Acknowledgement

We are profoundly grateful to all those who have contributed, both directly and indirectly, to the development of this project. Their influence has been instrumental in shaping our work and enhancing our experience throughout the course of our study.

First and foremost, we extend our heartfelt gratitude to Dr. Pintu Pal, our esteemed Principal, whose support and encouragement provided us with the invaluable opportunity to undertake this project at TECHNO INDIA HOOGHLY. Dr. Pal's visionary leadership and commitment to academic excellence have been a source of inspiration and guidance.

We are also deeply appreciative of Prof. Abhijit Ghosh, whose unwavering support, cooperation, and motivational guidance have significantly contributed to the success of this endeavour. Prof. Ghosh's constant presence, insightful advice, and encouragement have been pivotal in navigating the challenges and achieving our project goals.

Additionally, we would like to express our sincere thanks to the almighty for granting us the strength and perseverance to complete this project. Our heartfelt appreciation also goes to our parents, whose moral support, understanding, and sacrifices have been the cornerstone of our academic and personal growth.

We are equally grateful to our friends, who have been a constant source of support and camaraderie. Sharing our daily experiences with them and receiving their valuable suggestions have greatly enhanced the quality of our work.

To all who have played a role in this journey, we offer our deepest thanks and appreciation.

Abstract

TrackIt is a sophisticated and comprehensive financial management application crafted to streamline the tracking of both income and expenses for individuals and businesses alike. This cutting-edge tool is designed with user-friendliness in mind, offering an intuitive interface that simplifies the process of inputting and categorizing financial transactions. With **TrackIt**, users gain a clear and detailed overview of their financial health, facilitating more informed decision-making and enhanced financial stability.

TrackIt excels in providing real-time updates on account balances, ensuring that users always have the most current view of their financial standing. This feature is essential for maintaining an accurate and up-to-date understanding of one's finances. Alongside this, the app offers detailed reports on spending patterns. These reports delve into various aspects of financial behaviour, helping users gain insight into their expenditure habits and making it easier to identify areas where adjustments can be made.

TrackIt offers real-time balance updates and detailed reports on spending patterns, helping users understand their financial behaviour and make informed decisions. The app includes budget planning tools and customizable alerts to prevent overspending, while also managing daily, weekly, monthly, and yearly expenses with ease.

Users can categorize expenses, upload images of bills, add location details, and specify amounts. This data is securely stored, allowing users to view and sort expenses by different timeframes. The app also supports sharing bills and managing group expenses, and displays financial data through interactive graphs.

With features like income tracking, expense limits, and automatic calculations, **TrackIt** helps reduce manual work and provides a streamlined approach to financial management. By consolidating all financial information in one place, **TrackIt** empowers users to enhance their savings strategies and achieve financial stability efficiently.

Introduction

In this project we propose a website known as "**Trackit**" which will help to manage our income and expenses. It also acts as an indicator or reminder to remind the users of updating their expenses and paying their bills for the month. Due to some conflict or some other stress we sometimes forget about the payments and bills that we have to pay. This site will help you to make a note for what are the things we have to pay for by the end of month and/or year like gas bills, phone bill, electricity bill, taxes and some other personal expenses. In this fast-moving world this mobile application and website will be very useful for people who have a family and especially for business persons. Budgeting is an integral part of the society. Budget Tracking involves recording and analysing the incomes and expenses of a person or an organization over a particular period of time. Today, since we are living in a fast paced society, many people are looking forward to efficient ways to budget their time and money. During recent years, some research has been carried out on household budget. It has been noted that in most cases, budget management is being done mentally and never being put on paper which makes Budget Tracking very difficult. Tracking income and expenses is essential for managing personal finances or running a business. Here's an introduction to help you get started

Objective and Scope of the Project

Objective

The primary objective of the "TrackIt" project is to develop a comprehensive personal finance management tool. This tool aims to help users efficiently track their income, expenses, and budget, offering clear insights into their financial status. By doing so, the project seeks to empower users to make informed financial decisions, achieve their financial goals, and maintain overall financial health .

Scope

The scope of the "TrackIt" project encompasses several key features and functionalities:

1. **Income and Expense Tracking**:
   - **Detailed Recording**: Users can record all their income sources, such as salaries, freelance earnings, and investments, in a structured manner.
   - **Expense Categorization**: Users can categorize expenses into predefined categories like bills, groceries, entertainment, and more, allowing for better analysis and control.
2. **Budget Management**:
   - **Budget Creation**: Users can create monthly, quarterly, or annual budgets tailored to their financial goals and spending habits.
   - **Monitoring and Adjustment**: The tool provides real-time updates on budget performance, allowing users to adjust their budgets as needed to stay on track.
3. **Financial Reports**:
   - **Comprehensive Reports**: Users can generate detailed reports that summarize income, expenses, and budget performance over specific periods.
   - **Visual Representations**: Graphs and charts provide visual insights into financial data, making it easier to understand spending patterns and financial trends.
4. **Account Management**:
   - **Multiple Account Support**: Users can manage different types of accounts, such as savings, cash, and investment accounts, within the tool.
   - **Account Addition and Deletion**: Users can add new accounts or delete old ones as their financial situation changes.
5. **User-Friendly Interface**:
   - **Intuitive Design**: The interface is designed to be user-friendly, ensuring that users with varying levels of financial literacy can navigate and use the tool effectively.

Technological Background

HTML (HyperText Markup Language)

HTML provides the foundation for the "TrackIt" web application, structuring content and defining the layout. HTML elements are used to create forms for user input, organize financial data, and display charts and graphs. HTML5 semantic tags, like `<header>`, `<footer>`, `<section>`, and `<article>`, enhance accessibility and SEO, ensuring that the application is both userfriendly and search engine optimized.

CSS (Cascading Style Sheets)

CSS is used to style the HTML content, enhancing the visual appeal and ensuring a responsive design. With CSS3, advanced features like transitions, animations, and media queries are utilized to create a dynamic and adaptive user interface. The "TrackIt" project uses CSS to maintain a consistent look and feel across different devices, ensuring a seamless user experience.

JavaScript (JS)

JavaScript adds interactivity and dynamic behavior to the "TrackIt" application. It handles clientside scripting, enabling real-time updates, form validation, and dynamic content manipulation without requiring a page reload. JavaScript libraries and frameworks, such as jQuery or Chart.js, may be used to create interactive charts and graphs that provide visual insights into the user's financial data.

PHP (Hypertext Preprocessor)

PHP is the server-side scripting language used in "TrackIt" to manage the application's backend. It processes user inputs, interacts with the database, and generates dynamic content. PHP scripts handle tasks such as user authentication, financial data storage and retrieval, and budget calculations. By embedding PHP within HTML, the application can deliver personalized and dynamic web pages based on user interactions.

Git and GitHub

Version control is managed using Git, a distributed version control system that tracks changes in the source code. GitHub, a platform for hosting Git repositories, is used for collaborative development. It allows multiple developers to work on the "TrackIt" project simultaneously, manage changes, and resolve conflicts efficiently. GitHub also facilitates issue tracking, code reviews, and continuous integration, ensuring that the project maintains high quality and reliability.

Integration of Technologies

In "TrackIt", HTML structures the content, CSS styles it, JavaScript adds interactivity, and PHP handles server-side logic and database interactions. Git and GitHub ensure efficient version control and collaboration. This combination of technologies creates a robust, user-friendly web application for comprehensive financial tracking,budget planning and data visualization.

Problem Approach

Many individuals face challenges in managing their finances effectively due to a variety of factors. Here are some key issues that contribute to these difficulties:

1. **Lack of Basic Financial Literacy:** Many people do not have a strong foundation in financial literacy. This means they may not understand fundamental concepts such as budgeting, saving, investing, or the impact of interest rates on debt. Without this knowledge, it's challenging to make informed financial decisions, leading to mismanagement of funds. This lack of understanding can result in poor financial habits, making it difficult to achieve long-term financial stability and goals.

2. **Inadequate Tracking of Expenses:** Without proper tracking mechanisms, individuals can easily lose sight of where their money is going. This often leads to overspending, as people may not realize how much they are spending on non-essential items. Tools like expense tracking apps or maintaining a spending diary can help, but many people do not use these tools consistently.

3. **Difficulty in Creating and Sticking to a Budget:** Budgeting is a fundamental aspect of financial management, but many people struggle with it. Creating a realistic budget requires an understanding of income, expenses, and financial goals. Sticking to a budget requires discipline and often involves making difficult choices about spending. Without a budget, it's easy to spend more than one earns, leading to financial instability.

4. **Insufficient Savings for Emergencies and Future Goals:** A significant number of individuals do not save adequately for emergencies or future goals. Emergency savings are crucial to cover unexpected expenses such as medical emergencies or car repairs. Additionally, saving for long-term goals like buying a house, education, or retirement is essential for financial security. Many people, however, find it challenging to save regularly due to various financial pressures and priorities.

5. **Challenges in Managing and Paying Off Debts:** Debt management is another common challenge. Whether it's credit card debt, student loans, or mortgages, managing and paying off debts can be overwhelming without a clear plan. High-interest rates can quickly compound debt, making it even harder to pay off. Developing a debt repayment strategy, such as the snowball or avalanche method, can help, but it requires knowledge and discipline. Without a proper strategy, debt can spiral out of control, affecting credit scores and limiting future financial opportunities.

Solution Approach

The "TrackIt" project aims to address these challenges by providing a robust tool that offers:

1. **Comprehensive Tracking:** Users can track all their income sources and expenses in one place, allowing for a detailed overview of their financial status. This holistic tracking helps identify areas where spending can be optimized.

2. **Budget Planning:** The tool helps users create realistic budgets based on their financial goals and track their progress over time. It also provides alerts for overspending, ensuring users stay within their planned budgets.

3. **Visual Insights:** Graphs and charts provide clear visual representations of financial data, making it easier to understand spending patterns. These insights empower users to make informed decisions about their finances quickly.

4. **Multiple Accounts Management**: Users can manage different accounts, including savings, cash, and investments, to monitor their overall financial health. This feature allows for easy tracking and comparison of various financial assets in one unified platform.

5. **User-Friendly Design**: The intuitive interface ensures users can easily navigate and use the tool without extensive financial knowledge. It includes guided prompts and tips to help users make the most out of their financial planning journey.

Feasibility Study

Summary:

This comprehensive feasibility study assesses the viability and practicality of **‘Track It’**, an online finance tracking website. Our in-depth analysis evaluates technical, economic, operational, and social aspects to determine whether the project is worth pursuing. We examine the project's potential benefits, risks, and challenges to ensure a well-informed decision-making process.

Technical Feasibility:

Technical feasibility assesses a project's practicality from a technical perspective. It evaluates compatibility with existing systems, available tools, technical expertise, scalability, and performance. A technically feasible solution can be implemented, maintained, and supported within given constraints, ensuring a stable and efficient outcome, and minimizing technical risks and challenges.

- The feasibility to produce outputs in a given time.
- Response time under certain conditions.
- Ability to process a certain volume of the transaction at a particular speed.
- Facility to communicate data.

Economic Feasibility:

Economic feasibility assesses a project's financial practicality, evaluating costs versus benefits. It considers initial investment, operating costs, revenue, return on investment (ROI), and break-even point. A project is economically feasible if it generates sufficient financial returns to justify investment, ensuring a positive impact on the organization's financial health.

- Cost of Hardware and Software.
- Cost of Software to be acquired to build and run the product is a onetime cost.
- Database is the major part of hardware and software cost.
- Cost saving decision is needed.

Operational Feasibility:

Operational feasibility assesses a project's ability to be implemented and integrated into existing operations. It evaluates the impact on workflows, processes, and resources, considering factors such as staffing, training, and infrastructure. A project is operationally feasible if it can be successfully executed and sustained within the organization's existing framework.

- If there is sufficient support for the management from the user?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

Social Feasibility:

Social feasibility assesses a project's acceptance and impact on stakeholders, including users, customers, and communities. It evaluates factors such as social norms, cultural values, and potential resistance to change. A project is socially feasible if it aligns with stakeholder needs and values, ensuring adoption and minimizing negative social consequences.

- **Team Dynamics**: Establish clear roles, respect, and open communication to foster a positive, productive, and inclusive team environment.

- **Shared Goals**: Unite under a common objective, motivating each other to ensure collective success and a sense of accomplishment.

- **Learning Environment**: Encourage experimentation, feedback, and growth, promoting continuous learning, improvement, and innovation.

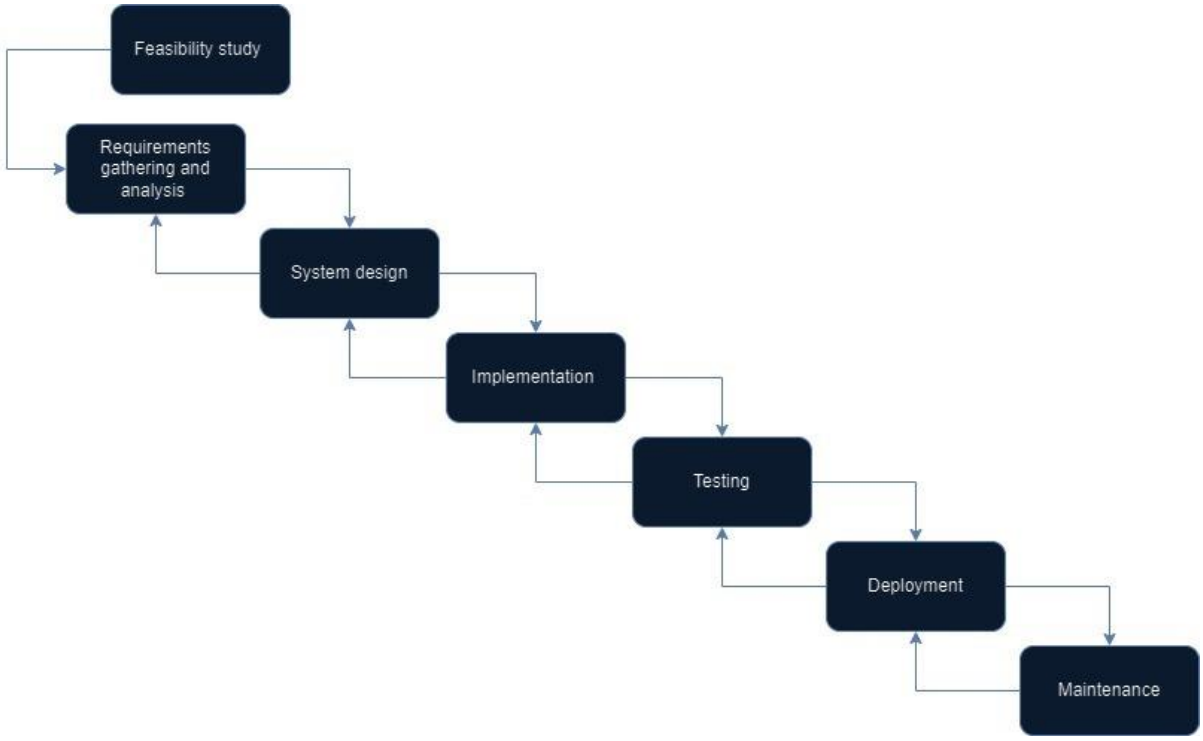
- **Social Impact**: Assess the project's potential social impact, ensuring it aligns with societal values, needs, and expectations.

Software Development Life Cycle (SDLC)

The project will follow the Iterative Waterfall Model, a variant of the traditional Waterfall model that incorporates iterative cycles to allow for refinement and improvements based on feedback and evolving requirements. This model is structured into sequential phases, with each phase providing a distinct focus while allowing for iterative development and testing. Here's how the iterative waterfall model will be applied:

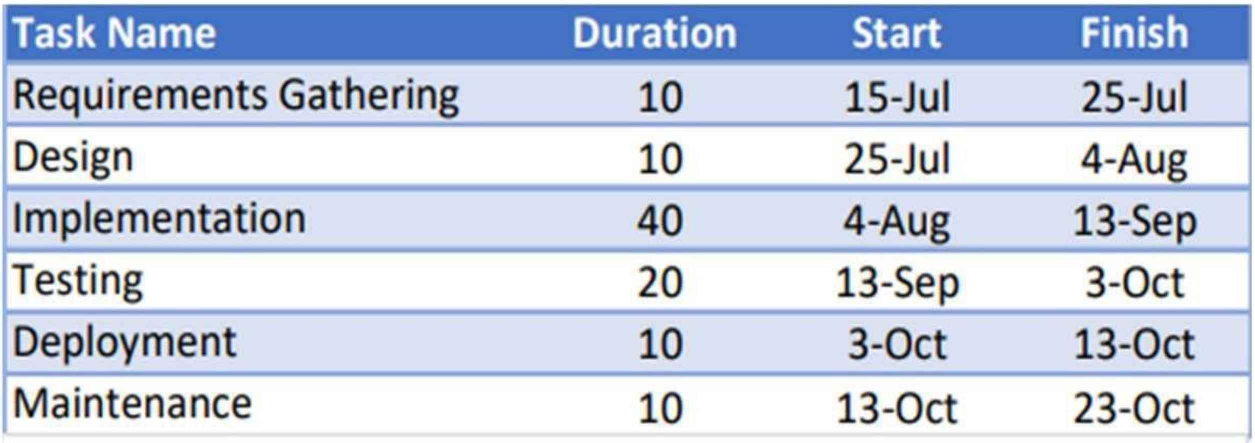
1. **Feasibility Study**
    - **Objective:** Evaluate the feasibility of the project in terms of technical, financial, and operational aspects before proceeding to the development phase.
    - **Iteration Aspect:** Feasibility findings may be revisited and updated throughout the project to accommodate any changes in scope or constraints.
2. **Requirements Gathering and Analysis**
    - **Objective**: Collect and analyse all the requirements for the project. This involves understanding user needs, business requirements, and any constraints.
    - **Iteration Aspect**: Requirements may be revisited and refined through iterations to ensure accuracy and completeness.
3. **System Design**
    - **Objective**: Develop the system architecture and design based on the gathered requirements. This includes both high-level and detailed design.
    - **Iteration Aspect**: Design may be iteratively updated based on feedback and new insights gained during the development phase.
4. **Implementation**
    - **Objective**: Code the system according to the design specifications. This phase involves developing the software components and integrating them.
    - **Iteration Aspect**: Iterative cycles involve implementing features, then reviewing and revising them based on testing and feedback.
5. **Testing**
    - **Objective**: Test the system to ensure that it meets the specified requirements and is free of defects. This phase includes unit testing, integration testing, and system testing.
    - **Iteration Aspect**: Testing is performed iteratively, with each iteration focusing on different aspects of the system. Feedback from testing leads to revisions and improvements.
6. **Deployment**
    - **Objective**: Deploy the system to the production environment, making it available to end-users.
    - **Iteration Aspect**: Deployment may occur in stages, with iterative releases to gather user feedback and make incremental improvements.
7. **Maintenance**
    - **Objective**: Provide ongoing support and maintenance to address issues, perform updates, and ensure the system continues to operate effectively.
    - **Iteration Aspect**: Maintenance includes iterative updates and enhancements based on user feedback and changing requirements.

The iterative nature of this model allows for flexibility and responsiveness to changes, making it suitable for projects where requirements may evolve or become clearer over time. Each iteration provides an opportunity to refine the project, ensuring better alignment with user needs and expectations.

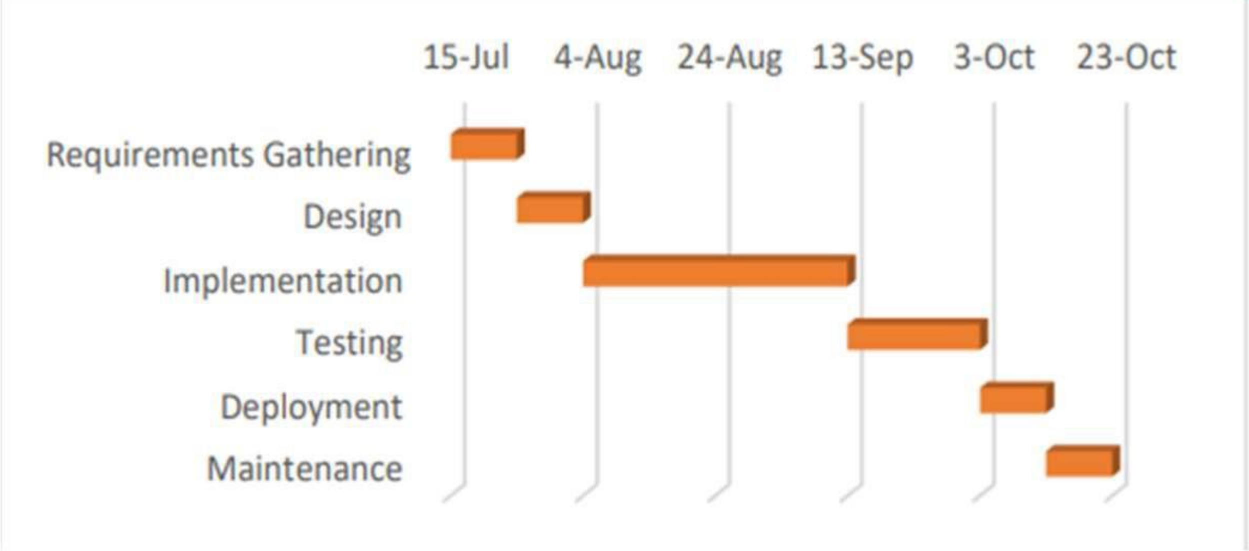


Gantt Chart

A Gantt chart is a visual project management tool that provides a clear overview of a project's timeline. It represents tasks as horizontal bars, where the length of each bar corresponds to the task's duration and its position indicates the start and end dates. Gantt charts excel at displaying task dependencies, milestones, and resource allocation. This visual representation makes it easy to identify potential bottlenecks, track progress, and make adjustments to the project schedule. By providing a clear picture of the project timeline, Gantt charts facilitate effective communication among team members and stakeholders

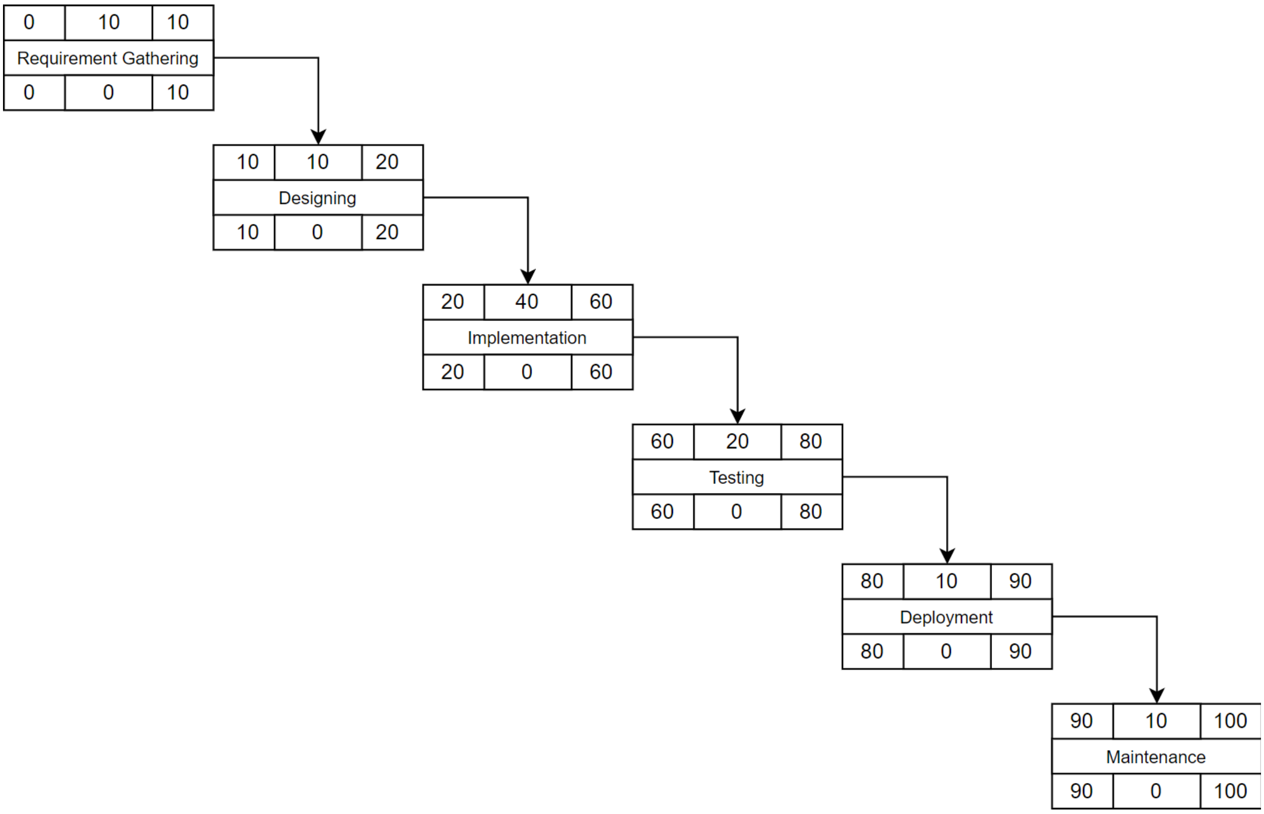


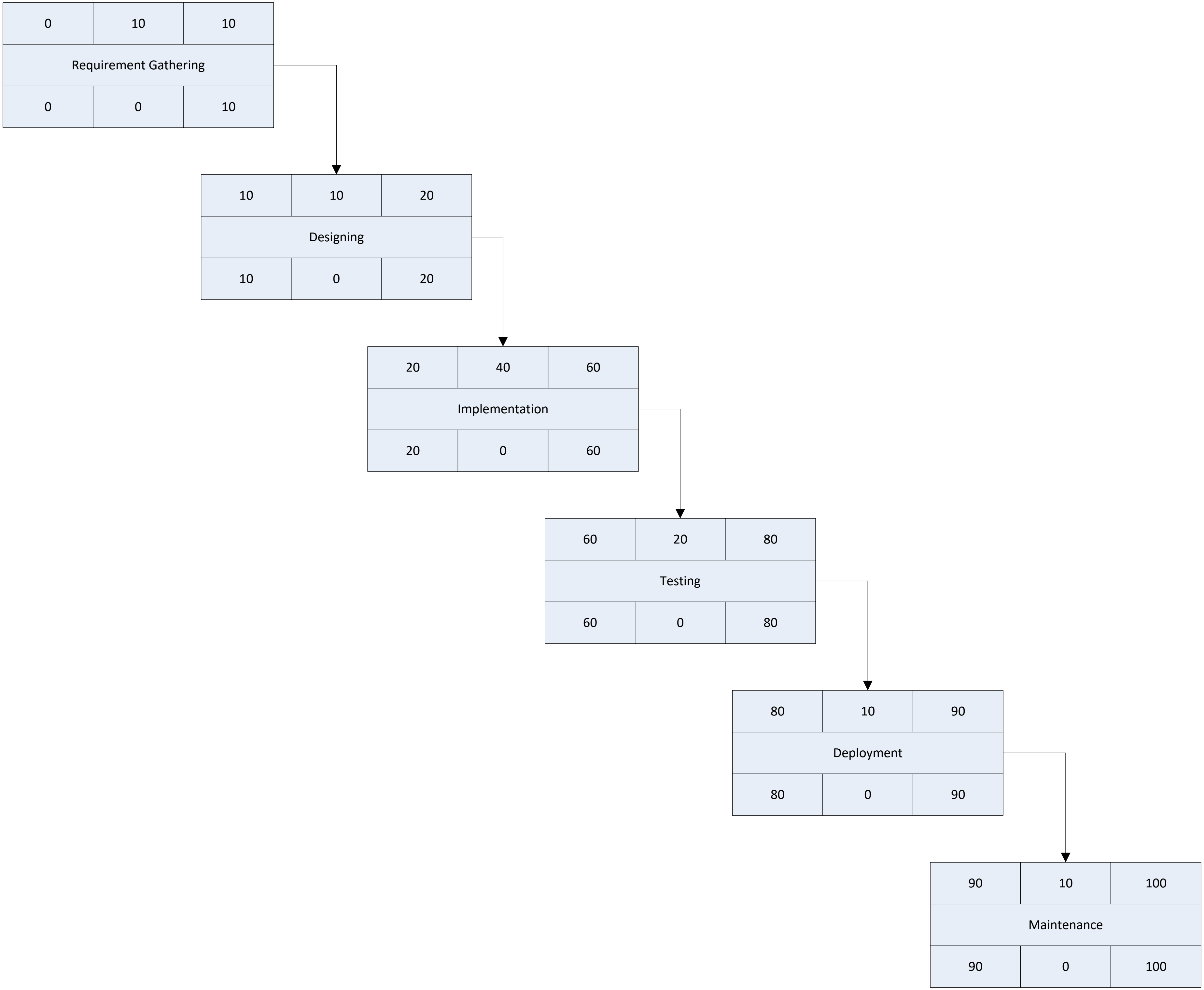
Task Name	Duration	Start	Finish
Requirements Gathering	10	15-Jul	25-Jul
Design	10	25-Jul	4-Aug
Implementation	40	4-Aug	13-Sep
Testing	20	13-Sep	3-Oct
Deployment	10	3-Oct	13-Oct
Maintenance	10	13-Oct	23-Oct



Pert Chart

A PERT (Program Evaluation and Review Technique) chart is a network diagram used to analyze and represent the tasks involved in completing a project. It focuses on identifying the critical path, which is the sequence of tasks that directly impact the project's overall duration. PERT charts use nodes to represent events or milestones and arrows to indicate tasks and their dependencies. By visualizing task relationships, PERT charts help project managers identify potential delays, assess the impact of uncertainties, and calculate the probability of project completion within a specific timeframe. This information is invaluable for making informed decisions, allocating resources efficiently, and managing project risks




0 10 10
Requirement Gathering
0 0 10
10 10 20
Designing
10 0 20
20 40 60
Implementation
20 0 60
60 20 80
Testing
60 0 80
80 10 90
Deployment
80 0 90
90 10 100
Maintenance
90 0 100

Software Requirements Specification (SRS)

Purpose

The purpose of this document is to describe the software requirements for the “TrackIt” finance tracking website. This website is designed to help users manage their income and expenses, provide reminders for bill payments, and offer insights into their financial status through detailed reports and visualizations. The intended audience includes end-users, developers, and project managers.

Scope

The scope of the "TrackIt" project encompasses the development of a personal finance management tool that offers the following functionalities:

- Income and Expense Tracking
- Budget Management
- Financial Reports
- Account Management
- User-Friendly Interface

Product Perspective

TrackIt is a standalone web application designed to help users manage their personal finances. It integrates functionalities such as income and expense tracking, budget management, financial reporting, and account management to provide a comprehensive personal finance management solution.

Product Functions

- **Income and Expense Tracking:** Users can record and categorize their income and expenses.
- **Budget Management:** Users can create, monitor, and adjust budgets.
- **Financial Reports:** Users can generate detailed financial reports with visual representations.
- **Account Management:** Users can manage multiple accounts and their details.
- **User Authentication:** Users can register, log in, and manage their accounts securely.

Functional Requirements

1. **User Authentication and Authorization**
    - Users must be able to register and log in to the website.
    - User details must be encrypted during registration.
    - There will be two types of users: Admin and User.
2. **Income and Expense Tracking**

- Users can record all their income sources (e.g., salaries, freelance earnings, investments).
- Users can categorize expenses (e.g., bills, groceries, entertainment).

3. **Budget Management**
   - Users can create monthly, quarterly, or annual budgets.
   - The system provides real-time updates on budget performance.
   - Users can adjust their budgets as needed.
4. **Financial Reports**
   - Users can generate detailed reports summarizing income, expenses, and budget performance.
   - The system provides visual representations (graphs and charts) of financial data.
5. **Account Management**
   - Users can manage multiple accounts (e.g., savings, cash, investment accounts).
   - Users can add new accounts or delete old ones.

Non-Functional Requirements

1. **Usability**
   - The interface must be intuitive and easy to navigate.
   - The design should cater to users with varying levels of financial literacy.
2. **Security**
   - All user data must be encrypted.
   - The system must adhere to data protection regulations.

Software and Hardware Requirements

1. **Software Requirements**
   - **Operating System:** Windows, macOS, or Linux for development**.**
   - **Web Server:** Apache**.**
   - **Database:** MySQL**.**
   - **Programming Language:** HTML, CSS, JavaScript, PHP.
   - **Browser Compatibility:** Latest versions of Chrome, Firefox, Safari, and Edge.
   - **Development Tools:** Visual Studio Code or any preferred IDE, Git for version control.
2. **Hardware Requirements**
   - **Processor:** Intel Core i3 or higher.
   - **RAM:** 4GB or higher.
   - **Storage:** 128GB or higher.

Cost analysis is done by COCOMO Model

- Taking software project as organic type as team size is adequately small, the problem is well understood and has been solved in the past and also team members have nominal experience regarding the problem.
- a, b, c and d are constants for organic type system and the corresponding values are shown below:

Values			
a	**b**	**c**	**d**
2.4	1.05	2.5	0.38

- KLOC- Kilo Lines of Codes is the estimated size of the software product.
- EAF - Effort Adjustment Factor. The factors and the corresponding values for developing this project are shown below, considering the values are high for organic type system below:

Factors	Values
Software reliability(f1)	1.0
Application Database(f2)	1.0
Product complexity(f3)	1.0
Runtime Performance(f4)	1.0
Memory Constrints(f5)	1.0
Volatility of Virtual Machine(f6)	1.0
Turnaround time(f7)	1.0
Analyst capability(f8)	1.0
Application experience(f9)	1.13
S/w Engineer capability(f10)	1.0
Virtual machine experience(f11)	1.0
Programming language experience(f12)	1.07
Application of s/w engineering methods(f13)	1.0
Use of software tools(f14)	1.0
Required development schedule(f15)	1.0

COCOMO 1

Here LOC = 2000

Therefore KLOC = 2000/1000 = 2

For Organic:

Effort:

$a \times (\text{KLOC})^{b}$ PM

$= 2.4 \times 2^{1.05}$ PM

= 4.969271 PM

Development Time:

$c \times (\text{Effort})^{d}$ Months

$= 2.5 \times (4.969271)^{0.38}$ Months

= 4.5976 Months

≈ 5 Months

<u>COCOMO 2:</u>

Effort Adjustment Factor (EAF):

= f1×f2×f3×f4×f5×f6×f7×f8×f9×f10×f11×f12×f13×f14×f15

= 1.0×1.0×1.0×1.0×1.0×1.0×1.0×1.0×1.13×1.0×1.0×1.07×1.0×1.0×1.0

= 1.2091

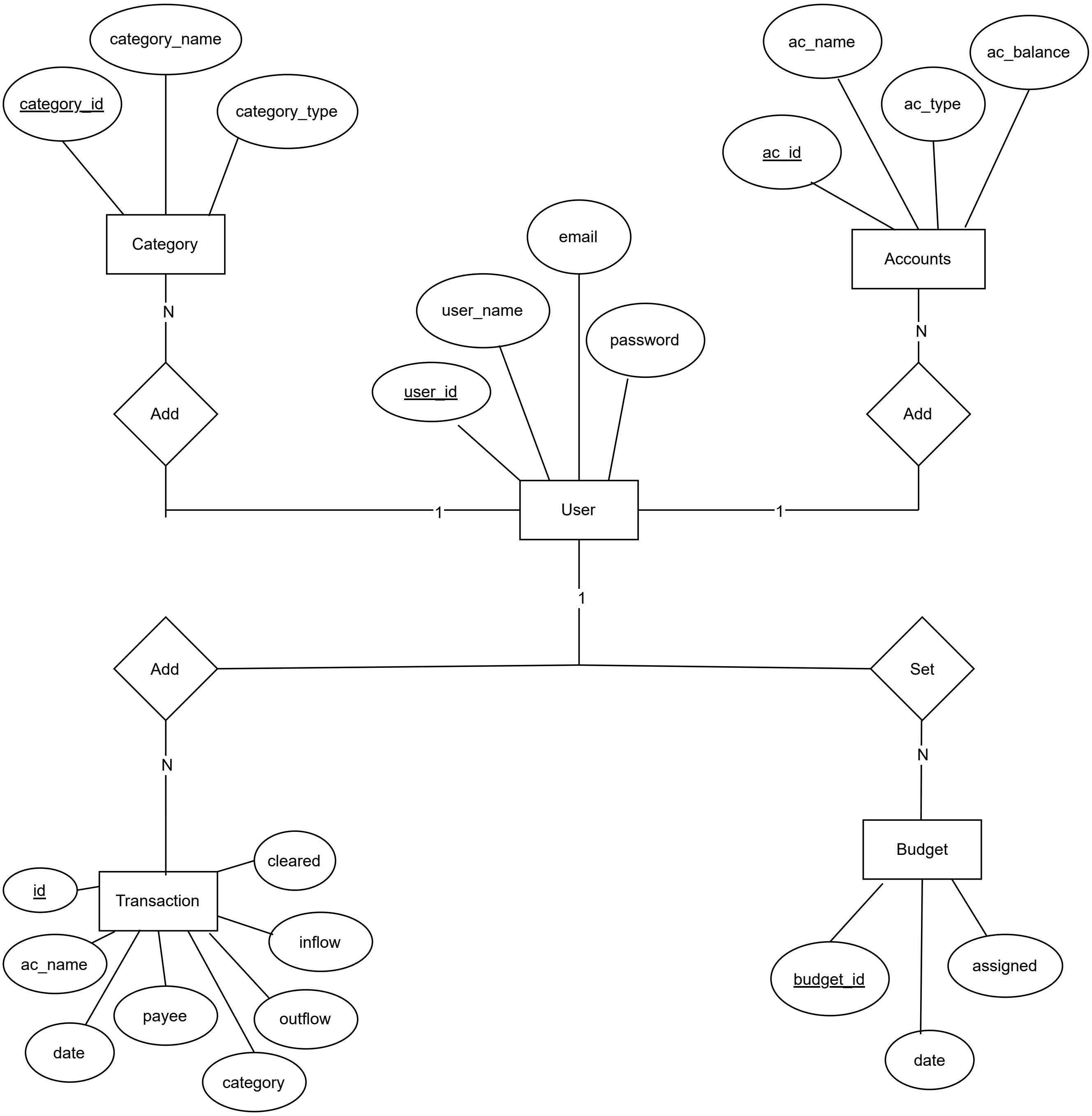
Effort:

$= 3.2 \times (\text{KLOC})^{1.05} \times \text{EAF}$ PM

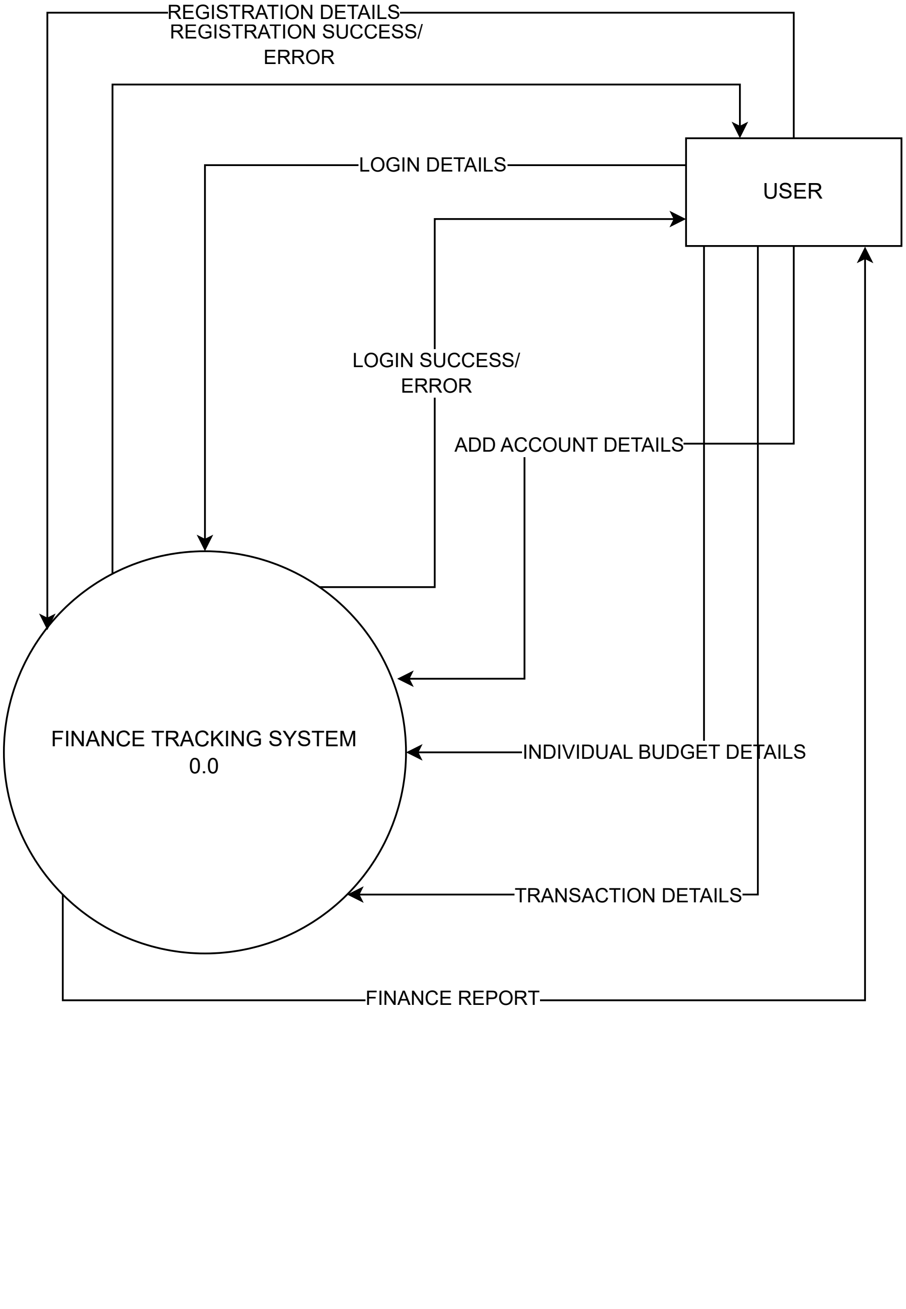
$= 3.2 \times (2)^{1.05} \times 1.2091$ PM

= 8.0111 PM

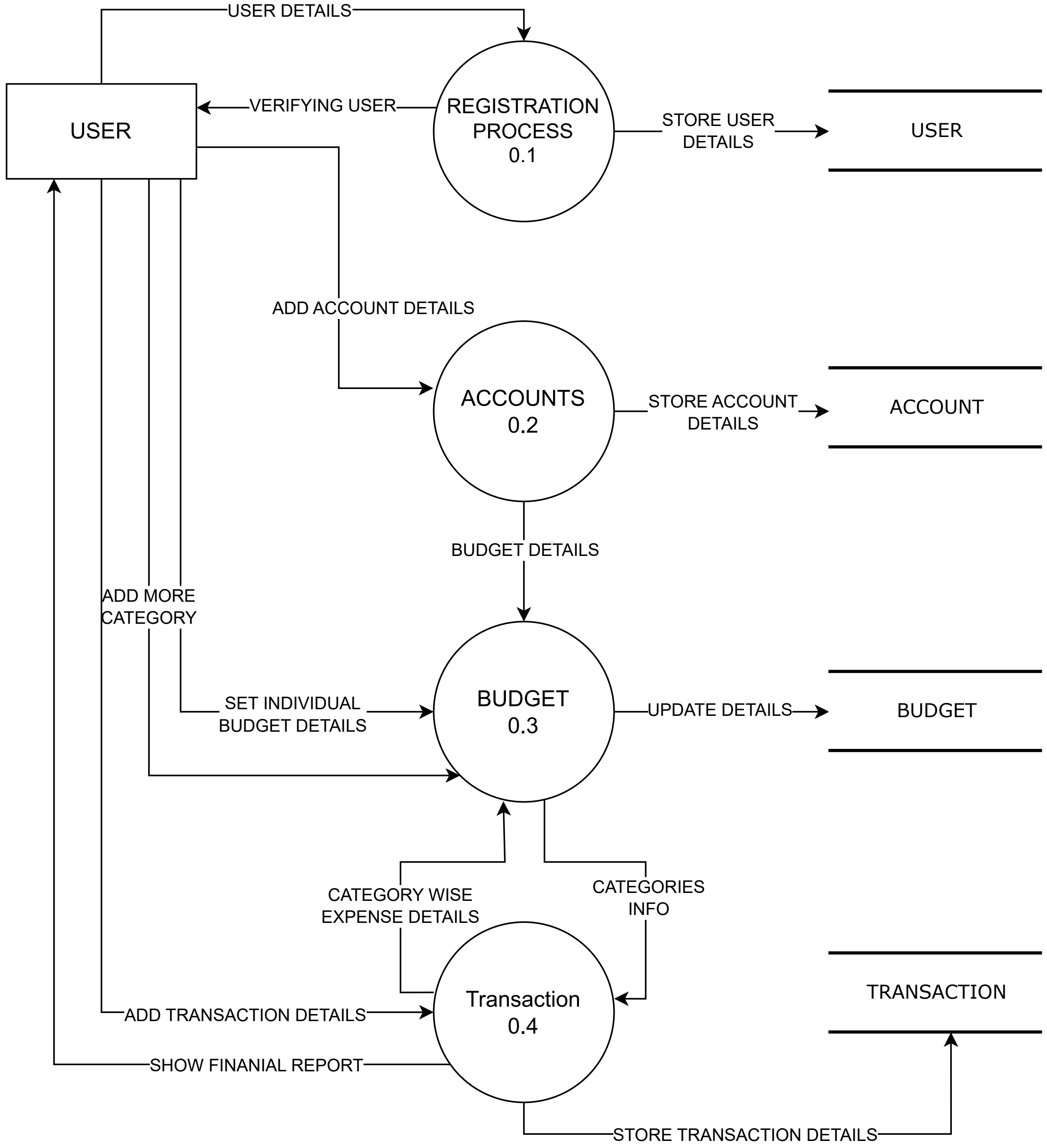
ER Digram


category_name
category_id
category_type
Category
N
Add
ac_name
ac_balance
ac_type
ac_id
Accounts
N
Add
email
user_name
password
user_id
User
1
1
1
Add
Set
N
N
Transaction
cleared
id
inflow
ac_name
outflow
payee
date
category
Budget
budget_id
assigned
date

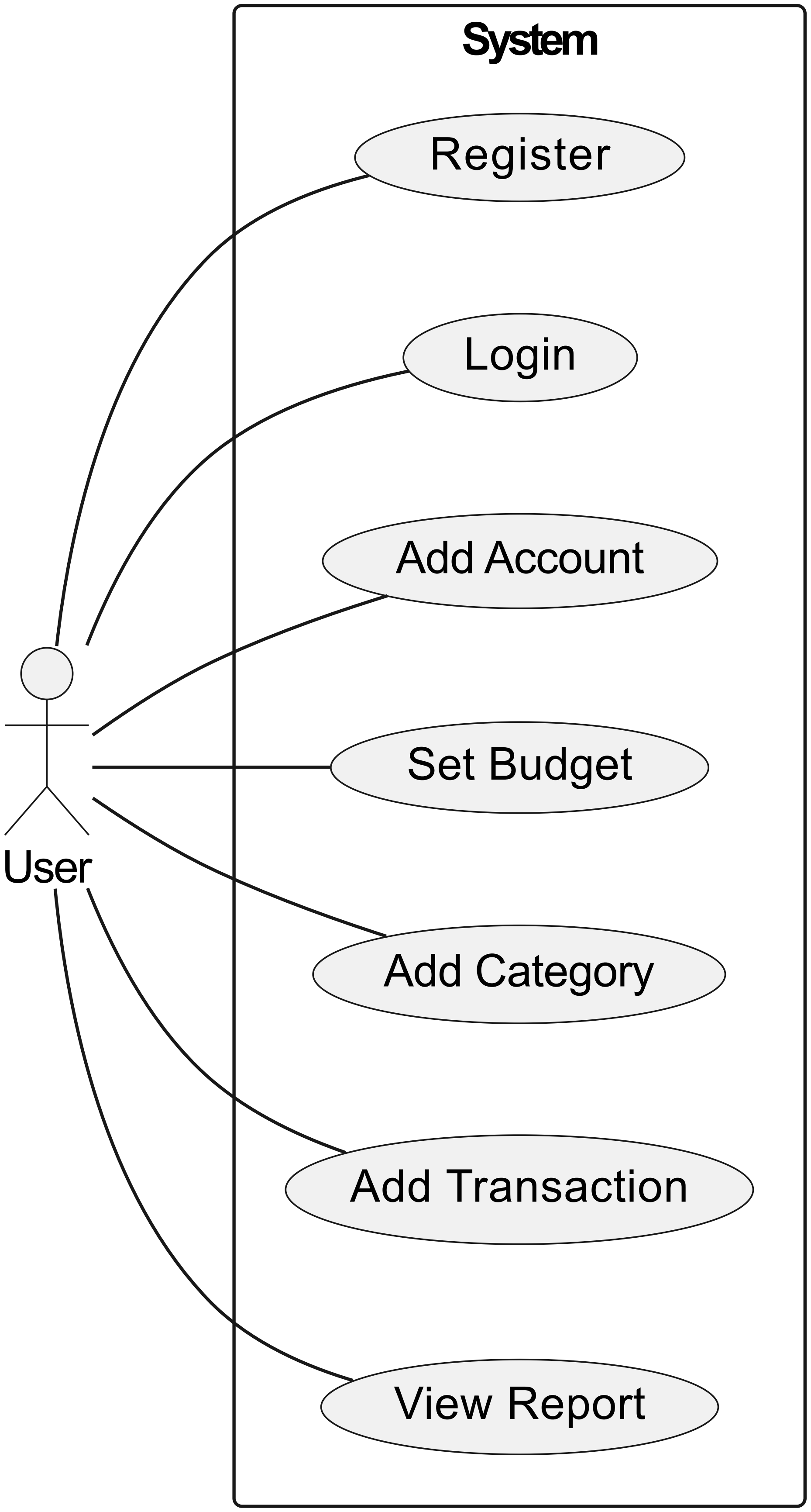
Level 0 DFD


REGISTRATION DETAILS
REGISTRATION SUCCESS/ ERROR
LOGIN DETAILS
USER
LOGIN SUCCESS/ ERROR
ADD ACCOUNT DETAILS
FINANCE TRACKING SYSTEM 0.0
INDIVIDUAL BUDGET DETAILS
TRANSACTION DETAILS
FINANCE REPORT

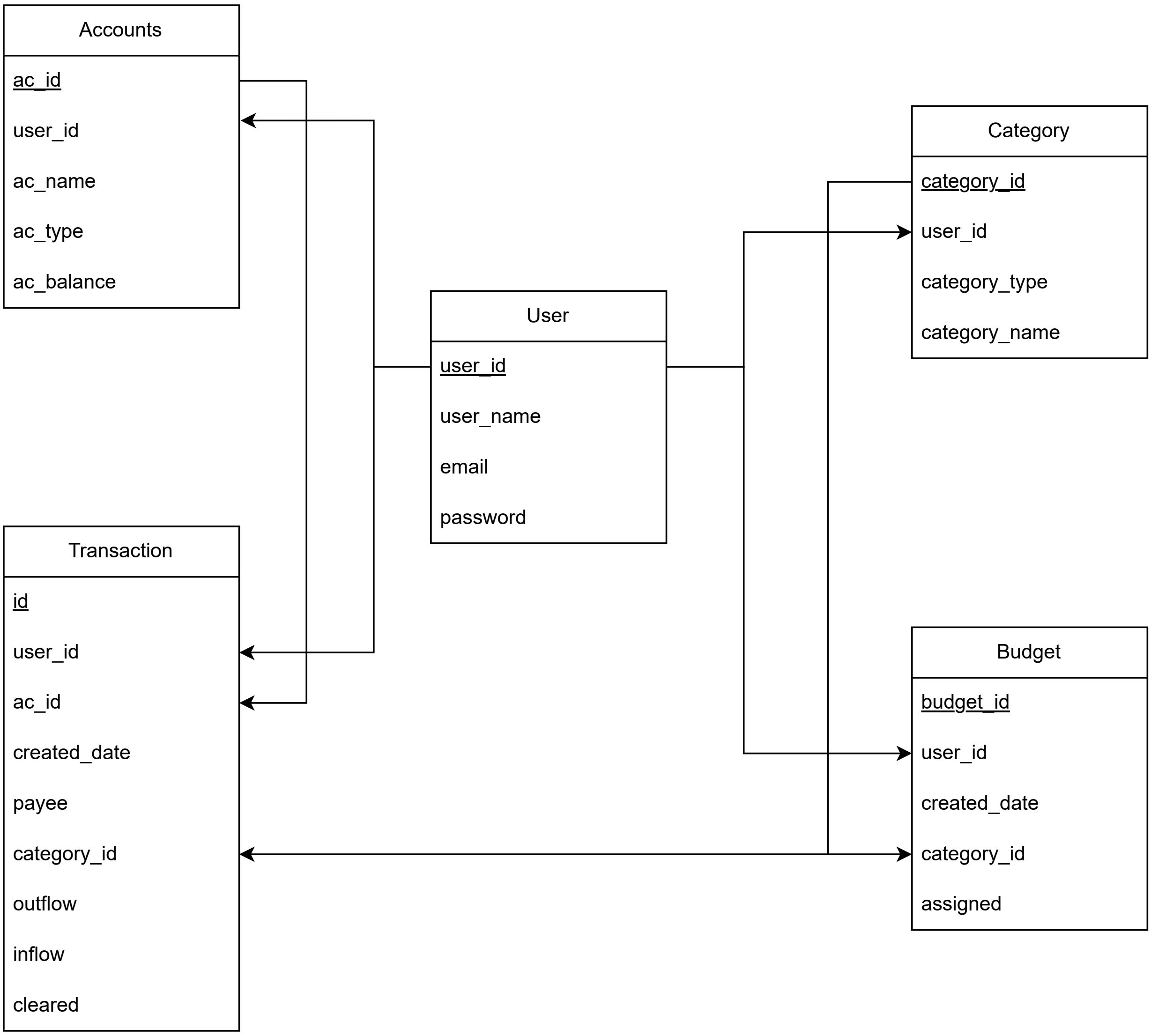
Level 1 DFD


USER DETAILS
USER
VERIFYING USER
REGISTRATION PROCESS 0.1
STORE USER DETAILS
USER
ADD ACCOUNT DETAILS
ACCOUNTS 0.2
STORE ACCOUNT DETAILS
ACCOUNT
BUDGET DETAILS
ADD MORE CATEGORY
SET INDIVIDUAL BUDGET DETAILS
BUDGET 0.3
UPDATE DETAILS
BUDGET
CATEGORY WISE EXPENSE DETAILS
CATEGORIES INFO
Transaction 0.4
TRANSACTION
ADD TRANSACTION DETAILS
SHOW FINANIAL REPORT
STORE TRANSACTION DETAILS

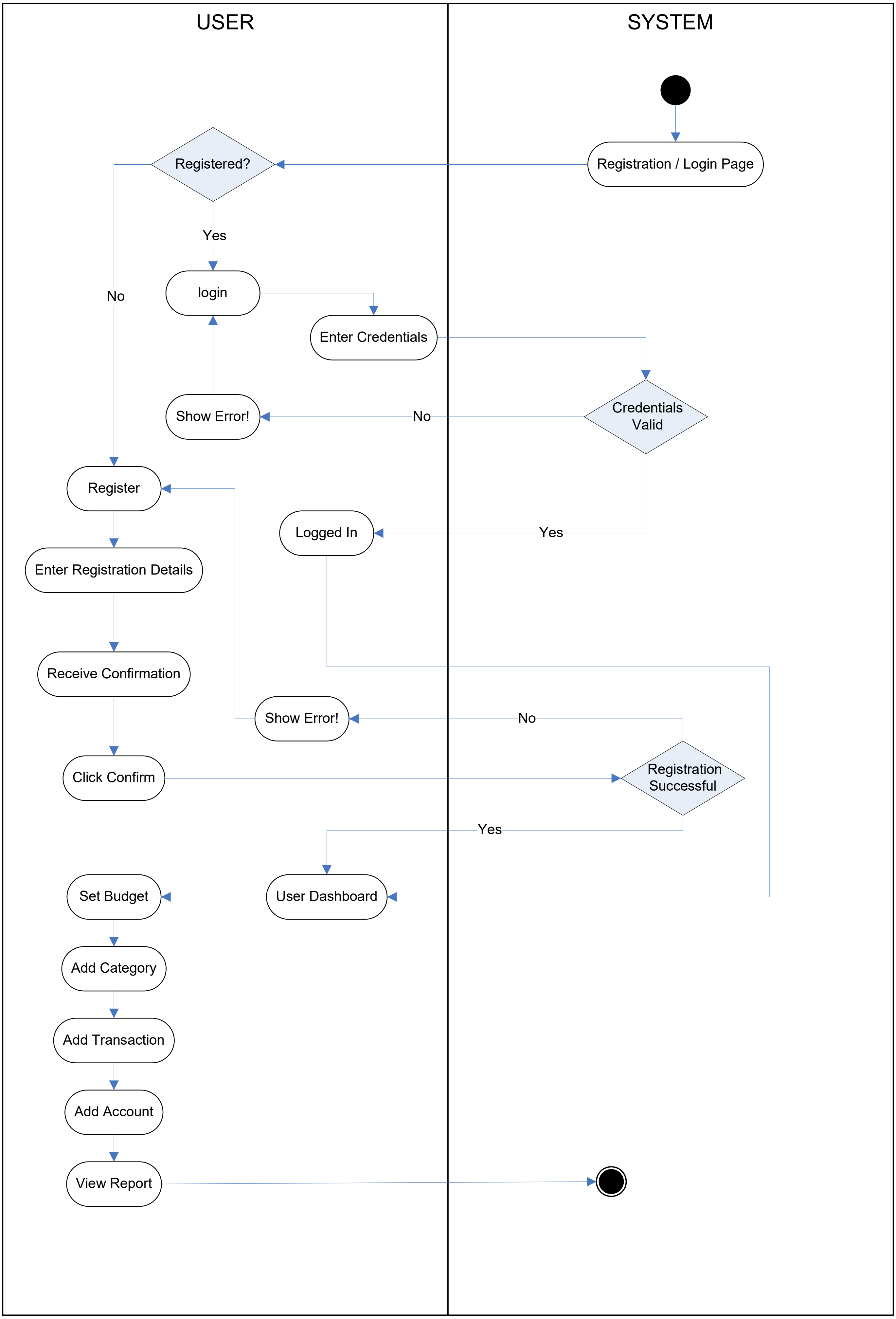
Use Case Diagram



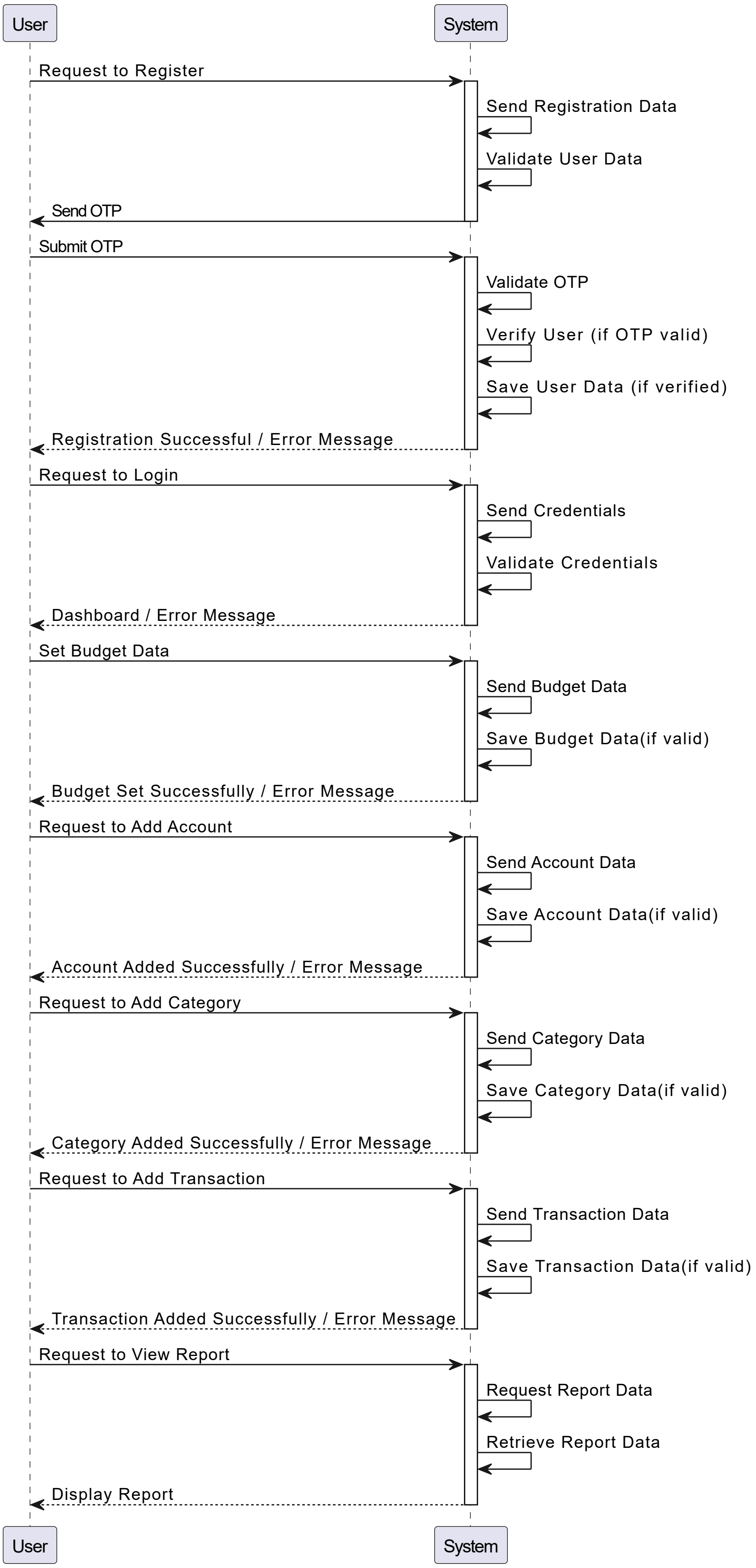
Database Schema

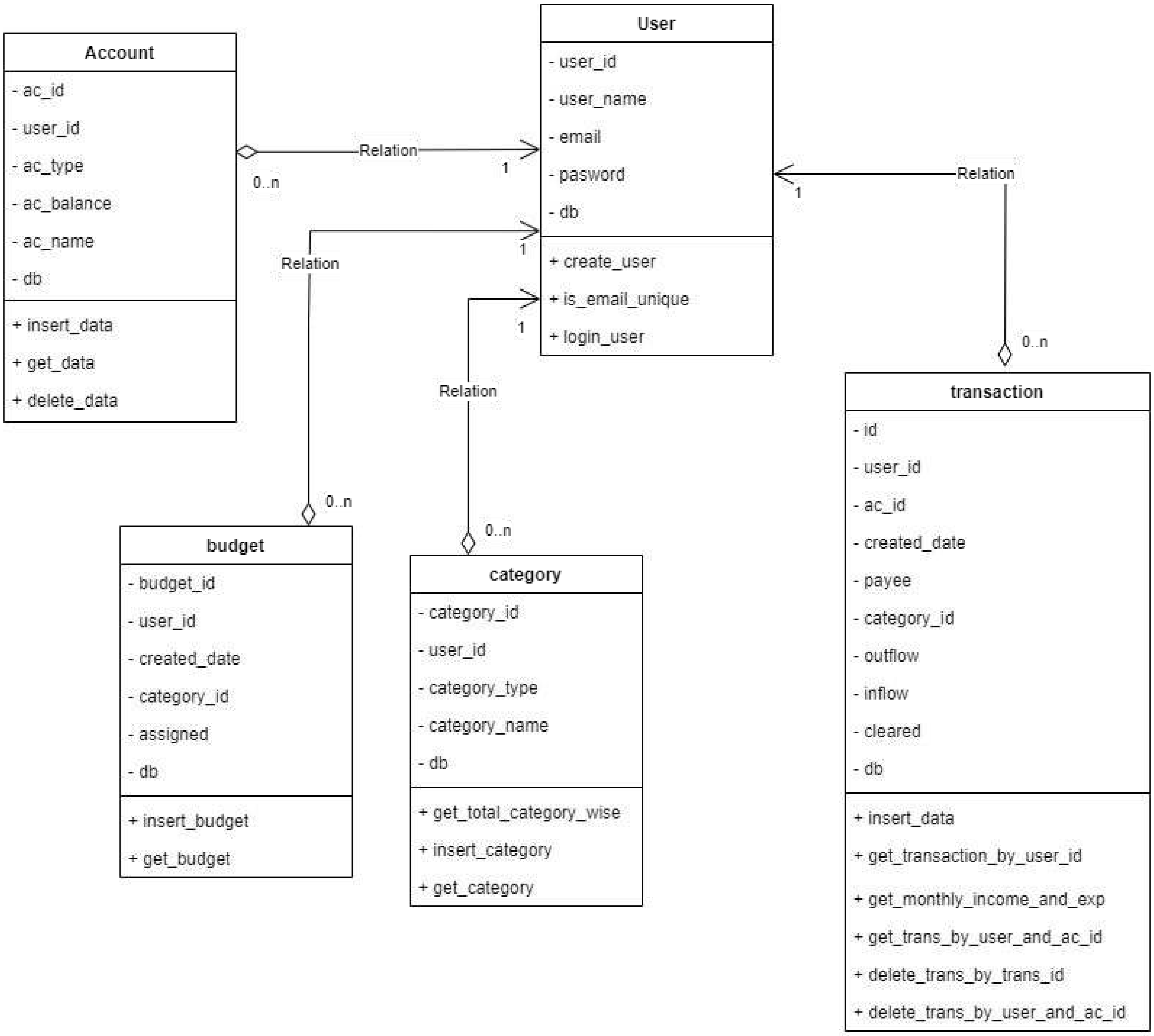


Activity Diagram



Sequence Diagram


User
System
Request to Register
Send Registration Data
Validate User Data
Send OTP
Submit OTP
Validate OTP
Verify User (if OTP valid)
Save User Data (if verified)
Registration Successful / Error Message
Request to Login
Send Credentials
Validate Credentials
Dashboard / Error Message
Set Budget Data
Send Budget Data
Save Budget Data(if valid)
Budget Set Successfully / Error Message
Request to Add Account
Send Account Data
Save Account Data(if valid)
Account Added Successfully / Error Message
Request to Add Category
Send Category Data
Save Category Data(if valid)
Category Added Successfully / Error Message
Request to Add Transaction
Send Transaction Data
Save Transaction Data(if valid)
Transaction Added Successfully / Error Message
Request to View Report
Request Report Data
Retrieve Report Data
Display Report
User
System


Account
- ac_id
- user_id
- ac_type
- ac_balance
- ac_name
- db
+ insert_data
+ get_data
+ delete_data
User
- user_id
- user_name
- email
- pasword
- db
+ create_user
+ is_email_unique
+ login_user
budget
- budget_id
- user_id
- created_date
- category_id
- assigned
- db
+ insert_budget
+ get_budget
category
- category_id
- user_id
- category_type
- category_name
- db
+ get_total_category_wise
+ insert_category
+ get_category
transaction
- id
- user_id
- ac_id
- created_date
- payee
- category_id
- outflow
- inflow
- cleared
- db
+ insert_data
+ get_transaction_by_user_id
+ get_monthly_income_and_exp
+ get_trans_by_user_and_ac_id
+ delete_trans_by_trans_id
+ delete_trans_by_user_and_ac_id
Relation
0..n
1
Relation
1
0..n
Relation
1
0..n
Relation
1
0..n

Testing & Analysis

Unit Testing

Unit testing focuses on verifying individual components or modules of the software. Each function or method is tested in isolation to ensure that it works as expected. Unit testing is typically done early in the development process and often automated using tools like JUnit (for Java), PHP Unit (for PHP), or Jest (for JavaScript).

Integration Testing

Integration testing validates the interaction between different modules or components of the software. After the individual units have been tested, integration testing ensures that these components work together as expected. It helps identify issues related to the interaction, such as data flow problems or interface mismatches, ensuring the system's cohesion.

System Testing

System testing evaluates the complete and fully integrated system to verify that it meets the specified requirements. It tests the entire system in a real-world environment, focusing on end-to-end functionality, performance, security, and usability. System testing includes various subtypes, such as performance and security testing.

Acceptance Testing

Also known as User Acceptance Testing (UAT), acceptance testing is conducted to ensure that the system satisfies business requirements and is ready for deployment. This is typically done by the end users or stakeholders to validate that the software behaves as expected and meets their needs. It’s the final testing phase before release and can include both alpha and beta testing.

Recovery Testing

Recovery testing assesses the system’s ability to recover from failures, such as crashes, hardware malfunctions, or network failures. It checks how well the system can restore itself after encountering an issue and how quickly it returns to normal operation. This type of testing ensures that the software is resilient and fault-tolerant.

Functional Testing

Functional testing focuses on verifying that the software's functionalities operate according to the specified requirements. This type of testing checks whether each feature of the system performs as expected, covering input/output, user commands, data manipulation, and interactions with external systems. It ensures that the system functions in line with user expectations.

Hardware/Software Testing

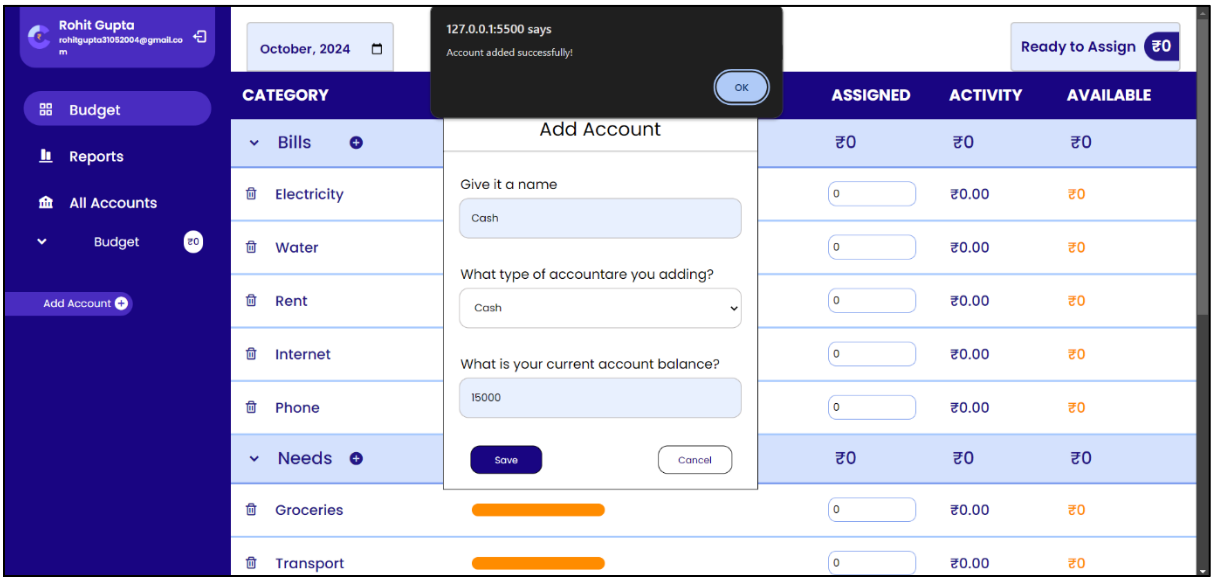
Hardware/software testing ensures that the software is compatible with the hardware on which it runs. It tests the interactions between hardware components and software systems, verifying that they work together without errors. This type of testing is crucial for systems with specific hardware dependencies, such as embedded systems, IoT devices, or specialized hardware setups.

Security Testing

Security testing identifies vulnerabilities, risks, and potential threats in the system. It ensures that the application is protected from attacks like SQL injections, cross-site scripting (XSS), and unauthorized access. Security testing includes penetration testing, risk assessments, and testing for data encryption, authentication, and access controls to prevent breaches.

Test Case 5: Add New Account

Test Name	Unit Testing
Test Description	Verify that a user can successfully add a new account.
Objective	Ensure that users can add accounts to manage finances.
Input	Enter account name, account type and initial amount and press save.
Expected Result	The account is added successfully, and a confirmation message is displayed.
Output	Passed



127.0.0.1:5500 says

Account added successfully!

OK

Add Account

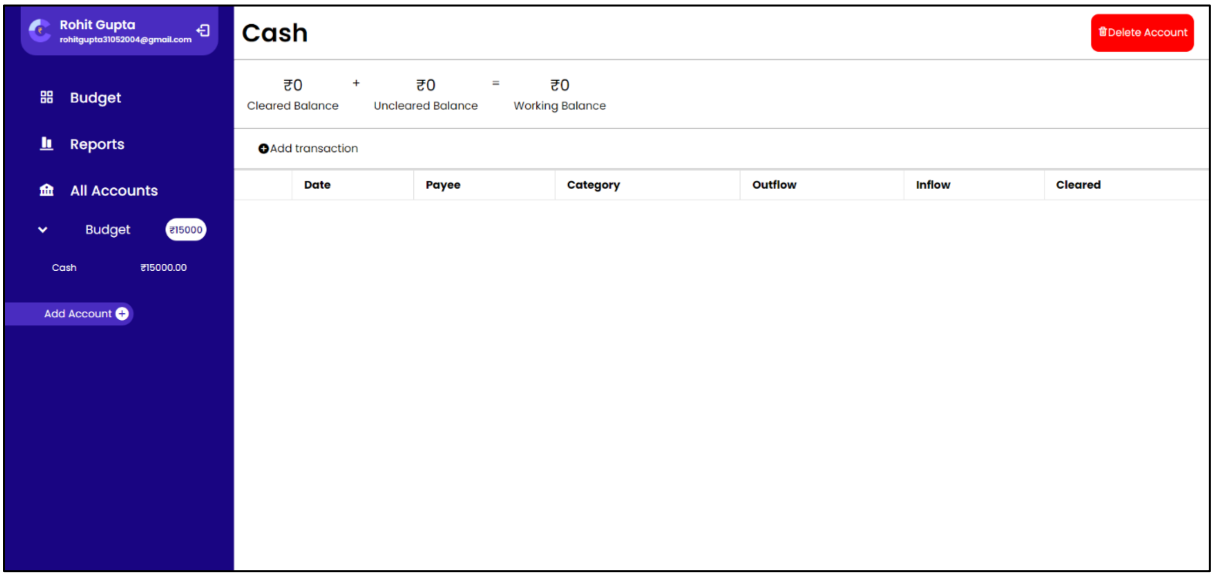
Give it a name: Cash

What type of accountare you adding?: Cash

What is your current account balance?: 15000

Save Cancel

CATEGORY	ASSIGNED	ACTIVITY	AVAILABLE
Bills	₹0	₹0	₹0
Electricity	0	₹0.00	₹0
Water	0	₹0.00	₹0
Rent	0	₹0.00	₹0
Internet	0	₹0.00	₹0
Phone	0	₹0.00	₹0
Needs	₹0	₹0	₹0
Groceries	0	₹0.00	₹0
Transport	0	₹0.00	₹0



Cash

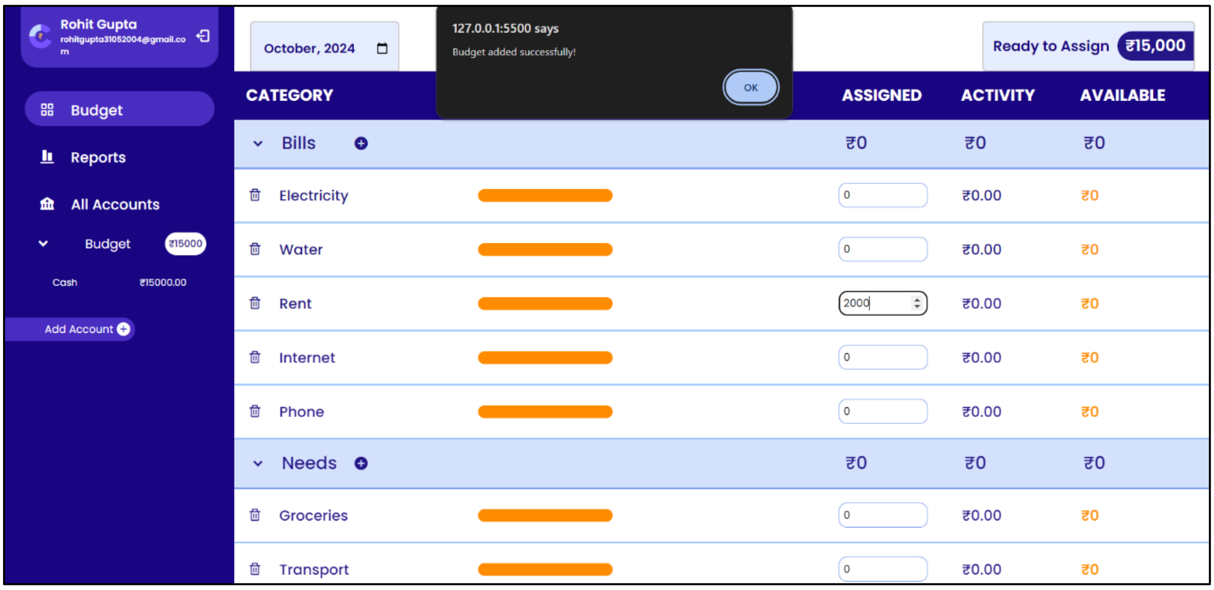
₹0 Cleared Balance + ₹0 Uncleared Balance = ₹0 Working Balance

Add transaction

Date	Payee	Category	Outflow	Inflow	Cleared

Test Case 6: Add Budget

Test Name	Unit Testing
Test Description	Verify that a user can successfully set a budget for a specific category.
Objective	Ensure that users can add and set budgets for their expense categories.
Input	Select the budget category and enter the budget amount.
Expected Result	The budget is saved successfully, and a confirmation message is displayed. The budget should now be visible in the budget summary.
Output	Passed



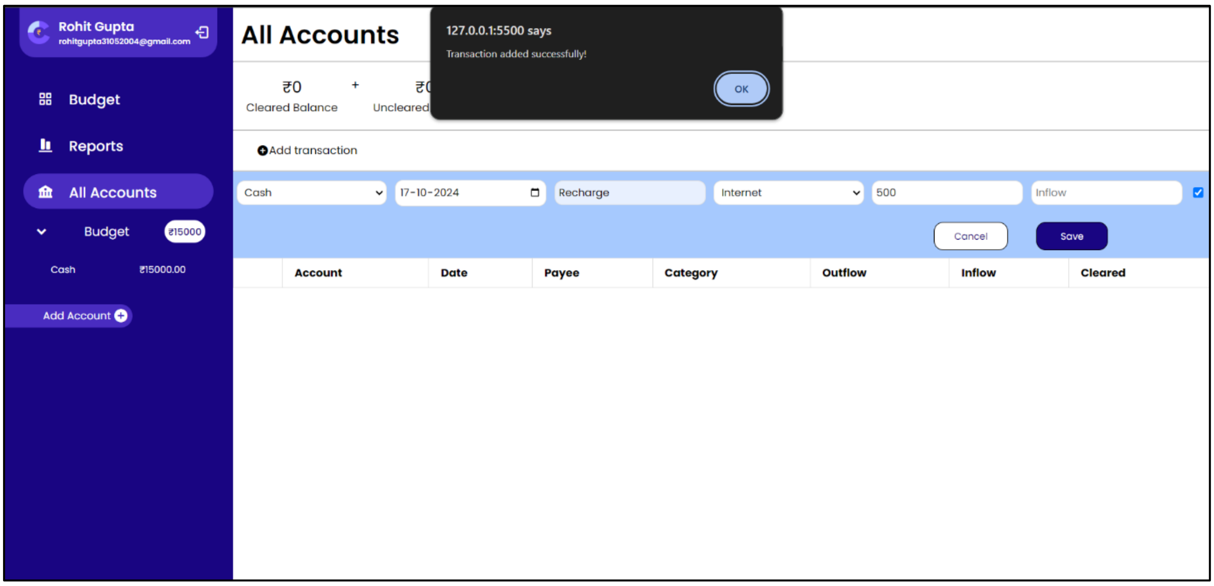
127.0.0.1:5500 says

Budget added successfully!

CATEGORY	ASSIGNED	ACTIVITY	AVAILABLE
Bills	₹0	₹0	₹0
Electricity	0	₹0.00	₹0
Water	0	₹0.00	₹0
Rent	2000	₹0.00	₹0
Internet	0	₹0.00	₹0
Phone	0	₹0.00	₹0
Needs	₹0	₹0	₹0
Groceries	0	₹0.00	₹0
Transport	0	₹0.00	₹0

Test Case 7: Add Transaction

Test Name	Unit Testing
Test Description	Verify that a user can successfully add a transaction.
Objective	Ensure that users can add income and expense transactions accurately.
Input	Select account and enter transaction details. i.e., date, amount, type (inflow/outflow), date, category, cleared status.
Expected Result	The transaction is saved successfully, and the balance updates accordingly
Output	Passed



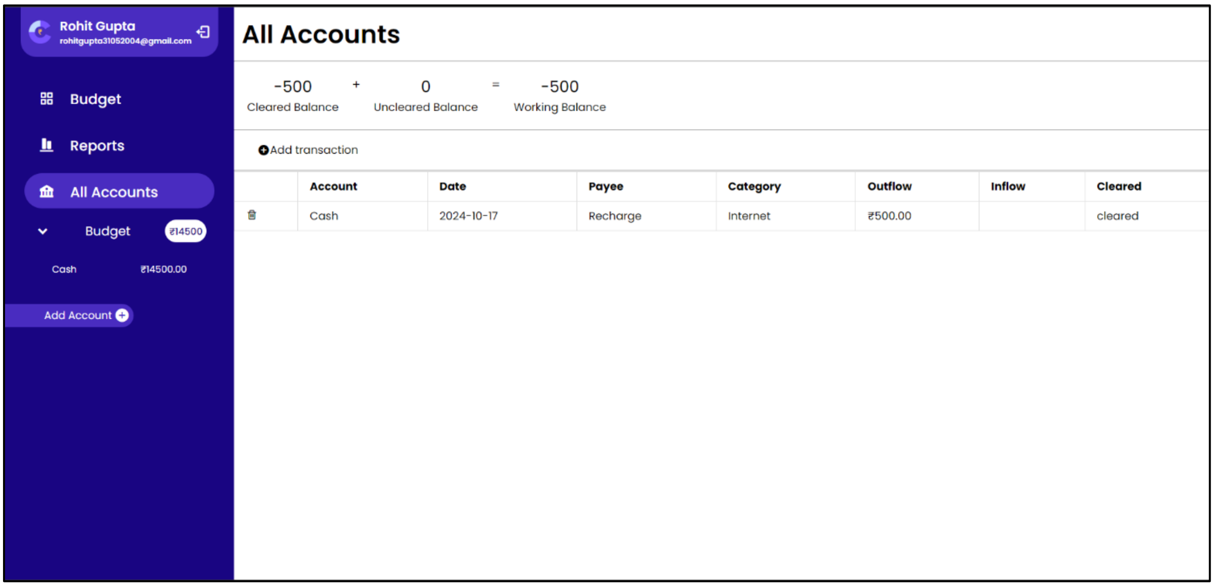
127.0.0.1:5500 says

Transaction added successfully!

₹0 Cleared Balance + ₹0 Uncleared

Cash	17-10-2024	Recharge	Internet	500	Inflow	☑

	Account	Date	Payee	Category	Outflow	Inflow	Cleared

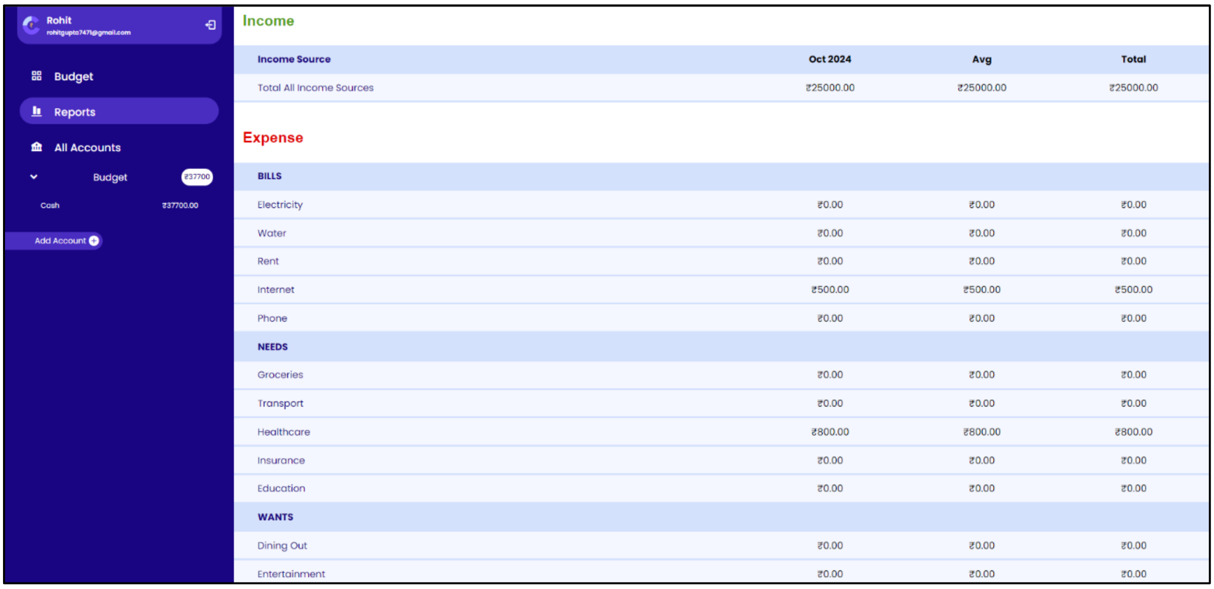


-500 Cleared Balance + 0 Uncleared Balance = -500 Working Balance

	Account	Date	Payee	Category	Outflow	Inflow	Cleared
🗑	Cash	2024-10-17	Recharge	Internet	₹500.00		cleared

Test Case 8: View Category-wise Spending and Report

Test Name	Unit Testing
Test Description	Verify that the user can view category-wise spending data and generate monthly financial report.
Objective	Ensure that users can view their spending based on different categories and monthly report.
Input	Select a date range.
Expected Result	Category-wise spending data and monthly report is displayed in a chart and table format respectively.
Output	Passed



Rohit
rohitgupta7471@gmail.com

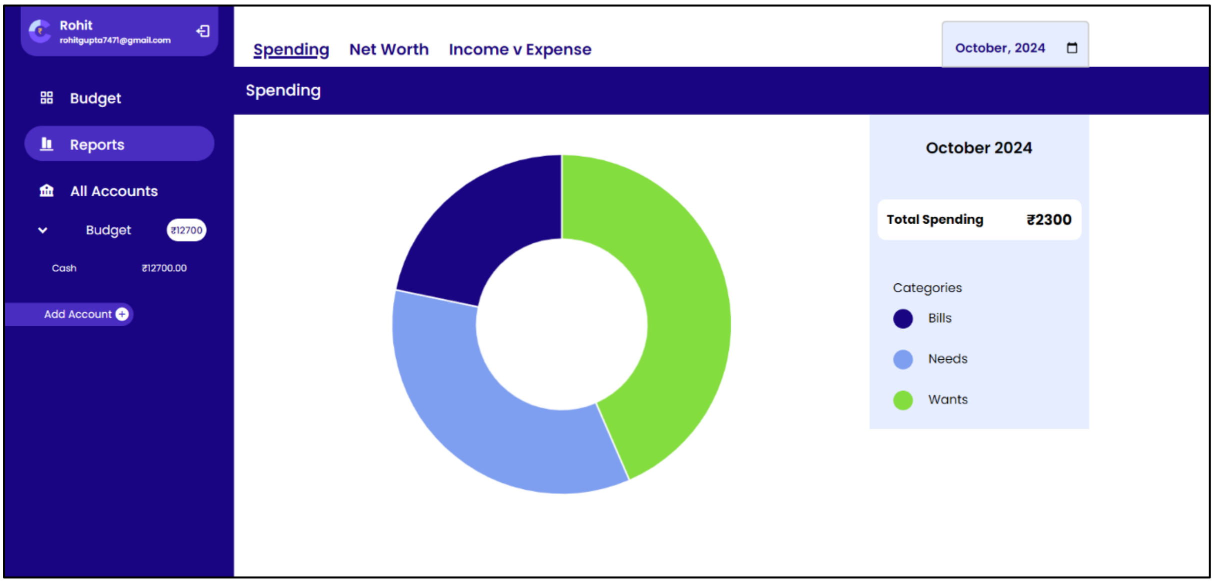
Budget
Reports
All Accounts
Budget ₹37700
Cash ₹37700.00
Add Account

Income

Income Source	Oct 2024	Avg	Total
Total All Income Sources	₹25000.00	₹25000.00	₹25000.00

Expense

BILLS			
Electricity	₹0.00	₹0.00	₹0.00
Water	₹0.00	₹0.00	₹0.00
Rent	₹0.00	₹0.00	₹0.00
Internet	₹500.00	₹500.00	₹500.00
Phone	₹0.00	₹0.00	₹0.00
NEEDS			
Groceries	₹0.00	₹0.00	₹0.00
Transport	₹0.00	₹0.00	₹0.00
Healthcare	₹800.00	₹800.00	₹800.00
Insurance	₹0.00	₹0.00	₹0.00
Education	₹0.00	₹0.00	₹0.00
WANTS			
Dining Out	₹0.00	₹0.00	₹0.00
Entertainment	₹0.00	₹0.00	₹0.00



Rohit
rohitgupta7471@gmail.com

Budget
Reports
All Accounts
Budget ₹12700
Cash ₹12700.00
Add Account

Spending Net Worth Income v Expense

October, 2024

Spending

October 2024

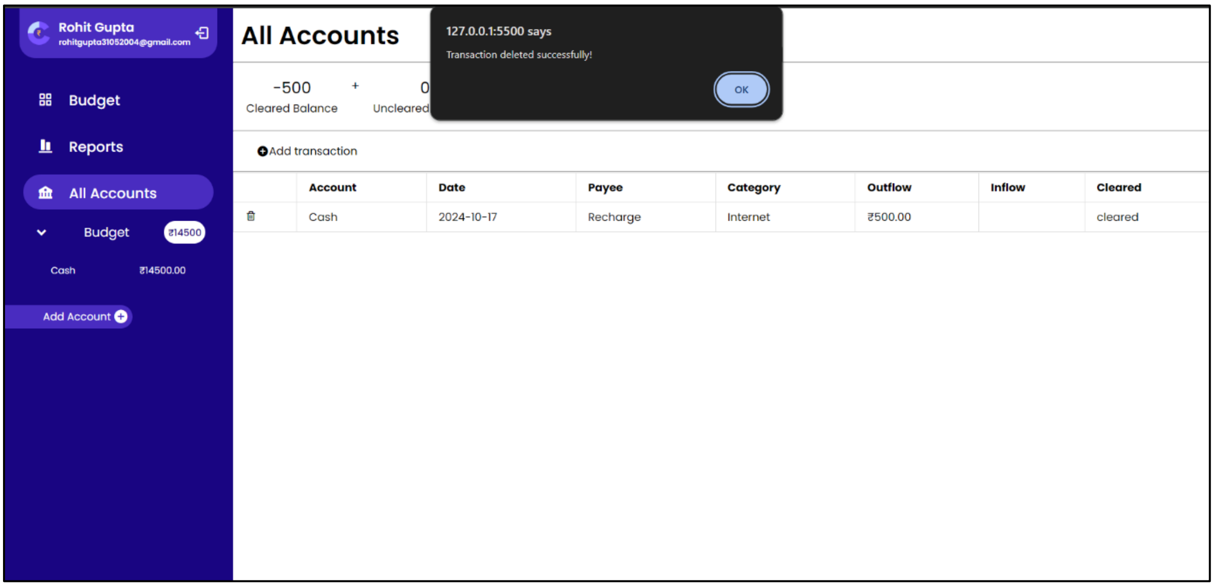
Total Spending ₹2300

Categories

● Bills
● Needs
● Wants

Test Case 8: Delete Transaction

Test Name	Unit Testing
Test Description	Verify that a user can delete a transaction.
Objective	Ensure that users can remove transactions.
Input	Click the delete button next to the transaction you want to delete.
Expected Result	The transaction is deleted successfully, and the balance updates accordingly.
Output	Passed



All Accounts

127.0.0.1:5500 says

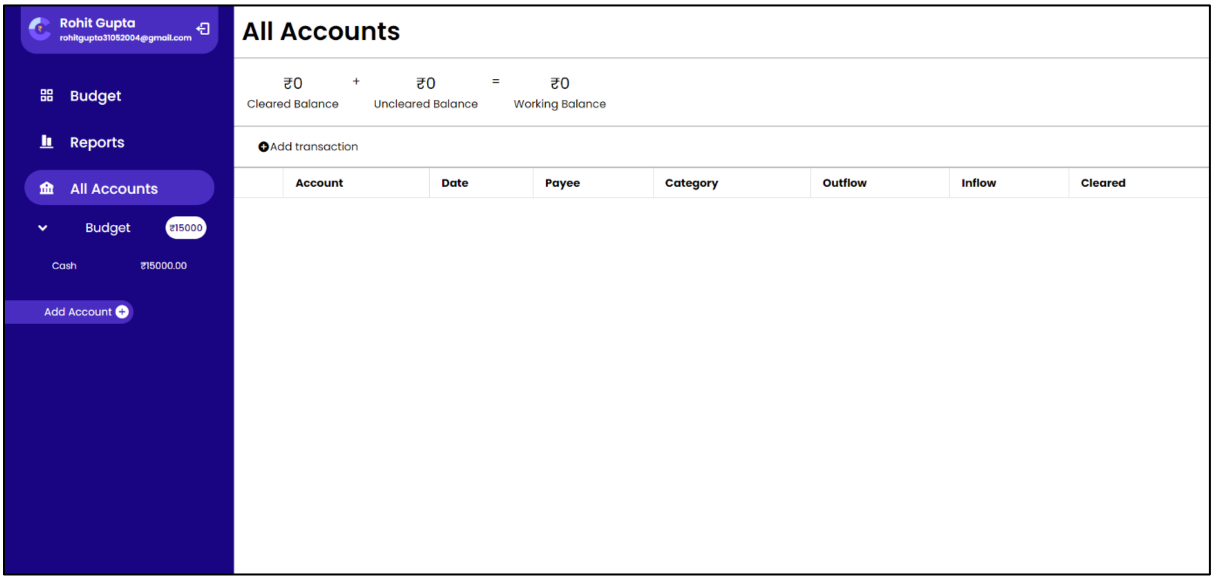
Transaction deleted successfully!

OK

-500 Cleared Balance + 0 Uncleared

Add transaction

	Account	Date	Payee	Category	Outflow	Inflow	Cleared
🗑	Cash	2024-10-17	Recharge	Internet	₹500.00		cleared



All Accounts

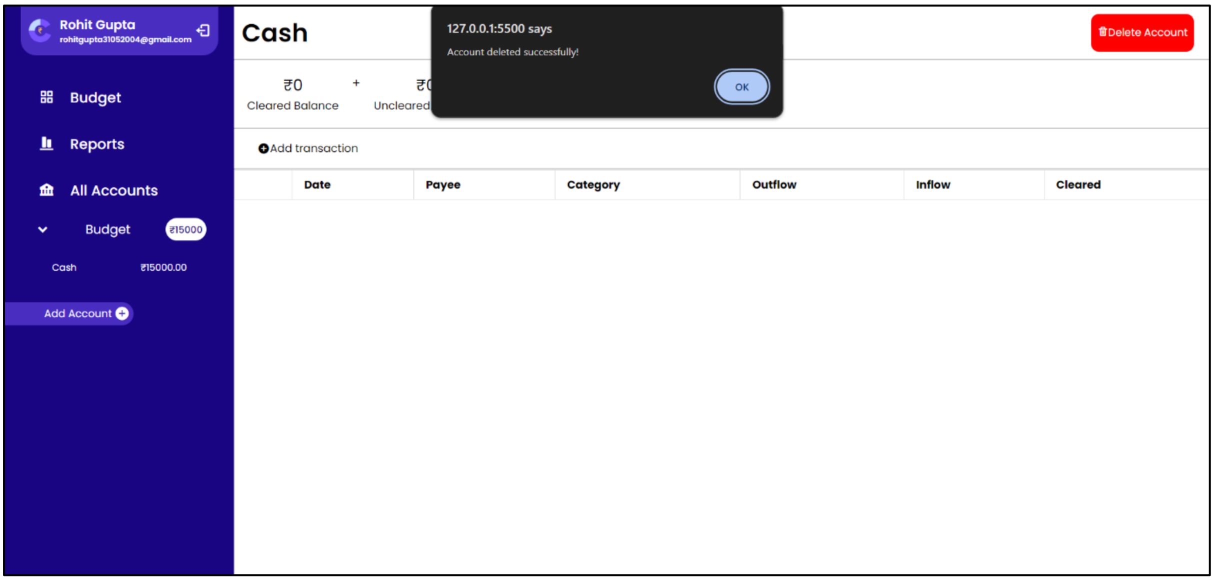
₹0 Cleared Balance + ₹0 Uncleared Balance = ₹0 Working Balance

Add transaction

	Account	Date	Payee	Category	Outflow	Inflow	Cleared

Test Case 9: Delete Account

Test Name	Unit Testing
Test Description	Verify that a user can successfully delete an account.
Objective	Ensure that users can delete an account from their account list.
Input	Select the account to delete from the account list, click the delete button next to it, and confirm the deletion action when prompted.
Expected Result	The account is deleted successfully, a confirmation message is displayed, and the account is no longer visible in the account list.
Output	Passed



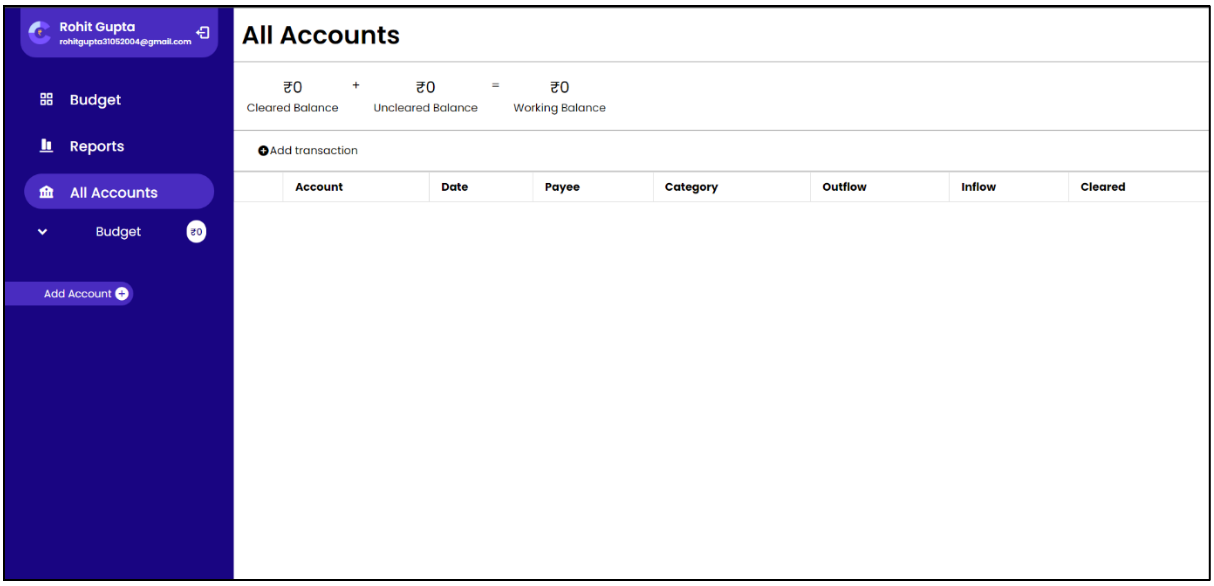
127.0.0.1:5500 says

Account deleted successfully!

₹0 Cleared Balance + ₹0 Uncleared

Add transaction

	Date	Payee	Category	Outflow	Inflow	Cleared



All Accounts

₹0 Cleared Balance + ₹0 Uncleared Balance = ₹0 Working Balance

Add transaction

	Account	Date	Payee	Category	Outflow	Inflow	Cleared