button	Button	Required	Input	Enabled	Takes the user to the Add Funds interface.
Add Expense Button	Button	Required	Input	Enabled	Takes the user to the Add Expense interface.
Welcome Message	Text	Optional	Output	Displayed for after Login	Shows a welcome message to users

Table 38 Expense/Budget Selection interface components

#### 3.3.1.1.5 Adding Expense interface for User's

The User adding Expense Interface enables users to record their expenses by selecting categories, entering amounts, and adding notes. It ensures accurate tracking of personal finances and provides immediate feedback on successful entries.

Field Name	Format	Requirements	I/O	Constraints	Comments
Expense Type	Dropdown	Required	Input	Limited to predefined Categories	Allows user to select predefined categories like Food, Entertainment, etc.
Amount	Numeric (Currency)	Required	Input	Must be positive number	Ensures accurate tracking of expenses.
Date	Date picker	Required	Input	Must be a valid date and in DD-MM-YY format	Specifies when the expense was made.
Notes	Text	Optional	Input	Up to 250 characters	Users can add details about the expense.
Add Expense Button	Button	Required	Input	Enabled when required fields are filled	Saves the expense entry to user's account; error if input invalid.
Delete Expense Button	Button	Optional	Input	Enabled for an exist expense	Delete an expense entry from the user account
Confirmation Message	Text	Conditional	Output	Displayed upon successful entry or deletion	Displays "Expense added/deleted successfully" after valid entry or deletion.
Error Message	Text	Conditional	Output	Displayed upon incorrect entry	"Invalid expense entry" if inputs are incorrect

Warning Message	Text	Conditional	Output	Displayed if amount exceeds budget	"Warning: Amount exceeds budget" shown when expense surpasses budget.
-----------------	------	-------------	--------	------------------------------------	---

Table 39 Adding Expenses interface Components.

### 3.3.1.1.6 User's Add Funds interface

The *User Add Funds Interface* enables users to add money to their account by specifying an amount, selecting, or entering a source of income, and adding notes if desired. This interface ensures accurate tracking of income and updates the user's account balance, providing immediate feedback upon successful entry.

Field Name	Format	Requirements	I/O	Constraints	Comments
Source of Income	Dropdown	Required	Input	Limited to predefined Sources	Allows user to select predefined sources of income like Salary, gift, etc.
Amount	Numeric (Currency)	Required	Input	Must be positive number	Amount of money user wants to add.
Date	Date picker	Required	Input	Must be a valid date and in DD-MM-YY format	Specifies the date the funds are added.
Notes	Text	Optional	Input	Up to 250 characters	Users can add details about the source.
Add Funds Button	Button	Required	Input	Enabled when required fields are filled	Adds funds to the user's account; error if input invalid.
Confirmation Message	Text	Conditional	Output	Displayed upon successful entry	Displays "Funds added successfully" after valid entry.
Error Message	Text	Conditional	Output	Displayed upon incorrect entry	"Invalid Funds entry" if inputs are incorrect

Table 40 User's Add Funds interface Components.

### 3.3.1.1.7 Admin Financial management interface

The Admin Financial Management Interface allows administrators to manage user financial entries, including expenses and funds. Admins can update amounts, modify notes, and filter transactions by type or date. Confirmation messages are provided after successful updates, ensuring effective oversight of user accounts.

Field Name	Format	Requirements	I/O	Constraints	Comments
Transaction Type	Dropdown	Required	Input	Options include Expense, Fund	Admins can filter entries by type (e.g., Expense, Fund).
Amount	Numeric (Currency)	Required	Input/Output	Must be positive number	Admins can view and edit amounts for both expenses and funds.
Source/Category	Dropdown/Text	Required	Input	Must match predefined categories or sources	Allows selection or editing of fund sources or expense categories (e.g., salary, rent).
Date Range Filter	Date Picker	Optional	Input	Must be valid date range in DD-MM-YY format	Allows filtering of entries within specific date ranges.
Notes	Text	Optional	Input/Output	Up to 250 characters	Admins can view and edit user notes related to the transaction.
Add/Edit Transaction Button	Button	Required	Input	Enabled when valid entries are present	Saves or updates the transaction entry; error if input is invalid.
Confirmation/Update Message	Text	Conditional	Output	Only displayed upon successful entry or edit	Displays "Transaction added/updated successfully" for confirmation.
Error Message	Text	Conditional	Output	Displayed upon invalid transactions	"Transaction update failed" if editing is incorrect

Table 41 Admin Financial management interface Components.

**3.3.1.1.8 Analysis expenses interface**

In this interface, users will be able to see the analysis of their expenses based on their transactions. Here are the fields that will be in this interface:

Field Name	Format	Requirements	I/O	Constraints	Comments
Date Range	Date picker	Required	input	The start date cannot be after the end date.	Enables analysis of expenses over specific periods
Category Filter	Dropdown	Optional	input	Only categories that exist in the system can be selected	Allows users to view spending by specific categories
Total Expenses	Numeric (currency format)	Conditional	output	N/A	Summarizes the total amount spent and converts different currencies into the default, helping users see overall expenditure.
Percentage	Numeric (percentage format)	Conditional	Output	N/A	Calculate the percentage of each category expense to the whole total expenses of the period
Expense Trend Graph	Visual chart	Conditional	Output	Dynamic based on the user's date and category selection.	Help users visually track spending patterns across chosen periods.
Transactions List	List (table view with details like Date, Amount, Category, Payment Method)	Conditional	Output	Limited to selected filters.	Provides detailed insight into each transaction within the analysis scope.
Export option	Button (exports to CSV, Excel, or PDF)	Conditional	Input	N/A	Enables users to keep a record or conduct further external analysis

Confirmation Message	Text	Conditional	Output	N/A	Displays confirmation after export or analysis.
Error Message	Text	Conditional	Output	N/A	Displays messages for invalid input or no transactions available.

Table 42 : Analysis expense interface components.

### 3.3.1.1.9 Account Linking Interface

This interface should provide the services required to initiate a link between the Tracker and bank account for a better and more convenient way of viewing transactions.

The following fields are necessary for the start of linking an account:

Field Name	Format	Requirements	I/O	Constraints	Comments
Bank Name	Dropdown	Required	Output	Only official banks are selectable	Displays and allows selection of official banks
Link	Button	Required	Input	Must choose a bank first	Redirects to selected bank's interface
Confirmation Message	Text	Conditional	Output	Displays upon successful entry	Displays "Redirecting to the bank's official page" after successful connection is established



Error Message	Text	Conditional	Output	Displays if an error occurs during the redirection process	Displays “Error: Connection was cut off during linking process”
---------------	------	-------------	--------	--	---

Table 43 Account Linking Interface Components.

### 3.3.1.1.10 Password Recovery Interface

This interface should assist in renewing the account’s password by either recovering the old one or creating a new one. The main key requirement is the email address associated with the account.

The following fields will suffice for creating this interface:

Field Name	Format	Requirements	I/O	Constraints	Comments
Email	Text	Required	Input	Must be a valid email address and registered in the database	The email address is written for the system to search for
Password Reset	Button	Required	Input	Must input the email address before pressing	Sends a verification request to the written email address
Verification Code	Text	Required	Input	The exact code sent to the email address must be written	Verifies the email address’ association with the password reset request
Verify	Button	Required	Input	Accessible after a code is written	Cross-references the input code with the one sent to the email address
Confirmation Message	Text	Conditional	Output	Displays upon successful entry	Displays “Password has been reset” after a successful process
Error Message	Text	Conditional	Output	Displays if the code entered is incorrect	Displays “Error: Verification code is incorrect”

Table 44 Password Recovery Interface Components.

4.

# Chapter 4: SDS

## 4.1 Introduction

Expense Tracker Software Design Specification (SDS) Document the Software Requirements Specification (SRS) is translated into the data, architecture, interfaces, and components needed to develop and finish the Expense Tracker system in this document, the Software Design Specification (SDS). It describes how the entire system is set up and developed to meet the needs of the Expense Tracker and guarantee a productive application. There will be two stages to the SDS:

- **Initial Design Phase:** This phase, which consists of the first five sections, is devoted to designing the data interfaces and system components.
- **Detailed idea Phase:** We build on the original idea in this phase by offering a comprehensive demonstration that includes component designs and a thorough system blueprint.

The subjects pertaining to the expense are introduced in this section of the SDS. It outlines the document's purpose, scope, definitions, acronyms, abbreviations, and references used.

### 4.1.1 Purpose

The Software Design Specification (SDS) document aims to offer a complete description of the Expense Tracker system, including its user interface, architectural design, and data implementation. The system designs should clearly show how the system responds to user interactions and meets the criteria provided in the previous Software criteria Specification (SRS) document.

4.1.2 Scope

The Software Requirements Specification (SRS) document will detail all of the design elements that are aligned with the proposed services and traits that distinguish the Expense Tracker from other systems. This paper contains the system's architecture, prototypes, and logical design to demonstrate the interface flow, as well as extensive descriptions and pseudocode for the system's major features. It also includes interactions between entities, data processes, and data modeling using sequence diagrams, context data flow diagrams, data flow diagrams (DFD), and entity-relationship diagrams (ERD).

4.1.3 Definitions, Acronyms, and Abbreviations

Terminology	Definition
Software	The programs, routines, and symbolic languages that control the functioning of the hardware and direct its operation.
Database	A database is an organized collection of structured information, or data, typically stored electronically in a computer system.
Use case diagram	Used to show the relation between a set of actions (cases) that might occur in the system with the user (actor).
Budget	A financial plan that sets limits on spending within various categories (e.g., groceries, rent) based on income and financial goals.
Financial Management	The practice of overseeing and controlling income, expenses, and budgeting to achieve personal or financial goals.

Table 45 Terminology

Acronym	Definition
SDS	Software Design Specifications
SRS	Software Requirements Specifications
IEEE	Institute of Electrical and Electronics Engineers

ETA	Expense Tracker Application
UI	User Interface
PDF	Portable Document Format
APP	Mobile Application

Table 46 Acronyms

**4.1.4 References**

References used in this document:

[1] Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education Limited.

[2] IEEE Computer Society. (1998). IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE.

[3] Pressman, R. S. (2014). Software Engineering: A Practitioner's Approach (8th ed.). McGraw-Hill Education.

## 4.2 System overview

ETA is a comprehensive financial management solution that helps users monitor and manage their expenses and income. This section describes administrators' and users' basic system functionalities.

### 4.2.1 Administrator Skills

- 1- The Administrator oversees financial records and system user interactions.
- 2- Evaluation and management of user transactions, including expenses and income, are essential. As needed, modify, approve, or reject transactions.
- 3- Classifying financial data by category or date period for administrative purposes.
- 4- Contributing financial data for system reliability.
- 5- Monitoring application utilization and system integrity.

### 4.2.2 User Skills

- 1- User resources help individuals manage their money. Essential functions include Registration and Authentication: Create a secure account and recover lost passwords using a dependable mechanism.
- 2- Use two-factor authentication to secure logins.
- 3- Manage expenses and income by adding, modifying, and removing expenditures with date, amount, payment method, and remarks.
- 4- Fund accounts with income sources and information.
- 5- Budgeting and notifications: Set category budgets and get notifications when spending exceeds criteria.
- 6- Get expense alerts and financial summaries.
- 7- Financial Visualization: Chart and graph expenses.
- 8- Export PDF or CSV reports for selected dates.
- 9- Automate transaction import by connecting bank accounts.
- 10- Manage multi-currency transactions with quick conversion.

### 4.2.3 System Functions

- 1- The safe, user-centric system has features to improve usability and data management, including a mobile interface for simple financial monitoring.
- 2- Encrypt any sensitive data.
- 3- Connecting to external banking APIs simplifies data import.
- 4- Offline data export and reporting tools.

## 4.3 Design Considerations

This section covers the assumptions, dependencies, and general restrictions that affect the Expense Tracker Application (ETA)'s design and development.

### 4.3.1 Assumptions and Dependencies

System assumptions and dependencies are classed as:

#### 4.3.1.1 Related Software/Hardware

- The Expense Tracker targets Android and iOS smartphones.

Local data storage using SQLite will make application database administration lightweight and efficient.

Android Studio and Xcode will be used for Android and iOS development. Debugging and UI design are efficient on both platforms.

Figma and Adobe XD help build user-friendly UI/UX.

#### 4.3.1.2 Operating System

- Compatible with Android 8.0 (Oreo) and iOS 12+.
- The development team will code, test, and deploy on Windows, MacOS, and Linux to ensure cross-platform compatibility.

#### 4.3.1.3 End-User Characteristics

- The software targets average person's managing personal money.
- Basic mobile app literacy and financial terms (budgets, costs, categories) are required.
- The system offers guided onboarding and tooltips for non-technical users.

#### 4.3.1.4 Potential Functional Changes

- Based on user input, more currencies or statistics may be added.
- Changes will improve user experience without impacting workflow. To maintain stability, development timeframes and testing processes will limit major alterations.

### 4.3.2 General Constraints

This section explores software design constraints:

#### 4.3.2.1 Hardware/Software Environment

- The program needs a minimum of 2GB RAM and 200MB storage for optimal performance on mobile devices.
- Developers require 8GB RAM or more to use Android Studio or Xcode.
- To test performance across setups, several cellphones will be used.

#### 4.3.2.2 Interface Needs

- To serve varied users, the design will emphasize simplicity and clarity:
- Use legible fonts and sizes.
- Icons and color-coded visualizations will help consumers navigate crucial functions.
- Drop-down menus, toggles, and auto-fill fields reduce user effort.
- Labels, Tooltips: Clear labeling and contextual tooltips improve usability.

#### 4.3.2.3 Repository and Distribution Needs

- SQLite will handle local storage for offline access.
- Users may export reports to CSV or PDF for external data analysis.
- Secure API integrations will import transactions if bank account linkage is enabled.

#### 4.3.2.4 Security Needs

- Users will be authenticated using strong passwords and optional two-factor authentication.
- User data including account passwords and financial information will be secured using AES-256.
- A password recovery procedure will need email or phone verification.

#### 4.3.2.5 Performance Standards

- The program will keep response times for basic tasks like inputting costs and creating reports under 1 second.
- To guarantee reliable performance during thousands of transactions, stress testing will be done.
- App battery life and data use will be improved.

#### 4.3.2.6 Validation and Verification Needs

- A thorough system test will verify functional requirements and user scenarios.
- Automated and human testing will find and fix bugs early in development. User acceptability testing (UAT) ensures product meets customer expectations.



## 4.4 User Interface Design

### 4.4.1 Overview of User Interface

The Expense Tracker applications offers a smooth financial management experience from the user's point of view. Registering, logging in, tracking spending, setting budgets, viewing reports, and analyzing spending trends are all available to users. Users are able to add expenses, categorize entries, set budget limitations, and export financial data through the interface. Notifications, confirmation messages, and alerts—such as reminders to record daily costs or cautions if they over predetermined budget limits—provide users with instant feedback.

Examples of feedback:

1. Alerts of extraordinary spending or budgetary restrictions.
2. Success and error messages that let users make corrections or validate actions (e.g., "Expense added successfully" and "Invalid entry").
3. Visual representations (charts and graphs) that summarize spending patterns and facilitate simple tracking.

### 4.4.2 Interface Design Rules

Interface rules of design for consistency and usability include:

- Consistency in the application's color schemes, fonts, and icons to guarantee user-friendly navigation.
- Layout minimalism, with distinct areas for important tasks (such as viewing reports or adding spending), guarantees simple access to essential features.
- Readability is ensured by accessibility through the use of bigger fonts and colors with strong contrast.
- Action feedback in the form of prompt textual or graphic reactions to user actions.
- Mobile-friendly layout changes to accommodate different screen sizes because it's a mobile app.

### 4.4.3 Screen Images

Based on the interfaces we mentioned in our previous SRS document we will sketch an initial layout of the user perspective.

#### 1) Start-up Selection Interface



Figure 12 Start-up Selection Interface

#### 1.1) Registration interface



Figure 13 Registration interface

1.1) Login interface

Login

Email

Password

Back

Forget Password

Login

Figure 14 Login interface

1.2.1) Password recovery interface

Password Recovery

Email

send reset link

Enter Verification code

Verify

Password Recovery

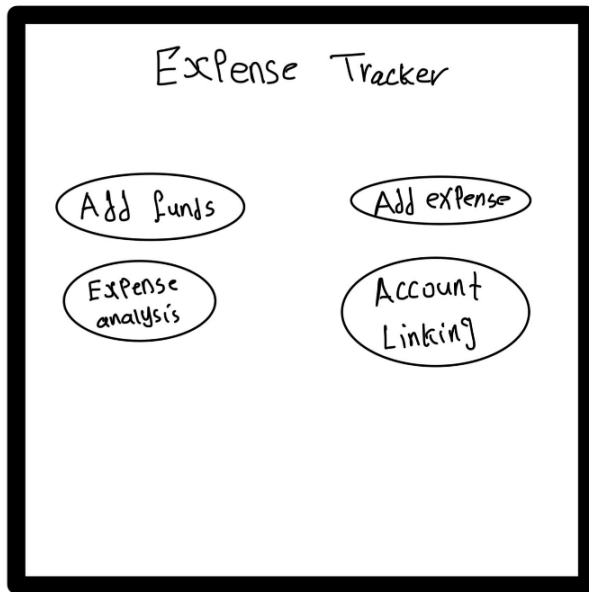
New Password

confirm Password

Reset

Figure 15 Password recovery interface

2) Selection interface



*Figure 16 Selection interface*

2.1) Adding expense interface

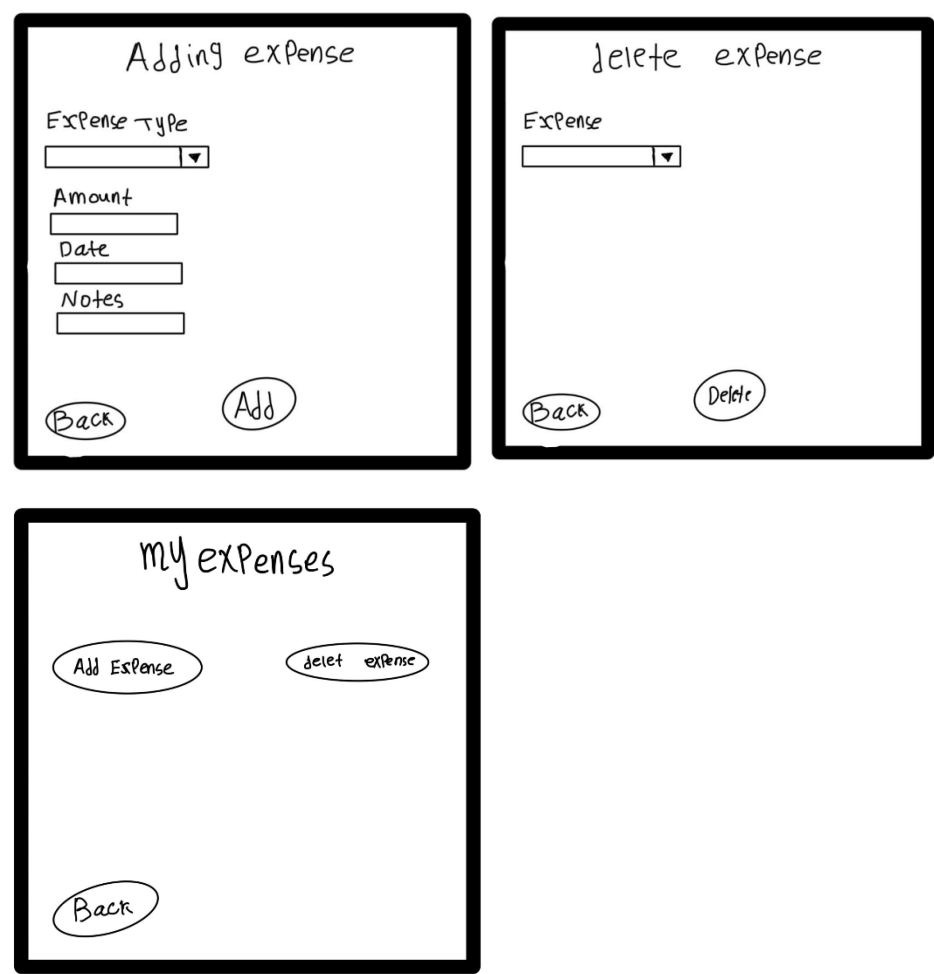


Figure 17 Adding expense interface

## 2.2) Add funds interface

Adding Fund

Fund Type

Amount

Date

Notes

Back Add

Figure 18 Add funds interface

## 2.3) bank account linking interface

Bank Account Linking

Bank

Back Bank website

Figure 19 bank account linking interface

## 2.4) Analysis expenses interface

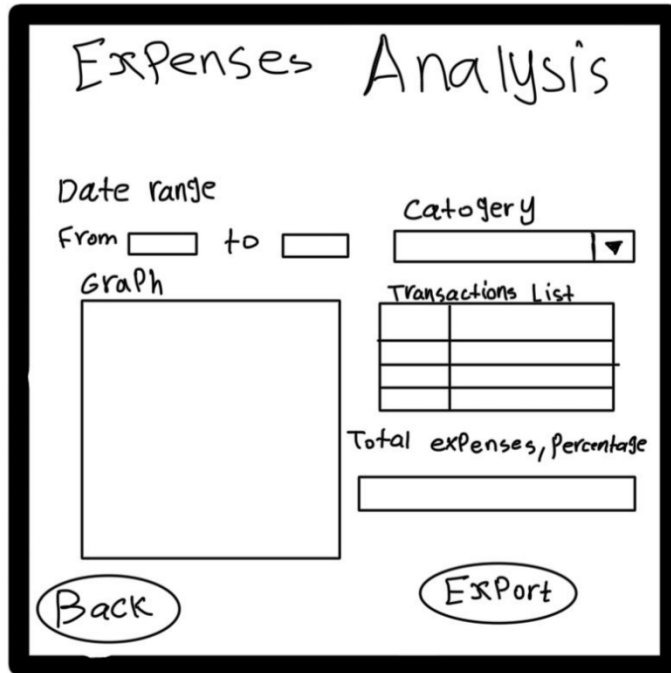


Figure 20 Analysis expenses interface

### 4.4.4 Screen Objects and Actions

In this section we will explain the objects and actions of the previous interfaces. One common button that included in the most of the interfaces is the Back button which navigates to the previous interface.

#### 4.4.4.1.1 Selection Interface

- **Screen Objects:**
  - **Login Button**
    - **Action:** Navigates to the Login Interface.
  - **Register Button**
    - **Action:** Navigates to the Registration Interface.
  - **Logo/Text:**
    - **Action:** Displays app branding (no user interaction).

#### 4.4.4.1.2 Registration Interface

- **Screen Objects:**
  - **First Name, Last Name (Text Fields):**
    - **Action:** Accepts user input (letters only).
  - **Email (Text Field):**
    - **Action:** Accepts user email (validates format).
  - **Phone Number (Numeric Field):**
    - **Action:** Accepts numeric input only.
  - **Password & Confirm Password (Password Fields):**
    - **Action:** Accepts and masks user input; validates for matching passwords.
  - **Register Button:**
    - **Action:** Submits the form if all fields are valid.
  - **Error Message (Text):**
    - **Action:** Displays issues with input (e.g., "Invalid email format").

#### 4.4.4.1.3 Login Interface

- **Screen Objects:**
  - **Email (Text Field):**
    - **Action:** Accepts user email (validates format).
  - **Password (Password Field):**
    - **Action:** Accepts and masks input (validates against stored password).
  - **Login Button:**
    - **Action:** Authenticates user and redirects based on role (Admin/User).
  - **Forgot Password (Link/Button):**
    - **Action:** Navigates to the Password Recovery Interface.
  - **Error Message (Text):**
    - **Action:** Displays login errors (e.g., "Invalid credentials").

#### 4.4.4.1.4 Password Recovery Interface

- **Screen Objects:**
  - **Email (Text Field):**
    - **Action:** Accepts registered email for verification.
  - **Send Reset link Button:**
    - **Action:** Sends a verification request.
  - **Verification Code (Text Field):**
    - **Action:** Accepts the code sent to the email.
  - **Verify Button:**
    - **Action:** Verifies code and resets password.
  - **Error Message (Text):**
    - **Action:** Displays errors with verification.
  - **Password & Confirm Password (Password Fields):**
    - **Action:** Accepts and masks user input; validates for matching passwords.
  - **Confirmation Message (Text):**
    - **Action:** Displays success after resetting password.



#### 4.4.4.1.5 Selection Interface

- **Screen Objects:**
  - **Add Funds Button:**
    - **Action:** Navigates to the Add Funds Interface.
  - **Add Expense Button:**
    - **Action:** Navigates to the Adding Expense Interface.
  - **Account linking:**
    - **Action:** Navigate to the account linking Interface.
  - **Expense analysis:**
    - **Action:** Navigate to the analysis expenses Interface.
  - **Welcome Message (Text):**
    - **Action:** Displays user greeting (no interaction).

#### 4.4.4.1.6 Adding Expense Interface

- **Screen Objects:**
  - **Delete Expense Button:**
    - **Action:** Navigate to delete expense interface.
  - **Add Expense Button:**
    - **Action:** Navigate to add expense interface.

#### 4.4.4.1.7 Add expense

- **Expense Type (Dropdown):**
  - **Action:** Allows user to select from predefined categories (e.g., Food, Transport).
- **Amount (Numeric Field):**
  - **Action:** Accepts positive numbers.
- **Date (Date Picker):**
  - **Action:** Allows selection of a valid date.
- **Notes (Text Field):**
  - **Action:** Allows optional input (up to 250 characters).
- **Add Button:**
  - **Action:** Saves the expense if inputs are valid.
- **Error Message (Text):**
  - **Action:** Displays issues with input (e.g., "Invalid expense entry").
- **Warning Message (Text):**
  - **Action:** Displays warnings if expense exceeds budget.

#### 4.4.4.1.8 Delete expense

- **Expense list (Dropdown):**
  - **Action:** Allows users to select from existing expenses.
- **Delete Expense Button:**
  - **Action:** Deletes selected expense entry.

#### 4.4.4.1.9 Add Funds Interface

- **Screen Objects:**
  - **Source of Income (Dropdown):**
    - **Action:** Allows user to select income source (e.g., Salary, Gift).
  - **Amount (Numeric Field):**
    - **Action:** Accepts positive numbers.
  - **Date (Date Picker):**
    - **Action:** Allows selection of a valid date.
  - **Notes (Text Field):**
    - **Action:** Allows optional input (up to 250 characters).
  - **Add Funds Button:**
    - **Action:** Saves the funds entry if inputs are valid.
  - **Error Message (Text):**
    - **Action:** Displays issues with input (e.g., "Invalid funds entry").

#### 4.4.4.1.10 Account Linking Interface

- **Screen Objects:**
  - **Bank Name (Dropdown):**
    - **Action:** Allows selection of a bank.
  - **Link Button:**
    - **Action:** Initiates the linking process with the selected bank.
  - **Confirmation Message (Text):**
    - **Action:** Displays success for linking.
  - **Error Message (Text):**
    - **Action:** Displays errors during linking.

#### 4.4.4.1.11 Analysis Expenses Interface

- **Screen Objects:**
  - **Date Range (Date Picker):**
    - **Action:** Filters analysis based on a specific time frame.
  - **Category Filter (Dropdown):**
    - **Action:** Filters analysis by expense categories.
  - **Total Expenses (Text):**
    - **Action:** Displays the calculated total of expenses.
  - **Percentage (Text):**
    - **Action:** Displays the percentage of each category in total expenses.
  - **Expense Trend Graph (Chart):**
    - **Action:** Dynamically updates based on selected filters.
  - **Transactions List (Table):**
    - **Action:** Displays filtered transaction details.
  - **Export Button:**
    - **Action:** Exports data as CSV or PDF.

#### 4.4.4.2 Other Interfaces

This section describes additional interfaces not covered in the prior sections. Each interface includes details about its design, enabling technologies, protocols, and interaction specifics.

##### 4.4.4.2.1 Notifications and Alerts Interface

**Description:**

Provides users with real-time notifications about budgets, spending limits, and reminders to log expenses.

**Interface Design:****1. Technology Used:**

- **Mobile Notifications API** for push notifications (e.g., Firebase Cloud Messaging for Android/iOS).
- **Backend:** REST API to send and manage notification data.
- **Frontend:** React Native or equivalent framework to display notifications.

**2. Protocol:**

- **Request Format:** JSON over HTTPS (for sending notification data).
- **Response Format:** JSON acknowledgment.

**3. Error Conditions:**

- **No Internet:** Display an error message, "Unable to fetch notifications. Check your internet connection."
- **Invalid User ID:** Server responds with an error message: `{"error": "User not found"}`.

**4. Initiation and Closure:**

- Notifications are initiated from the backend based on triggers (e.g., user exceeds a budget limit).
- Users can clear notifications on the app, and read statuses are updated in the backend.

#### 4.4.4.2.2 Data Export Interface

**Description:**

Enables users to export financial data (expenses, budgets) to formats like CSV or PDF.

**Interface Design:****1. Technology Used:**

- **Backend:** REST API for data generation.
- **Frontend:** Client-side library for file downloads (e.g., FileSaver.js).
- **Export Formats:** CSV (comma-separated values) and PDF (via libraries like PDFKit or jsPDF).

**2. Protocol:**

- **Request Format:** JSON over HTTPS.
- **Response Format:**
  - CSV: Plain text data with comma-separated fields.
  - PDF: Binary data stream for PDF rendering.

**3. Error Conditions:**

- **Invalid Date Range:** Server responds with `{"error": "Invalid date range selected"}`.
- **Unsupported Format:** Server responds with `{"error": "Export format not supported"}`.

**4. Initiation and Closure:**

- User triggers export via a button in the **Analysis Expenses Interface**.
- The backend generates the requested file, which is downloaded to the user's device.

#### 4.4.4.2.3 Currency Conversion Interface

**Description:**

Allows users to input expenses in different currencies and converts them to the default currency based on real-time exchange rates.

**Interface Design:****1. Technology Used:**

- **API:** Third-party currency exchange APIs (e.g., OpenExchangeRates or XE Currency API).
- **Backend:** RESTful services to fetch and cache exchange rates.
- **Frontend:** Input fields with dropdowns for currency selection.

**2. Protocol:**

- **Request:** The system sends the amount, source currency, and target currency to the currency conversion API.
- **Response:** The API returns the converted amount and the current exchange rate.

**3. Error Conditions:**

- **API Timeout:** Displays a message, "Unable to fetch exchange rates. Please try again later."
- **Unsupported Currency:** Displays an error, "Selected currency is not supported."

**4. Initiation and Closure:**

- Initiated when a user selects a non-default currency during expense entry.
- Exchange rates are fetched and displayed, and the converted amount is stored in the system.

#### 4.4.4.2.4 Bank Account Integration Interface

**Description:**

Links users' bank accounts to automatically import transactions for analysis.

**Interface Design:****1. Technology Used:**

- **API:** Bank API integration (e.g., Plaid, Yodlee).
- **Authentication:** OAuth 2.0 for secure access.

**2. Protocol:**

- **Handshake:** OAuth handshake to link user accounts.

- **Request:** The system sends the bank ID and OAuth token to the bank's API for verification.
- **Response:** The API confirms the connection and provides the linked account details (e.g., account number or identifier).

### 3. Error Conditions:

- **Invalid Credentials:** Displays "Authentication failed. Please check your credentials."
- **Connection Timeout:** Displays "Unable to connect to the bank. Try again later."

### 4. Initiation and Closure:

- Users initiate the linking process via the **Account Linking Interface**.
- Transactions are imported after a successful link and displayed in the **Analysis Expenses Interface**.

## 4.5 System Architecture

This part offers a summary of the Expense Tracker app structure by outlining how its tasks and roles are divided among different parts and elements of the system. The goal is to explain the design approach and demonstrate how every component of the system works together to achieve the necessary functionality, while also explaining why the selected architectural design approach was chosen.

### 4.5.1 Architectural Design Approach

The Expense Tracker uses a modular design for scalability, maintainability and simplicity. The architecture is based on key functional modules focusing on user interactions and efficient data management. The system is designed around a component-based model and each module covers certain functionalities e.g. user management, expense tracking, budget monitoring and reporting.

This approach ensures:

**Separation of Concerns:** Each module performs its task independently, massively improving system clarity and reducing interdependencies.

**Ease of Maintenance:** Updates and feature addition are streamlined without interfering with other modules.

**Scalability:** The architecture can easily support new features e.g. external APIs interactions.

Secure data handling, fast communication between components and a user-friendly interface are the priorities of the design. The architectural model ensures a smooth data flow from user inputs to data storage and retrieval processes, strengthening the system's reliability and usability.

4.5.2 Architectural Design

This section describes the architectural structure of the Expense Tracker. It will include a system diagram with the main components and interactions, a description of each subsystem's purpose and functionality, and the relationships between subsystems.

4.5.2.1 System Diagram

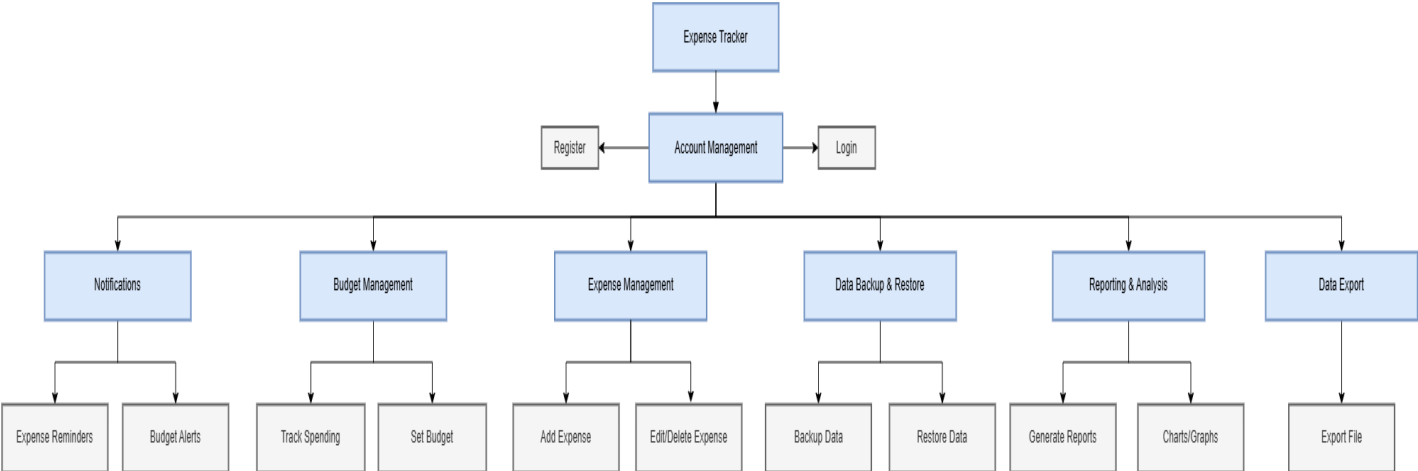


Figure 21 System Diagram

4.5.2.2 Subsystem Overview

- **User Management:** Handles user registration, login, and session management.
- **Expense Management:** Manages expense entry, editing, deletion, and categorization.
- **Budget Management:** Supports budget creation, monitoring, and alerts.
- **Reporting & Analysis:** Generates visual and tabular reports for user insights.
- **Notifications:** Sends reminders for expense logging and budget thresholds.
- **Data Backup & Restore:** Facilitates secure data backups and recovery processes.
- **Export Data:** Enables users to select specific data (e.g., expenses, budgets) and export it in formats like CSV or PDF.

4.5.2.3 Interconnections and Relationships

- 1- User Management connects to all modules as it governs user access
- 2- Expense Management provides data for Budget Management and Reporting & Analysis.
- 3- Notifications rely on data from Budget Management and Expense Management.
- 4- Data Backup & Restore interacts with all other subsystems to ensure data integrity.
- 5- Data Export connects with Expense Management and Budget Management to gather data for export.

4.5.3 Subsystem Architecture

This section will describe the system's functionality and significant processes and how the data is saved on the Expense Tracker database server.

4.5.3.1 User Management Subsystem

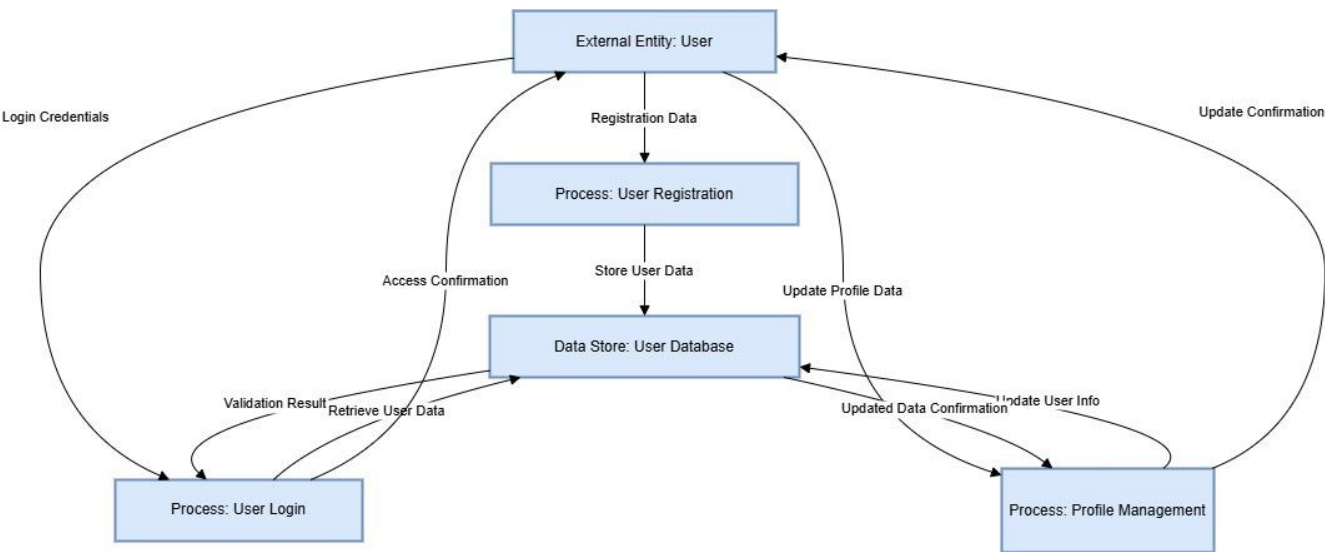


Figure 22 User Management Subsystem DFD

The DFD showcases the User Management Subsystem, where the User interacts with processes like Registration, Login, and Profile Management. These processes store, retrieve, and update information in the User Database, ensuring secure data flow for account creation, access validation, and profile updates.



4.5.3.2 Expense Management Subsystem

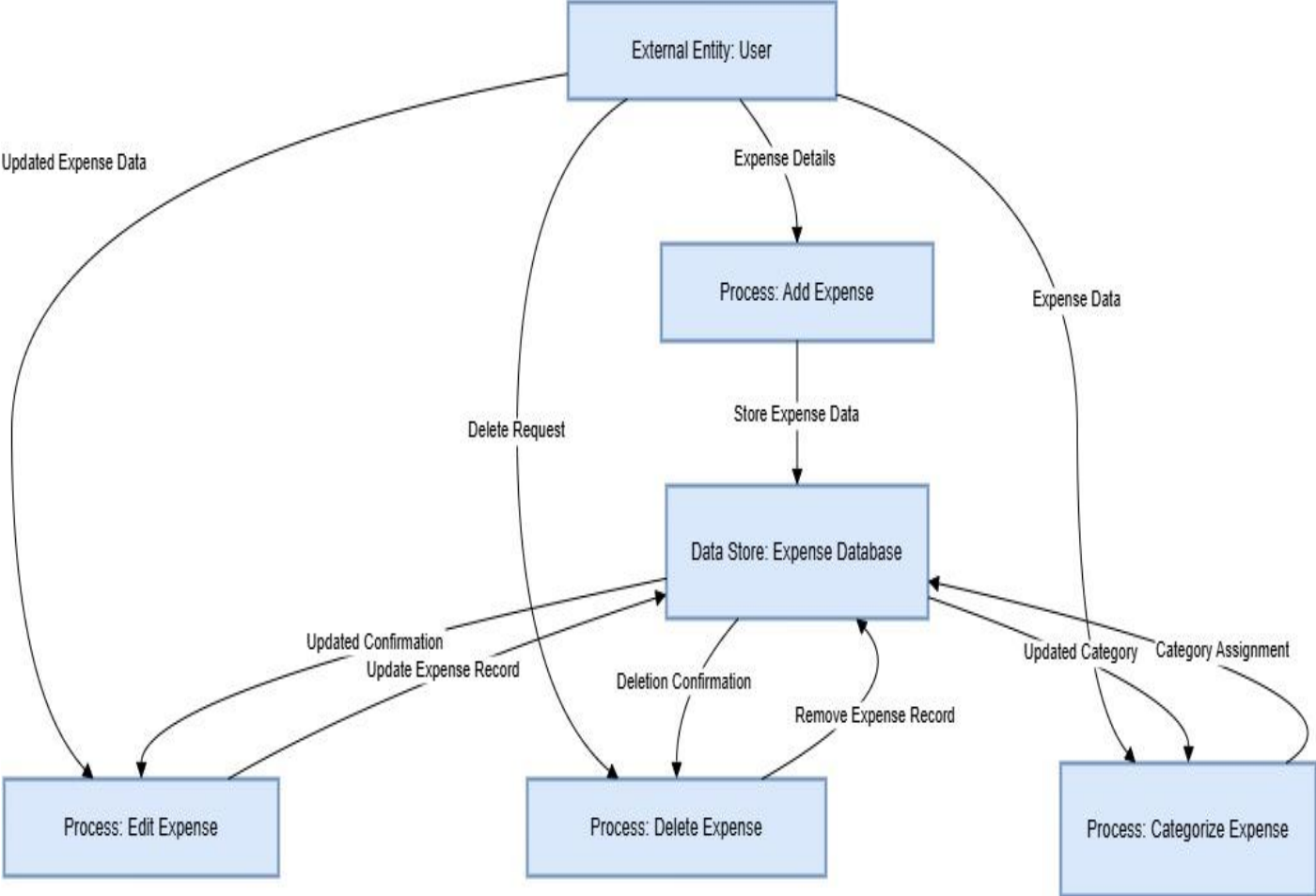


Figure 23 Expense Management Subsystem DFD

The DFD for the Expense Management Subsystem shows how the User interacts with processes like Add Expense, Edit Expense, Delete Expense, and Categorize Expense. These processes store, update, or remove data in the Expense Database, ensuring efficient management of expense records and categories.

4.5.3.3 Budget Management Subsystem

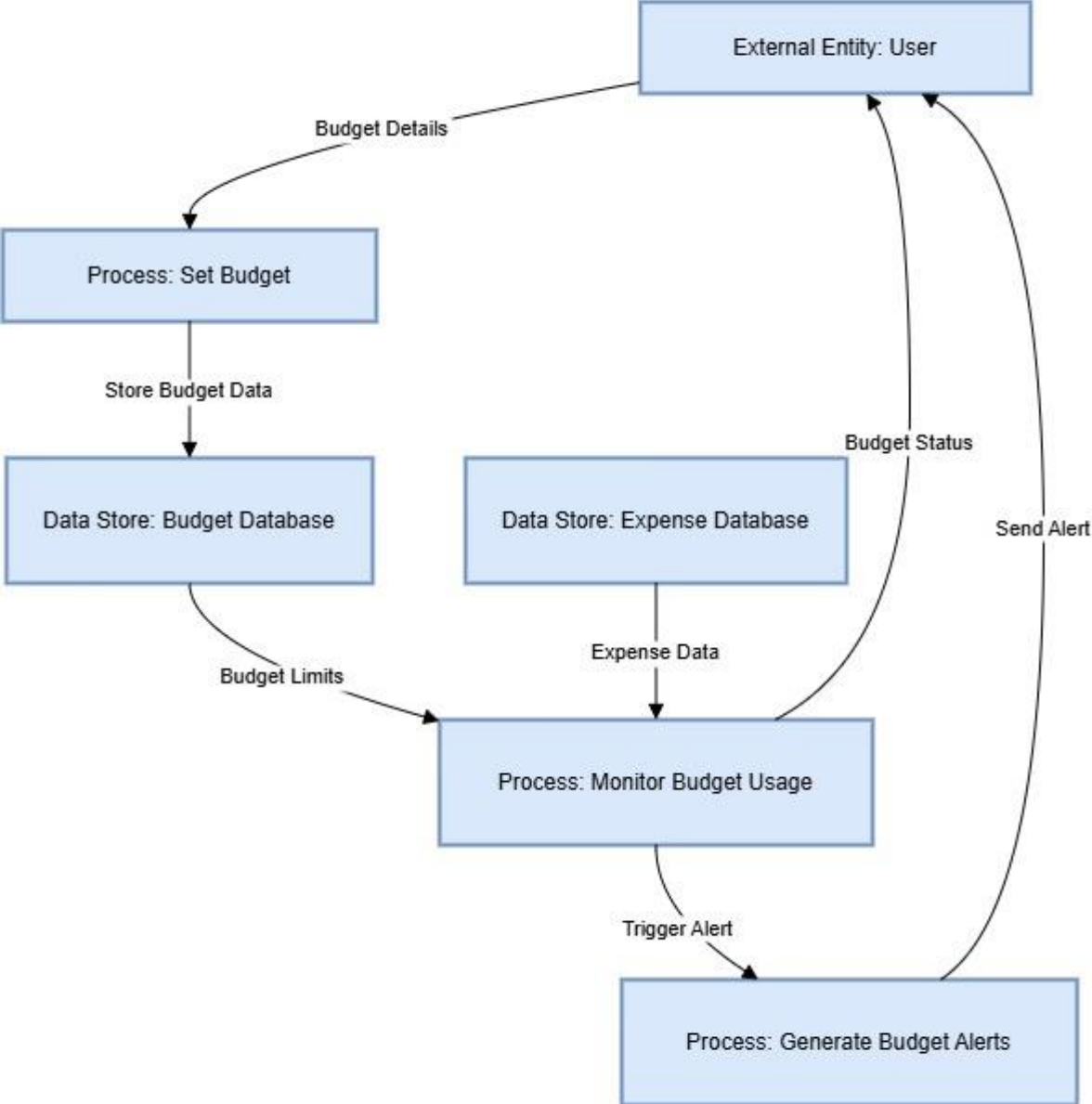


Figure 24 Budget Management Subsystem DFD

The Budget Management Subsystem DFD shows how the User sets budgets, which are stored in the Budget Database. Expenses from the Expense Database are compared to budget limits to monitor usage. If thresholds are exceeded, alerts are generated and sent to the user.

4.5.3.4 Reporting & Analysis Subsystem

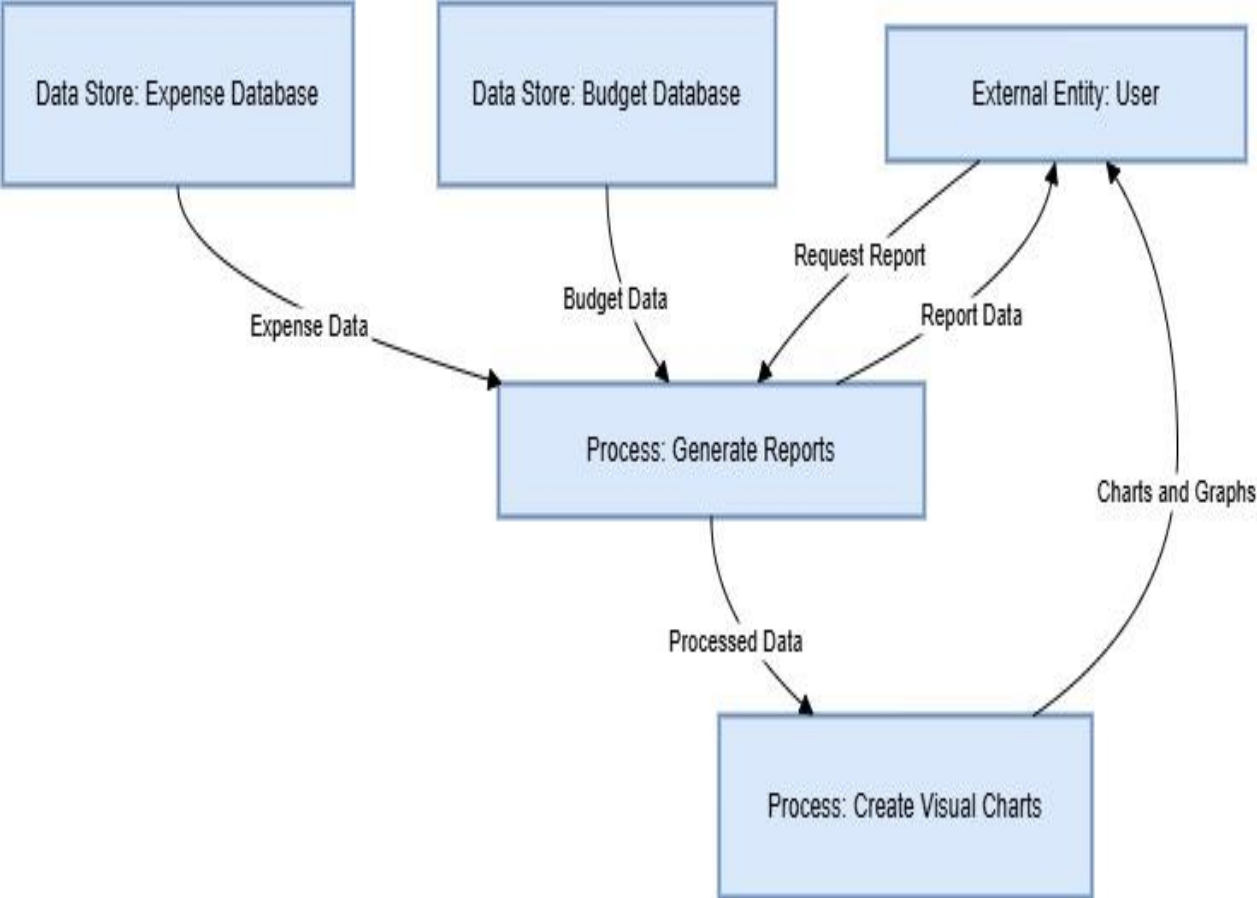


Figure 25 Reporting and Analysis Subsystem DFD

The Reporting and Analysis Subsystem DFD illustrates how the User requests reports based on data from the Expense Database and Budget Database. The processed data generates visual charts, which are presented to the user for spending analysis.

4.5.3.5 Notification Subsystem

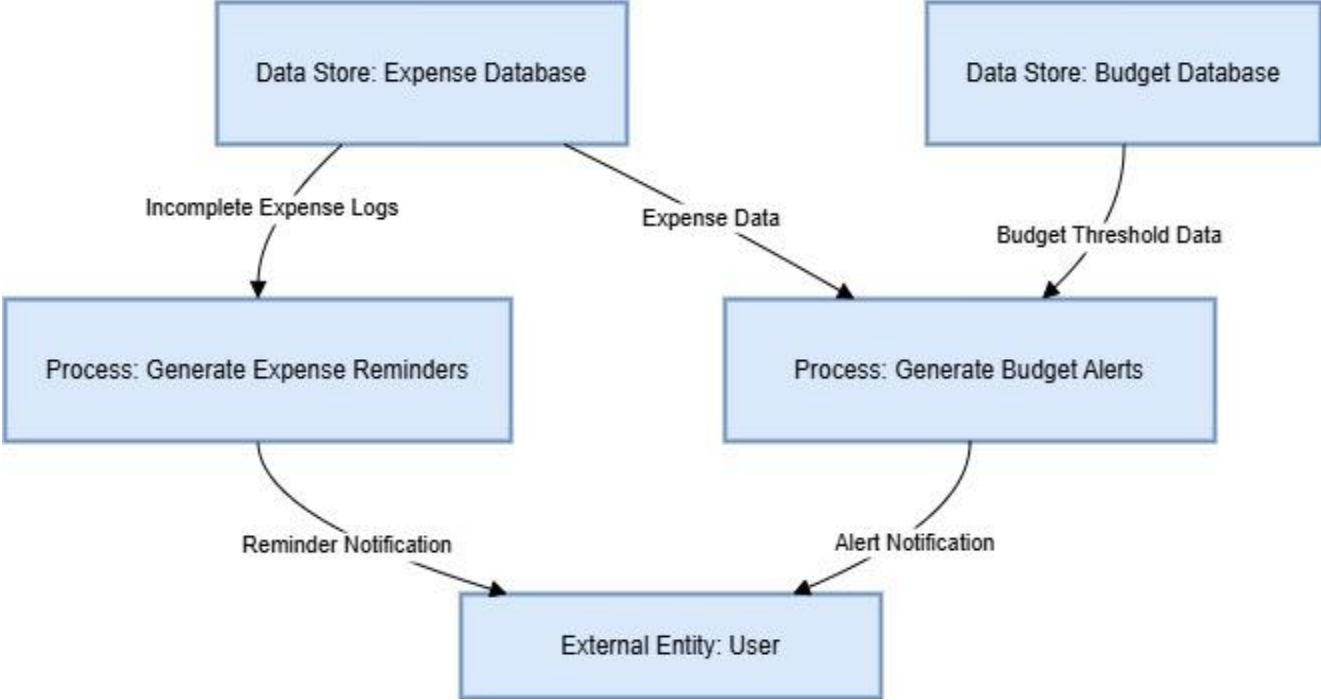


Figure 26 Notifications Subsystem DFD

The Notifications Subsystem DFD demonstrates how the system generates Expense Reminders using data from the Expense Database and Budget Alerts by comparing expense and budget data from the respective databases. Notifications are then sent to the User to ensure timely updates and budget adherence.

#### 4.5.3.6 Data Backup & Restore Subsystem

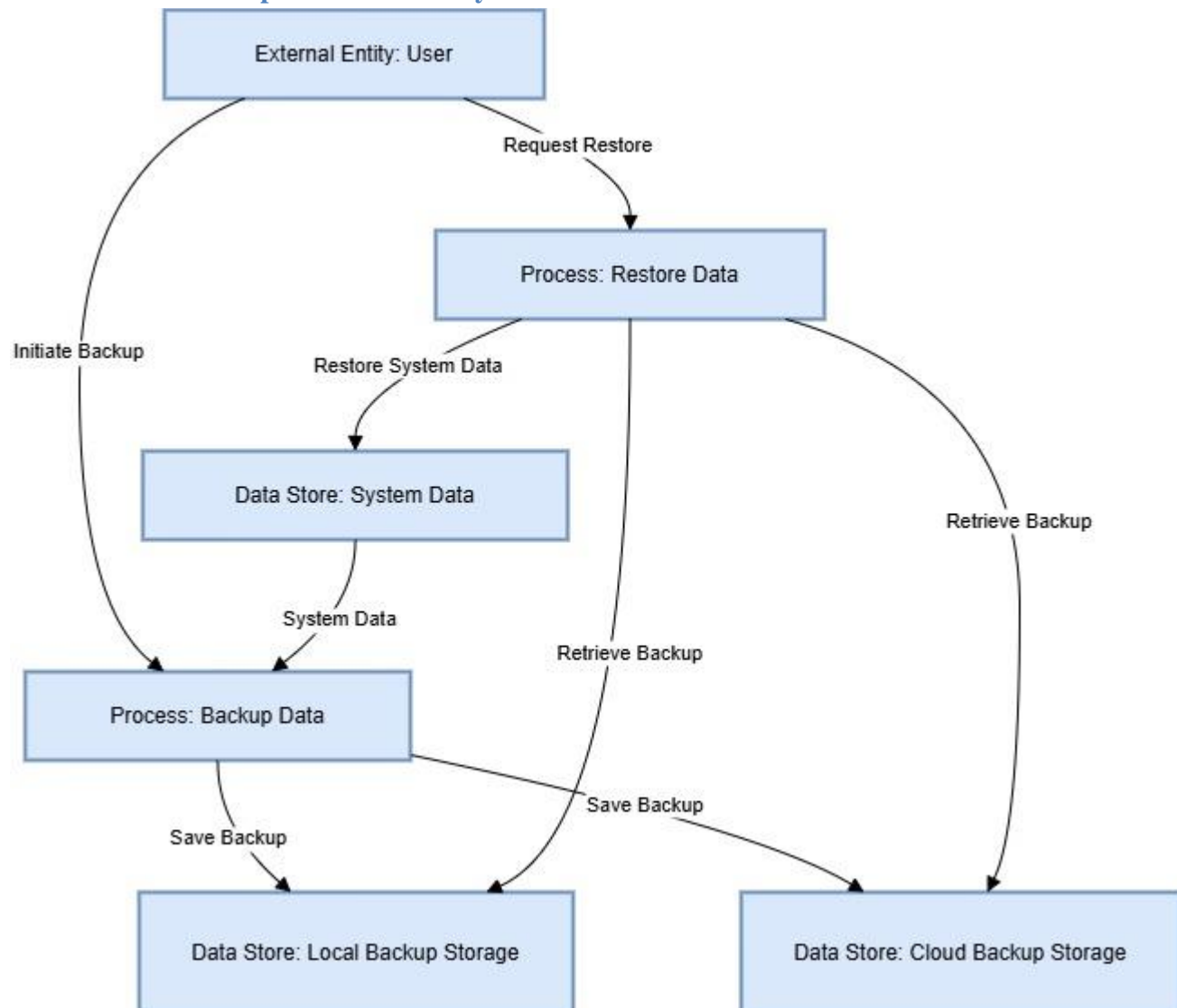


Figure 27 Data Backup & Restore Subsystem DFD

The Data Backup & Restore Subsystem DFD shows how the User initiates data backups, which save system data to both Local Backup Storage and Cloud Backup Storage. When a restore is requested, the system retrieves the backup from the specified storage and restores the data to the System Data repository.

4.5.3.7 Export Data Subsystem

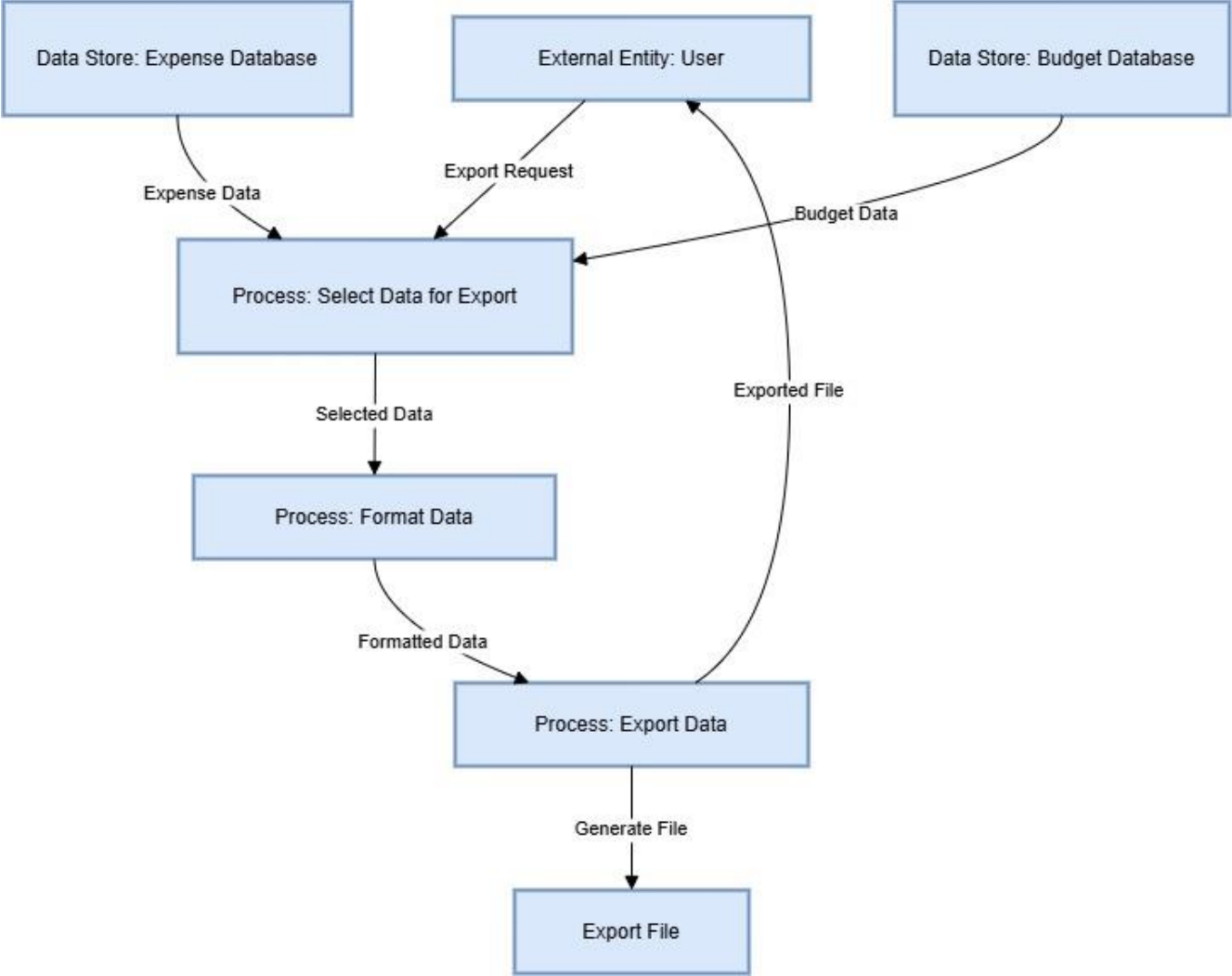


Figure 28 Export Data Subsystem DFD

The Export Data Subsystem DFD shows how the User selects data for export from the Expense Database and Budget Database. The system formats the selected data and generates an export file in a format like CSV or PDF, which is then delivered to the User and stored as an Export File.

## 4.6 Data Design

This section provides data description, data dictionary and database description

### 4.6.1 Data Description

This subsection provides Our Expense Tracker database entities, their fields including the data type and constraint for each field including if they are a primary or foreign key.

Entity	Attribute	Attribute Type	P K	F K	Constraints
User	user_id	INT	✓		AUTO_INCREMENT, NOT NULL
	first_name	VARCHAR(50)			NOT NULL
	last_name	VARCHAR(50)			NOT NULL
	email	VARCHAR(100)			NOT NULL , UNIQUE
	phone_number	VARCHAR(15)			NOT NULL
	password	VARCHAR(255)			NOT NULL
	role	ENUM('user','admin')			DEFAULT 'user'
	password_reset_token	VARCHAR(255)			NULL
	token_expiry	TIMESTAMP			NULL
Category	category_id	INT	✓		AUTO_INCREMENT, NOT NULL
	Name	VARCHAR(50)			NOT NULL , UNIQUE
	description	VARCHAR(100)			NULL
Bank	bank_id	INT	✓		AUTO_INCREMENT, NOT NULL
	bank_name	ENUM('Al Rajhi Bank', 'Saudi National Bank', 'Riyad Bank', 'Bank Albilad', 'Alinma Bank', 'Banque Saudi Fransi', 'Arab National Bank')			NOT NULL , UNIQUE
	bank_url	VARCHAR(255)			NOT NULL
LinkedAccount	linked_account_id		✓		AUTO_INCREMENT, NOT NULL
	user_id	INT		✓	FOREIGN KEY REFERENCES User(user_id), NOT NULL
	bank_id	INT		✓	FOREIGN KEY REFERENCES Bank(bank_id), NOT NULL
	link_date	TIMESTAMP			DEFAULT CURRENT_TIMESTAMP
Budget	budget_id	INT	✓		AUTO_INCREMENT, NOT NULL

	user_id	INT		✓	FOREIGN KEY REFERENCES User(user_id), NOT NULL
	amount	DECIMAL(10,2)			NOT NULL
	current_balance	DECIMAL(10,2)			NOT NULL
	period_type	ENUM ('WEEKLY','MONTHLY', 'CUSTOM')			NOT NULL
	start_date	DATE			NOT NULL
	end_date	DATE			NOT NULL
	status	ENUM('active', 'expired', 'archived')			DEFAULT 'active'
Expense	Expense_id		✓		AUTO_INCREMENT, NOT NULL
	user_id	INT		✓	FOREIGN KEY REFERENCES User(user_id), NOT NULL
	category_id	INT		✓	FOREIGN KEY REFERENCES Category(category_id), NOT NULL
	amount	DECIMAL(10,2)			NOT NULL
	currency	VARCHAR(3)			DEFAULT 'SAR'
	exchange_rate	DECIMAL(10,4)			DEFAULT 1.0
	payment_method	ENUM('cash', 'credit', 'debit', 'other')			DEFAULT 'cash'
	date	DATE			NOT NULL
	notes	VARCHAR(250)			NULL
Income	income_id	INT	✓		AUTO_INCREMENT, NOT NULL
	user_id	INT		✓	FOREIGN KEY REFERENCES User(user_id), NOT NULL
	source	VARCHAR(50)			NOT NULL
	amount	DECIMAL(10,2)			NOT NULL
	currency	VARCHAR(3)			DEFAULT 'SAR'
	exchange_rate	DECIMAL(10,4)			DEFAULT 1.0
	date	DATE			NOT NULL
	notes	VARCHAR(250)			NULL
Notifications	notification_id	INT	✓		AUTO_INCREMENT, NOT NULL
	user_id	INT		✓	FOREIGN KEY REFERENCES User(user_id), NOT NULL
	type	ENUM('warning', 'info', 'reminder')			DEFAULT 'info'



	message	VARCHAR(250)			NOT NULL
	status	ENUM('unread', 'read')			DEFAULT 'unread'
	date	TIMESTAMP			DEFAULT CURRENT_TIMESTAMP

Table 47: Data Base Description Table.

#### 4.6.2 Data Dictionary

Entity	Attribute	Description
User	user_id	Unique identifier for each user, primary key, auto-incremented.
	first_name	User's first name, required.
	last_name	User's last name, required.
	email	User's email address, required and unique.
	phone_number	User's phone number, required.
	password	Hashed user password, required.
	role	User's role (e.g., 'user' or 'admin'), defaults to 'user'.
	password_reset_token	Token for password reset functionality, nullable.
	token_expiry	Expiry date and time for the reset token, nullable.
Category	category_id	Unique identifier for each category, primary key, auto-incremented.
	name	Name of the category, required and unique.
	description	Description of the category, optional.
Bank	bank_id	Unique identifier for each bank, primary key, auto-incremented.
	bank_name	Name of the bank (predefined list), required and unique.
	bank_url	URL for the bank's website, required.
LinkedAccount	linked_account_id	Unique identifier for each linked account, primary key, auto-incremented.
	user_id	Foreign key referencing `user_id` in User, required.
	bank_id	Foreign key referencing `bank_id` in Bank, required.
	link_date	Date the account was linked, defaults to current timestamp.
Budget	budget_id	Unique identifier for each budget, primary key, auto-incremented.
	user_id	Foreign key referencing `user_id` in User, required.
	amount	Budgeted amount, required.
	current_balance	Current balance remaining in the budget, required.
	period_type	Type of budget period (e.g., 'WEEKLY', 'MONTHLY', 'CUSTOM'), required.
	start_date	Start date of the budget period, required.
	end_date	End date of the budget period, required.
	status	Status of the budget (e.g., 'active', 'expired', 'archived'), defaults to 'active'.
Expense	expense_id	Unique identifier for each expense, primary key, auto-incremented.

	user_id	Foreign key referencing `user_id` in User, required.
	category_id	Foreign key referencing `category_id` in Category, required.
	amount	Amount of the expense, required.
	currency	Currency of the expense, defaults to 'SAR'.
	exchange_rate	Exchange rate applied to the currency, defaults to 1.0.
	payment_method	Payment method (e.g., 'cash', 'credit', 'debit', 'other'), defaults to 'cash'.
	date	Date of the expense, required.
	notes	Additional notes for the expense, optional.
Income	income_id	Unique identifier for each income entry, primary key, auto-incremented.
	user_id	Foreign key referencing `user_id` in User, required.
	source	Source of the income, required.
	amount	Amount of the income, required.
	currency	Currency of the income, defaults to 'SAR'.
	exchange_rate	Exchange rate applied to the currency, defaults to 1.0.
	date	Date of the income, required.
	notes	Additional notes for the income, optional.
Notifications	notification_id	Unique identifier for each notification, primary key, auto-incremented.
	user_id	Foreign key referencing `user_id` in User, required.
	type	Type of notification (e.g., 'warning', 'info', 'reminder'), defaults to 'info'.
	message	Content of the notification message, required.
	status	Read/unread status of the notification, defaults to 'unread'.
	date	Date and time of the notification, defaults to the current timestamp.

Table 48 Data Dictionary Table.

## ❖ Functions:

• **Trigger after\_expense\_insert**

- Function: Automatically updates the `current_balance` in the `Budget` table after an expense is added, and if the balance falls below zero, adds a warning notification.
- Parameters: Operates on each row of `Expense` after insertion.
- Process:
  - Deducts `amount * exchange_rate` from the `current_balance` in `Budget`.
  - Checks the updated balance; if negative, inserts a warning in `Notifications`.

• **Trigger after\_income\_insert**

- Function: Automatically updates the `current_balance` in the `Budget` table after an income entry is added.
- Parameters: Operates on each row of `Income` after insertion.
- Process:

- o Adds amount \* exchange\_rate to the current\_balance in Budget.

4.6.3 Database Description

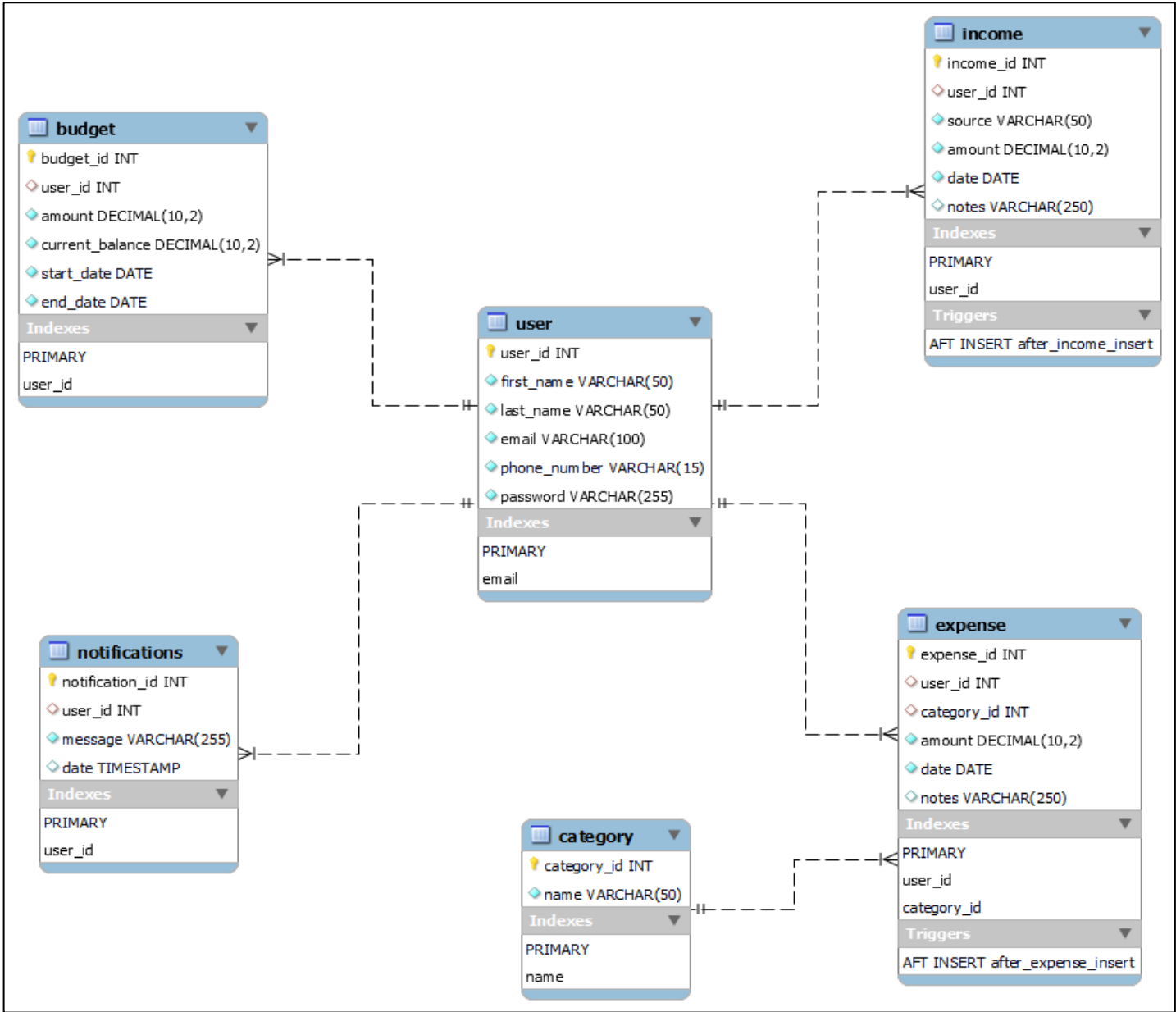
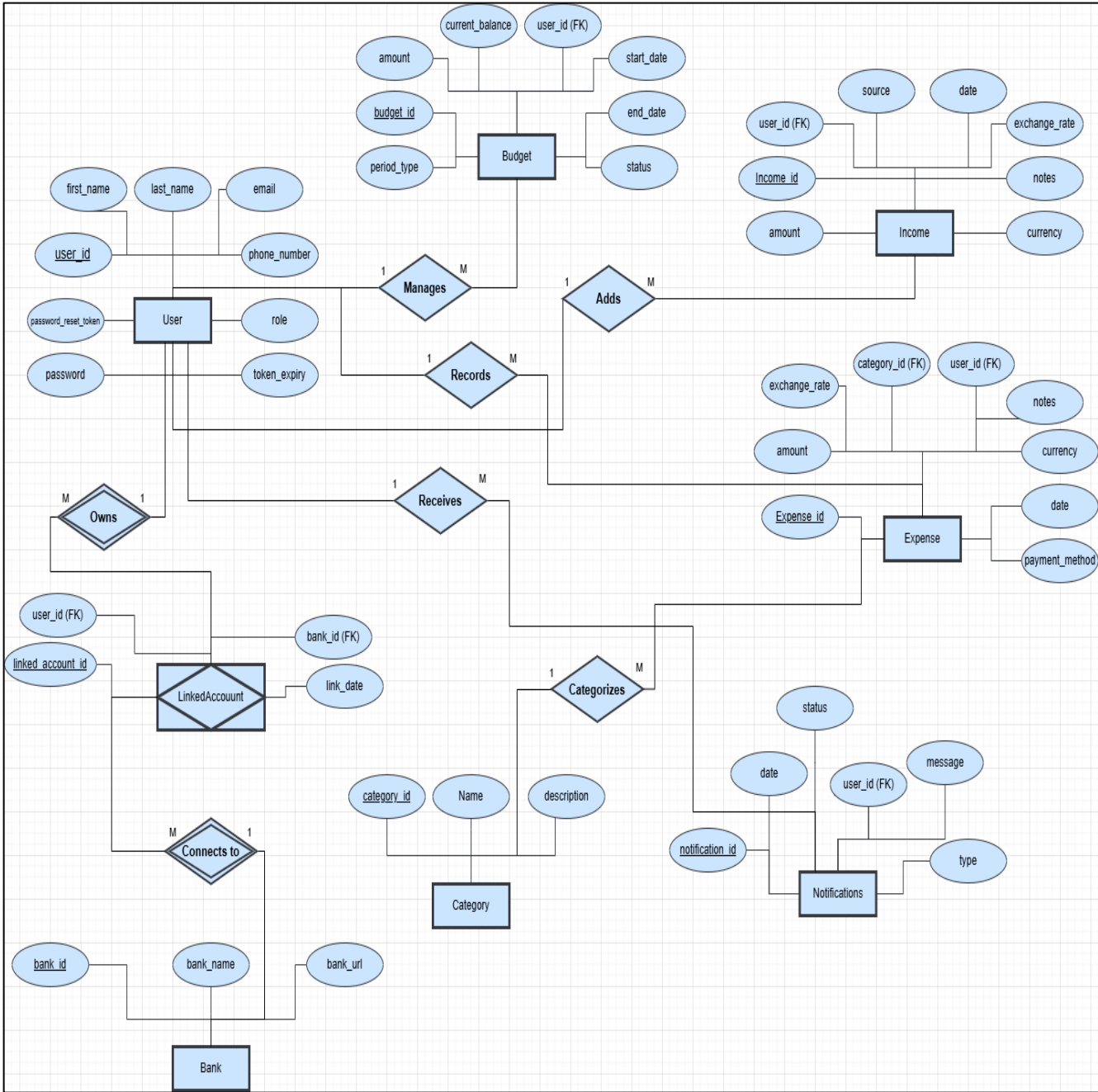


Figure 29: Class Diagram

Figure 30: Entity Relationship Diagram.



\* For better quality of the ERD go here: [ERD.pdf](#)

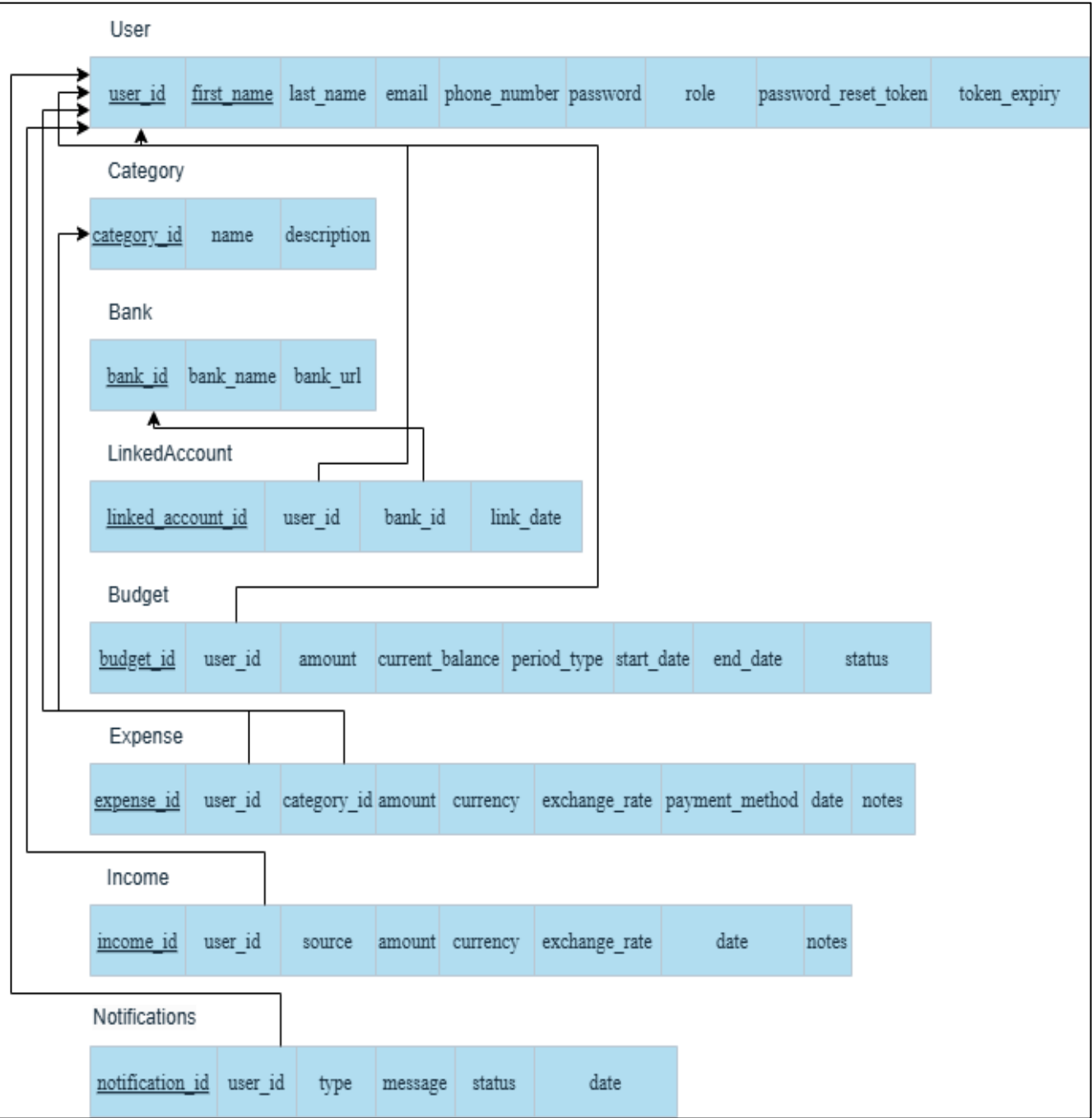


Figure 31: Relational Schema diagram.

\* For better quality of the Schema go here: [RelationalSchema.pdf](#)

## 4.7 Component Design

This section includes the pseudocode of Expense Tracker. It provides each component in the system by algorithm.

### 4.7.1 Common Functions

These functions are accessible by both User and Admin roles.

#### 4.7.1.1 Login

```
Function login() {  
    Enter email and password  
    If email and password match records in database  
        Then  
            If user is admin  
                Redirect to Admin Mode  
            Else  
                Redirect to User Mode  
        Else  
            Show error message "Invalid email or password, please try again."  
    }  
}
```

#### 4.7.1.2 Forgot Password

```
Function ForgotPassword() {  
    Enter email  
  
    If email exists in database  
        Then send password reset link to email  
    Else  
        Show error message "Invalid email address, please try again."  
    }  
}
```

#### 4.7.1.3 Change Password

```
Function ChangePassword() {  
    Enter old password  
  
    If old password is valid  
        Then enter new password and confirm password  
  
        If new password == confirm password  
            If new password length >= 8
```

Update password in database (hash new password)

Show confirmation message "Password changed successfully."

Else

Show error message "Password must be at least 8 characters long."

Else

Show error message "New password doesn't match the confirmed password."

Else

Show error message "Invalid old password."

}

#### **4.7.2 Admin Functions**

These Functions only accessible by Admins.

##### **4.7.2.1 Delete Transaction**

Function DeleteTransaction() {

If admin clicks on "Delete" button

Then display confirmation message "Are you sure you want to delete this transaction?"

If admin clicks on "Yes"

Then delete transaction from database

Show confirmation message "Transaction deleted successfully."

Else

Close confirmation window

}

##### **4.7.2.2 View User Transactions**

Function ViewUserTransactions() {

Set transaction\_count = 0

If admin clicks on "View Transactions" button

If transaction\_count > 0

Then display all user transactions

Else

Display message "No transactions available."

}

#### 4.7.2.3 View Reports

Function ViewReports() {

If admin clicks on "View Reports" button

Then display detailed financial reports (income, expenses, etc.)

Allow filtering by date range

}

#### 4.7.2.4 Manage User Accounts

Function ManageUserAccounts() {

If admin clicks on "Manage Users"

Then display list of all users with options to edit or delete

If admin clicks on a user

Then display user details and allow admin to edit or deactivate account

If admin clicks on "Delete User"

Then display confirmation message "Are you sure you want to delete this user?"

If admin clicks "Yes"

Then delete user account

Show confirmation message "User account deleted successfully."

}

### 4.7.3 User Functions

These Functions only accessible by Users.

#### 4.7.3.1 Register

Function Register() {

Enter first name, last name, email, phone number, password, confirm password

If any fields are missing



Show red asterisk next to missing fields

Else if email format is incorrect

Show error message "Invalid email format!"

Else if phone number format is incorrect

Show error message "Invalid phone number format!"

Else if password != confirm password

Show error message "Passwords do not match!"

Else if password length < 8

Show error message "Password must be at least 8 characters long."

Else

Check if email already exists in the User table

If email exists

Show error message "Email already in use."

Else

Create account and redirect to login page

}

#### 4.7.3.2 Add Expense

Function AddExpense() {

Select expense category (e.g., Food, Entertainment)

Enter amount

Select date

Enter notes (optional)

If amount <= 0

Show error message "Amount must be a positive value."

Else if amount exceeds budget

Show warning message "Warning: Expense exceeds budget."

Else

Save expense entry to database

Show confirmation message "Expense added successfully."

}

#### 4.7.3.3 Add Funds

Function AddFunds() {

Select source of income (e.g., Salary, Gift)

Enter amount to add

Select date of income

Enter notes (optional)

If amount  $\leq 0$

Show error message "Amount must be a positive value."

Else

Add funds to user account

Show confirmation message "Funds added successfully."

}

#### 4.7.3.4 View Expense Analysis

Function ViewExpenseAnalysis() {

Select date range for analysis

Select category filter (optional)

If no transactions match filters

Show message "No transactions found."

Else

Display total expenses, category-wise breakdown, and trends

Display graphical reports (e.g., pie chart, bar chart)

Allow export to CSV, Excel, or PDF

Show confirmation message "Analysis exported successfully."

}

#### 4.7.3.5 Link Bank Account

Function LinkBankAccount() {

    Select bank from dropdown list of supported banks

    If a bank is selected

        Then redirect to bank's official website to complete the linking process

        Show confirmation message "Redirecting to the bank's official page."

    Else

        Show error message "Please select a valid bank."

}

## 4.8 Detailed System Design

In this section we will state the classifications, definitions, responsibilities, constraints and composition of every system component. We will also discuss the uses/interactions, resources, interface/exports and detailed subsystem design.

### 4.8.1 Classification

This section includes the classification of all system components

Component	Classification
Common Components	
Login	Function
Forgot Password	Function
Change Password	Function
Admin Components	
Delete Transaction	Function
View User Transactions	Function
View Reports	Function
Manage User Accounts	Function
User Components	
Register	Function
Add Expense	Function
Add Funds	Function
View Expense Analysis	Function
Link Bank Account	Function

*Table 49: Components Classification*

### 4.8.2 Definition

This section includes the definition of all system components

Component	Definition
Common Components	
Login	Verifies user email and password, redirecting them to appropriate user or admin pages.
Forgot Password	Sends a reset link to the user's email for password recovery.
Change Password	Allows users to update their password after verification of the old password.
Admin Components	
Delete Transaction	Allows admins to delete a user transaction with confirmation.
View User Transactions	Displays all transactions associated with a user for administrative purposes.
View Reports	Generates and displays financial reports filtered by date range.
Manage User Accounts	Enables management of user accounts, including editing or deactivation.
User Components	
Register	Facilitates new user registration by collecting required details and validating inputs.
Add Expense	Records a new expense with details such as category, amount, date, and notes.
Add Funds	Adds a new income entry with details like source, amount, and date.
View Expense Analysis	Provides expense summaries, trends, and graphical visualizations for a selected date range.
Link Bank Account	Guides users to link their accounts with a supported bank via a secure process.

Table 50: Components Definitions.

4.8.3 Responsibilities

This section includes the responsibilities of all system components

Component	Responsibility
Common Components	
Login	Authenticates user credentials and determines the appropriate system role.
Forgot Password	Validates email and generates a reset link for recovering account access.
Change Password	Verifies old password and updates it with a securely hashed new password.
Admin Components	
Delete Transaction	Ensures accurate deletion of user transactions with user confirmation.
View User Transactions	Retrieves and displays user-specific transactions for review or audit.
View Reports	Summarizes financial data and trends into reports for analysis.
Manage User Accounts	Oversees user account actions, including updates, deletions, and role management.
User Components	
Register	Ensures successful creation of a user account while validating input fields and handling errors.
Add Expense	Logs expenses into the database while ensuring budget compliance and triggering warnings if needed.
Add Funds	Updates the user's budget balance and records income sources.
View Expense Analysis	Provides insights into user spending patterns, including total, category-wise breakdowns, and trends.
Link Bank Account	Redirects users to the bank's official website and stores linked account details securely.

Table 51: Components Responsibilities.

#### 4.8.4 Constraints

This section shows the constraints on components in the system

Component	Constraints
Common Components	
Login	Requires email and password to match database records; encrypted passwords stored securely must meet policy criteria.
Forgot Password	Email must exist in the system; token validity is time-limited, requiring action within the expiry window.
Change Password	Old password must match the stored hash; new passwords must be at least 8 characters and meet defined complexity.
Admin Components	
Delete Transaction	Admin access is mandatory; deleted data cannot violate the referential integrity of related records.
View User Transactions	Large datasets might delay responses; transaction records must be associated with a valid user.
View Reports	Date range filters must be valid; system must handle large datasets efficiently without performance degradation.
Manage User Accounts	Admin role is required; sensitive information like user passwords cannot be directly modified.
User Components	
Register	Unique email and valid inputs are mandatory; phone numbers and emails must meet format constraints.
Add Expense	Must select a valid category; amount must be positive; expenses exceeding budget trigger warnings.
Add Funds	Source must be predefined or manually input; amount must be positive; dates cannot be in the future.
View Expense Analysis	Date ranges must be valid; expense trends depend on transaction history availability.
Link Bank Account	Bank name must match a predefined list; secure API redirection is required for successful linking.

Table 52: Components Constraints.

#### 4.8.5 Composition

This section will include Pre-conditions and post-conditions of each component

Component	Pre-condition	Post-condition
Common Components		
Login	User enters valid email and password; database connection is active.	User is authenticated and redirected to the appropriate dashboard (user/admin).
Forgot Password	User submits a registered email; database verifies token generation.	Password reset link is sent to email; token is saved in the system with expiry metadata.
Change Password	User provides the correct old password; new password meets security criteria.	Password is updated in the database, and the user is notified of success.
Admin Components		
Delete Transaction	Admin selects a valid transaction and confirms deletion.	Transaction is deleted from the database; a confirmation message is displayed.
View User Transactions	Admin selects a user; user has associated transactions in the database.	Transactions are retrieved and displayed in a list or table.
View Reports	Valid date range and filters are provided.	Financial reports are generated, summarized, and optionally visualized graphically.
Manage User Accounts	Admin accesses the management interface and selects a user to edit.	User details are updated or deactivated with confirmation of the action.
User Components		
Register	User provides valid and unique information; all required fields are filled.	New user account is created, and the user is redirected to the login page.
Add Expense	User provides a positive amount and selects a valid category.	Expense is recorded in the database; budget balance is updated if applicable.



Add Funds	User provides valid details for source, amount, and date.	Fund entry is saved in the database, and the budget balance is updated accordingly.
View Expense Analysis	User applies filters like date range or category.	Expense trends and breakdowns are displayed graphically or as tabular data.
Link Bank Account	User selects a valid bank and initiates redirection.	Account is linked; confirmation is displayed upon success or error upon failure.

Table 53: Components Composition.

4.8.6 Uses/Interactions

In the following subsections, an illustration of every component’s interaction and collaboration with other components is shown through sequence diagrams.

4.8.6.1 Common Components

4.8.6.1.1 Login

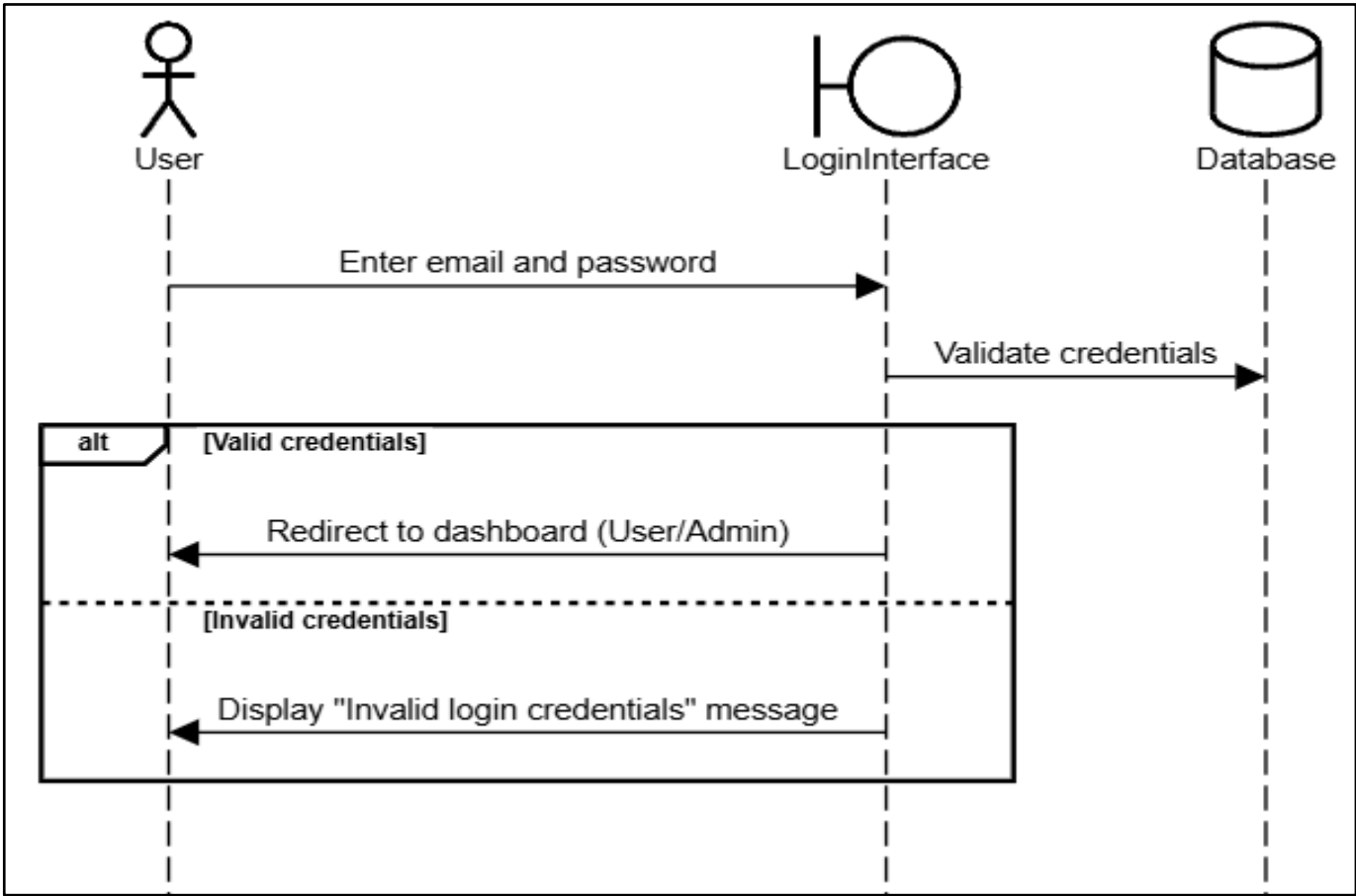


Figure 32: Login Sequence Diagram

4.8.6.1.2 Forgot Password

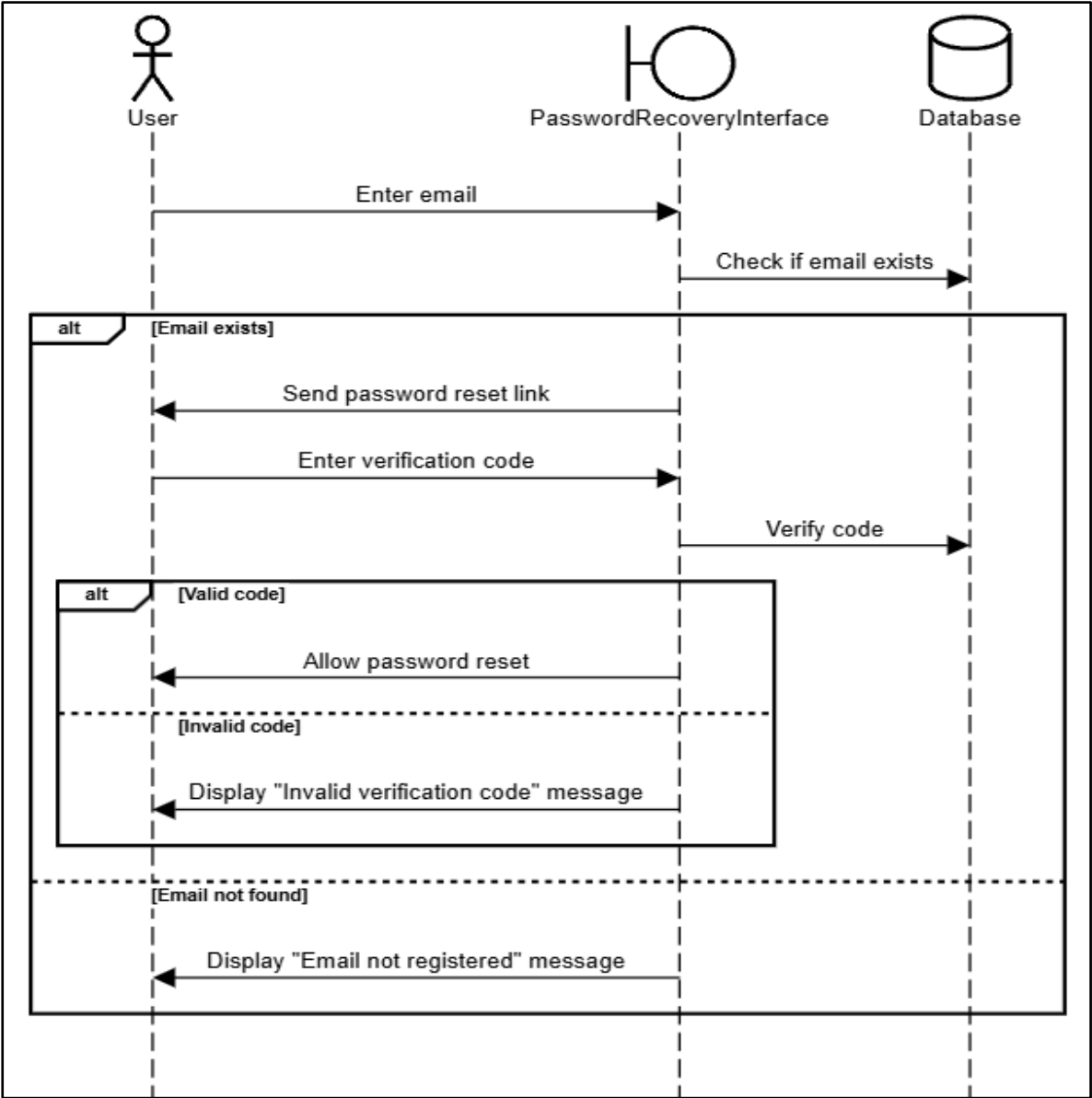


Figure 33: Forgot Password Sequence Diagram

4.8.6.1.3 Change Password

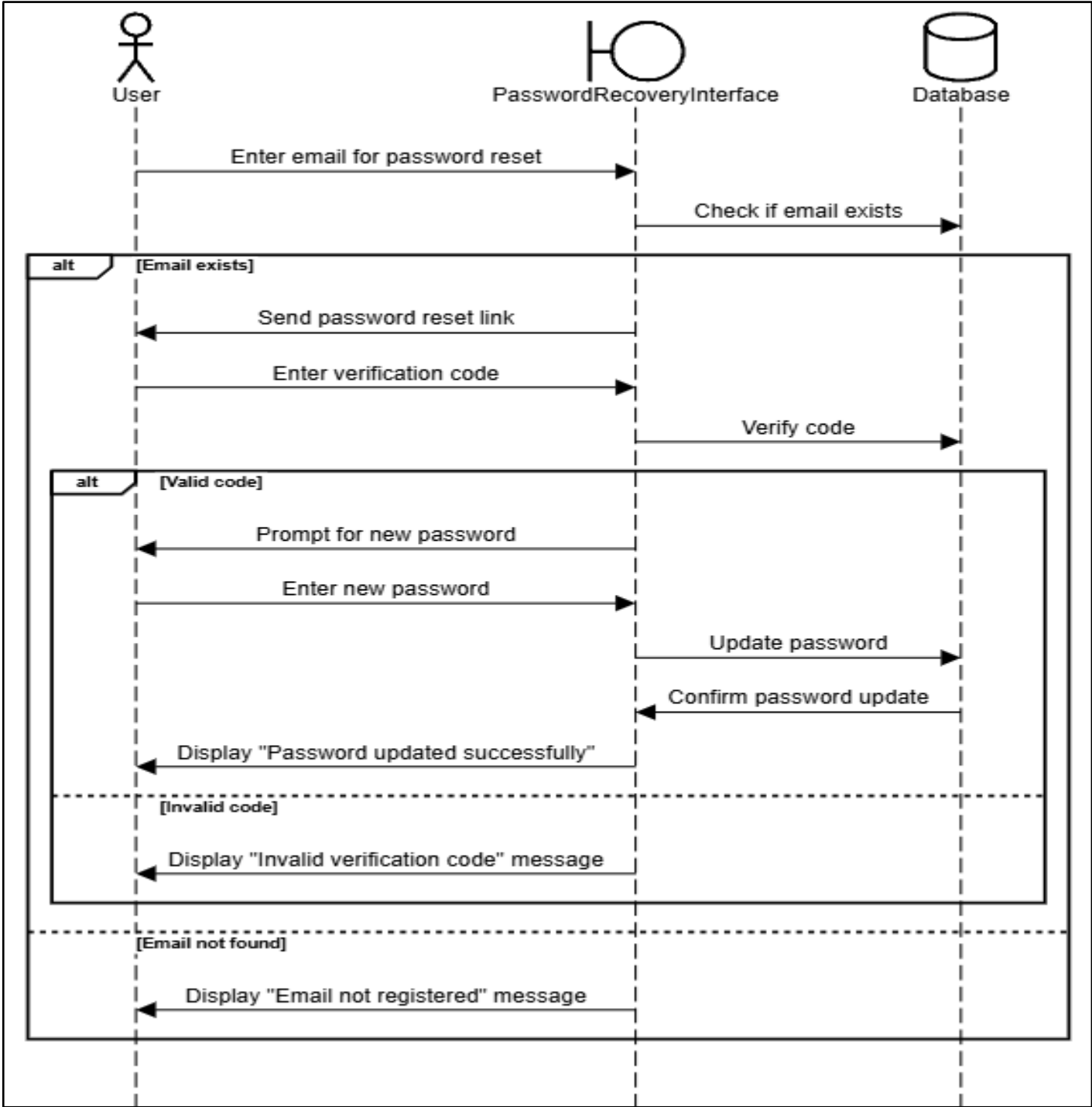


Figure 34: Change Password Sequence Diagram

4.8.6.2 Admin Components

4.8.6.2.1 Delete Transaction

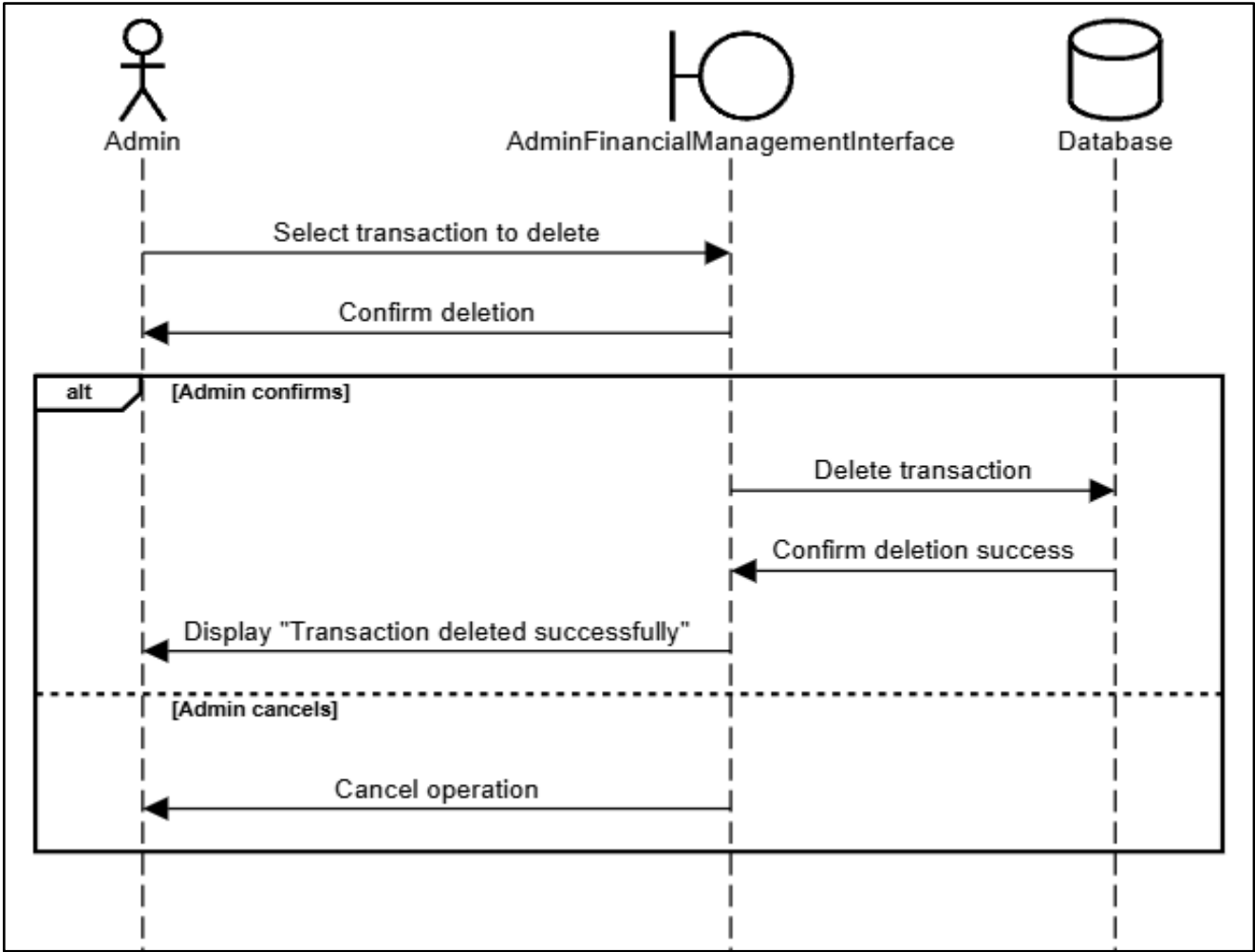


Figure 35: Delete Transaction Sequence Diagram

4.8.6.2.2 View User Transactions

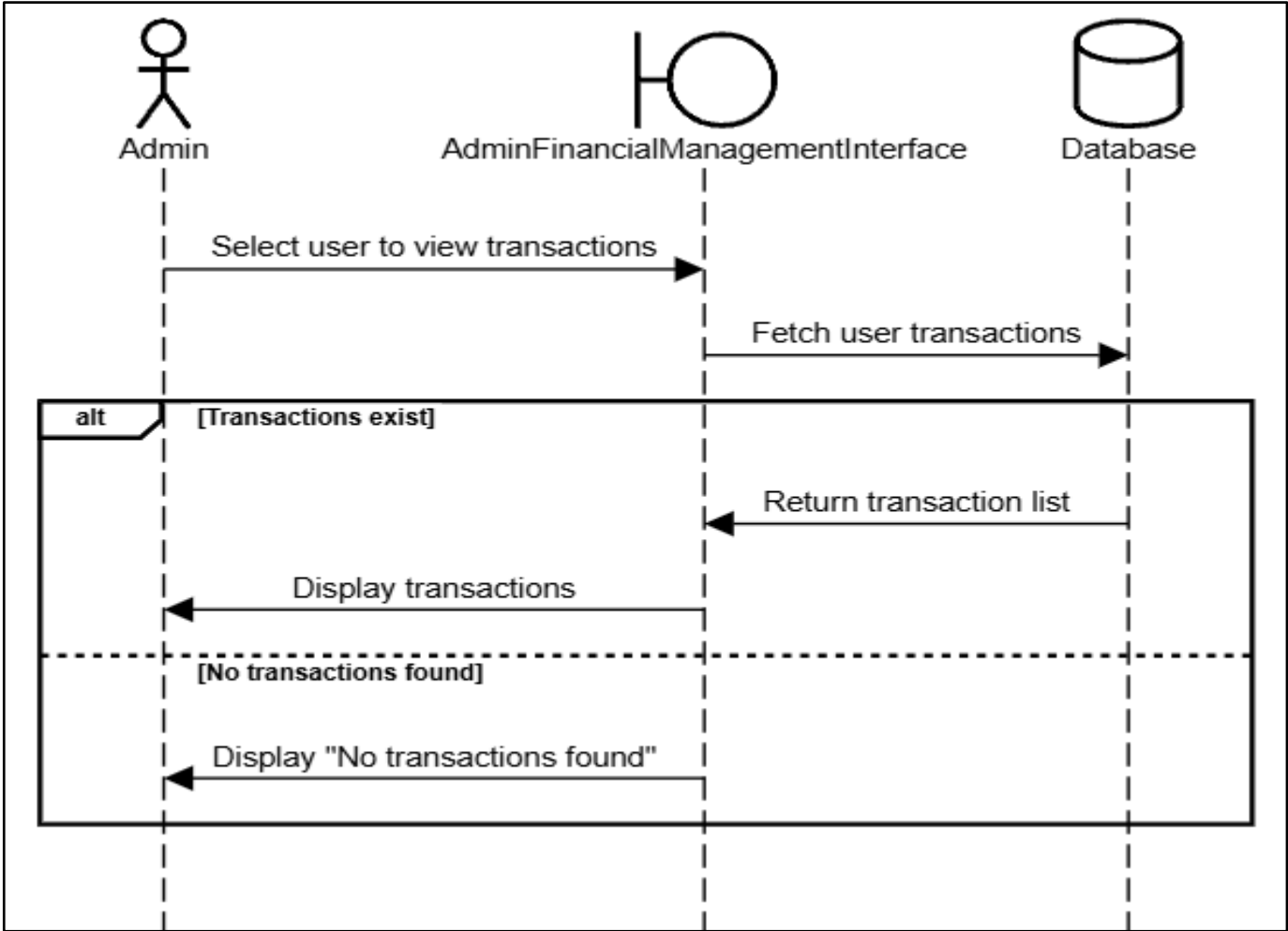


Figure 36: View User Transaction Sequence Diagram

4.8.6.2.3 View Reports

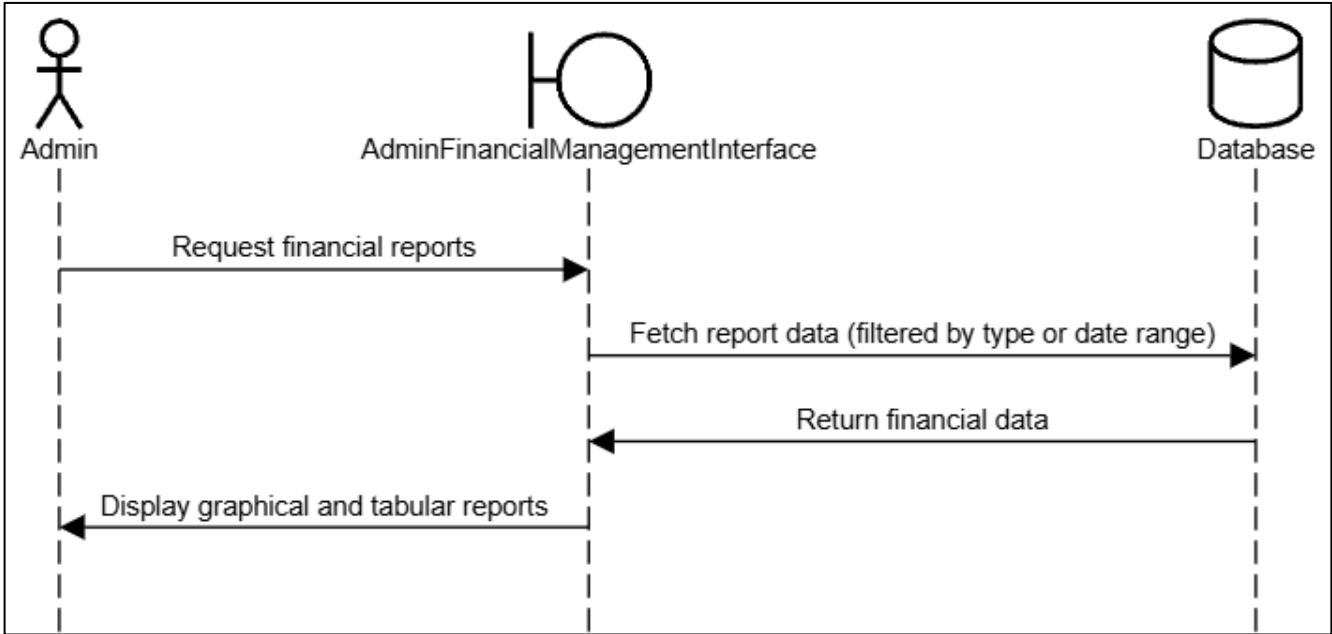


Figure 37: View Reports Sequence Diagram

4.8.6.2.4 Manage User Accounts

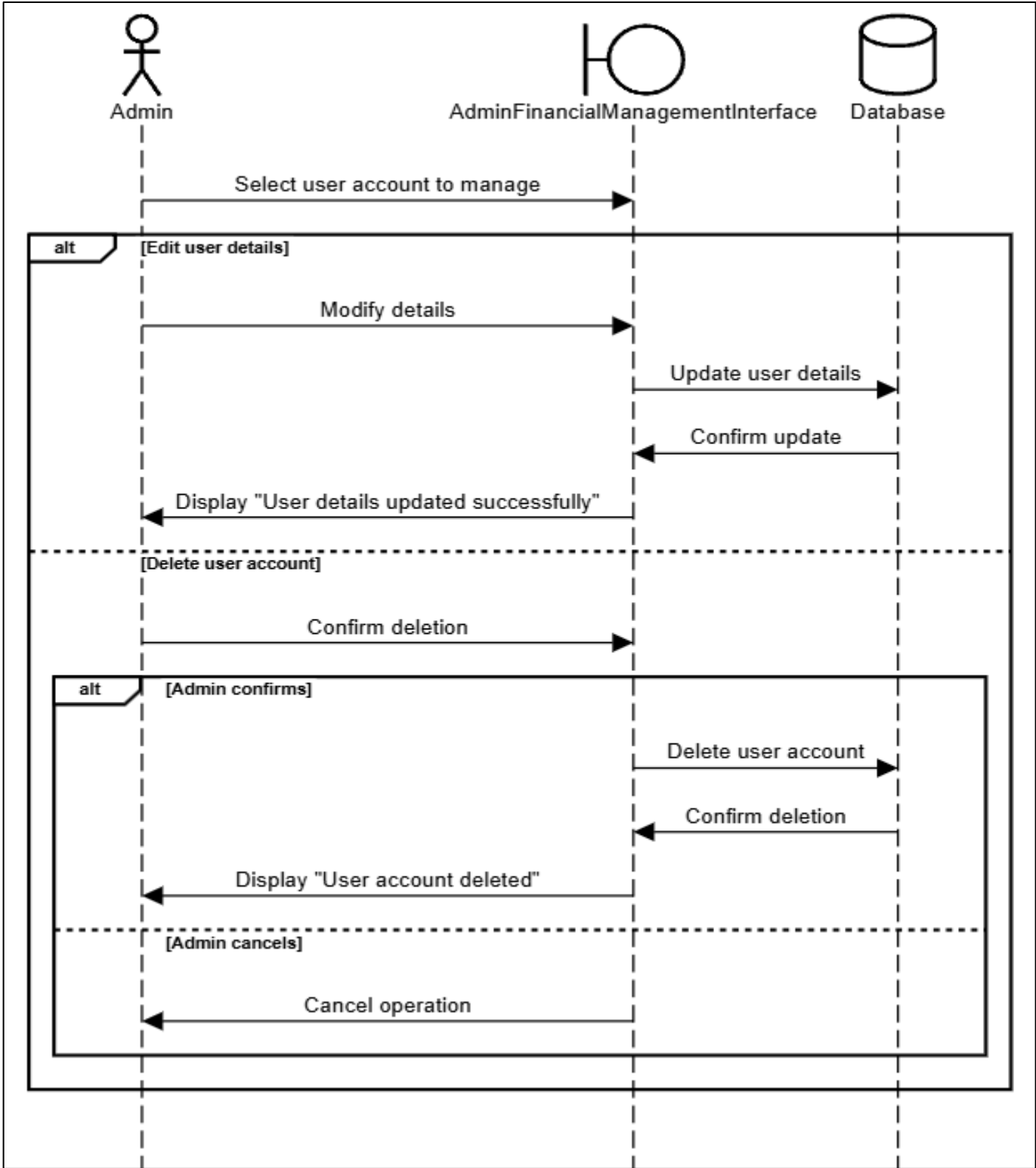


Figure 38: Manage User Accounts Sequence Diagram

4.8.6.3 User Components

4.8.6.3.1 Register

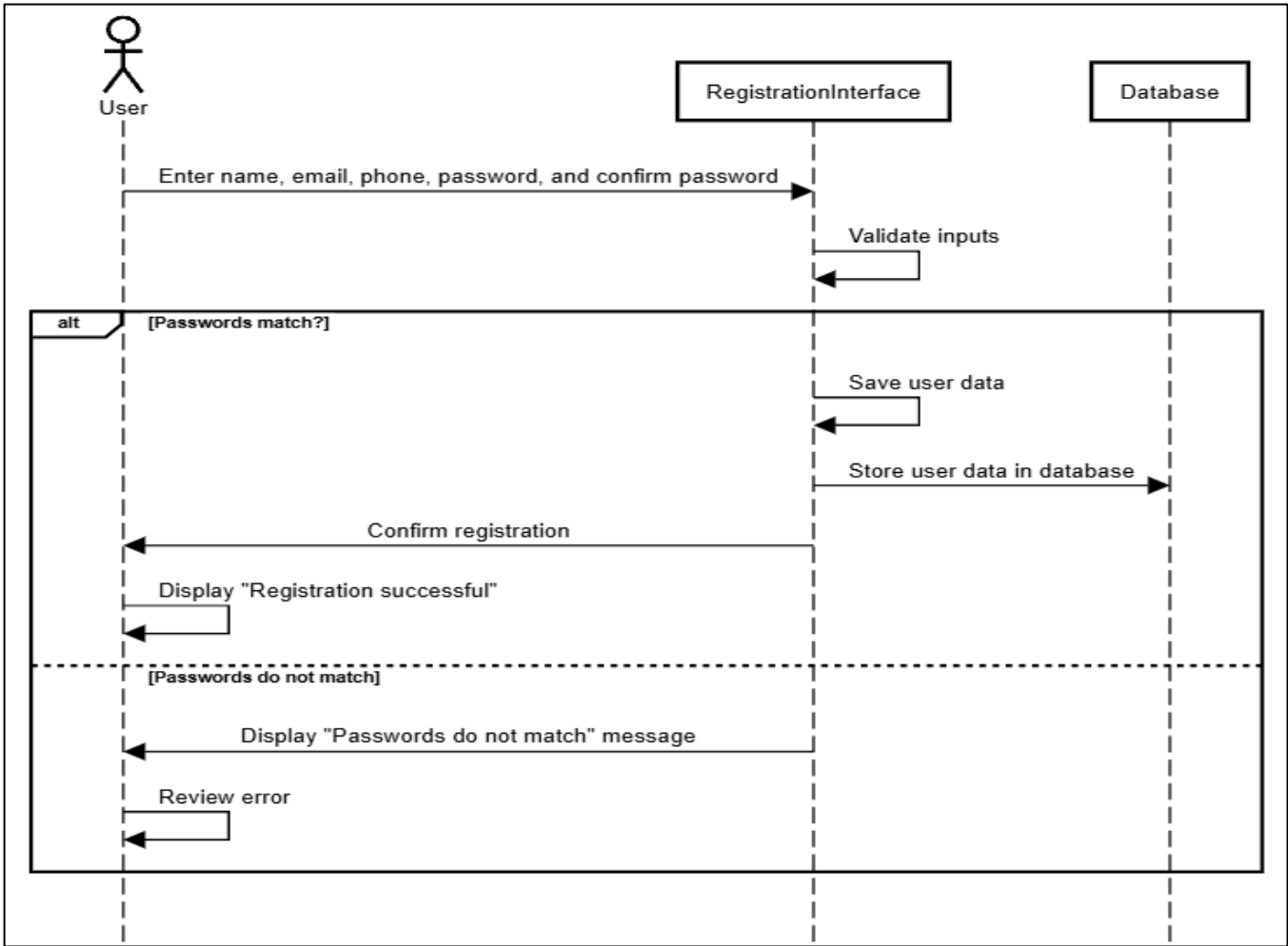


Figure 39: Register Sequence Diagram



4.8.6.3.2 Add Expense

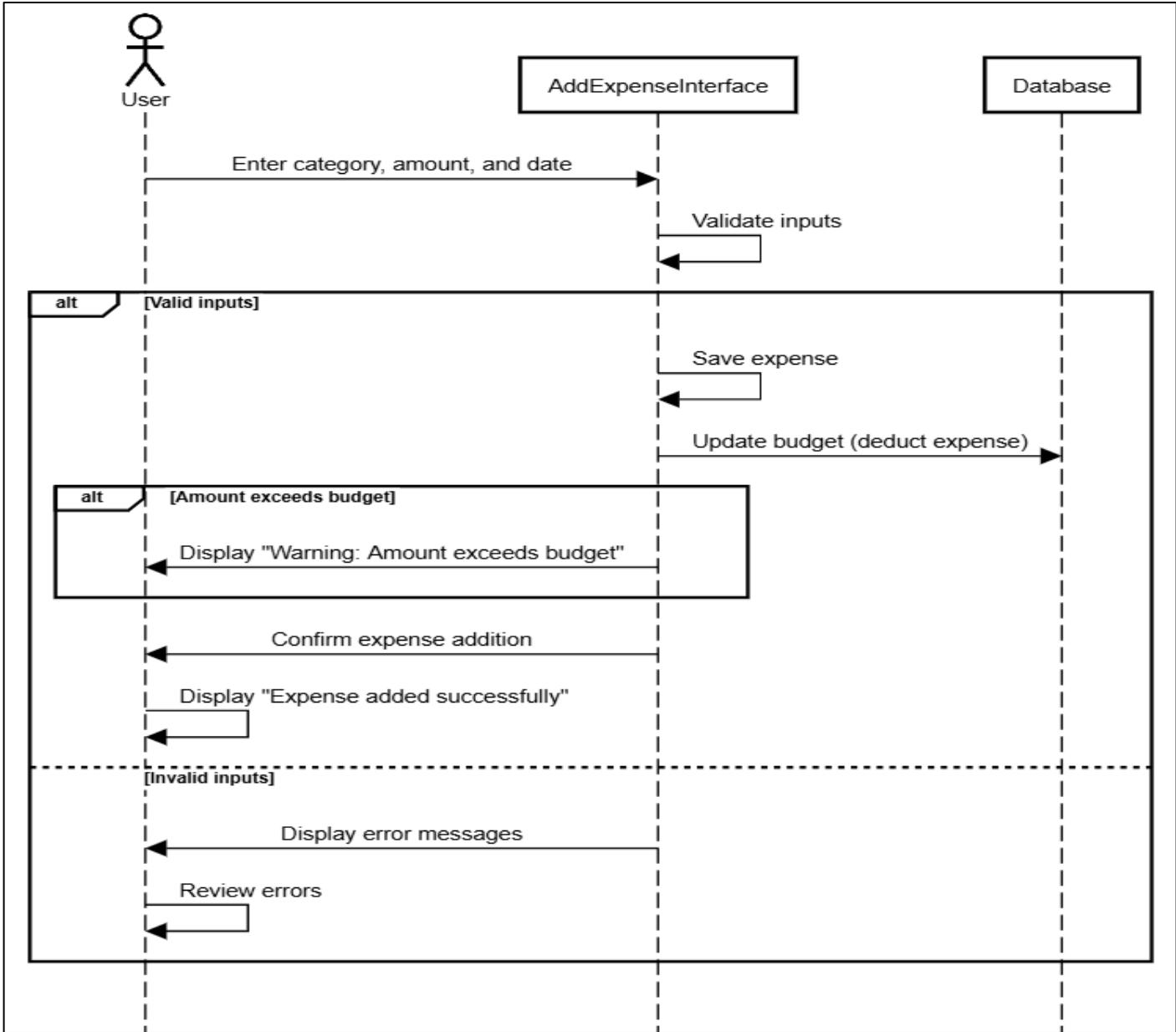


Figure 40: Add Expense Sequence Diagram

4.8.6.3.3 Add Funds

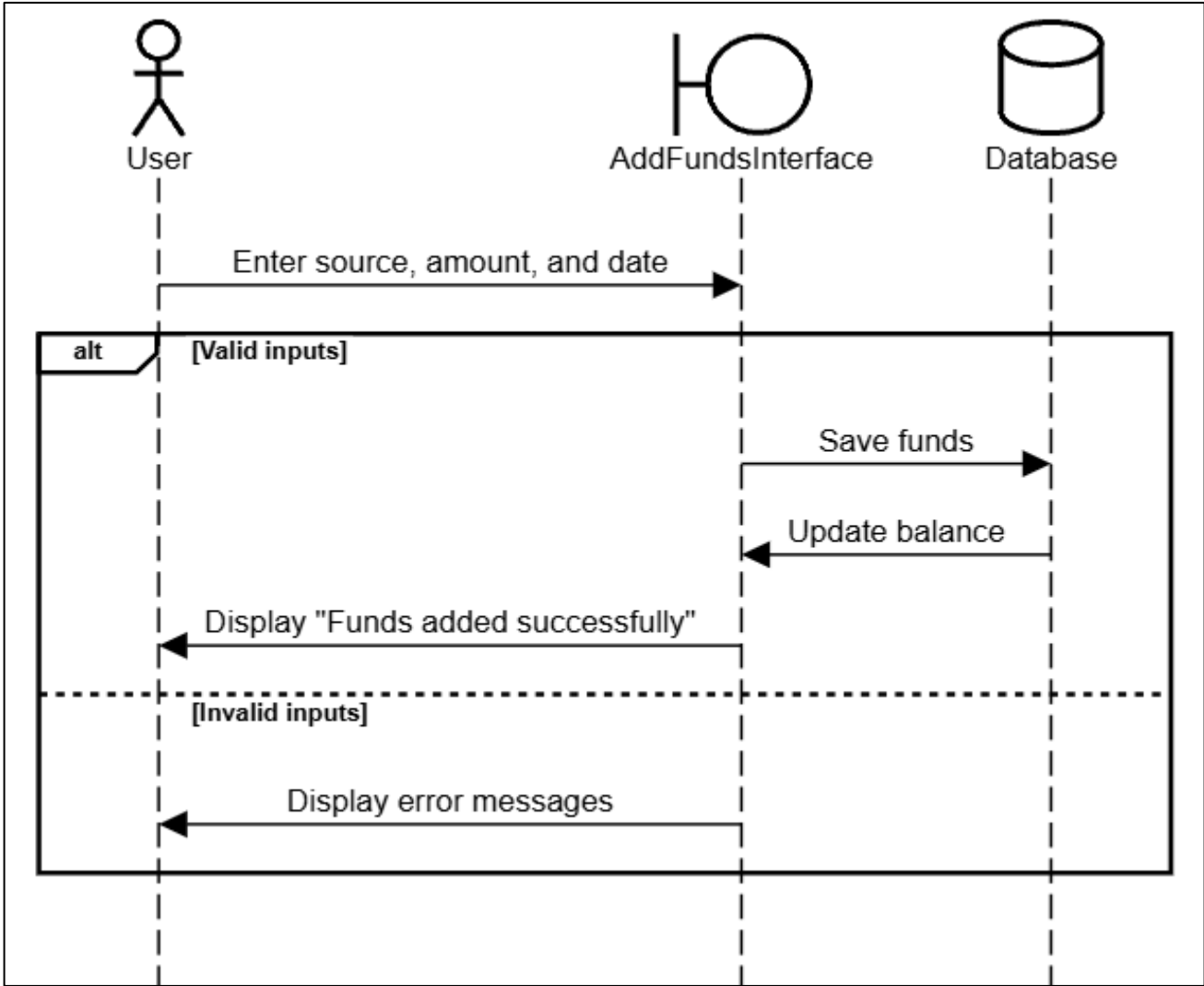


Figure 41: Add Funds Sequence Diagram

4.8.6.3.4 View Expense Analysis

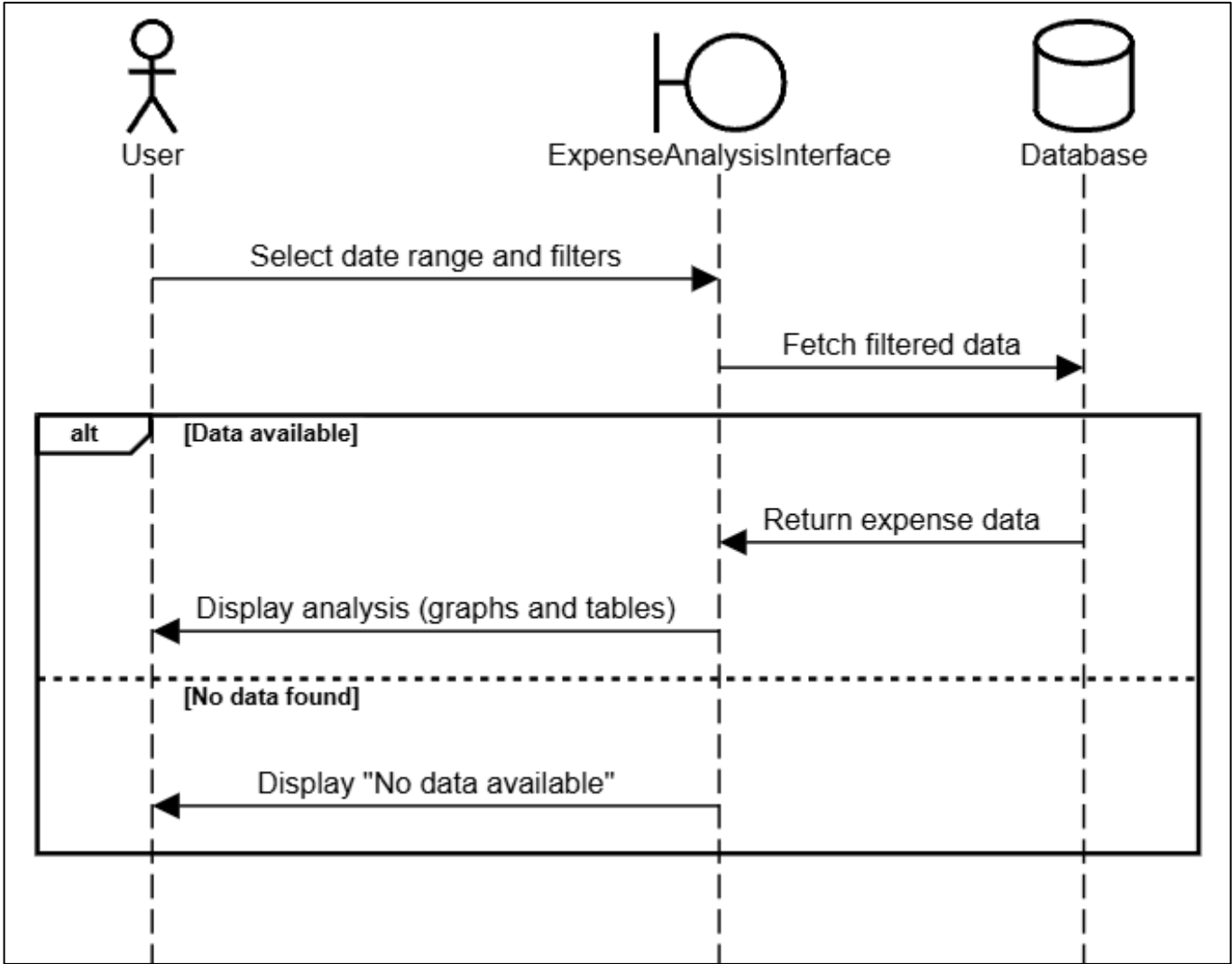


Figure 42: View Expense Analysis Sequence Diagram

4.8.6.3.5 Link Bank Account

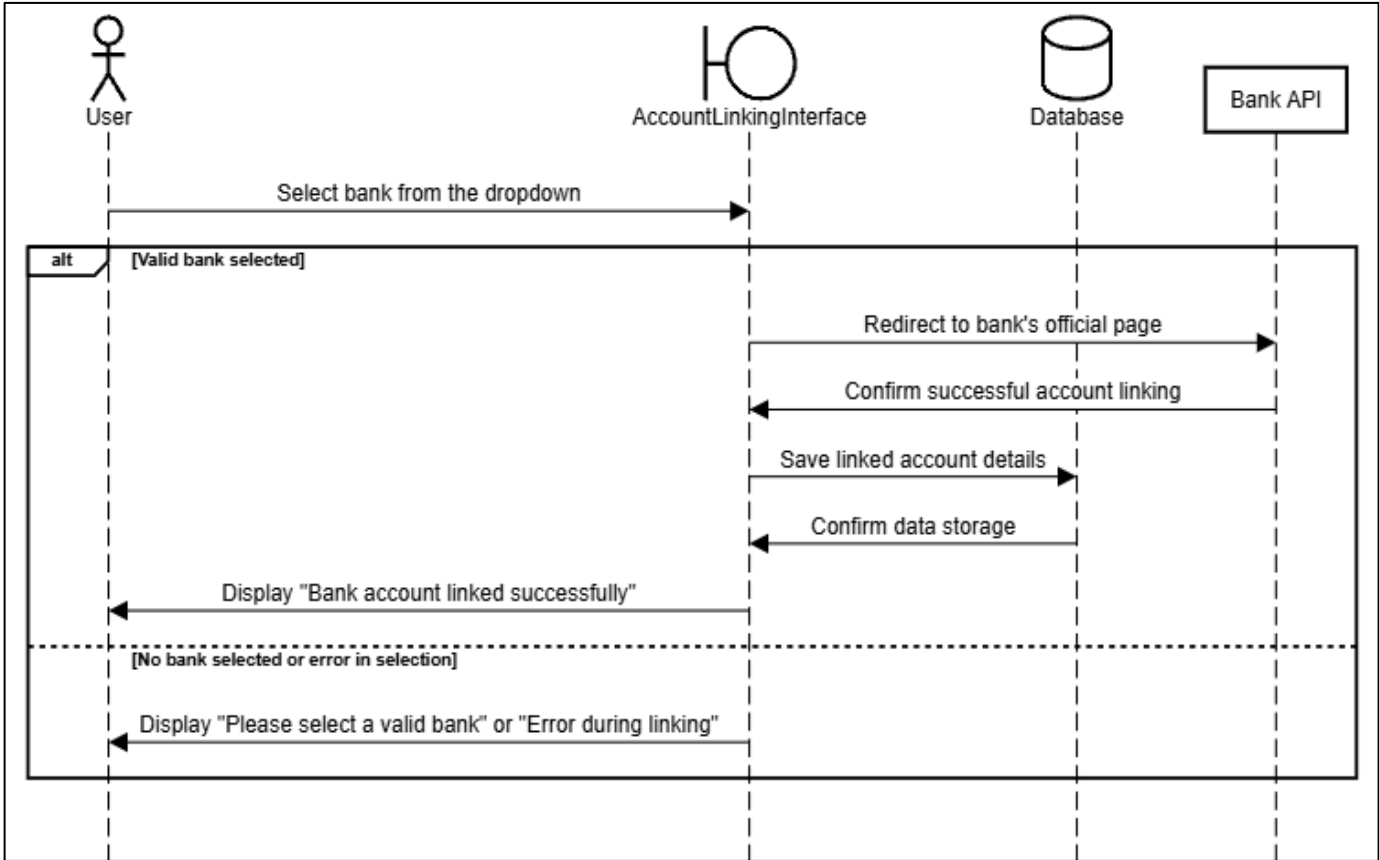


Figure 43: Link Bank Account Sequence Diagram

4.8.7 Resources

All external entity resources that are managed, affected, or required for the Expense Tracker are defined in this section.

Type	Resource
Database	mysql
Memory storage	1 GB
OS	Any OS

Table 54: External Resources

Table below shows deadlock situations that may occur in Expense Tracker:

Deadlock Situation	Description	Solution
Trigger-based update on the same table	Multiple triggers attempting to update the same table simultaneously, leading to a deadlock when both operations attempt to modify the same record concurrently.	Use row-level locking to lock the specific row before updating it, preventing simultaneous modifications to the same record. Ensure all operations occur within a single transaction to maintain consistency.
Locking order inconsistency in concurrent transactions	Two or more transactions acquire locks on different rows or tables in an inconsistent order, leading to a circular wait situation.	Ensure that all transactions acquire locks on rows and tables in a consistent order to prevent circular dependencies that can lead to deadlocks.
Simultaneous updates on related data	Simultaneous updates on related data (such as updates to balances and linked records) may cause conflicts if they happen at the same time, leading to a deadlock.	Use explicit transactions to ensure that related updates happen sequentially within a single transaction. Row-level locking can help avoid conflicts on shared resources.
Multiple dependent operations with conflicting locks	Multiple operations trying to update dependent data (e.g., updating budget and inserting notifications) may deadlock if locks are not managed carefully.	Ensure that dependent operations are executed within the same transaction, and use consistent lock ordering to avoid conflicting locks on the same resources.

Table 55: Deadlock Situations

4.8.8 Processing

This subsection includes a description of precisely how each component performs the duties necessary to fulfill its responsibilities. The following subsections include each component description, input, output and constraints for each interface.

4.8.8.1 Common Functions

4.8.8.1.1 Login

Login	
Description	Verifies user credentials and redirects to the appropriate dashboard based on role (admin or user).
Inputs	- Email - Password
Outputs	- Successful login redirection - Error message ("Invalid credentials")
Constraints	- Email must be registered and follow a valid format - Password must match the stored encrypted password

Table 56: Login Processing Table.

4.8.8.1.2 Forgot Password

Forgot Password	
Description	Assists users in recovering their account by verifying the reset token and allowing a password update.
Inputs	- Email - Verification token - New password
Outputs	- Reset link sent - Confirmation message - Error message ("Invalid or expired token")
Constraints	- Email must exist in the database - Token must be valid and unexpired - Password must meet complexity requirements

Table 57: Forgot Password Processing Table.

#### 4.8.8.1.3 Change Password

Change Password	
Description	Allows users to update their password after verifying the old password.
Inputs	<ul style="list-style-type: none"><li>- Old password</li><li>- New password</li><li>- Confirm new password</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Success message ("Password updated")</li><li>- Error messages ("Invalid old password", "Passwords do not match")</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Old password must match the stored hash</li><li>- New password must meet complexity requirements (e.g., min 8 characters)</li></ul>

Table 58: Change Password Processing Table.

#### 4.8.8.2 Admin Functions

##### 4.8.8.2.1 Delete Transaction

Delete Transaction	
Description	Allows an admin to delete a user's transaction from the database.
Inputs	<ul style="list-style-type: none"><li>- Transaction ID</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Success message ("Transaction deleted")</li><li>- Error message ("Deletion failed")</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Only admins can delete transactions</li><li>- Referential integrity must be maintained for linked records</li></ul>

Table 59: Delete Transaction Processing Table.

##### 4.8.8.2.2 View User Transactions

View User Transactions	
Description	Displays all transactions associated with a selected user for administrative review or audit purposes.
Inputs	<ul style="list-style-type: none"><li>- User ID</li><li>- Filter criteria (optional, e.g., date range, transaction type)</li></ul>

Outputs	<ul style="list-style-type: none"><li>- List of transactions</li><li>- Message if no transactions are found ("No transactions available")</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Transactions must belong to a valid user</li><li>- System must handle large datasets efficiently</li></ul>

Table 60: View User Transactions Processing Table.

4.8.8.2.3 View Reports

View Reports	
Description	Generates detailed financial reports for income and expenses over a selected period.
Inputs	<ul style="list-style-type: none"><li>- Date range</li><li>- Filters (optional, e.g., category, transaction type)</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Graphs (e.g., pie charts, bar graphs)</li><li>- Tabular reports</li><li>- Export options (CSV, PDF)</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Date range must be valid</li><li>- System must handle large datasets without performance issues</li></ul>

Table 61: View Reports Processing Table.

4.8.8.2.4 Manage User Accounts

Manage User Accounts	
Description	Allows admins to manage user accounts, including editing details, deactivating accounts, or deleting users.
Inputs	<ul style="list-style-type: none"><li>- User selection</li><li>- Edit criteria (e.g., name, email, role, status)</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Success message ("User updated")</li><li>- Error message ("Update failed")</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Only admins can access this function</li><li>- Sensitive data (e.g., passwords) cannot be directly edited</li></ul>

Table 62: Manage User Accounts Processing Table.



4.8.8.3 User Functions

4.8.8.3.1 Register

Register	
Description	Allows new users to create an account by providing the required information.
Inputs	<ul style="list-style-type: none"><li>- First name</li><li>- Last name</li><li>- Email</li><li>- Phone number</li><li>- Password</li><li>- Confirm password</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Success message ("Account created")</li><li>- Error message ("Invalid email format")</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Email must be unique and valid</li><li>- Password must meet complexity requirements (min 8 characters)</li></ul>

Table 63: Register Processing Table.

4.8.8.3.2 Add Expense

Add Expense	
Description	Allows users to log their expenses, including category, amount, and additional notes.
Inputs	<ul style="list-style-type: none"><li>- Expense category</li><li>- Amount</li><li>- Date</li><li>- Notes (optional)</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Success message ("Expense added")</li><li>- Warning message ("Expense exceeds budget")</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Amount must be positive</li><li>- Notes must not exceed 250 characters</li></ul>

Table 64: Add Expense Processing Table.

4.8.8.3.3 Add Funds

Add Funds	
Description	Allows users to add funds to their account, specifying the source, amount, and additional details.
Inputs	<ul style="list-style-type: none"><li>- Source</li><li>- Amount</li><li>- Date</li><li>- Notes (optional)</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Success message ("Funds added")</li><li>- Error message ("Invalid amount")</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Amount must be positive</li><li>- Notes must not exceed 250 characters</li></ul>

Table 65: Add Funds Processing Table.

4.8.8.3.4 View Expense Analysis

View Expense Analysis	
Description	Provides visual and tabular insights into spending patterns, trends, and summaries.
Inputs	<ul style="list-style-type: none"><li>- Date range</li><li>- Category filter (optional)</li></ul>
Outputs	<ul style="list-style-type: none"><li>- Graphical reports (e.g., bar charts, pie charts)</li><li>- Expense trends</li><li>- Export options (CSV, PDF)</li></ul>
Constraints	<ul style="list-style-type: none"><li>- Date range must be valid</li><li>- Analysis requires transaction history</li></ul>

Table 66: View Expense Analysis Processing Table.

4.8.8.3.5 Link Bank Account

Link Bank Account	
Description	Allows users to link their bank accounts securely for automatic transaction imports.
Inputs	- Bank selection
Outputs	- Success message ("Bank account linked") - Error message ("Linking failed")
Constraints	- Bank must be from a predefined list - Secure API redirection is required

Table 67: Link Bank Account Processing Table.

4.8.9 Interface/Exports

System developers must understand each subsystem’s functionalities, including inputs, outputs, preconditions, postconditions, constraints, and processing details, to ensure the system functions properly. This includes validating user credentials, processing data like expense amounts, and adhering to constraints such as role-based access. Sections 8.1 to 8.8 provide detailed component descriptions, responsibilities, and constraints, while Section 8.6 illustrates interactions between components with sequence diagrams. Section 8.10 offers a detailed description of each software component, including complex diagrams showing component structure, behavior, and information flow, ensuring a comprehensive understanding of system integration. Together, these define the necessary interfaces and exports for seamless system implementation.

4.8.10 Detailed Subsystem Design

This subsection will demonstrates the details of each component’s structure or control flow. This is shown below by activity diagrams for each component, to document the internal structure of the subsystem.

4.8.10.1 Common Functions

4.8.10.1.1 Login

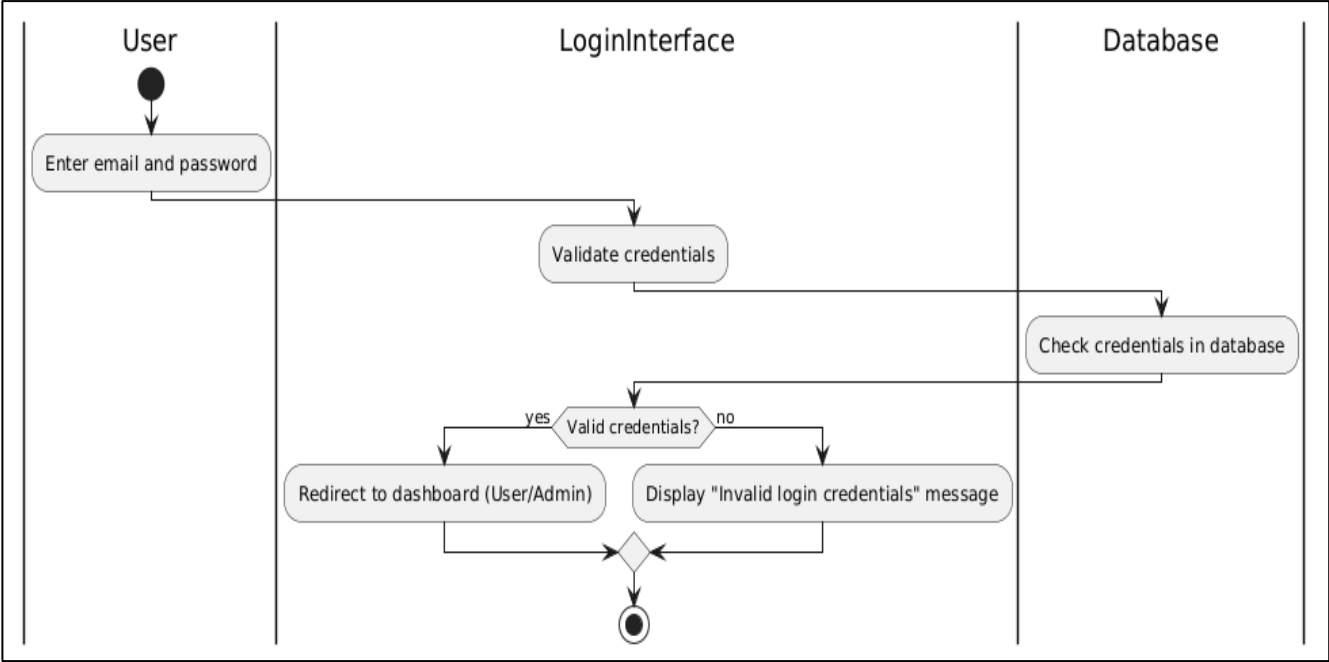


Figure 44: Login Activity Diagram.

4.8.10.1.2 Forgot Password

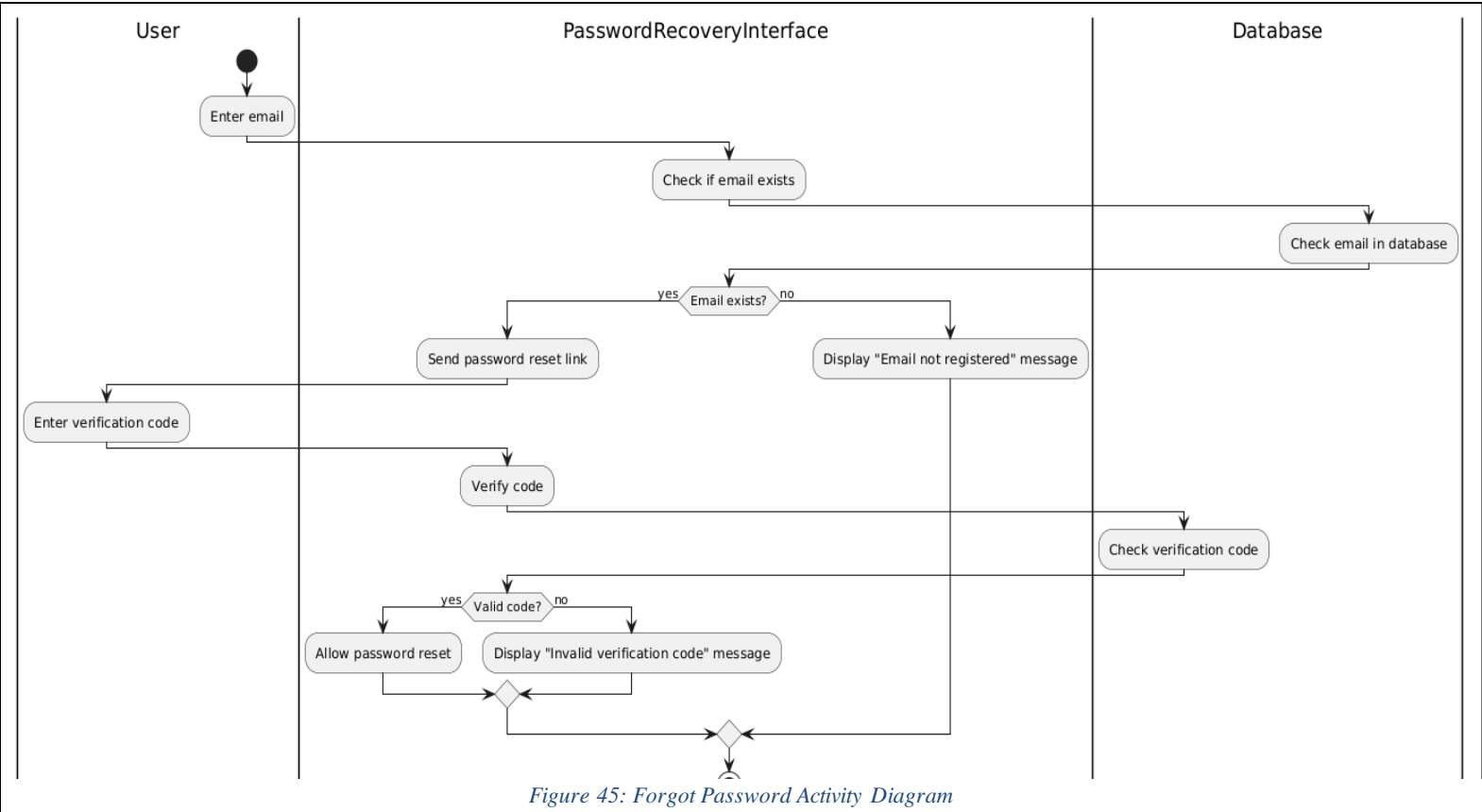


Figure 45: Forgot Password Activity Diagram

4.8.10.1.3 Change Password

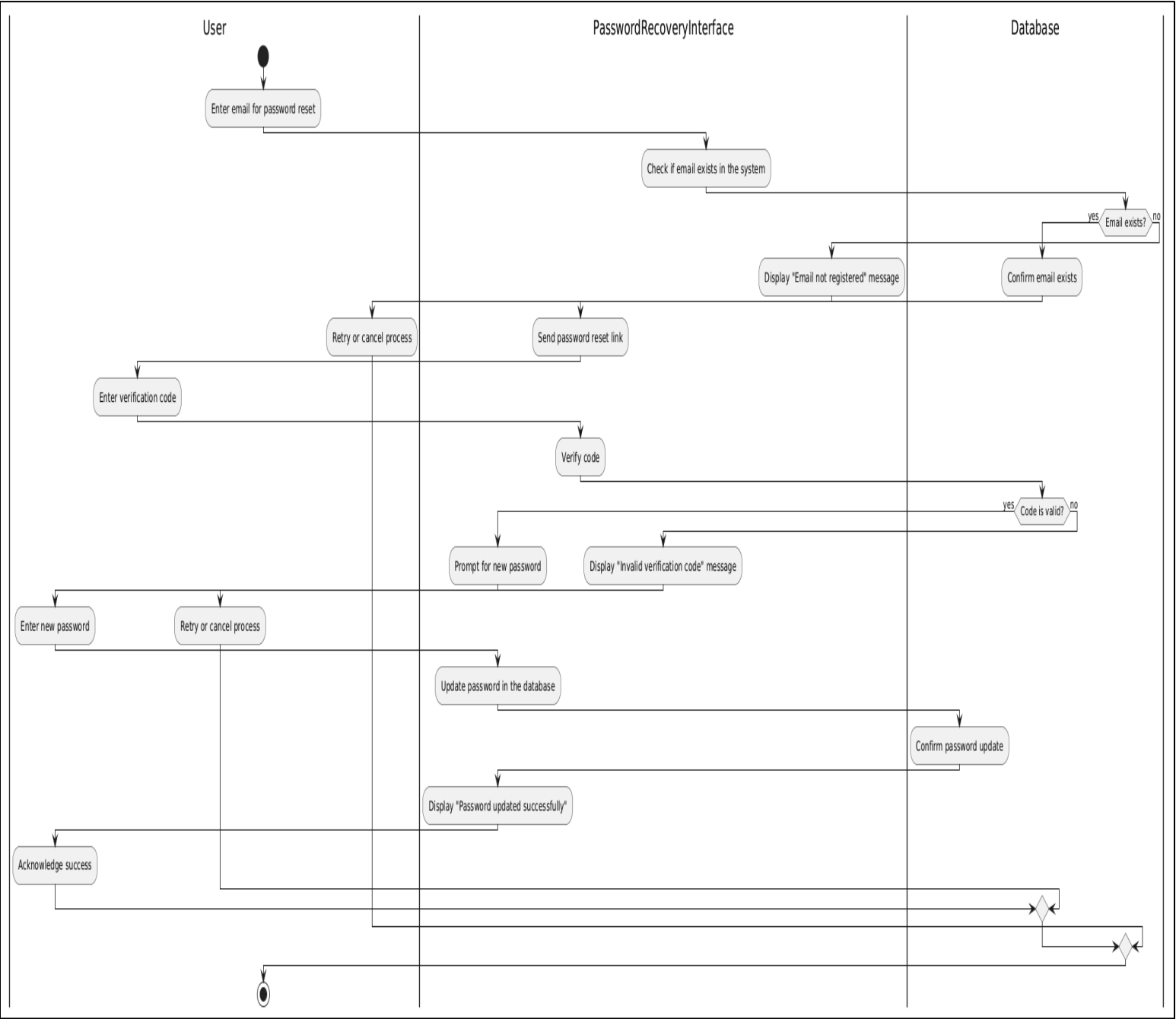


Figure 46: Change Password Activity Diagram

4.8.10.2 Admin Functions

4.8.10.2.1 Delete Transaction

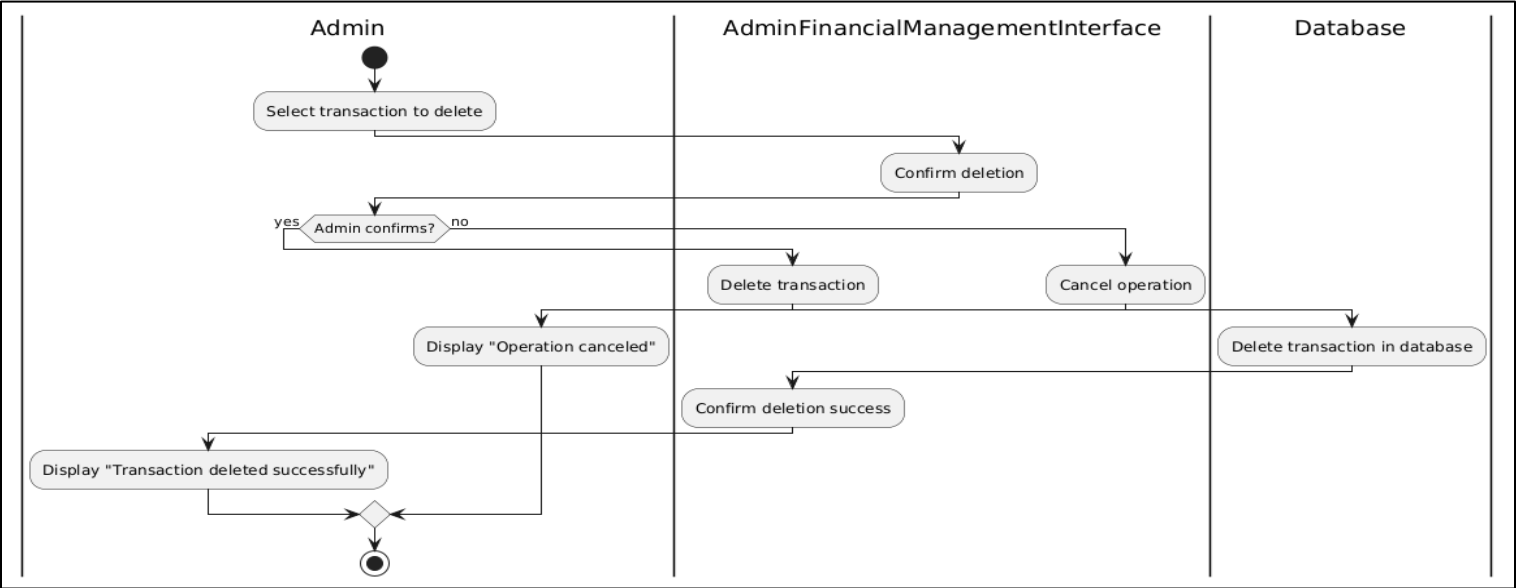


Figure 47: Delete transaction Activity Diagram.

4.8.10.2.2 View User Transactions

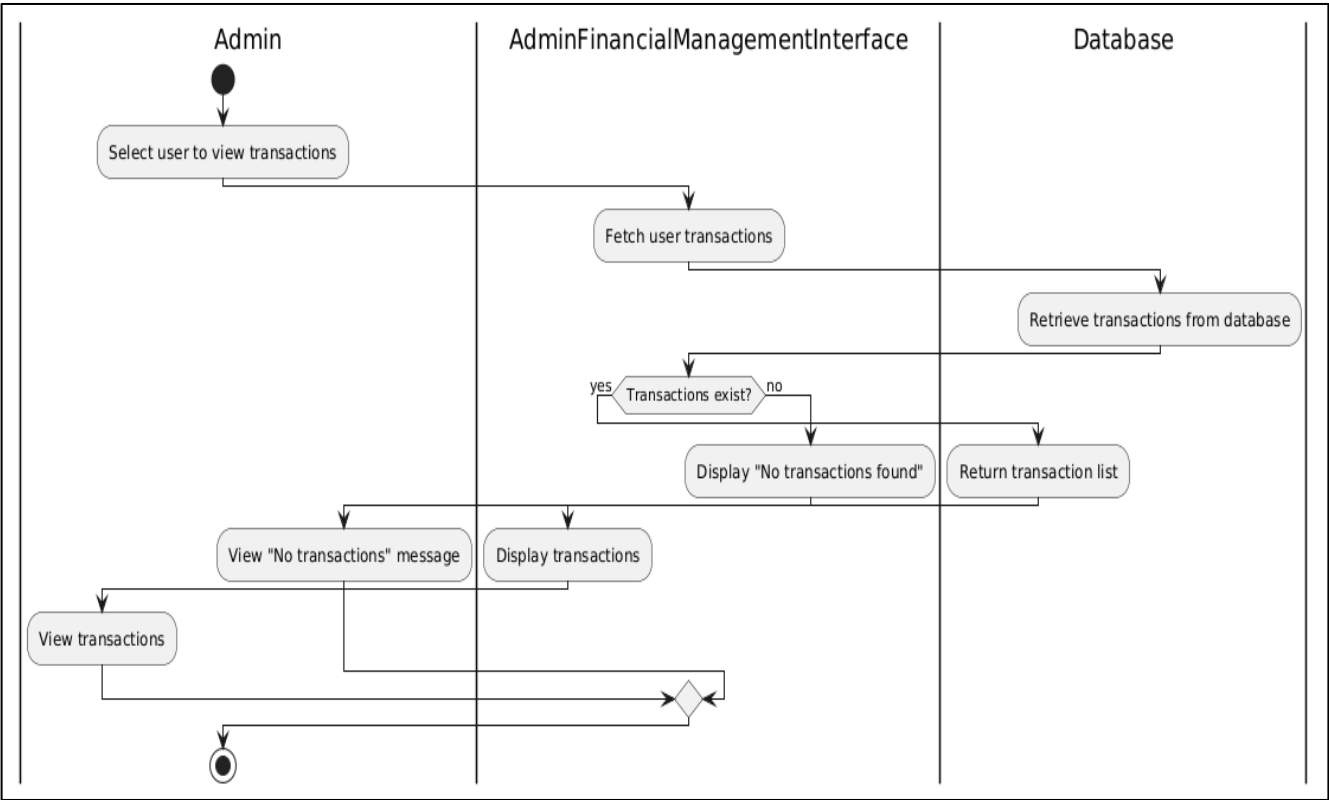


Figure 48: View User Transactions Activity Diagram.

4.8.10.2.3 View Reports

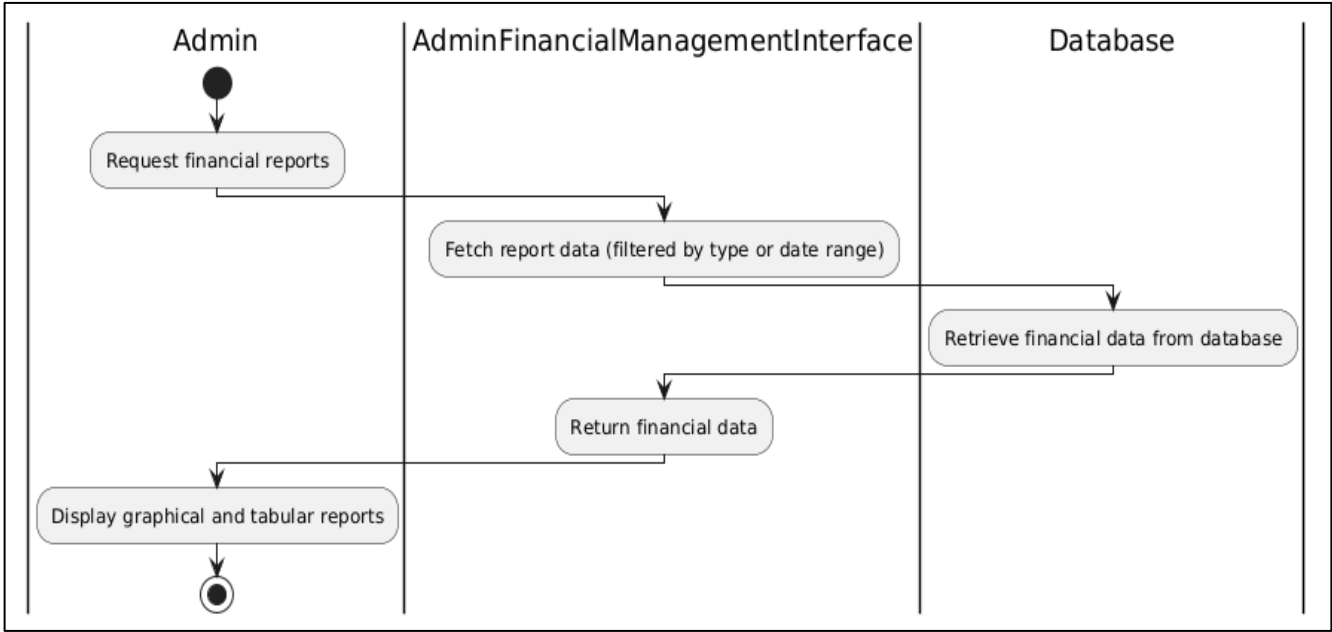


Figure 49: View Reports Activity Diagram.

4.8.10.2.4      Manage User Accounts

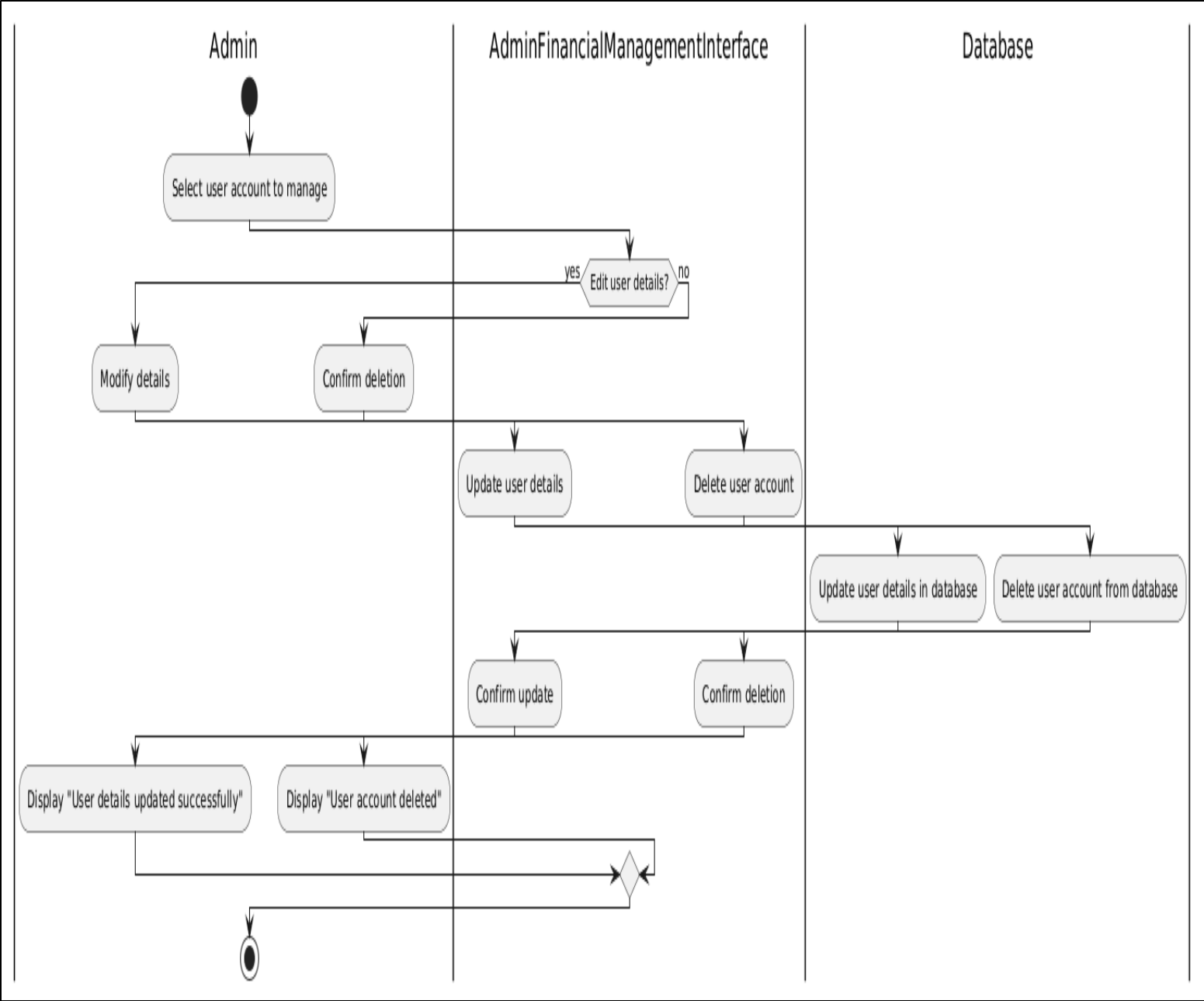


Figure 50: Manage User Accounts Activity Diagram.



4.8.10.3 User Functions

4.8.10.3.1 Register

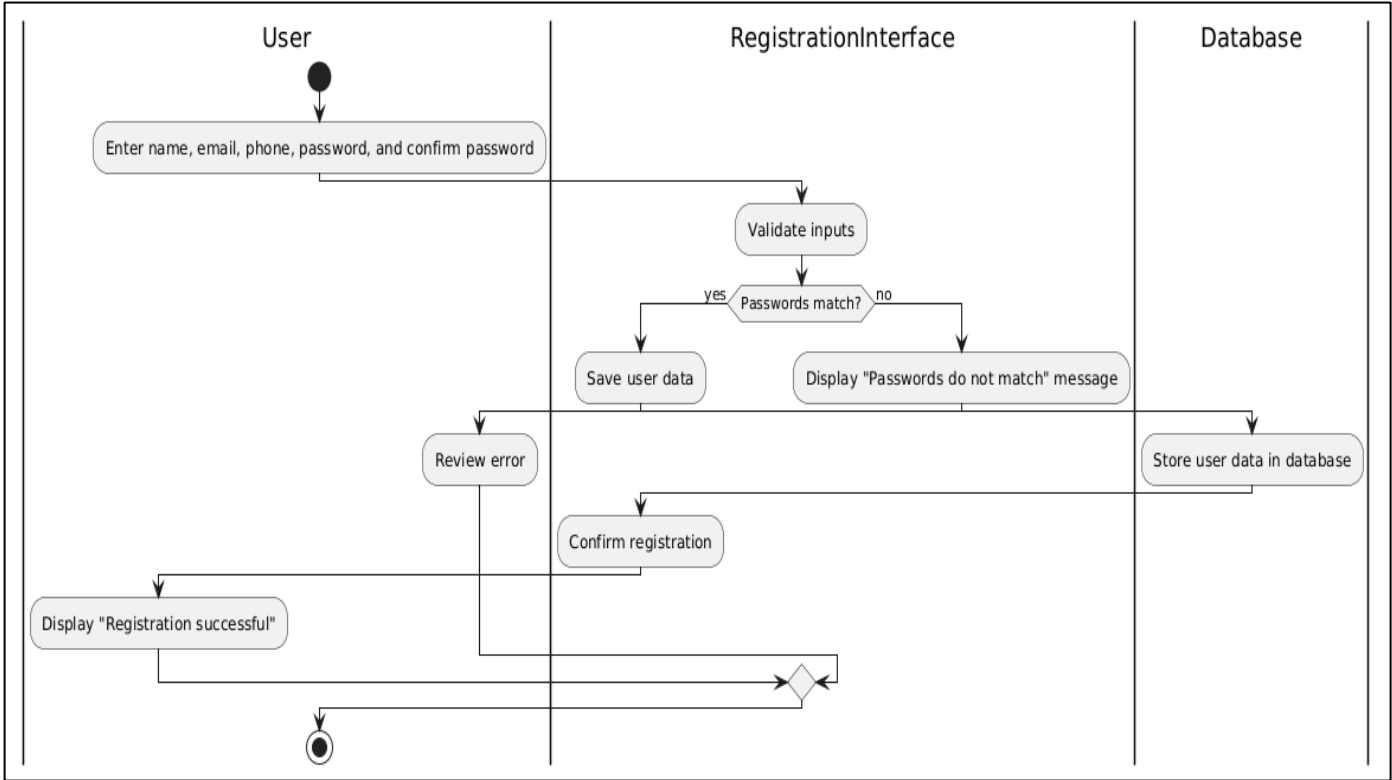


Figure 51: Register Activity Diagram.

4.8.10.3.2 Add Expense

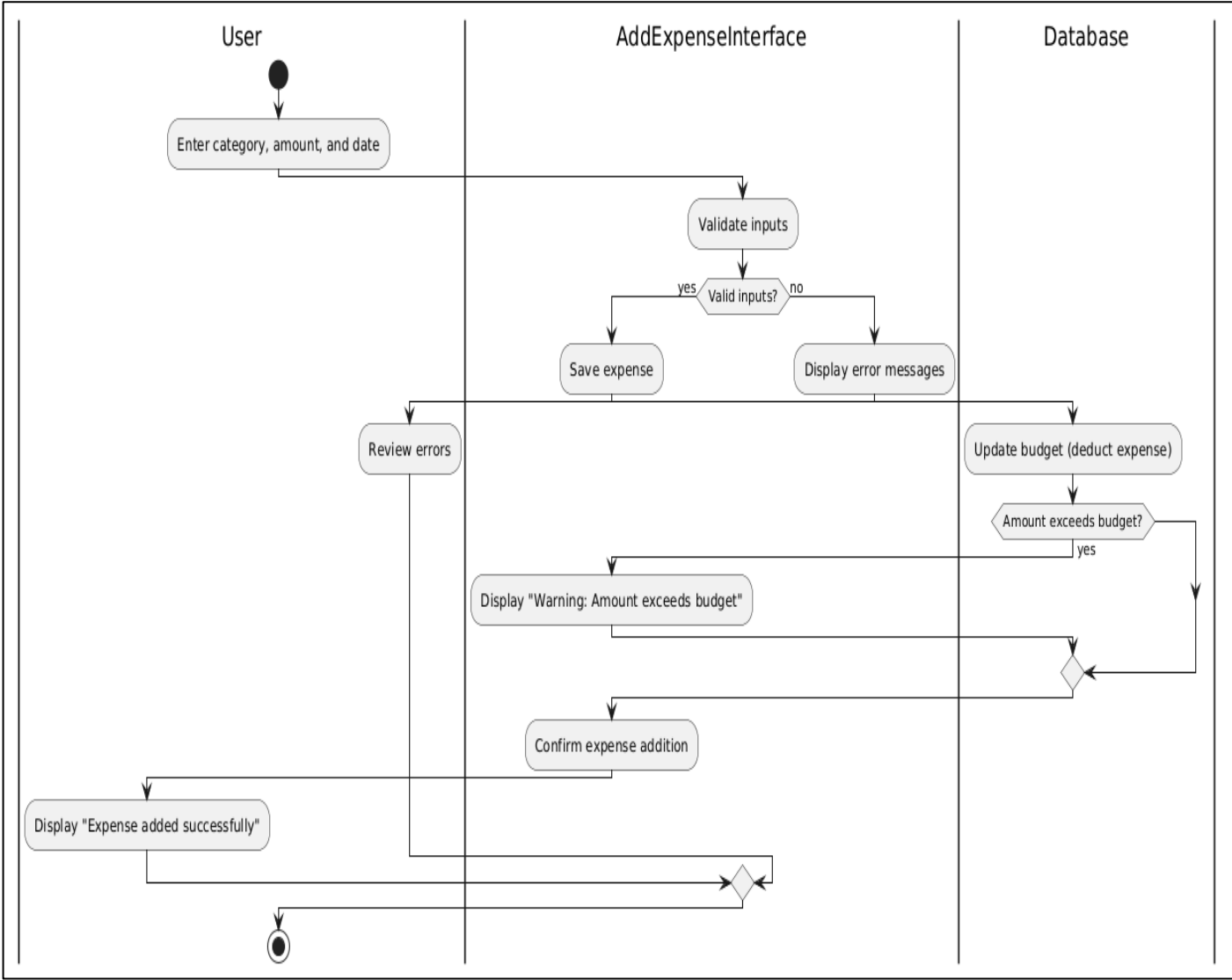


Figure 52: Add Expense Activity Diagram

4.8.10.3.3 Add Funds

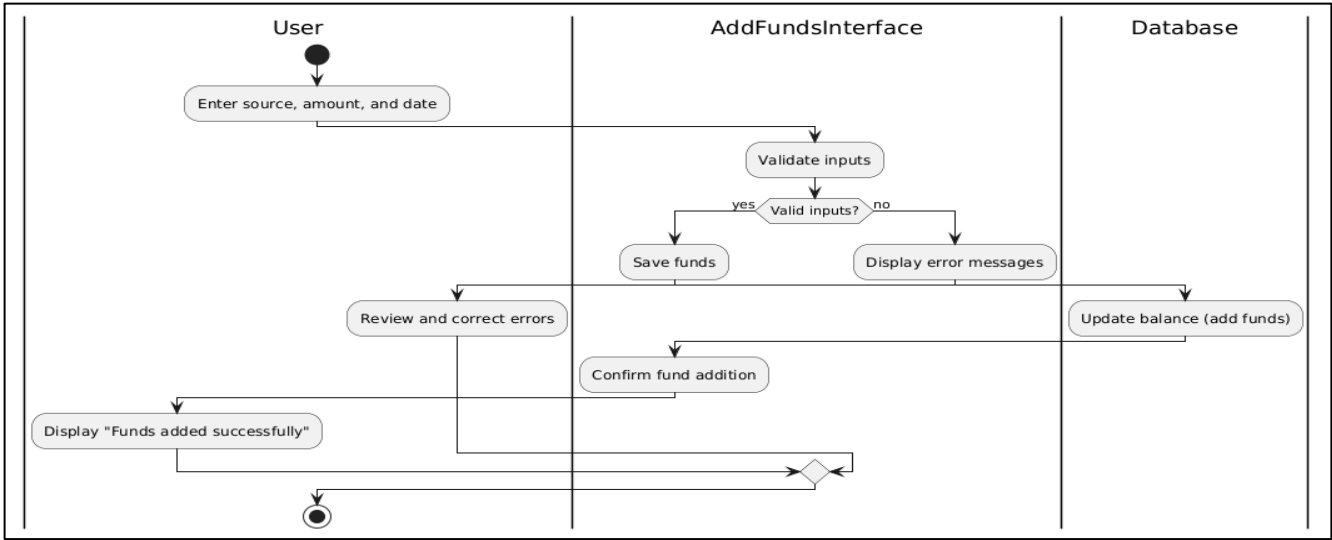


Figure 53 Add Funds Activity Diagram

4.8.10.3.4 View Expense Analysis

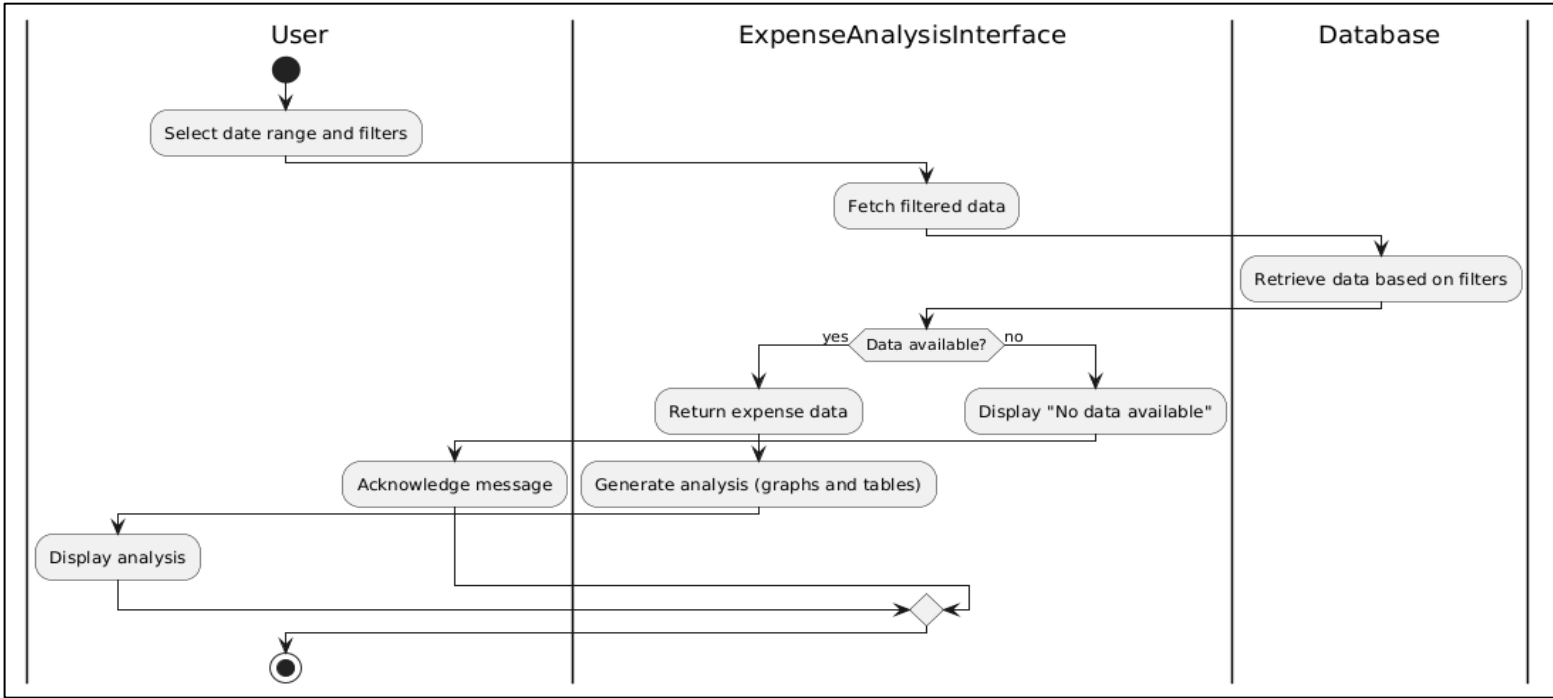


Figure 54:View Expense Analysis Activity Diagram.

2.1.1.1 Link Bank Account

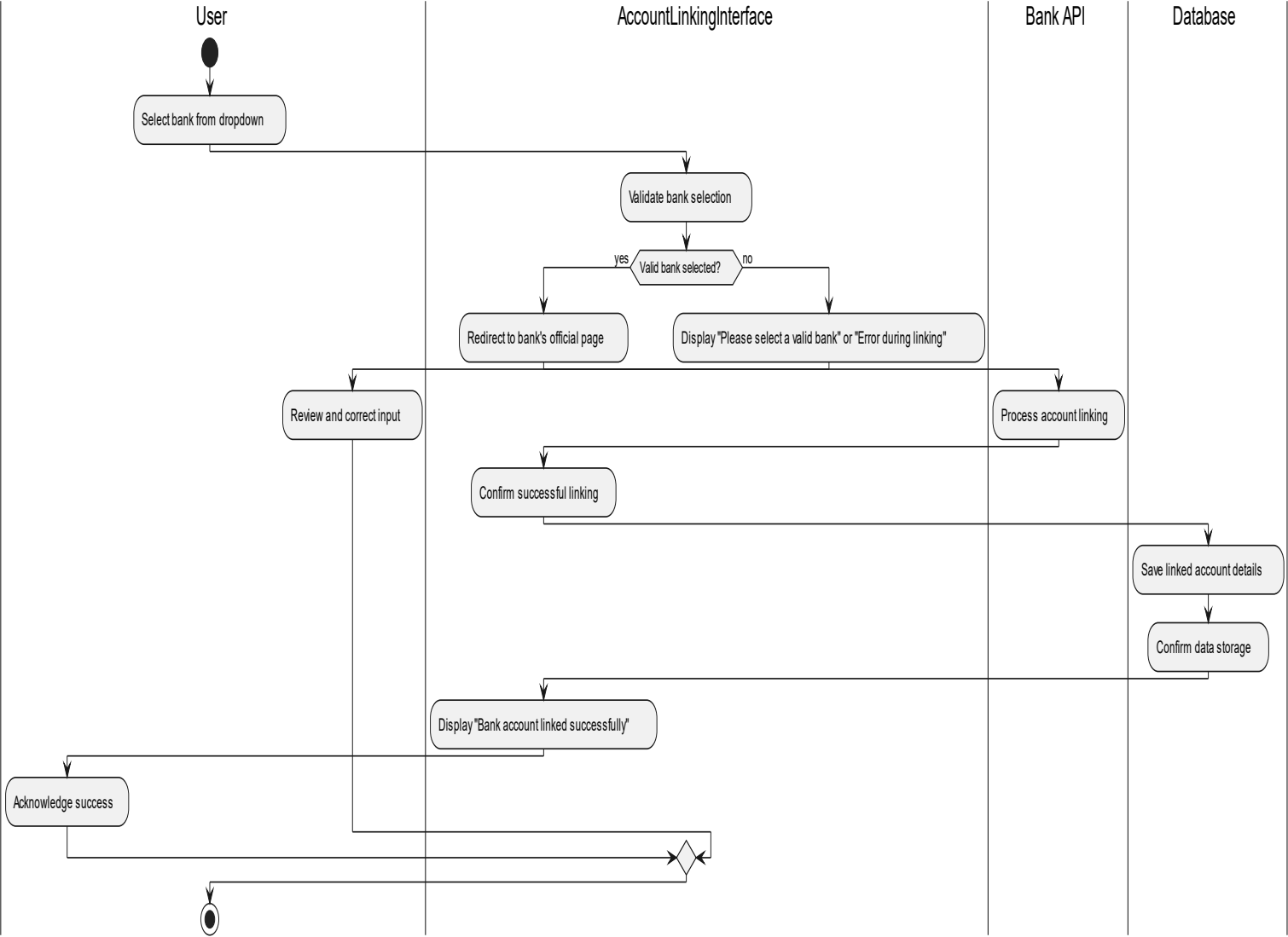


Figure 55: Link Bank Account Activity Diagram.

\*To see all activity Diagrams in a better Resolution Enter here: [Activity Diagrams](#)

## 4.9 Other Design Features

Every design feature in Expense Tracker has been showcased thoroughly in the previous sections and subsections of the current Software Design Document.

### 4.10 Requirements Traceability Matrix

Associated ID in SRS	Technical Assumptions and/or Customer Needs	Functional Requirement	System Component	Associated ID in SDS	User Type
3.1.1	Users need a secure way to register and access their accounts.	Allow users to register and log in securely.	Registration and Login Module	7.1.1	Standard User
3.1.1	Users may forget their credentials and need account recovery.	Provide password recovery options using email verification.	Password Recovery Interface	7.1.2	Standard User
3.1.2	Users require easy tracking of their expenses by category.	Allow users to add, edit, and delete expenses by category.	Expense Management Interface	7.2.1	Standard User
3.1.3	Users need to manage budgets and monitor their limits.	Allow users to set budgets and get alerts when limits are exceeded.	Budgeting Module	7.3.1	Standard User
3.1.4	Users expect visual insights into their spending.	Provide graphs and reports to analyze expenses.	Analytics and Reporting Dashboard	7.4.1	Standard User
3.1.5	Users handling multiple currencies require conversion support.	Convert foreign expenses to default currency in real time.	Currency Conversion System	7.5.1	Standard User

3.1.6	Admins need to oversee and manage user data effectively.	Enable admins to manage user accounts and transactions.	Admin Management Interface	7.6.1	Admin
3.1.7	Users need reminders to log expenses daily.	Send notifications to remind users to log expenses regularly.	Notification System	7.7.1	Standard User
3.1.8	Users want to export financial data for offline analysis.	Allow users to export data in PDF or CSV formats.	Data Export Module	7.8.1	Standard User
3.1.9	Users need to link their accounts to manage finances across platforms.	Allow integration with external bank accounts for automatic transaction imports.	Account Linking Interface	7.9.1	Standard User
3.1.10	Users need real-time alerts when nearing or exceeding budgets.	Notify users when spending limits are exceeded.	Budget Alert System	7.10.1	Standard User
3.1.11	Admins need access to detailed reports on system usage.	Provide admins with reports on user activities and system performance.	Admin Reporting Module	7.11.1	Admin
3.1.12	Users want to analyze spending habits over time.	Generate detailed expense trends based on past data.	Expense Analysis Dashboard	7.12.1	Standard User

Table 68 Requirements Traceability Matrix

5.

## Chapter 5: STS

## 5.1 Introduction

In this section we are going to discuss the objectives, testing strategy and scope as well as the definitions, acronyms and references used in this report.

### 5.1.1 Objectives

The objective of this Software Test Plan (STP) is outlining the scope, resources, approach, and schedule of all test's activities for the Expense Tracker application. The objective is to make sure that the system meets the non-functional and functional requirements specified in the SRS and additionally adheres to the design concepts outlined in the SDS. This plan details just how testing will verify system functionality, reliability, and performance to satisfy user expectations.

### 5.1.2 Testing Strategy

The testing strategy focuses on functional, integration, and performance tests across all subsystems, including User Management, Expense Management, Budget Management, Reporting & Analysis, Notifications, Data Backup & Restore, and Export Data. Each test level has defined objectives, deliverables, and schedules. This iterative process ensures all requirements are met while maintaining system reliability. Moreover, sections 2,3, and 4 will go over the test items and their performance and the features to be tested and the ones that will not be tested respectively. Additionally, the approach for implementing the test plan will be discussed in section 5. This STP will also go over the pass/fail criteria, the testing process with scheduling, and environmental requests.

### 5.1.3 Scope

The scope includes testing all functional modules, including:

- Core Functionality (e.g., expense entry, budget setting).
- Non-Functional aspects (e.g., performance, security, usability).
- Interactions between subsystems as specified in the SDS. Testing will be iterative across multiple phases, from unit testing to system testing.

### 5.1.4 Reference Material

The following documents formed the basis of this test plan:

1. **Software Project Management Plan (SPMP):** Provided timelines, resources, and milestones.
2. **Software Requirements Specification (SRS):** Defined the system requirements.
3. **Software Design Specification (SDS):** Detailed architectural and subsystem design.
4. IEEE 829-1998 Standard for Software Test Documentation.



### 5.1.5 Definitions and Acronyms

Terminology	Definition
Functional Testing	Testing that verifies that an implementation of some function operates correctly.
Non-Functional Testing	A type of Software Testing that is performed to verify the non-functional requirements of the application
Subsystem	A distinct, functional unit within a larger system.
Test Case	A set of conditions and inputs used to verify a specific functionality or feature of the system.
Defect	Any deviation from the expected outcome or behavior of the system.

*Table 69 Terminology Table*

Acronym	Definition
SPMP	Software Project Management Plan
SRS	Software Requirements Specification
SDS	Software Design Specification
STP	Software Test Plan
IEEE	Institute of Electrical and Electronics Engineers
DFD	Data Flow Diagram
CSV	Comma-Separated Values
PDF	Portable Document Format
UAT	User Acceptance Testing

*Table 70 Acronyms Table*

## 5.2 Test Items

All items in Expense Tracker will be tested alongside the stored data. Of course, this includes the GUI elements which will help verify the program's integrity. All information regarding requirements is documented in the Software Requirements Specification (SRS) while the information regarding the program's design is documented in the Software Design Specification (SDS).

### 5.2.1 Program Modules

The tests will be conducted in "Expense Tracker", with a detailed description written in section 5 of this document.

### 5.2.2 Job Control Procedures

The components will be tested one by one before properly testing the functions. Interface testing comes next, followed by security and performance testing. Finally, acceptance testing will verify if the program is fit for public launch by eliminating leftover bugs or errors.

### 5.2.3 User Procedures

All user procedure testing will be done in "Expense Tracker", with a detailed description of the tests written in section 5 of this document.

### 5.2.4 Operator Procedures

Operators are advised to do the following:

1. Test individual and grouped functionalities.
2. Test system performance and how it handles production-like load conditions.
3. Ensure application security against unauthorized access.

Help Desk procedures:

1. Handle user-reported issues efficiently.
2. Ensure high-difficulty issues are escalated to and handled by the appropriate teams.
3. Provide self-service options and FAQs for users to resolve common issues.
4. Log incidents from issues or requests by users for tracking.

### 5.3 Features To Be Tested

All the functions specified in the SRS will be tested alongside the functions documented in the SDS.

### 5.4 Features Not To Be Tested

All the functions specified in the SRS will be tested alongside the functions documented in the SDS.

### 5.5 Approach

This section details all the necessary information regarding the testing of the “Expense Tracker” program’s functionality, and it is split into the following subsections:

- Component Testing
- Integration Testing
- Job Stream Testing
- Conversion Testing
- Interface Testing
- Security Testing
- Recovery Testing
- Performance Testing
- Regression Testing
- Acceptance Testing
- Beta Testing

#### 5.5.1 Component Testing

In this test, each component will be tested separately in order to determine whether it achieves the expected results. This method of testing helps to validate and verify the functionalities of each component as well as addressing any newfound bugs or errors and resolve them prior to conducting any other tests.

**Technique:** Test each component independently.

**Completion criteria:** Each component achieves the expected performance and results prior to integration.

### 5.5.1.1 Common Functions

#### 5.5.1.1.1 Login

Test ID	Login
Prerequisite(s)	<ul style="list-style-type: none"><li>- Email must be registered and follow a valid format.</li><li>- Password must match the stored encrypted password.</li></ul>
Test Procedure	Click on “Login” button with each of the following: <ul style="list-style-type: none"><li>a. Correct email and password</li><li>b. Correct email and incorrect password</li><li>c. Incorrect email and correct password</li><li>d. Incorrect email and password</li></ul>
Expected Result	<ul style="list-style-type: none"><li>a. Access either admin interface or user interface depending on privileges</li><li>b. An error will be displayed: “Invalid email or password, please try again.”</li></ul>
Actual Result	Same as expected result
Verified (yes/no)	Yes

*Table 71 “Login” Table*

#### 5.5.1.1.2 Forgot Password

Test ID	Forgot Password
Prerequisite(s)	<ul style="list-style-type: none"><li>- Account must be registered.</li><li>- Token must be valid and unexpired</li><li>- Password must meet complexity requirements</li></ul>
Test Procedure	Click on “Send” button with each of the following: <ul style="list-style-type: none"><li>a. Correct email</li><li>b. Incorrect email</li></ul>
Expected Result	<ul style="list-style-type: none"><li>a. Password reset link will be sent to email</li><li>b. An error will be displayed: “Invalid email, please try again.”</li></ul>
Actual Result	Same as expected result
Verified (yes/no)	Yes

*Table 72 “Forgot Password” Table*

### 5.5.1.1.3 Change Password

Test ID	Change Password
Prerequisite(s)	<ul style="list-style-type: none"> <li>- Old password must match the stored hash</li> <li>- New password must meet complexity requirements</li> </ul>
Test Procedure	<p>Click on “Confirm” button with each of the following:</p> <ol style="list-style-type: none"> <li>Correct old, confirm, and new password</li> <li>Correct old, confirm, and new password but doesn’t meet complexity requirements</li> <li>Correct old and new password but not confirm password</li> <li>Incorrect old password</li> </ol>
Expected Result	<ol style="list-style-type: none"> <li>Password successfully changed with a confirmation message “Password changed successfully”</li> <li>An error will be displayed: “Password must be at least 8 characters long.”</li> <li>An error will be displayed: “New password doesn’t match the confirmed password.”</li> <li>An error will be displayed: “Invalid old password.”</li> </ol>
Actual Result	Same as expected result
Verified (yes/no)	Yes

Table 73 "Change Password" Table

### 5.5.1.2 Admin Functions

#### 5.5.1.2.1 Delete Transaction

Test ID	Delete Transaction
Prerequisite(s)	<ul style="list-style-type: none"> <li>- Have a registered admin account</li> </ul>
Test Procedure	<ul style="list-style-type: none"> <li>- Select transaction</li> <li>- Click “Delete” button</li> <li>- Click “Yes” after pop-up “Are you sure you want to delete this transaction?”</li> </ul>
Expected Result	<ul style="list-style-type: none"> <li>- Delete transaction from database and display confirmation message: “Transaction deleted successfully.”</li> </ul>
Actual Result	Same as expected result
Verified (yes/no)	Yes

Table 74 "Delete Transaction" Table

### 5.5.1.2.2 View User Transactions

Test ID	View User Transactions
Prerequisite(s)	<ul style="list-style-type: none"><li>- Have a registered admin account</li><li>- User transactions exist in the database</li></ul>
Test Procedure	<ul style="list-style-type: none"><li>- Click on “View Transactions” button</li></ul>
Expected Result	<ul style="list-style-type: none"><li>- User transactions will be displayed</li><li>- No user transactions exist and display message: “No transactions available.”</li></ul>
Actual Result	Same as expected result
Verified (yes/no)	Yes

Table 75 "View User Transactions" Table

### 5.5.1.2.3 View Reports

Test ID	View Reports
Prerequisite(s)	<ul style="list-style-type: none"><li>- Have a registered admin account</li></ul>
Test Procedure	<ul style="list-style-type: none"><li>- Click “View Reports” button</li></ul>
Expected Result	<ul style="list-style-type: none"><li>- Detailed financial reports are displayed alongside filtering by date range</li></ul>
Actual Result	Same as expected result
Verified (yes/no)	Yes

Table 76 "View Reports" Table

### 5.5.1.2.4 Manage User Accounts

Test ID	Manage User Accounts
Prerequisite(s)	- Have a registered admin account
Test Procedure	<ul style="list-style-type: none"> <li>- Click on “Manage Users”</li> <li>- Click on any user</li> <li>- Click on “Delete User”</li> <li>- Click “Yes”</li> </ul>
Expected Result	<ul style="list-style-type: none"> <li>- Displays list of all users with options to edit or delete</li> <li>- Displays user details and allow admin to edit or delete account</li> <li>- Displays confirmation message “Are you sure you want to delete this user?”</li> <li>- Deletes user and displays confirmation message “User account deleted successfully.”</li> </ul>
Actual Result	Same as expected result
Verified (yes/no)	Yes

Table 77 "Manage User Accounts" Table

### 5.5.1.3 User Functions

#### 5.5.1.3.1 Register

Test ID	Register
Prerequisite(s)	- None
Test Procedure	<ul style="list-style-type: none"> <li>- Enter first name, last name, email, phone number, password, and confirm password correctly</li> <li>- Enter most fields with a few empty</li> <li>- Enter all fields correctly except email</li> <li>- Enter all fields correctly except phone number</li> <li>- Enter all fields correctly but password doesn't match confirm password</li> <li>- Enter all fields but password doesn't fulfill requirements</li> <li>- Enter all fields but email currently exists in the User table</li> </ul>
Expected Result	<ul style="list-style-type: none"> <li>- Account is created and user is redirected to login page</li> <li>- Red asterisks show up at empty fields</li> <li>- Display error message “Invalid email format!”</li> <li>- Display error message “Invalid phone number format!”</li> <li>- Display error message “Passwords do not match!”</li> </ul>

	<ul style="list-style-type: none"> <li>- Display error message “Password must be at least 8 characters long.”</li> <li>- Display error message “Email already in use.”</li> </ul>
<b>Actual Result</b>	Same as expected result
<b>Verified (yes/no)</b>	Yes

Table 78 "Register" Table

### 5.5.1.3.2 Add Expense

Test ID	Add Expense
<b>Prerequisite(s)</b>	<ul style="list-style-type: none"> <li>- Registered user account</li> </ul>
<b>Test Procedure</b>	<ul style="list-style-type: none"> <li>- Select expense category</li> <li>- Enter amount to add               <ul style="list-style-type: none"> <li>a. Positive value</li> <li>b. Negative value</li> <li>c. Amount exceeds budget</li> </ul> </li> <li>- Select date</li> <li>- Enter notes (optional feature)</li> </ul>
<b>Expected Result</b>	<ul style="list-style-type: none"> <li>- Expense is successfully saved to database with confirmation message “Expense added successfully.”</li> <li>- Display error message “Amount must be a positive value.”</li> <li>- Display warning message “Warning: Expense exceeds budget.”</li> </ul>
<b>Actual Result</b>	Same as expected result
<b>Verified (yes/no)</b>	Yes

Table 79 "Add Expense" Table

### 5.5.1.3.3 Add Funds

Test ID	Add Funds
<b>Prerequisite(s)</b>	<ul style="list-style-type: none"> <li>- Registered user account</li> </ul>
<b>Test Procedure</b>	<ul style="list-style-type: none"> <li>- Select source of income</li> <li>- Enter amount to add               <ul style="list-style-type: none"> <li>a. Positive value</li> <li>b. Negative value</li> </ul> </li> <li>- Select date of income</li> </ul>



	<ul style="list-style-type: none"> <li>- Enter notes (optional feature)</li> </ul>
<b>Expected Result</b>	<ul style="list-style-type: none"> <li>- Adds funds to user account and display message “Funds added successfully.”</li> <li>- Display error message “Amount must be a positive value.”</li> </ul>
<b>Actual Result</b>	Same as expected result
<b>Verified (yes/no)</b>	Yes

Table 80 "Add Funds" Table

#### 5.5.1.3.4 View Expense Analysis

Test ID	View Expense Analysis
<b>Prerequisite(s)</b>	<ul style="list-style-type: none"> <li>- Registered user account</li> <li>- Expense was added</li> </ul>
<b>Test Procedure</b>	<ul style="list-style-type: none"> <li>- Select date range for analysis</li> <li>- Select category filter (optional feature)</li> <li>- Display total expense, category-wise breakdown, and trends</li> <li>- Display graphical reports</li> <li>- Export to CSV, Excel, or PDF files</li> </ul>
<b>Expected Result</b>	<ul style="list-style-type: none"> <li>- Expense analysis is displayed with different formats, data is exported to CSV, Excel or PDF files, and displays message “Analysis exported successfully.”</li> <li>- No transactions match filters and displays message “No transactions found.”</li> </ul>
<b>Actual Result</b>	Same as expected result
<b>Verified (yes/no)</b>	Yes

Table 81 "View Expense Analysis" Table

#### 5.5.1.3.5 Link Bank Account

Test ID	Link Bank Account
<b>Prerequisite(s)</b>	<ul style="list-style-type: none"> <li>- Registered user account</li> </ul>
<b>Test Procedure</b>	<ul style="list-style-type: none"> <li>- Select bank from dropdown list of supported banks</li> <li>- Click “Link” button</li> </ul>

<b>Expected Result</b>	<ul style="list-style-type: none"> <li>- Displays message “Redirecting to the bank’s official page.” and user is redirected to selected bank’s official website to complete the linking process</li> <li>- Displays error message “Please select a valid bank” if no bank is selected when clicking “Link” button</li> </ul>
<b>Actual Result</b>	Same as expected result
<b>Verified (yes/no)</b>	Yes

Table 82 "Link Bank Account" Table

### 5.5.2 Integration Testing

After confirming the functionality of each component, the integration testing is conducted to ensure that the interactions between components work as expected. Completing this test indicates that the complete system is fully functional.

**Technique:** Testing the functions incrementally to determine bugs and errors more efficiently.

**Completion criteria:** Every component is added and combined with no bugs remaining.

#### 5.5.2.1 Admin

Test ID	Admin
<b>Prerequisite(s)</b>	<ul style="list-style-type: none"> <li>- Logged in to admin account</li> </ul>
<b>Test Procedure</b>	<ul style="list-style-type: none"> <li>- The following procedure <ol style="list-style-type: none"> <li>1. Access to “Transaction list” and do the component’s test cases for “Delete Transaction” as mentioned in (section 5.1.2.1)</li> <li>2. Access to “Transaction list” and do the component’s test cases for “View User Transactions” as mentioned in (section 5.1.2.2)</li> <li>3. Access to “Reports” and do the component’s test cases for “View Reports” as mentioned in (section 5.1.2.3)</li> <li>4. Access to “Users” and do the component’s test cases for “Manage User Accounts” as mentioned in (section 5.1.2.4)</li> </ol> </li> </ul>
<b>Expected Result</b>	<ol style="list-style-type: none"> <li>1. Same as the previous result mentioned in (section 5.1.2.1)</li> </ol>

	<ol style="list-style-type: none"> <li>2. Same as the previous result mentioned in (section 5.1.2.2)</li> <li>3. Same as the previous result mentioned in (section 5.1.2.3)</li> <li>4. Same as the previous result mentioned in (section 5.1.2.4)</li> </ol>
<b>Actual Result</b>	Same as expected result
<b>Verified (yes/no)</b>	Yes

Table 83 Admin Table

### 5.5.2.2 User

Test ID	User
<b>Prerequisite(s)</b>	- Logged in to user account
<b>Test Procedure</b>	<ul style="list-style-type: none"> <li>- The following procedure</li> <li>1. Access to “Register page” and do the component’s test cases for “Register” as mentioned in (section 5.1.3.1)</li> <li>2. Access to “Expenses” and do the component’s test cases for “Add Expense” as mentioned in (section 5.1.3.2)</li> <li>3. Access to “Funds” and do the component’s test cases for “Add Funds” as mentioned in (section 5.1.3.3)</li> <li>4. Access to “Expenses” and do the component’s test cases for “View Expense Analysis” as mentioned in (section 5.1.3.4)</li> <li>5. Access to “Account” and do the component’s test cases for “Link Bank Account” as mentioned in (section 5.1.3.5)</li> </ul>
<b>Expected Result</b>	<ol style="list-style-type: none"> <li>1. Same as the previous result mentioned in (section 5.1.3.1)</li> <li>2. Same as the previous result mentioned in (section 5.1.3.2)</li> <li>3. Same as the previous result mentioned in (section 5.1.3.3)</li> <li>4. Same as the previous result mentioned in (section 5.1.3.4)</li> <li>5. Same as the previous result mentioned in (section 5.1.3.5)</li> </ol>
<b>Actual Result</b>	Same as expected result

Verified (yes/no)	Yes
----------------------	-----

Table 84 User Table

5.5.3    **Conversion Testing**

This test focuses on the system’s ability to transfer or convert data from an old format to a new one during migration/upgrades.

**Technique:** Ensure data compatibility with newer system versions, and check for data completeness and accuracy post-upgrades.

**Completion criteria:** Data remains complete and accurate to the old format version after migration.

5.5.4    **Job Stream Testing**

This test is conducted to validate proper job sequencing, data transfers, and ensure proper workflow between job sequences. The purpose of this test is to determine if the system components function correctly and dependently to produce the required output without slowing down performance.

**Technique:** Documenting results based on dependencies between jobs by ensuring jobs wait for outputs of other jobs to take as inputs, and check for workflow speed and integrity.

**Completion criteria:** Job workflow is smooth, and performance doesn’t slow from dependencies.

5.5.5    **Interface Testing**

After completing the integration testing, interface testing is done to detect faults throughout the entire system to ensure that every interaction is working as expected and all errors within the program are handled. The test is performed as shown below:

**Login****Admin:**

- Enter email and password
- If email and password are valid then login to admin homepage
- Else, an error message will be displayed if email and/or password are invalid

**User:**

- Enter email and password
- If email and password are valid then login to user homepage
- Else, an error message will be displayed if email and/or password are invalid

**Mentioned in section 5.1.1.1**

<b>Forgot Password</b>	- Mentioned in section 5.1.1.2
<b>Change Password</b>	- Mentioned in section 5.1.1.3
<b>Register</b>	- Mentioned in section 5.1.3.1
<b>Add Expense</b>	- Mentioned in section 5.1.3.2

*Table 85 Interface Testing Table*

### 5.5.6 Security Testing

Security testing is done to ensure that all the users' information and data are safe from unauthorized access.

**Technique:** No entry to the application is granted without a valid and registered account.

**Completion criteria:** Entry is granted for authorized users only.

### 5.5.7 Recovery Testing

Recovery testing looks at how well the system recovers from crashes, hardware failures, or any disaster that affects the software. It involves documenting the system's ability to resume functionality as normal as well as ensuring data integrity to avoid unwanted data loss.

**Technique:** Crashing, forced shutdowns, network disruption, and data recovery techniques such as backup restorations.

**Completion criteria:** System functions safely and data is preserved in the worst-case scenarios.

### 5.5.8 Performance Testing

Performance testing is conducted with the goal of the application's speed and responsiveness to be at a high level for user convenience. The user should be able to access the application at all times, and the program should handle and stabilize massive user load in order to prevent unnecessary crashes that would ruin the user's experience.

**Technique:** Push the application to its limits in ways that test its response time, availability, portability, and scalability to ensure that the performance lives up to users' expectations.

**Completion criteria:** All the test cases are complete within an acceptable time.

### 5.5.9 Regression Testing

In regression testing, a thorough check and testing is conducted on functionalities to determine if any accidental changes done in the present affected existing functionalities.

**Technique:** Perform double-testing on components and/or the entire system to find out if any bugs or errors have occurred after present modifications.

**Completion criteria:** The software remains the same with no hidden bugs remaining post-modifications.

### 5.5.10 Acceptance Testing

The acceptance testing verifies whether the application meets the demands of users and clients or not and determines if it is ready for delivery.

**Technique:** Collect feedback from users to determine whether the application is ready or requires adjustments.

**Completion criteria:** Application meets expectations and is released for the public.

### 5.5.11 Beta Testing

Beta testing aims to eliminate any remaining traces of bugs and errors by sharing the software with a number of users for collecting feedback from multiple perspectives. The goal is to enhance communications with customers to help detect bugs that weren't found during development and testing.

**Technique:** Share the software to a select number of end-users to increase chances of bug detection while collecting feedback for further improvements.

**Completion criteria:** Software is bug-free and error-free.

## 5.6 Pass / Fail Criteria

The criteria for determining whether a test passes or fails are based on the following conditions:

- 1- All functional requirements, as defined in the SRS, are met without deviation.
- 2- Non-functional benchmarks (e.g., performance, security, usability) are achieved.
- 3- The actual results match the expected results for all defined test scenarios.
- 4- No critical or severe defects are identified.
- 5- Results fall within acceptable tolerances ranges.

### 5.6.1 Suspension Criteria

Testing activities will be suspended if any of the following conditions occur:

- A critical defect is identified that prevents the execution of more test cases (e.g., system crashes, data loss).
- Crucial test resources (e.g., data, hardware, or software) become unavailable.
- A major subsystem fails (e.g., User Management, Budget Management) to meet basic requirements.

### 5.6.2 Resumption Criteria

Testing will resume once the following conditions are met:

- The critical defects identified during suspension have been resolved.
- All dependent subsystems are functioning correctly.
- Required test resources are restored and operational

### 5.6.3 Approval Criteria

Testing will be considered complete and approved if:

- All critical subsystems achieve a 100% pass rate.
- At least 90% of all test cases have passed without critical or major defects.
- Any remaining minor defects are documented and await resolution if they do not affect core functionality.
- All performance benchmarks, as outlined in the SRS are met or exceeded.

## 5.7 Testing Process

(Identify the methods and criteria used in performing test activities. Define the specific techniques and procedures for each type of test. Define the detailed criteria for evaluating test results.)

5.7.1 Test Deliverables

The deliverable document that will be produced from the testing process is the Software Test Plan (STP). After passing the test phases, the project will be ready for delivery.

5.7.2 Testing Tasks

The testing process involves the following tasks:

Once the SRS and SDS are complete, testing will focus on creating test cases and running various checks, including unit, integration, interface, security, performance, and regression tests. will also be conducted to ensure the system meets the user’s needs. Throughout the process, all activities and issues will be carefully tracked, and the system will be tested in different environments to ensure reliability and maintainability.

5.7.3 Responsibilities

The responsibilities for the testing process are distributed as follows:

- Testers: Design, execute, and document test cases, and report issues.
- Developers: Address issues identified during testing and implement fixes.
- Project Manager: Oversee the testing process and ensure timely completion.
- End-Users: Participate in UAT to provide feedback on usability and functionality.

5.7.4 Resources

The resources required for testing include:

Resource Category	Description
Hardware	High-performance computers with internet connectivity for testing
Software	NetBeans, MySQL Workbench, and other relevant tools.
Support	None
Human	All the team members depend on their knowledge and skills, skilled testers, developers, and project manager.

Table 86 Resources Table



5.7.5 Schedule

Managing the project in a correct way to get satisfying results. Therefore, the high-level schedule for testing activities is as follows:

Task	Date
Software Requirement Specification (SRS)	Nov 2, 2024
Software Design Specification (SDS)	Nov 16, 2024
Develop test cases	Nov 18, 2024
Execute testing activities using various testing approaches	Nov 20, 2024
Troubleshoot errors occurred during testing	Nov 22, 2024
Modify the system accordingly	Nov 24, 2024
Modify Software testing plan (STP)	Nov 28, 2024

Table 87 Schedule Table

## 5.8 Environmental Requirements

The environment required to properly test the Expense Tracker application is described in this section.

### 5.8.1 Hardware

The hardware devices required to test the application are:

- High speed and secure internet connection.
- Personal computers
- Smartphones

### 5.8.2 Software

The software needed to apply testing on the application are:

- NetBeans
- MySQL Workbench
- Windows 10 or any operating system

### 5.8.3 Security

The testing environment security and asset protection requirements:

- Securing test environments to protect sensitive user data, particularly during testing bank account integration and user authentication.
- Using encrypted communication protocols (e.g., HTTPS).
- Testing on virtual machines or isolated devices to prevent unauthorized access during testing.

### 5.8.4 Tools

The tools employed in the testing effort are:

- Firebase Test Lab: Test the app on various virtual devices and configurations.

### 5.8.5 Publications

The documents and publications that are required to support the software test plan document (STP) are:

- Software Requirement Specification (SRS)
- Software Design Specification (SDS)
- Test plan and test case templates for structured testing.
- IEEE Std 829-1998 for testing documentation standards.

### 5.8.6 Risks and Assumptions

In this section, we will identify constraints on testing, risks and assumptions associated with testing tasks, and a contingency plan for each risk factor.

#### Constraints:

- Limited availability of team members to perform extensive testing during peak academic periods.
- Testing device unavailability due to budget constraints.
- Potential delays in acquiring third-party APIs for bank account integration

#### Risks:

- **Risk:** Failure to simulate real-world scenarios for bank integration testing.
  - **Mitigation:** Use mock APIs and simulated data to mimic real-world interactions.
- **Risk:** Incompatibility of the application on certain Android/iOS versions.
  - **Mitigation:** Regularly update development environments and test on a range of devices/emulators.
- **Risk:** Miscommunication between developers and testers.
  - **Mitigation:** Weekly sync meetings and clearly documented test cases.
- **Risk:** Lack of time to fully test all scenarios.
  - **Mitigation:** Prioritize critical test cases and automate routine ones.






#### Contingency Plan:

- Have backup devices and simulated environments for urgent needs.
- Maintain a buffer period in the project schedule for extended testing if issues arise.

## 5.9 Change Management Procedures

To obtain approval, any changes to the test plan must be followed by a sequence of steps. Any modifications that the client or team members would want to see will be suggested to the team and the manager for discussion. The plan will be changed with recorded changes if everyone agrees with the suggested adjustment.

### 5.10 Plan Approvals

#	Name	Signature	Date
1	Dr. Nehad Ibrahim		
2	Fawaz Altahini		29/11/2024
3	Ali Al hushayyish		29/11/2024
4	Sultan Almutairi		29/11/2024
5	Osama Almutairi		29/11/2024
6	Abdulrahman Alqahtani		29/11/2024

*Table 88 Plan Approvals*

# Appendix A: Status Report 1



## CS 411 – Software Engineering

Term 1 – 2024/2025

### Expense Tracker Status Report #1

Period from September 12, 2024, to October 17, 2024.

- ✓ Group #: 2
- ✓ Members of the group:

#	Name	ID	Role
1	Fawaz Altahini	2220003471	Leader
2	Ali Al hushayyish	2220005429	Member
3	Sultan Almutairi	2220003546	Member
4	Osama Almutairi	2220006826	Member
5	Abdulrahman Alqahtani	2220004752	Member

- ✓ **Tasks planned to work on during the specified period:**

1. Submit project proposal.
2. Submit the project management plan.

- ✓ **Status of each task including the names of members worked on each task:**

1. Project proposal and its status is completed by all team members.
2. Project management plan and its status is completed by all team members.

- ✓ **Difficulties:**

1. Communication, since we are all students, we don't have a formal meeting place.
2. Documenting, we didn't access the document at one time instead we charged one of us to prepare the documents.

✓ **Pending tasks:**

1. Project requirements.
2. Project design.
3. Project test plan.
4. Delivery of the project.
5. Presentation.

✓ **Next planned tasks:**

1. Submit the project requirements plan.
2. Submit the project design plan.
3. Build the code and database.
4. Submit the project test plan.
5. Present the project.

✓ **Number of group meetings you had during the specified time:**

Task No.	Task	No. of meetings
1	Project Proposal	1, Face to face
2	Project management plan	1, Face to face

✓ **Attached document(s):**

No attached documents.

✓ **Additional notes:**

None.

----- For instructor use only -----

Date:

Comments:

# Appendix B: Status Report 2





**CS 411 – Software Engineering  
Term 1 – 2024/2025**

**Expense Tracker Status Report #2**

Period from October 18, 2024, to November 30, 2024.

- ✓ Group #: 2
- ✓ Members of the group:

#	Name	ID	Role
1	Fawaz Altahini	2220003471	Leader
2	Ali Al hushayyish	2220005429	Member
3	Sultan Almutairi	2220003546	Member
4	Osama Almutairi	2220006826	Member
5	Abdulrahman Alqahtani	2220004752	Member

✓ **Tasks planned to work on during the specified period:**

3. Submit the Project Requirements (SRS).
4. Submit the Project Design (SDS).
5. Submit the Project Test Plan (STS).
6. Submit the second status report.

✓ **Status of each task including the names of members worked on each task:**

3. Submit the Project Requirements and its status is completed by all team members.
4. Submit the Project Design and its status is completed by all team members.
5. Submit the Project Test Plan and its status is completed by all team members.
6. Submit the second status report and its status is completed by all team members

✓ **Difficulties:**

3. Meeting the deadlines, Since we all are students we faced difficulties meeting the deadlines because of our studying.

- 4. Group meetings, one of our deliveries occurred on vacation so some team members were traveling and it led to poor communication.

✓ **Pending tasks:**

- 6. Delivery of the project.
- 7. Presentation.

✓ **Next planned tasks:**

- 6. Submit the whole project.
- 7. Present the project.

✓ **Number of group meetings you had during the specified time:**

Task No.	Task	No. of meetings
1	Project Requirements	1, Face to face
2	Project Design	1, Face to face
3	Project Test Plan	1, Face to face
4	Status Report	1, Zoom

✓ **Attached document(s):**

No attached documents.

✓ **Additional notes:**

None.

----- For instructor use only -----

Date:

Comments:

# Appendix C: Comments And Changes

Changes Table:

Comment	Change	Page
"Dependency is missing"	Task dependency	<a href="#">18</a>
“The use-case diagram is missing.”	Use-Case Diagrams	<a href="#">40/41</a>
“The use-case descriptions are not included.”	Use-Case Description	<a href="#">42</a>
“ the Class Diagram missed”	Class Diagram	<a href="#">91</a>

Table 89: Changes Table.

# Appendix D: Samples of Code

```

public boolean sendPasswordResetToken(String email, String token) throws SQLException {
    String query = "SELECT user_id FROM User WHERE email = ?";
    PreparedStatement stmt = connection.prepareStatement(string: query);
    stmt.setString(i: 1, string: email);
    ResultSet rs = stmt.executeQuery();

    if (rs.next()) {
        // If email exists, store the password reset token
        int userId = rs.getInt(string: "user_id");
        String updateQuery = "UPDATE User SET password_reset_token = ? WHERE user_id = ?";
        PreparedStatement updateStmt = connection.prepareStatement(string: updateQuery);
        updateStmt.setString(i: 1, string: token);
        updateStmt.setInt(i: 2, i1: userId);
        updateStmt.executeUpdate();
        return true; // Email exists and token is saved
    }
    return false; // Email doesn't exist
}

// Method to verify the reset token and reset the password
public boolean verifyTokenAndResetPassword(String token, String newPassword) throws SQLException {
    // Find user with the matching reset token
    String query = "SELECT user_id FROM User WHERE password_reset_token = ?";
    PreparedStatement stmt = connection.prepareStatement(string: query);
    stmt.setString(i: 1, string: token);
    ResultSet rs = stmt.executeQuery();

    if (rs.next()) {
        // If token is correct, reset the password
        int userId = rs.getInt(string: "user_id");
        String updatePasswordQuery = "UPDATE User SET password = ? WHERE user_id = ?";
        PreparedStatement updatePasswordStmt = connection.prepareStatement(string: updatePasswordQuery);
        updatePasswordStmt.setString(i: 1, string: newPassword);
        updatePasswordStmt.setInt(i: 2, i1: userId);
        updatePasswordStmt.executeUpdate();

        // Optionally, clear the token after successful password reset
        String clearTokenQuery = "UPDATE User SET password_reset_token = NULL WHERE user_id = ?";
        PreparedStatement clearTokenStmt = connection.prepareStatement(string: clearTokenQuery);
        clearTokenStmt.setInt(i: 1, i1: userId);
        clearTokenStmt.executeUpdate();
        return true; // Password reset successfully
    }
    return false; // Invalid token
}

```

Figure 56: Password Recovery Algorithm.

```
// Method to get the bank's URL from the database
private String getBankUrl(String bankName) throws SQLException {
    String query = "SELECT bank_url FROM Bank WHERE bank_name = ?";
    PreparedStatement stmt = dbConnector.connection.prepareStatement(query);
    stmt.setString(1, bankName);
    ResultSet rs = stmt.executeQuery();
    if (rs.next()) {
        return rs.getString("bank_url");
    }
    return null;
}

// Method to open the URL in the default browser
private void openBrowser(String url) {
    try {
        Desktop desktop = Desktop.getDesktop();
        URI uri = new URI(url);
        desktop.browse(uri);
    }
    catch (Exception e) {
        JOptionPane.showMessageDialog(parentComponent: this, "Error opening browser: " + e.getMessage(), title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
    }
}
```

*Figure 57: Linking To banks.*

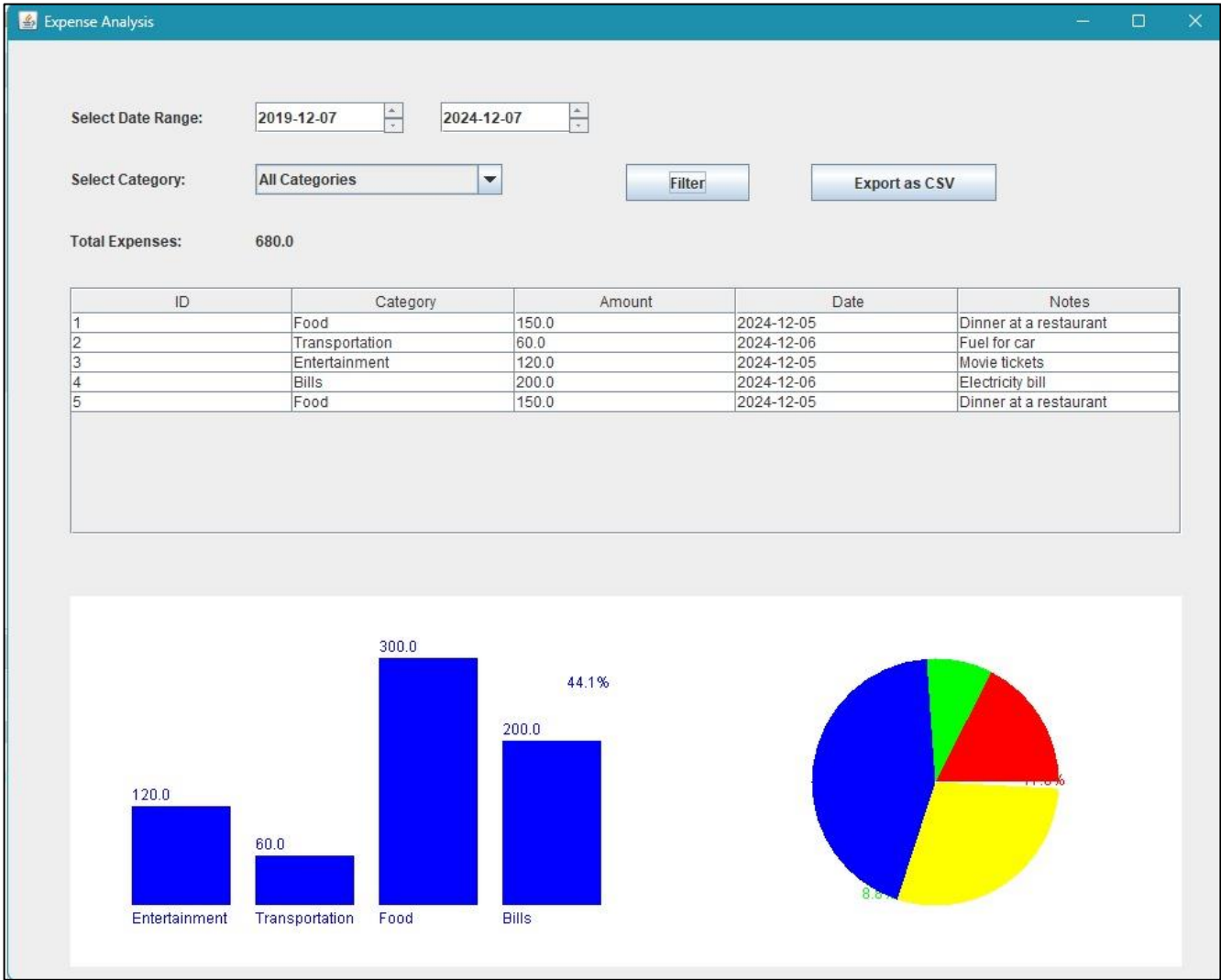


Figure 58: Expense Analysis.



Manage User Accounts

Manage User Accounts

User ID	Full Name	Email	Role
1	Ahmed Al-Fahad	ahmed.fahad@example.com	user
2	Fatimah Al-Saud	fatimah.saud@example.com	user
3	Sami Al-Mutairi	sami.mutairi@example.com	user
4	Rashid Al-Harbi	rashid.harbi@example.com	admin
5	abduLrahman Qahtani	abcd@gmail.com	user

Edit User

Delete User

Name:

Email:

Role:

user

Figure 59:Managing User Accounts.