# DimeSage

**A Project Report**

Submitted in partial fulfillment of the

Requirements for the award of Degree of

**BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)**

**By**

**Mr. Tanay Praveen Bhomia (03)**

**Seat Number :**

**Under the esteemed guidance of**

**Mrs. Vibhuti Sane**

**Assistant Professor**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**St. John College of Humanities and Sciences**

***(Affiliated to University of Mumbai)***

**Palghar, MAHARASHTRA , 401404**

**2023-2024**

**St. John college of Humanities and Sciences**

***(Affiliated to University of Mumbai)***

**PALGHAR – MAHARASHTRA - 401404**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

## <u>CERTIFICATE</u>

This is to certify that the project entitled, "**DimeSage**", is bonafide work of **Tanay Praveen Bhomia** bearing Seat. submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATIONTECHNOLOGY from University of Mumbai.

| | | |
|---|---|---|
| **Project Guide** | **External Examiner** | **HOD** |
| **Date** | **Internal Examiner** | **College Seal** |

# ABSTRACT

In this era of economic complexities, the financial well-being of students and others necessitates innovative solutions. Our study introduces a sophisticated financial management application, integrating Artificial Intelligence and Natural Language Processing, to provide a comprehensive suite of tools. By offering personalized budgeting, tracking financial goals, and analyzing spending patterns, the application empowers students to forge a path towards financial independence and responsibility. With dynamic charts and timely notifications, one can proactively adapt their financial strategies and make informed decisions.

Moreover, the application’s commitment to user empowerment extends to international financial dynamics. By seamlessly incorporating currency conversion, students engaging in cross-border transactions can effortlessly navigate global financial landscapes, underscoring the application's comprehensive scope and relevance in an increasingly interconnected world. Furthermore our commitment to a seamless and delightful user experience is evident in our meticulous focus on user interface. Every facet of the application’s interface is thoughtfully crafted to ensure user engagement and comprehension.

In summary, our innovative financial management application, fortified by the integration of AI and NLP, represents a pivotal step forward in equipping students and individuals with the tools they need to navigate the intricacies of modern economics. From personalized budgeting to international financial prowess, our application stands as a testament to our commitment to fostering fiscal empowerment and resilience in a complex and ever-evolving financial landscape.

# ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my Project in charge as well as our Project Guide **Ms. Vibhuti Sane** who gave us the golden opportunity to do this wonderful project on the topic **DimeSage**, which also helped me in doing a lot of research and I came across many new things. Also I express my sincere and heartfelt gratitude to our Principal **Dr. N. Sani** and our Head of Department **Ms. Trupti Bidaye** for the support which led to the successful accomplishment of the project. Also, I would like to thank our whole I.T department who helped me in every aspect of my project.

**Mr. Tanay Bhomia**

# DECLARATION

I hereby declare that the project entitled, **DimeSage** done at **St. John College of Humanities and Sciences**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge and other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as a final semester project as part of our curriculum.

**Mr. Tanay Bhomia**

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# INTRODUCTION

This chapter presents an extensive contextual backdrop, elucidating the project's positioning in relation to existing scholarly and practical endeavors within the field. By offering a concise synthesis of preceding research and undertakings, the groundwork is laid for the forthcoming project, establishing a clear continuum of knowledge progression. Concurrently, the section meticulously outlines the project's anticipated trajectory, encapsulating its intended objectives and distinctive contributions, thereby guiding the reader through the forthcoming phases with a comprehensive understanding of the project's overarching aspirations.

Embedded within this contextual framework is a vivid articulation of the project's imperative, underscoring the exigent rationale driving its execution. Through a judicious exploration of the project's scope, its boundaries and dimensions are delineated, concurrently unveiling the tangible real-world applications that underpin its significance. This synthesis thereby affirms the project's inherent value by showcasing its practical utility and relevance in addressing contemporary challenges, effectively fostering an appreciation for the project's symbiotic relationship with both theoretical advancements and practical exigencies in the broader landscape.

## 1.1 Background

Navigating the realm of spending and financial management has proven to be one of the most formidable challenges I've encountered. Despite numerous attempts, from Documenting expenses daily to Digital diaries, success has eluded me. Frustratingly, these efforts have fallen short, often due to the absence of crucial features or accessibility limitations. It's this personal struggle that ignited the spark for a project aimed at solving this deeply ingrained issue.

Recognizing the intricacies and intricacies of managing money in a world saturated with choices and temptations, I resolved to develop a solution that would not only address my own struggles but also empower others facing similar challenges. This project is born out of a genuine desire to create an effective and user-friendly platform that resonates with individuals like me who have wrestled with financial management and expense tracking

By drawing upon the latest advancements in technology, including Artificial Intelligence (AI) and Natural Language Processing (NLP), my aim is to provide everybody with a tool which makes tracking expenses a fun task rather than making it feel like a chore. I envision an intuitive application that not only simplifies expense tracking but also offers a comprehensive suite of features designed to tackle the multifaceted aspects of financial management. From budgeting and goal-setting to insightful analysis and tailored recommendations, this project seeks to provide a holistic and accessible solution to a pervasive problem.

## 1.2 Objective

The initial hurdle encountered with many existing expense tracking solutions revolved around the tiresome and repetitive nature of daily expense entry. The prevalent widget-based interfaces typically required users to input details like the purchased item, amount, and date—resulting in a monotonous process that gradually drained motivation. Consequently, the enthusiasm for daily expense tracking waned, leading to inconsistency and a lack of order in managing expenses.

However, it's important to note that the act of merely recording daily expenses doesn't inherently foster improved financial habits or enhance overall financial well-being. This project delves deeper, aiming to address these shortcomings by offering a dynamic solution that extends beyond mere data entry. It seeks to establish a personalized guidance system that assists users in cultivating healthier spending behaviors and achieving greater financial health.

Central to the project's mission is the integration of AI technology to provide invaluable support in enhancing expense management practices. The infusion of AI can offer personalized insights, suggestions, and strategies, thereby transforming the process into an engaging and rewarding endeavor. Beyond streamlining expense tracking, this project aspires to fill a crucial void in the realm of financial management. It envisions serving as a comprehensive hub that caters to a diverse range of financial needs. From efficiently tracking online subscription services to effectively managing ambitious financial goals, the application aims to be an encompassing resource—a one-stop destination for all financial requirements.

By addressing the challenges of redundancy, motivation, and guidance through innovative AI-driven features, this project endeavors to redefine expense tracking and financial management. The overarching goal is to empower users to take charge of their financial well-being, fostering a lasting positive impact on their habits and overall financial health. Another main thing that I had to keep in mind is that most users use their laptop a lot in many areas like offices and homes and I thought the laptop would be the perfect environment for building this application.

## 1.3 Structure of Web Apps

A web application, often referred to as a web app, is a software application that is accessed and used through a web browser over the internet. Unlike traditional desktop applications, which are installed on a user's computer, web apps run on remote servers and are accessed by users through their web browsers. Web apps have become increasingly popular due to their accessibility, cross-platform compatibility, and ease of maintenance.

The structure of a web app can be divided into several components, each serving a specific purpose. Here's an overview of the typical structure of a web app.

### 1.3.1 Client-Side (Front-End):

- This is the part of the web app that runs in the user's web browser and is responsible for the user interface and user interactions. It's built using web technologies like HTML, CSS, and JavaScript. The front-end communicates with the server-side components to fetch and display data.
- **HTML (Hypertext Markup Language):** Provides the structure and content of the web page.
- **CSS (Cascading Style Sheets):** Defines the visual styling and layout of the web page.
- **JavaScript:** Adds interactivity and dynamic behavior to the web app, such as handling user input, making AJAX requests, and updating the page without requiring a full page reload.

### 1.3.2 Server-Side (Back-End):

- This is the part of the web app that runs on the server and handles tasks like processing requests, managing databases, and performing business logic. It's built using various programming languages and frameworks.
- **Web Server:** Handles incoming requests from clients (web browsers), routes those requests to appropriate parts of the application, and sends back responses.
- **Application Logic:** Performs the core functionality of the web app, such as processing user input, managing data, and applying business rules.
- **Database:** Stores and retrieves data necessary for the application. Popular databases include MySQL, PostgreSQL, MongoDB, etc.
- **Server-Side Scripting:** Server-side code executes on the server before sending a response to the client. Common server-side scripting languages include Python, Ruby, Node.js, Java, PHP, etc.

### 1.3.3 API (Application Programming Interface):

- APIs are endpoints exposed by the server-side of the web app that allow other applications or services to interact with and consume the functionality of the app. They can be used for data retrieval, manipulation, and more. APIs often use standards like REST (Representational State Transfer) or GraphQL. APIs mainly help the application to communicate with various components of the application like Databases and other Modules, can be Chatbots and AI modules

### 1.3.4 Database Management:

- The server-side of the web app interacts with databases to store and retrieve data. The database stores user information, content, settings, and other relevant data. The choice of database technology depends on factors like data structure, scalability, and performance requirements.
- The sensitivity of the data being stored and the security measures required to protect it can influence the choice of a database system. Some databases offer advanced security features like encryption, access control, and auditing, which might be essential for applications handling confidential or regulated data. The level of compliance with data protection regulations and industry standards may also be a consideration.

### 1.3.5 Middleware:

- Middleware components provide additional functionality and services between the front-end and back-end. They might include authentication and authorization layers, caching mechanisms, logging, and security measures.

### 1.3.6 External Services and Libraries:

- Web apps often integrate with external services and libraries for various functionalities, such as payment processing, social media integration, analytics, and more.

### 1.3.7 Deployment and Hosting:

- A web app needs to be deployed on a server that can be accessed over the internet. It can be hosted on cloud platforms like AWS, Azure, Google Cloud, or on dedicated servers.

## 1.4 Applications

This project encapsulates a comprehensive array of tools and functionalities, geared towards elevating the realm of financial management across a diverse spectrum of user profiles. Designed to resonate with students embarking on their financial journey, professionals fine-tuning budgetary allocations, and individuals seeking to enhance their fiscal dexterity, the project's applications radiate far and wide.

Students can gain invaluable insights into expense tracking, cultivating financial mindfulness that will stand them in good stead throughout their academic journey. On the other end of the spectrum, professionals navigating complex career paths find a reliable companion in their pursuit of optimized financial planning and resource allocation. Beyond these realms, the project's reach extends to anyone eager to reinforce their financial prudence, transforming everyday consumers into informed decision-makers.

This project bridges the gap between financial aspiration and reality, resonating with each individual's specific goals and circumstances. Whether it's about instilling lifelong fiscal literacy, harnessing cutting-edge technology for dynamic budgetary adjustments, or simply attaining greater control over personal finances, this project emerges as a conduit for financial empowerment that transcends boundaries and resonates across myriad life trajectories.

## 1.5 Future Scope

With the ever-growing need for efficient expense management, this project holds immense potential to cater to the evolving requirements of users. As we delve into the future of this application, we can anticipate significant advancements, especially in areas where AI and machine learning play a pivotal role. These areas include:

- **Expense Analysis Enhancement**
  - Continuous improvement in AI-driven expense analysis is on the horizon. By integrating more advanced machine learning algorithms and sophisticated data analytics techniques, the application can offer predictive analytics capabilities. This will enable users to anticipate their future spending patterns based on historical data and prevailing economic trends.
- **Expense Categorization and Tagging**
  - The application's AI system will evolve to automatically categorize and tag expenses based on transaction descriptions. As it becomes more proficient, this feature will not only save users time but also provide increasingly accurate insights into their spending habits.
- **Better Natural Language Processing**
  - The application's Natural Language Processing (NLP) capabilities will continuously improve. It will adapt to recognize common words and phrases more effectively, enhancing the user experience and data storage functionality.

In addition to these enhancements, here are some key features that can be incorporated to make the application even more valuable for a wide range of users:

- **Expense Receipt Scanning**
  - Implement a feature that empowers users to scan and upload expense receipts. The AI system can then extract pertinent information from these receipts, automating the process of logging expense details accurately.

- **Integration with Financial Institutions**
    - Establish seamless integration with banks and financial institutions to fetch real-time transaction data. This integration will provide users with up-to-the-minute financial information, reducing the need for manual data entry and ensuring data accuracy.
- **Credit Score Monitoring**
    - Integrate credit score monitoring into the application, allowing users to track how their financial behaviors impact their credit scores. An AI-powered system can offer insights and tips to improve users' credit health, empowering them with financial knowledge.
- **Expense Sharing and Splitting**
    - Enhance user convenience by enabling easy expense sharing and splitting with friends or family members. Implement a feature that calculates and tracks shared expenses, particularly useful for group activities and shared bills.
- **Gamification for Financial Education**
    - Make financial management engaging by introducing gamification elements. Users can earn points or rewards for achieving financial goals and making smart financial decisions, transforming expense management into an enjoyable experience.
- **AI-Powered Investment Recommendations**
    - Develop a sophisticated AI-driven investment advisory feature. This feature will provide users with personalized investment recommendations based on their financial goals, risk tolerance, and real-time market conditions, helping them make informed investment decisions.

These proposed features align with the project's primary objective of making expense management not only efficient but also enjoyable. As the AI learns and adapts over time, the application will continuously evolve, offering users an ever-improving tool for managing and tracking their expenses with ease and confidence.

# SYSTEM ANALYSIS

This chapter plays a pivotal role in the dissertation, serving as a critical examination of the limitations and challenges associated with the current pool of expense analysis applications designed to help users track their financial expenditures. It is meticulously structured to offer readers a profound understanding of the problem domain, while also introducing essential elements and factors crucial to the project's overall success.

## 2.1 Existing System

Existing expense management systems are often criticized for their complexity. These systems typically feature intricate user interfaces, multi-layered menus, and a myriad of settings and options. This complexity can be overwhelming for users, especially those who are not tech-savvy or looking for a straightforward solution to manage their finances. Users might need extensive training or spend a significant amount of time learning how to use these systems effectively.

Furthermore, the complexity may result in users not fully utilizing all the features and capabilities of the system. As a result, they might miss out on valuable tools for budgeting, tracking expenses, or gaining insights into their financial health. This complexity can also lead to errors in data entry or configuration, potentially compromising the accuracy of financial records. The 2 main methods for managing expenses are as follows.

### 2.1.1 The Perils of Handwritten Solutions

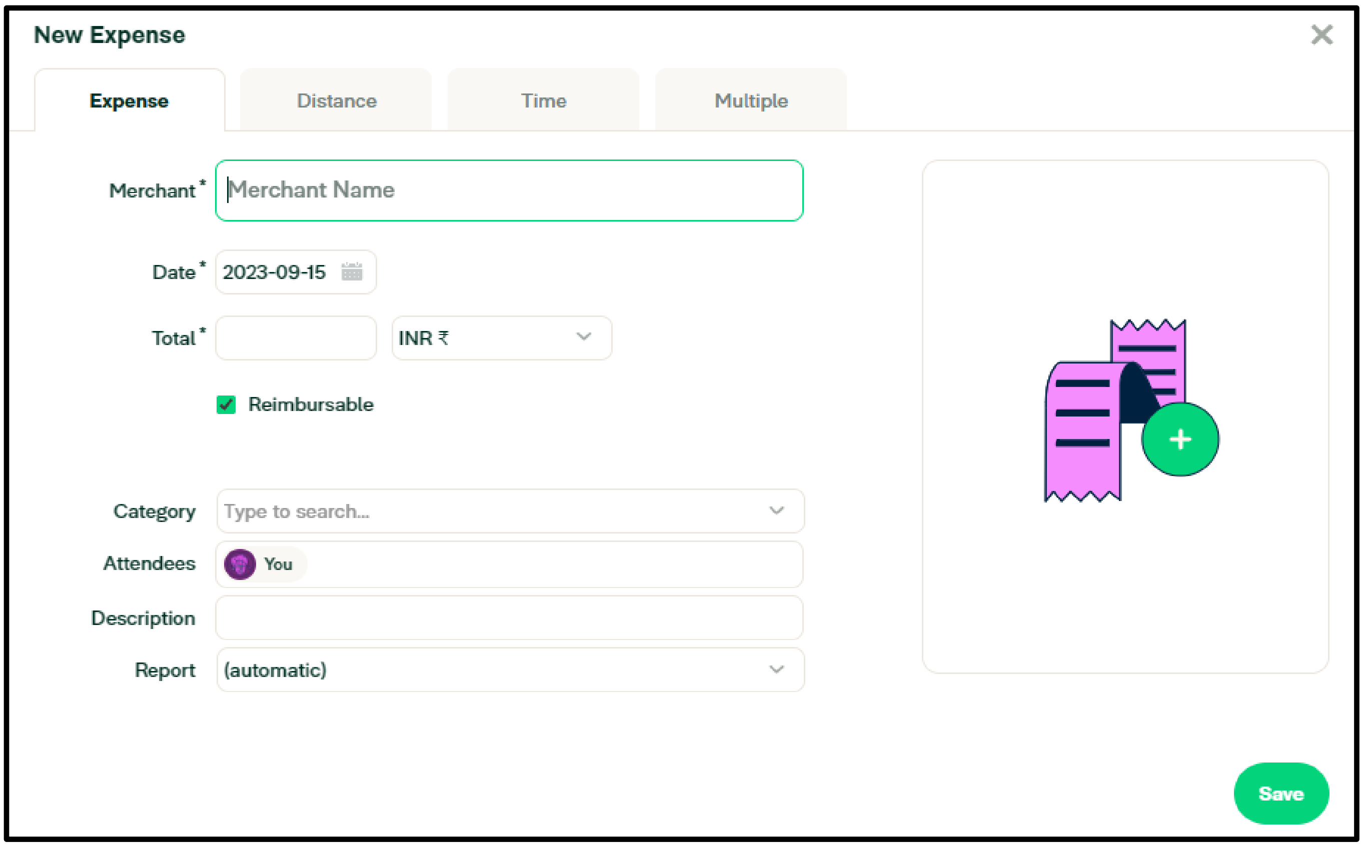
- While digital tools and apps have proliferated in recent years, some individuals still resort to age-old handwritten methods for managing expenses. These manual approaches, such as maintaining paper ledgers or spreadsheets, come with their own set of problems and limitations.
- Many people use small diaries which are kept in their pockets and are accessed whenever they want to maintain a record or add something.
- Handwritten expense tracking is bound to physical records, making it less accessible and mobile compared to digital solutions. Users may find it inconvenient to carry around physical ledgers or notebooks to record expenses when they are on the go.
- Handwritten expense tracking is inherently labor-intensive and prone to human errors. Users must painstakingly record each expense, income, and financial transaction, leaving ample room for oversights and inaccuracies. Moreover, as the volume of financial data grows, maintaining handwritten records can become increasingly burdensome and time-consuming.
- Additionally, handwritten methods often lack the sophistication required for comprehensive financial analysis. Users are left with static records that provide little insight into spending patterns, budget adherence, or potential cost-saving opportunities. The absence of dynamic features, such as automated alerts or personalized recommendations, makes it challenging for individuals to make informed financial decisions

## 2.1.2 Half Baked Apps - A fragmented Landscape

- The landscape of expense management applications is vast, offering users a plethora of choices. However, a significant portion of these applications falls into the category of what can be described as "half-baked" apps. These solutions typically exhibit a set of common shortcomings that undermine their effectiveness in helping users gain control over their finances.
- Most of the apps usually revolve around simple widgets which lets the user enter their expenses according to categories and dates but this is very tiresome and inefficient.
- Such apps often lack comprehensive features, offering only basic functionalities for expense tracking and budgeting. They may lack the sophistication to provide meaningful insights into spending patterns, leaving users in the dark about how to optimize their financial decisions. Additionally, user interfaces can be clunky and unintuitive, adding a layer of complexity that hinders users rather than empowering them.
- Furthermore, many of these applications fail to adapt to the unique needs and preferences of individual users. They provide generic templates and categories for budgeting and expense tracking, overlooking the diverse financial situations, goals, and spending habits that users have. This one-size-fits-all approach can lead to inaccurate budgeting and a lack of alignment between the app's features and the user's financial aspirations.

### 2.1.2.1 Manual Data Entry

- One of the most prominent drawbacks of many existing expense management systems is the reliance on manual data entry. Users are required to input their financial transactions, including expenses and income, manually into the system. This process is not only time-consuming but also prone to errors.
- This is done with the help of various widgets which let you enter expenses information like the date,category of the expense manually which is very slow, frustrating and inefficient at the same time
- Manual data entry can become a significant deterrent to users who may find it burdensome to record each transaction, especially when dealing with numerous expenses or frequent financial activity. The risk of errors further undermines the reliability of financial records, which can lead to incorrect budgeting and financial analysis.
- Moreover, manual data entry can discourage users from using the system consistently. Over time, users may become less motivated to enter their expenses diligently, leading to incomplete or inaccurate financial records.



| | |
|---|---|
| Merchant * | Merchant Name |
| Date * | 2023-09-15 |
| Total * | INR ₹ |
| | ✓ Reimbursable |
| Category | Type to search... |
| Attendees | You |
| Description | |
| Report | (automatic) |

*Figure 1* ***: A simple widget-based expense tracker***

### 2.1.2.2 Lack of Personalization

- Many existing expense management systems provide a one-size-fits-all approach to financial management. They offer pre-defined budgeting templates and categories that may not align with individual users' unique financial situations and goals.
- This lack of personalization can limit the effectiveness of these systems. Users may struggle to create budgets that accurately reflect their income, spending habits, and financial aspirations. Consequently, they might not achieve the level of financial control and optimization they desire.
- Personalization is crucial in financial management as it allows users to set realistic goals, allocate resources efficiently, and track progress effectively. Without this feature, existing systems may fall short in meeting the diverse needs of their user base.

### 2.1.2.3 Limited Mobile Accessibility

- While some existing expense management systems offer web-based platforms, not all of them provide user-friendly mobile applications. In today's fast-paced world, where individuals rely heavily on smartphones and tablets for various tasks, including financial management, limited mobile accessibility can be a significant drawback.

- Users often require the flexibility to manage their expenses on the go. This includes adding transactions, checking account balances, and receiving real-time notifications. Expense management systems that lack a mobile app or offer a subpar mobile experience may not meet these essential needs, leading to user dissatisfaction and reduced adoption rates.

#### 2.1.2.4 Privacy and Security Concerns

- Privacy and security are paramount in any financial management system. Users entrust these platforms with sensitive financial information, including transaction details, income sources, and savings goals. However, high-profile data breaches and privacy scandals in recent years have raised concerns about the safety of such data.
- Many existing expense management systems may not fully address these concerns, leaving users apprehensive about using these platforms. Even if the systems have robust security measures in place, the perception of potential risks can deter users from fully engaging with the platform.
- Addressing privacy and security concerns is not just a matter of implementing strong encryption and data protection measures; it also requires transparent communication with users about how their data is handled and protected.

#### 2.1.2.5 Insufficient Integration

- Effective financial management often involves the coordination of various financial tools and services. Users may need to link their expense management system with their bank accounts, credit cards, and other financial platforms to get a comprehensive view of their finances.
- However, many existing expense management systems lack the necessary integration capabilities. This means users must perform duplicate data entry, manually inputting transactions that are already available through their bank statements. This redundancy can be time-consuming and prone to errors.
- Furthermore, the lack of integration can limit the system's ability to provide users with a holistic view of their financial situation, making it challenging to make informed financial decisions.

#### 2.1.2.6 Lack of AI-driven Insights

- Advanced technologies, such as Artificial Intelligence (AI), have the potential to revolutionize expense management by providing users with personalized insights and recommendations. However, many existing systems do not leverage AI to its full potential.

- AI can analyze a user's spending patterns, identify opportunities for cost savings, and offer tailored budgeting recommendations. By understanding a user's financial goals and habits, AI-driven systems can provide valuable insights that empower users to make informed financial decisions.
- Existing systems that do not incorporate these AI-driven features may fall behind in helping users optimize their finances, potentially leading to less effective financial management.

#### 2.1.2.7 Incomplete Features

- Depending on the specific expense management system, certain features and functionalities may be missing or underdeveloped. For instance, some systems may lack robust reporting capabilities, making it difficult for users to visualize their spending patterns and financial trends over time.
- Other features, such as automated subscription tracking and currency conversion, are not always standard offerings in existing systems. This can limit the system's ability to provide a comprehensive solution for users who need to manage various aspects of their finances.
- Users often have diverse needs and preferences when it comes to expense management, and existing systems may not cater to all these requirements.

In conclusion, the existing systems in the realm of expense management exhibit various limitations and challenges that have motivated the development of the proposed expense management application. These limitations encompass issues related to complexity, manual data entry, lack of personalization, limited mobile accessibility, privacy and security concerns, insufficient integration, a lack of AI-driven insights, and incomplete features. The proposed application aims to address these shortcomings by providing a user-friendly, personalized, and feature-rich solution for managing expenses effectively and efficiently.

## 2.2 Proposed System

The world of finance has undergone a remarkable transformation in recent years, driven by the rapid digitization of transactions and the ever-evolving nature of modern finance. In this dynamic landscape, the significance of effective expense tracking has never been more pronounced. While existing expense management solutions abound, they often fall short of meeting the demands of today's users who seek a seamless, personalized, and technologically advanced experience. This chapter introduces my innovative expense tracker application, the culmination of extensive research and development. Rather than serving as a mere starting point, this application represents the apex of my dedication and effort to revolutionize how individuals and organizations interact with their financial resources, enabling them to make well-informed decisions with unwavering confidence

### 2.2.1 Effortless Expense Tracking

My proposed application sets out to redefine how users manage their expenses, departing from the laborious process of manual data entry. By seamlessly integrating with various financial platforms and leveraging cutting-edge Natural Language Processing (NLP) technology, users can effortlessly record transactions in a conversational manner. This innovative approach not only eliminates the tedium of manual entry but also significantly reduces the risk of errors, ensuring that financial records are consistently accurate and up to date. With this revolutionary feature, I aim to free users from the time-consuming and error-prone practice of manual data entry.

### 2.2.2 Tailored Financial Insights

At the heart of my application lies the power of Artificial Intelligence (AI), unlocking personalized financial insights and recommendations. This intuitive AI-driven feature ensures that users receive real-time guidance aligned with their unique financial goals, spending habits, and income sources. My application acts as a virtual financial advisor, empowering users to make well-informed decisions that resonate with their individual aspirations. By harnessing the capabilities of AI, I empower users to access a level of financial guidance and personalization that transcends the capabilities of traditional expense management systems.

### 2.2.3 Seamless Mobile Accessibility

Recognizing the dynamic nature of modern life, my application extends unparalleled convenience to users with its user-friendly mobile interface. Whether on the daily commute or during international travels, users can manage their expenses on the go. This includes the ability to record transactions, check account balances, and receive timely notifications, all from the palm of their hand. My commitment to mobile accessibility ensures that users remain connected to their financial data and resources at all times, eliminating the constraints of location or device.

### 2.2.4 Data Security Reinvented

As users entrust my application with their sensitive financial information, I prioritize data security and transparency above all. Robust encryption mechanisms, coupled with transparent communication protocols, ensure that users' financial data remains shielded from prying eyes while maintaining the highest level of transparency regarding data handling and protection. I understand that safeguarding financial data is paramount, and my robust security measures provide users with the peace of mind they deserve in today's data-driven world.

### 2.2.5 AI’s Guiding Hand

At the core of my application's functionality lies AI's transformative potential. It harnesses the ability to analyze users' spending patterns, identify opportunities for cost savings, and deliver tailored budgeting recommendations. By embracing users' financial goals and behaviors, this AI-driven system equips users with valuable insights, empowering them to navigate the financial terrain with confidence. My AI-driven insights serve as a constant companion, providing users with the guidance they need to achieve their financial aspirations.

### 2.2.6 All-in-One Solution

My application encompasses a diverse array of features, ranging from automated subscription tracking to seamless currency conversion and robust reporting capabilities. These features collectively form a comprehensive financial toolkit, adaptable to the diverse needs and preferences of my users, offering a holistic solution to manage every aspect of their financial portfolio. My commitment to providing a comprehensive solution ensures that users have all the tools they need at their fingertips to effectively manage their finances, eliminating the need for multiple disjointed financial applications.

## 2.3 Requirement Analysis

A Software Requirement Specification (SRS) serves as a comprehensive document that defines both the functional and non-functional requirements of a proposed software system. This document outlines the specific features and capabilities the software should possess. Additionally, it may incorporate a collection of use cases that detail the various ways users will interact with the software.The primary purpose of an SRS is to establish a clear and detailed understanding of how the software should operate. It acts as the foundational agreement between the client or customer and the developers or suppliers involved in the project. In the context of market-driven projects, these roles may be taken on by marketing and development divisions.

The SRS serves as a critical early-stage assessment of requirements, conducted before proceeding to more specific phases of system design. Its overarching goal is to preemptively address and mitigate issues, thereby reducing complications and costly revisions in later stages of the software development process. Essentially, it provides a blueprint that ensures everyone involved is on the same page regarding the software's functionality and behavior.

### 2.3.1 Purpose

The purpose of this SRS is to:

- Clearly define the functional and non-functional requirements of the application.
- Provide a detailed understanding of the system's capabilities and limitations.
- Serve as a reference document for both the development team and the client to ensure alignment with project objectives.

### 2.3.2 Types of Requirement Analysis

There are different types of Requirements which the developer has to satisfy for the user in order to build a good app. This part of the document plays an important role in explaining what are those requirements in the first place.

**Functional Requirements:**

These requirements provide a detailed description of what the system should do. Their formulation depends on factors like the specific type of software under development, the anticipated user base, and the organizational approach to requirement specification. While user requirements often express these needs in abstract terms, functional system requirements dive into the granular details of system functions, covering inputs, outputs, exceptional cases, and more.

**Non-Functional Requirements:**

This category introduces constraints on the services or functions offered by the system. Non-functional requirements encompass a wide array of factors, including timing constraints, development process stipulations, and compliance with standards. Importantly, these requirements tend to apply holistically to the entire system, guiding its overall performance and behavior rather than addressing individual features or services.

**Performance Requirements:**

Performance requirements set the benchmark for how effectively the system must execute specific functions under predefined conditions. Key metrics include factors like response time, throughput, execution speed, and storage capacity. These performance criteria are often intricately aligned with the system's ability to support essential end-user tasks seamlessly.

**Safety Requirements:**

Safety requirements come into play when considering worst-case scenarios and catastrophic failures that could potentially inflict extensive damage to the system or data. For instance, in the event of a disk crash, an effective recovery method should be in place to restore a previous database snapshot backed up to archival storage, typically tape. Subsequently, it should reconstruct a more current state by reapplying or redoing the operations of committed transactions from the backed-up log, up to the point of failure.

**System Requirements:**

These are the foundational building blocks that developers rely on to construct the system. System requirements are often categorized as either functional or supplemental requirements. Functional requirements articulate precisely what users need to accomplish their tasks, while supplemental (or non-functional) requirements encompass every other aspect not covered within functional requirements. Supplemental requirements, sometimes referred to as quality-of-service requirements, span diverse aspects such as system security, performance, and usability.

**Security Requirements:**

Security requirements hold paramount importance in safeguarding sensitive data and ensuring system integrity. They entail a rigorous process of assessing, justifying, and analyzing new software features before deployment. Security systems must include robust measures such as encryption and access controls to protect user data privacy and prevent unauthorized access or malicious use.

**Architectural Requirements:**

Architectural requirements provide a blueprint for how systems should integrate, define their structure, and specify their behavior. In software engineering, these are deemed architecturally significant requirements, as they wield a substantial impact on the software system's architecture.

**Portability Requirements:**

Portability testing focuses on evaluating how effortlessly a software component or application can transition from one environment to another. The results of portability testing illuminate the software's adaptability across diverse platforms and systems, shedding light on compatibility and ease of use across varying environments.

## 2.3.3 Feasibility Study

Feasibility studies serve as a methodical and objective process to uncover the inherent strengths and weaknesses of an existing business or a proposed venture. They delve into the opportunities and threats posed by the surrounding environment, assess the necessary resources for successful execution, and gauge the prospects for achieving desired outcomes. In essence, feasibility studies revolve around two fundamental criteria: the cost required for implementation and the anticipated value to be derived.

A well-structured feasibility study entails various critical components, starting with an exploration of the historical context of the business or project. It provides a comprehensive project description, offers insights into service aspects, furnishes essential accounting statements, outlines operational intricacies and management strategies, incorporates rigorous marketing research and associated policies, compiles indispensable financial data, and meticulously addresses legal requisites and tax obligations. Importantly, feasibility studies typically serve as a precursor to technical development and the eventual implementation of a project or business endeavor.

### 2.3.3.1 Objectives

- **Assess Viability**: Feasibility studies aim to determine the overall viability and practicality of a proposed project or business venture. This includes evaluating whether the project can be accomplished within the constraints of available resources and time.

- **Risk Evaluation:** One key objective is to identify and assess potential risks and challenges associated with the project. This involves considering external factors, market dynamics, and internal issues that may impact project success.
- **Cost-Benefit Analysis:** Feasibility studies help in conducting a thorough cost-benefit analysis. This objective involves quantifying the expected costs of the project against the anticipated benefits, including financial gains and strategic advantages.
- **Market Analysis:** Understanding the market and its dynamics is crucial. Feasibility studies aim to assess the target market, customer needs, competition, and demand for the proposed product or service.
- **Resource Evaluation:** These studies help in evaluating the resources required for the project, including financial, human, and technological resources. The objective is to ensure that the necessary resources are available or can be acquired.
- **Decision-Making:** Ultimately, the main goal of feasibility studies is to provide decision-makers with well-informed data and insights. This enables them to make informed decisions regarding whether to proceed with the project, modify its scope, or abandon it based on the findings of the study.

#### 2.3.3.2 Types of Feasibility

- **Economical Feasibility**
    - Economic analysis, often referred to as cost/benefit analysis, stands out as the most commonly employed approach for assessing the viability of a new system. This method revolves around the meticulous evaluation of the anticipated benefits and savings associated with a prospective system and the subsequent comparison with its associated costs. When the benefits convincingly outweigh the costs, the decision to proceed with the system's design and implementation is made.
    - Questions need to be answered during this phase of project development which clear the idea for the reader if the project will go beyond the costing or the budget which the creator has decided. Questions are as follows
        - Is this Project possible, with the give resource environment ?
            - Yes, The project is feasible with the given resource
        - What are the development and operational cost ?
            - The Project is for the Semester so adequate resources and funding is provided
    - **Cost Based Study**
        - It is important to identify cost and benefit factors, which can be organized as follows:
            - Development costs

        - Operating costs
      - This is an analysis of the costs to be incurred in the system and the benefits derivable out of the system
  - **Times-based Study**
      - This is an analysis of the time required to achieve a return on investment. Future value of the project is also a factor.
- **Technical Feasibility**
  - The assessment is rooted in an outline design that outlines the system's requirements, encompassing inputs, processes, outputs, files, programs, and procedures. This assessment can be quantified by considering data volumes, trends, update frequencies, and more. Its primary purpose is to estimate whether the new system will deliver adequate performance.
  - Technological feasibility, on the other hand, is conducted to determine the organization's readiness in terms of software, hardware, personnel, and expertise to successfully execute the project. When creating a feasibility report, it's essential to consider the following factors: a concise overview of the business to identify additional potential influencing factors, the specific segment of the business under examination, human and economic aspects, and potential problem-solving approaches.
  - Technical feasibility involves addressing various technological aspects and gathering pertinent information. This can be achieved by posing relevant questions to users and stakeholders.
      1) **Is this project feasible with current technology ?**
          - Yes, The Project is feasible with current advanced in AI , NLP and Other Machine learning technologies
      2) **What technical risk is there ?**
          - The most prominent risk in this technology is malfunctioning AI.
      3) **Availability of technology ?**
          - Technology is available widely on the internet from Open Source sources
      4) **Is it available locally ?**
          - Yes , The users can install the website
      5) **Can it be obtained ?**
          - The website will be available through the internet search engine.
      6) **Do we currently possess the necessary technology ?**
          - Yes, I have included the necessary tech in the project
      7) **What kind of technology is needed ?**

        - Following is the collection of software which includes both new and necessary technology required to build this WebApp
            i) Python
            ii) Flask
            iii) HTML
            iv) CSS
            v) JavaScript
            vi) VsCode
            vii) MySql
    8) **Is the required technology available in house ?**
        - Yes, The technology is required in house

- **Schedule Feasibility**
    - The success of a project hinges on its timely completion and its ability to become useful within a reasonable timeframe. To ascertain this aspect, it is imperative to estimate the system's development duration and assess whether it aligns with the prescribed time frame, often employing methods like the payback period. Schedule feasibility, in turn, serves as a yardstick to gauge the practicability of the project's timeline.
    - In our case, we need to evaluate whether the project's deadlines are both mandatory and achievable. Two key questions come into play:
        1) **Are There Any Constraints on the Schedule?**
            - Yes, the project is mandated to conclude by the end of the semester, encompassing the delivery of an operational system, comprehensive documentation, and a final presentation.
        2) **Can These Constraints Be Met?**
            - Affirmatively, the constraints are attainable and can be met within the stipulated time frame.
- **Operational Feasibility**
    - Certainly, the proposed system is designed with a fully user-friendly graphical user interface (GUI), making it easily understandable even for individuals with limited technical knowledge. Additionally, comprehensive training sessions have been conducted to familiarize users with the system's functionalities, ensuring their comfort and confidence in using the new system. Based on our assessment, clients are not only comfortable with the system but also delighted with its ability to significantly reduce their workload.
    - Operational feasibility encompasses a thorough examination to assess whether the organization's needs can be effectively met by implementing the project. It also evaluates

how well the project plan aligns with the requirements identified during the initial requirements analysis phase of system development. Operational feasibility serves as a gauge for how effectively the proposed system addresses existing challenges, capitalizes on identified opportunities from the scope definition, and fulfills the requirements outlined in the requirements analysis phase of system development.

- Questions again for the Operational feasibility are as follows
    - **If the system is developed will it be used ?**
        - Yes, The system is made for satisfying the users requirements in this particular sector
    - **Social Acceptability ?**
        - The Product is socially acceptable because of the lack of such features in any other expense tracking systems
    - **Legal Aspects and Government regulations ?**
        - Given that there are no trade secrets or proprietary information involved, there is no requirement for signing a non-disclosure agreement, especially since it is an integral component of E-business solutions. The expense tracker application operates by utilizing user inputs to provide tailored recommendations that align with their financial requirements. This recommendation mechanism operates akin to a voting system.
        - Each financial choice a user makes is associated with related expense categories or recommendations. When a user selects a particular choice, all associated expense categories or recommendations receive one vote. Upon completing the assessment, the application's results page will display expense categories or recommendations sorted by the number of votes they have received, offering users a prioritized view of their financial focus areas.

## 2.3.4 Planning and Scheduling

### 2.3.4.1 Project Plan

| Start Date | Activity | End Date |
|---|---|---|
| 27/07/2023 | Searching for Project Topics | 03/08/2023 |
| 04/08/2023 | Synopsis Submission | 11/08/2023 |
| 14/08/2023 | Title Finalization | 25/08/2023 |
| 26/08/2023 | Study of selected Project | 07/09/2023 |
| 08/09/2023 | Abstract Writing | 12/09/2023 |
| 15/09/2023 | Survey of Technologies | 25/09/2023 |
| 26/09/2023 | Requirement and Analysis | 03/10/2023 |
| 04/10/2023 | System Design | 25/10/2023 |
| 04/11/2023 | Semester V Documentation Submission | 10/11/2023 |
| 18/12/2023 | Finalization of GUI and Methodology | 08/01/2024 |
| 10/01/2024 | Coding and Implementation | 12/02/2024 |
| 14/02/2024 | Database Design | 29/02/2024 |
| 04/03/2024 | Testing | 16/03/2024 |
| 18/03/2024 | Black Book Masking | 23/03/2024 |
| 26/03/2024 | Black Book Submission | 30/03/2024 |

*Table 1: **Planning and Scheduling***

#### 2.3.4.2 Project Schedule

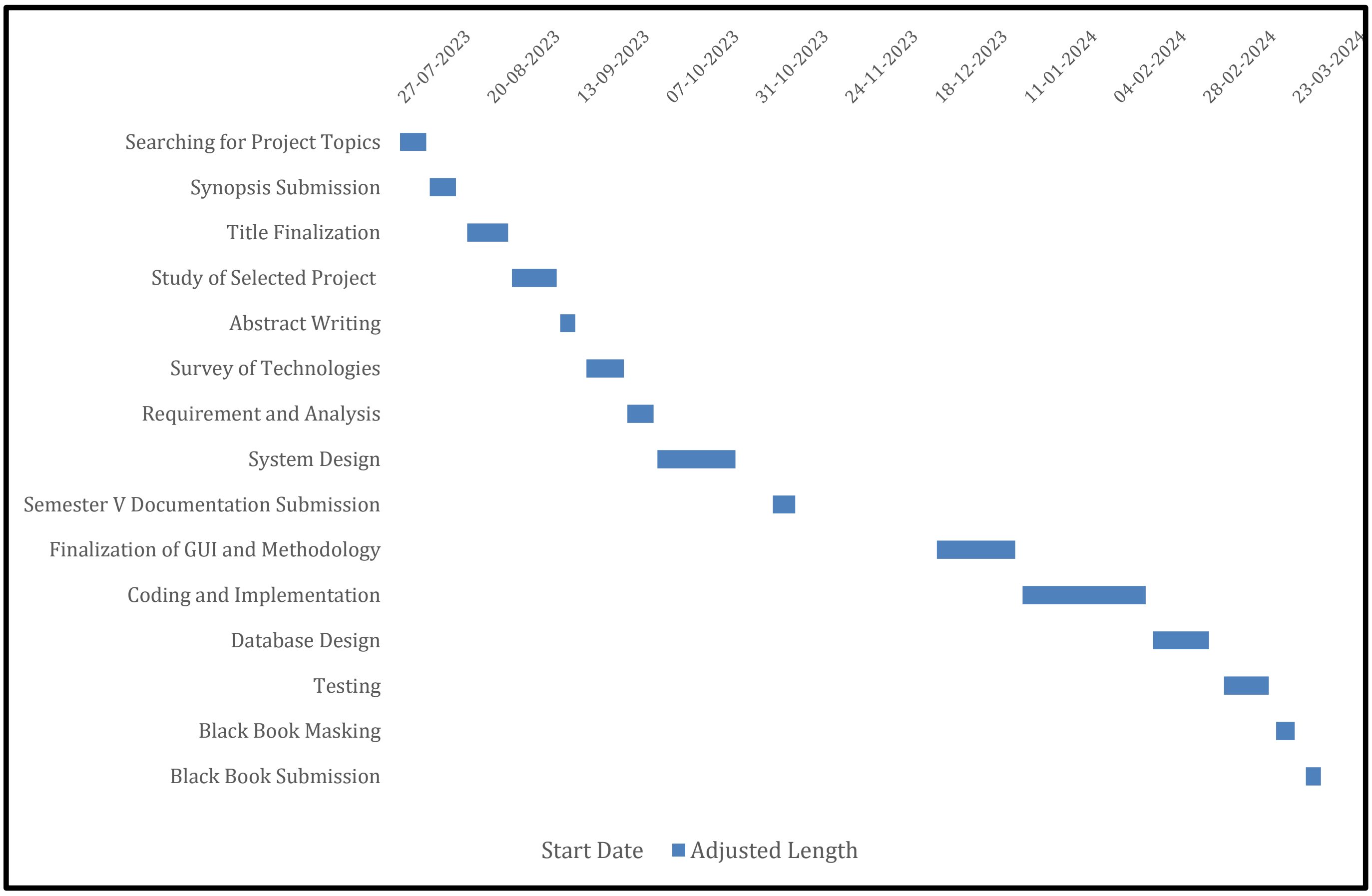


*Figure 2 : **Gantt Chart***

### 2.3.5 Preliminary Product Description

The expense tracker application begins with a user-friendly registration process where users provide essential details such as income, financial goals, and their preferred currency. Once registered, users complete their profiles, specifying their financial priorities, whether it's saving for a future goal or managing existing debts. The app offers flexibility in expense tracking, allowing users to focus on daily spending, monthly budgeting, or long-term financial planning, depending on their preferences.

The heart of the application lies in its ability to track and analyze each financial transaction, categorizing expenses, and identifying spending patterns. Leveraging this data, the app generates personalized recommendations designed to help users achieve their financial objectives. These recommendations can range from simple budget adjustments to investment opportunities or savings strategies.

For users seeking comprehensive budgeting assistance, the application provides a suite of tools to create, manage, and monitor budgets effectively. To ensure users stay informed about their financial activities, the app sends real-time alerts and feedback, including savings reminders and expense notifications. Upon completing the setup, users gain access to a personalized dashboard that offers insights into their financial health. This dashboard summarizes key metrics, provides a breakdown of spending habits, and tracks progress toward financial goals. With its user-centric approach, this expense tracker application empowers individuals to take control of their finances and make informed decisions with confidence.

## 2.4 Hardware Requirements

The development and deployment of the application necessitate specific hardware considerations to ensure optimal performance and user accessibility. While web applications are inherently versatile and can be accessed from a range of devices, it is essential to outline the recommended hardware specifications for users seeking an efficient and seamless experience.

### 2.4.1 User Hardware Specification

- **Device Diversity:** DimeSage is designed to accommodate various user devices like tablets, laptops, and desktop computers. As such, users can access the application from their preferred device, enhancing the convenience of expense management.
- **Internet Connection:** Users are expected to have a stable and reliable internet connection, supporting both Wi-Fi and mobile data networks. This ensures uninterrupted access to financial data and real-time updates.
- **Screen Adaptability:** The user interface of DimeSage is designed to be responsive, dynamically adjusting to diverse screen sizes and resolutions. This adaptability ensures an optimal user experience on screens of varying dimensions.
- **Input Devices:** Users interact with the web application using standard input devices, including touchscreens, keyboards, and mice. The user interface is responsive to touch inputs for mobile users, offering an intuitive experience.

### 2.4.2 Security and Accessibility

- **Security Awareness:** Users are educated about cybersecurity best practices, emphasizing the importance of secure browsing, password management, and data protection, given the financial nature of the application
- **Regular Backups:** To mitigate the risk of data loss, users are encouraged to perform periodic backups of their financial data or settings within the application

- **Accessibility:** The web application adheres to web accessibility guidelines (WCAG) to ensure inclusivity. Users with disabilities can access and use the application with ease.

### 2.4.3 Performance Considerations

- **Efficiency on Modest Hardware**: To cater to a wide user base, the application is optimized to perform efficiently on devices with modest hardware specifications. Resource-intensive processes that could compromise performance on older or budget devices are avoided.
- **Browser Updates:** Users are encouraged to keep their web browsers up to date to benefit from the latest features and security enhancements. Specific browser versions or settings, if required for optimal performance, are communicated to users.
- **Data Connection:** As DimeSage relies on cloud-based storage and real-time synchronization, users must have access to a reliable data connection. This ensures seamless data updates and access to financial information.

## 2.5 Software Requirements

The development and utilization of the DimeSage are contingent on specific software prerequisites that facilitate its functionality and accessibility. This section outlines the essential software requirements for users to engage with the application seamlessly.

### 2.5.1 User Software Specifications

- **Web Browser:** Users should have access to modern web browsers, such as Google Chrome, Mozilla Firefox, Apple Safari, or Microsoft Edge. The web application is optimized to perform efficiently on these browsers, ensuring consistent user experiences.
- **Operating System:** The web application's compatibility extends across various operating systems, including Windows, and macOS.
- **Internet Connectivity:** A stable and consistent internet connection is paramount for users to access and utilize the application correctly. Users can choose between Wi-Fi and mobile data networks based on their convenience.
- **JavaScript Enabled:** To enable dynamic and interactive features, users should have JavaScript enabled in their web browsers. This ensures the complete functionality of the application, including real-time updates and user interactions.

- **Cookies and Local Storage:** Users should allow cookies and local storage access in their browser settings. These mechanisms are utilized to enhance user experiences, such as remembering user preferences and login credentials.

### 2.5.2 Application Performance

- **Browser Updates:** Users are encouraged to keep their web browsers up to date to ensure compatibility and security. The application leverages the latest web technologies to provide an efficient and secure expense tracking experience.
- **Responsive Design:** DimeSage boasts a responsive design that adapts seamlessly to various screen sizes and resolutions. This ensures that users experience consistent functionality and user interfaces across devices.

### 2.5.3 Security and Data Privacy

- **Secure Browsing Practices:** Users are educated about secure browsing practices, including the importance of HTTPS connections and recognizing phishing attempts. These guidelines help safeguard sensitive financial information.
- **Data Encryption:** The application employs robust encryption protocols to protect user data during transmission and storage. This ensures the confidentiality and integrity of financial records
- **Authentication Mechanisms:** Users are required to create and maintain strong passwords for their accounts. Multi-factor authentication (MFA) may be available as an additional security layer to protect user accounts.

## 2.6 Justification of Platform

### 2.6.1 Why Web App

In the realm of modern software development, selecting the right platform is a critical decision that significantly influences the success and performance of an application. For the development and deployment of DimeSage, a web application was meticulously chosen as the optimal platform. This chapter delves into the rationale behind this pivotal decision, elucidating the advantages and considerations that make web application development the ideal choice.

#### 2.6.1.1 Universal Accessibility and Cross-Platform Compatibility

Web applications offer unparalleled accessibility to users across the globe. Unlike native applications, which are often limited to specific operating systems, web apps can be accessed from a multitude of devices and browsers. This universality ensures that users can manage their expenses seamlessly, regardless of their chosen device or platform.

#### 2.6.1.2 Cost-Efficiency and Scalability

Developing a web application aligns with prudent financial considerations. The absence of the need for multiple platform-specific development teams and the ability to roll out updates effortlessly reduce development costs. Moreover, web applications inherently possess scalability, accommodating an expanding user base without the need for extensive reengineering.

#### 2.6.1.3 Instant Updates and Maintenance

One of the distinctive advantages of web applications is the capacity for real-time updates and maintenance. When enhancements or bug fixes are required, they can be implemented swiftly and pushed to all users simultaneously. This ensures that users are always provided with an up-to-date and secure platform.

#### 2.6.1.4. Device-Agnostic User Experience

Web applications are designed to be device-agnostic, adapting to various screen sizes and resolutions. This adaptability ensures a consistent and user-friendly experience for individuals accessing the application from desktop computers, laptops, tablets, or smartphones. This enables the user to just hop on the website and then start the application wherever he or she wants to. This also removes the restriction from the user to only access the app on that particular platform which is a issue when tackling apps which are a part of the daily lives of the users

#### 2.6.1.5 Seamless Integration and API Utilization

The web application platform readily facilitates integration with external services, APIs, and databases. DimeSage leverages this capability to connect with financial institutions, payment gateways, and other relevant systems, offering users a comprehensive and holistic financial management experience.

#### 2.6.1.6 Rigorous Security Protocols

Security is paramount in financial management applications. Web application development allows for the implementation of robust security measures, including data encryption, authentication mechanisms, and protection against common web vulnerabilities. These safeguards are essential in safeguarding sensitive financial data.

#### 2.6.1.7 User-Friendly Updates

Web applications enable the delivery of user-friendly updates. Users are not burdened with the task of manually updating the application; instead, they benefit from new features and improvements seamlessly. This enhances user satisfaction and engagement.

#### 2.6.1.8 Analytical Insights and User Tracking

Web applications provide developers with valuable insights into user behaviors and interactions. DimeSage utilizes analytics to understand user preferences, optimize the application's performance, and offer tailored recommendations for financial management.

#### 2.6.1.9 Cloud Hosting and Data Backup

Web applications are ideally suited for cloud hosting, ensuring data availability and reliability. The application's financial data is securely backed up, minimizing the risk of data loss and enhancing disaster recovery capabilities.

#### 2.6.1.10 Adherence to Industry Standards

The development of DimeSage on the web application platform ensures adherence to industry standards and best practices. Compliance with web accessibility guidelines and data protection regulations promotes inclusivity and trust among users.

### 2.6.2 Choosing the Tech-Stack

The technological foundation of any software project is a cornerstone decision that shapes its development, performance, and overall success. In the context of DimeSage Expense Tracker, the careful selection of the technology stack serves as a pivotal step in realizing a robust, secure, and user-friendly financial management application. This chapter embarks on an exploration of the strategic considerations and rationale underpinning the choice of the tech stack for this innovative web application. From the programming languages to frameworks, libraries, and databases, every component plays a vital role in achieving the project's objectives. This chapter presents a comprehensive overview of the factors influencing the selection of each element within the technology stack, setting the stage for an in-depth exploration of the chosen tools and technologies.

#### 2.6.2.1 Flask

Flask, a lightweight and versatile Python web framework, has gained widespread popularity for its simplicity and flexibility in building web applications. While not as feature-rich as some heavyweight frameworks, Flask's minimalistic design and extensive ecosystem of extensions make it an excellent choice for web developers.

- **Key Benefits**
    a. **Simplicity and Minimalism:**
        - Flask's core philosophy is to keep things simple and minimal. It provides just the essentials needed to get a web application up and running. This simplicity allows developers to have greater control over the components they use.

    b. **Flexible and Extensible:**
        - Flask is known for its extensibility. It allows developers to choose the libraries and tools they need, giving them the freedom to build the application their way. Flask extensions cover a wide range of functionalities, from authentication to database integration.

    c. **Built-in Development Server:**
        - Flask includes a built-in development server, making it easy for developers to test and debug their applications during development. This server is efficient and convenient for local development.

    d. **Jinja2 Templating Engine:**
        - Flask employs the Jinja2 templating engine, which simplifies the generation of dynamic HTML content. It separates the presentation layer from the application logic, making templates highly customizable.

    e. **RESTful Support:**
        - Flask is well-suited for building RESTful APIs. Its simplicity and support for HTTP methods make it an excellent choice for creating web services and APIs.

    f. **Large and Active Community:**

- Flask has a thriving community of developers and enthusiasts who contribute to its ecosystem. This community support results in a wealth of extensions, tutorials, and resources available for Flask users.

g. **Werkzeug and WSGI Compliance:**
   - Flask is built on top of the Werkzeug WSGI toolkit, ensuring compliance with the Web Server Gateway Interface (WSGI) standard. This compatibility allows Flask applications to run on various web servers seamlessly.

h. **Microservices-Friendly:**
   - Flask's microframework nature makes it suitable for developing microservices architectures. It can be an excellent choice when building a complex, distributed system composed of loosely-coupled services.

### 2.6.2.2 HTML , CSS & JavaScript

HTML,CSS and JavaScript are and for the most part will be the base of making anything which will be related to websites. HTML forms the foundational structure of websites, acting as the canvas on which web content is created. This static framework is then brought to life through the dynamic duo of CSS and JavaScript. CSS, or Cascading Style Sheets, takes charge of the presentation, formatting, and layout, transforming the raw HTML into visually appealing and well-structured web pages. It defines how elements should be displayed, ensuring a harmonious visual experience for users. On the other hand, JavaScript takes the helm in controlling the behavior of various elements on the web page. It introduces interactivity and responsiveness, enabling features like real-time updates, data validation, and complex user interactions. Together, HTML, CSS, and JavaScript orchestrate the symphony of the web, crafting engaging, visually appealing, and highly functional user interfaces.

- **Key Benefits**

a. **Consistency across multiple browsers**
   - Not all browsers support all web pages or web applications, however, the implementation of HTML5 and CSS3 helps the designer to create a compatible site or system within all browsers.

b. **A better user experience**
   - HTML5 offers a wider range of design and presentation tools across media types, giving the developers greater scope to produce better web sites and web applications. This is vital from a business point of view, as user engagement and retention is key to increased site and system use and conversion. Creating an accessible and usable site or system means that users will be more likely to engage.

### 2.3.2.4 Bootstrap

Bootstrap is a popular front-end framework for building responsive, mobile-first web applications. It is an open-source project developed by Twitter and is now maintained by a community of developers. Bootstrap's pre-built components and templates allow developers to quickly build responsive web applications without having to write as much custom code. This can significantly speed up the development process.

- **Key Benefits**
  a. **Responsive design:**
     - Bootstrap provides a mobile-first design approach that makes it easy to build websites that look great on any device, from desktops to smartphones.

  b. **Pre-built UI components:**
     - Bootstrap includes a set of pre-built UI components, such as forms, buttons, tables, and navigation menus, that make it easy to create a consistent and professional-looking website.

  c. **Extensive documentation:**
     - Bootstrap has extensive documentation that provides clear and concise guidance on how to use its features, making it easy for developers of all skill levels to get started**.**
     - The documentation website makes everything very easy.

  d. **Customizable:**
     - Bootstrap is highly customizable and allows developers to modify the look and feel of their website using CSS and JavaScript.

  e. **Large developer community:**
     - Bootstrap has a large and active developer community that creates themes, plugins, and other tools to extend its functionality and make it easier to use.

### 2.3.2.6 VsCode

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 Build conference.

A preview build was released shortly thereafter. On November 18, 2015, the source of Visual Studio Code was released under the MIT License and made available on Github. Extension support was also announced. On April 14, 2016, Visual Studio Code graduated from the public preview stage and was released to the Web.Microsoft has released most of Visual Studio Code's source code on Github under the permissive MIT License, while the releases by Microsoft are proprietary freeware

- **Key Features**
    a. **Cross-Platform support**
        - VS Code is available for Windows, Linux, and macOS, making it a versatile tool for developers working on different platforms

    b. **Lightning and Fast**
        - VS Code is a lightweight editor that starts up quickly and consumes minimal system resources. This makes it a fast and efficient tool for coding and debugging

    c. **Integrated terminal:**
        - VS Code comes with an integrated terminal that allows developers to execute command-line operations without leaving the editor. This can save time and make it easier to switch between different tasks

    d. **Large extension ecosystem:**
        - VS Code has a large extension ecosystem that includes thousands of extensions developed by the community. These extensions provide additional functionality, such as language support, debugging tools, and code snippets.

    e. **IntelliSense:**
        - VS Code's IntelliSense feature provides context-aware code completion, code navigation, and other helpful suggestions as you type. This can help reduce errors and make coding more efficient

# SYSTEM DESIGN

The "System Design" chapter serves as a pivotal phase in the development of the proposed expense tracker application. This chapter focuses on translating the requirements and specifications outlined in the previous chapters into a comprehensive and actionable system design. It lays the foundation for the actual development and implementation stages of the project.

## 3.1 Module Division

Module division plays a pivotal role in structuring the project into manageable and comprehensible components, facilitating a clearer understanding for readers. This division dissects the project into distinct modules, each with its specific purpose and functionality, ensuring that readers can delve into the intricacies of the project with clarity and ease.

Through the module division, readers are provided with a comprehensive explanation of each module's intricacies, functionality, and interaction within the broader project framework. This approach enables readers to navigate through the project, gaining insights into how individual modules contribute to the project's overall objectives.

### 3.1.1 AI Modules

AI modules represent the heart of this project, aiming to address the limitations and challenges associated with existing expense management systems. These AI-driven modules have been meticulously designed to enhance the user experience and provide innovative solutions to common issues faced by users when managing their finances. By harnessing the power of artificial intelligence and machine learning, these modules offer personalized, insightful, and efficient ways for users to achieve better financial control and make informed decisions

#### 3.1.1.1 AI - Driven Budgeting | AI

The AI-driven budgeting feature stands as the backbone of our financial management system, offering users a meticulously personalized approach to maintaining their financial well-being. This feature's strength lies in its ability to conduct a comprehensive analysis of each user's unique financial circumstances, making it an invaluable tool for achieving financial goals and prudent spending practices.

**User Tailored Financial Analysis**

- AI-driven budgeting module begins by conducting an in-depth assessment of the user's financial landscape. This examination encompasses crucial aspects such as the user's monthly income, unalterable expenses (like rent or mortgage payments), and any established financial objectives. This user-centric analysis serves as the bedrock for the creation of a customized budgeting framework.

**Category-Specific Budgets**

- One of the hallmark features of this module is its capability to formulate personalized monthly budgets for each spending category. This means that users receive tailored spending limits and saving goals for various aspects of their financial lives, such as groceries, dining out, entertainment, or transportation.

**Practical Money Saving**

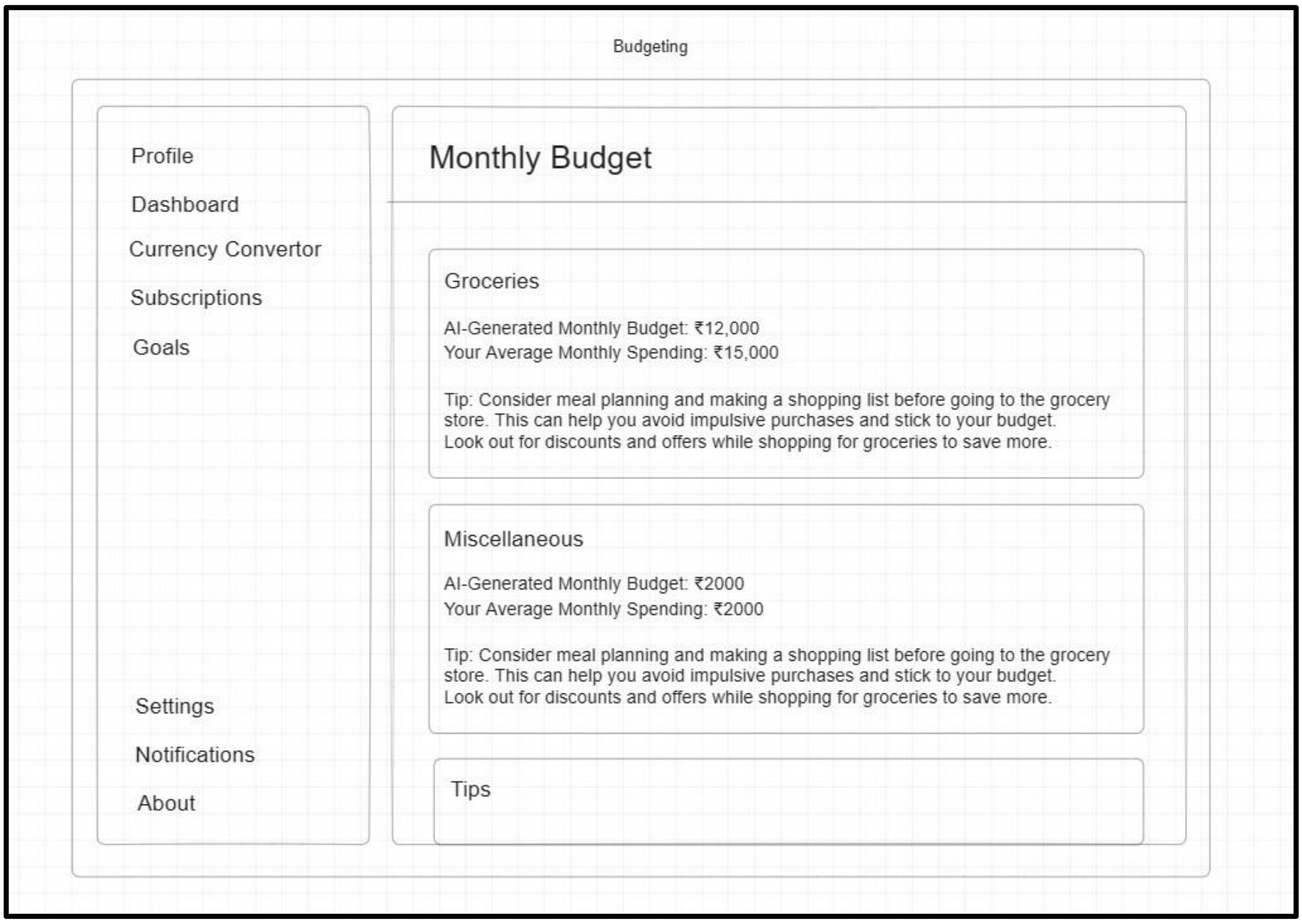
- Beyond the creation of personalized budgets, our system takes a step further by providing practical tips and insights to help users save money in specific spending categories. These tips are grounded in real-time financial data and AI-driven analysis.
- Following table provides an easy example where the AI provides a budget for the user for the categories like *Electronics* and *Grocery* which are the most common category of expenses for a user may he or she be a Student or a Parent.

**Category: Electronics**

| | |
|---|---|
| **Users Monthly Expenditure** | ₹15,000 |
| **AI Generated Budget** | ₹10,000 |
| **Tip** | To stay within budget, evaluate your electronics purchases carefully. Consider waiting for sales events or discounts before making major electronic acquisitions. Additionally, explore the option of refurbished or pre-owned devices, which can offer substantial savings without compromising quality. |

**Category: Groceries**

| | |
|---|---|
| **Users Monthly Expenditure** | ₹4,500 |
| **AI Generated Budget** | ₹4,000 |
| **Tip** | Optimize your grocery shopping by planning meals ahead of time and creating shopping lists. Stick to buying essential items and avoid impulsive purchases. Buying in bulk for non-perishable items can also yield long-term savings on your grocery bill. |



Groceries

AI-Generated Monthly Budget: ₹12,000
Your Average Monthly Spending: ₹15,000

Tip: Consider meal planning and making a shopping list before going to the grocery store. This can help you avoid impulsive purchases and stick to your budget.
Look out for discounts and offers while shopping for groceries to save more.

Miscellaneous

AI-Generated Monthly Budget: ₹2000
Your Average Monthly Spending: ₹2000

Tip: Consider meal planning and making a shopping list before going to the grocery store. This can help you avoid impulsive purchases and stick to your budget.
Look out for discounts and offers while shopping for groceries to save more.

Tips

*Figure 3 :* ***Base Design for AI Driven Budget Page***

### 3.1.1.2 Smart Expense Analysis | AI

The AI-driven Smart Expense Analysis module is a pivotal component of the DimeSage application, designed to provide users with a comprehensive analysis of their financial expenditures. Leveraging advanced artificial intelligence and machine learning techniques, this module offers real-time insights into users' spending patterns, budget adherence, and overall financial health.

**Expense Categorization and Tracking**

- The Expenses are automatically categorized and then are used to track users spending habits.
- **Example**
    a. For instance, in the "Dining Out" category, it might be noticed that John's dining expenses tend to increase during the holiday season (November and December) and summer months (June and July). However, he manages to spend less during the beginning of the year (January to March) and towards the end of the year (September to December).

**Budget Adherence Monitoring**

- Monitors Expneses against predefined budgets which may be AI generated or User generated not only this but this also considers the Goals which the user has set for various reasons and then compares it to the spending habits of the user.
- **Example**
    a. If John has set a budget of ₹4,000 per month for dining out and his AI-generated monthly budget is ₹3,500, the system will inform him of the positive deviation, highlighting that he's saving ₹500 per month in this category.

**Financial Health Score Calculation**

- The **Budget Monitoring** and **Expense Tracking** make this feature possible by generating financial scores for the user. It works on mainly 2 things which are
    a. If the Expenditure of the user is according to the Budgets which are provided to them or created by the user itself
    b. If the user is able to save something for the Goals which they have set.
- The system calculates and provides users with a financial health score. This score summarizes their overall financial well-being based on factors such as income, expenses, savings, and adherence to budget goals.
- This is all done using an algorithm which combines 4 main things and observation to create a final score for the users. This algorithm will learn as the database expands and grows over time, Not only this but the budget reading of the AI will also get better as more users use this application
- The Algorithm which will be used for calculating this is as follows
    **a. Income vs Expense Ratio (50%) :** Calculate the ratio of a user's total monthly income to their total monthly expenses. A higher ratio indicates better financial health.
        - **Excellent:** Ratio > 2.0
        - **Good:** Ratio between 1.0 and 2.0
        - **Fair:** Ratio between 0.5 and 1.0
        - **Poor:** Ratio < 0.5

    **b. Saving Rate (30%) :** Calculate the percentage of a user's income that they save each month, including contributions towards their goal funds. A higher savings rate contributes positively to their score.
        - **Excellent:** Savings Rate > 20%
        - **Good:** Savings Rate between 10% and 20%
        - **Fair:** Savings Rate between 5% and 10%

- **Poor:** Savings Rate < 5%

c. **Budget Adherence (20%) :** Assess how well a user sticks to their budget for various expense categories. users who consistently stay within their budget receive a higher score
   - **Excellent:** Adherence > 90%
   - **Good:** Adherence between 70% and 90%
   - **Fair:** Adherence between 50% and 70%
   - **Poor:** Adherence < 50%

d. **Goal Funding (10%) :** Consider whether the user has set and achieved their financial goals. Users who successfully save towards their goal receive a higher score
   - **Excellent:** All goals achieved
   - **Good:** More than 50% of goals achieved
   - **Fair:** Less than 50% of goals achieved
   - **Poor:** No goals set or achieved

e. **Working of this Algorithm :**
   - **User Profile**

| | |
|---|---|
| **Monthly Income** | ₹2,80,000 |
| **Monthly Expenses** | ₹1,75,000 |
| **Monthly Savings** | ₹84,000 |
| **Budget Adherence** | 85% |
| **Financial Goals** | Goal 1: Vacation Fund - Target Amount: ₹1,40,000, Monthly Contribution: ₹10,500<br><br>Goal 2: Emergency Fund - Target Amount: ₹3,50,000, Monthly Contribution: ₹7,000<br><br>Goal 3: New Laptop Fund - Target Amount: ₹70,000, Monthly Contribution: ₹5,250<br><br>Goal 4: Retirement Fund - Target Amount: ₹35,00,000, Monthly Contribution: ₹17,500 |

- **Calculations**

| | |
|---|---|
| **Income vs Expenses \| 50%** | Income vs. Expenses Ratio = ₹2,80,000 / ₹1,75,000 = 1.6 (Good) |
| **Savings Rate \| 30%** | Savings Rate = (₹84,000 / ₹2,80,000) * 100% = 30% (Excellent) |
| **Budget Adherence \| 20%** | Budget Adherence = 85% (Good) |
| **Goal Fund Savings \| 10%** | Goals Achieved: Goal 1 (Vacation Fund) and Goal 2 (Emergency Fund) have been achieved.<br><br>Goals In Progress: Goal 3 (New Laptop Fund) and Goal 4 (Retirement Fund) are in progress. |

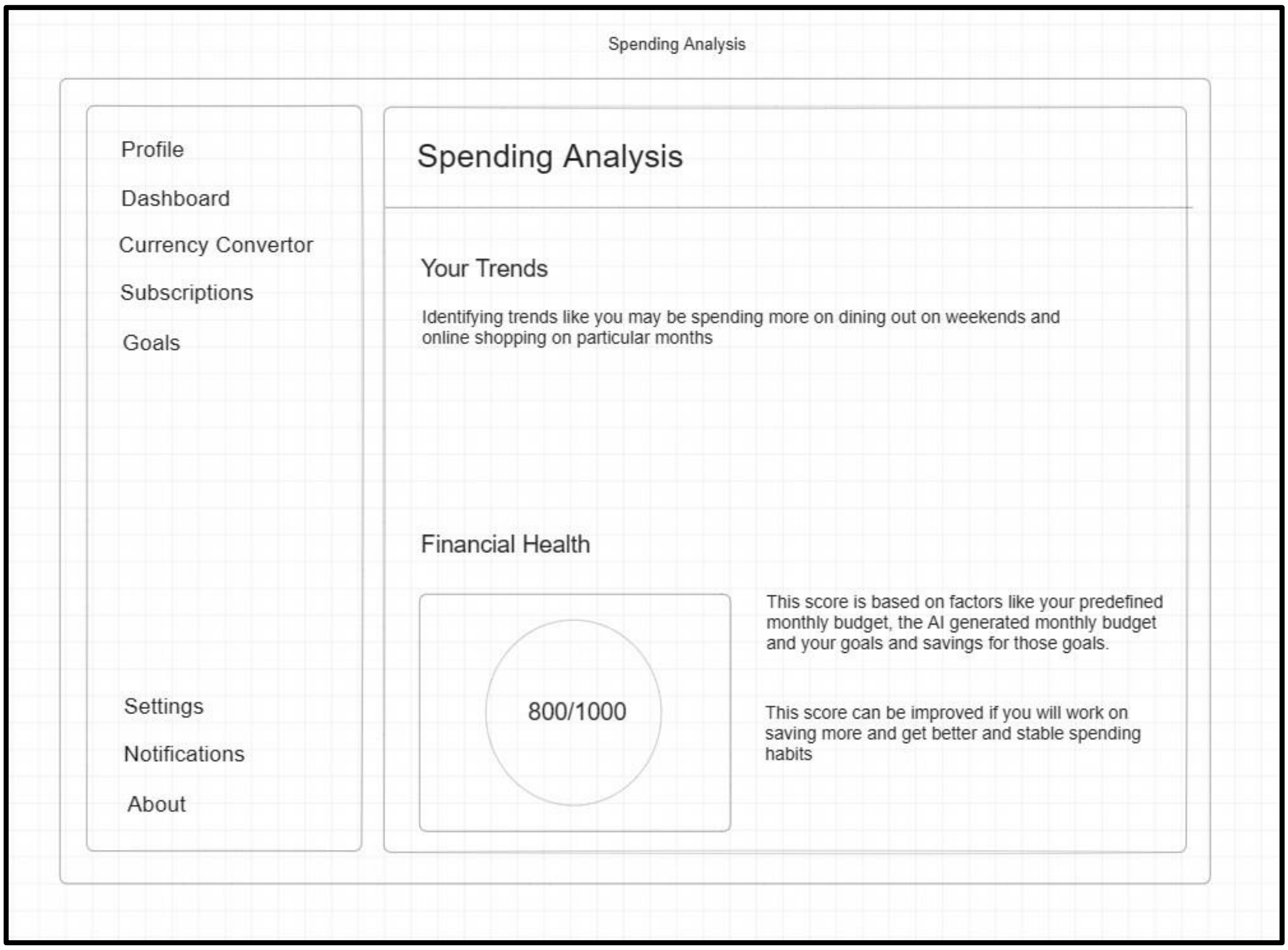
- **Financial Health Score**

| | |
|---|---|
| **Income vs Expense Ratio** | Good (Score: 8/10) |
| **Saving Rate** | Excellent (Score: 9/10) |
| **Budget Adherence** | Excellent (Score: 9/10) |
| **Goal Fund Saving** | Good (Score: 8/10) |

- **Total Score Calculation**

  Total Score = (0.5 * 8) + (0.3 * 9) + (0.2 * 8) + (0.1 * 9) = 4 + 2.7 + 1.6 + 0.9 = 9.2

- **Financial Health Level -** Based on this the financial score the user will get is 9.2 and the user falls into “Good” financial health level.
- **Summary :** This user demonstrates good financial health. They have a balanced income-to-expenses ratio, save a significant portion of their income, and adhere reasonably well to their budget. Additionally, they have successfully achieved two financial goals (Vacation Fund and Emergency Fund), contributing to their overall financial health score.



Identifying trends like you may be spending more on dining out on weekends and online shopping on particular months

This score is based on factors like your predefined monthly budget, the AI generated monthly budget and your goals and savings for those goals.

This score can be improved if you will work on saving more and get better and stable spending habits

*Figure 4 : **Basic Design for Spending Analysis window***

### 3.1.2 NLP Modules

As previously highlighted, traditional expense management systems tend to employ interfaces reliant on widgets, which can be both inefficient and time-consuming for users. In our pursuit of an improved user experience, we have devised a more user-centric approach. We enable users to effortlessly input data by typing in a natural language format, thus eliminating the need for cumbersome widgets. This innovative method not only enhances efficiency but also significantly simplifies the data entry process. The underlying AI technology then takes over, making the entire procedure smoother and more user-friendly.

This works just like any other chatbot and lets you enter the expenses seamlessly. Following is a bit more detailed explanation.

#### 3.1.2.1 Chatbot for recording expenses

The chatbot is designed to engage users in a conversational manner, making the process of recording expenses more intuitive and user-friendly. Users can interact with the chatbot just as they would with a human conversation. For instance, they can provide expense details by simply typing or speaking in natural language sentences. For example, a user might input, "I spent $50 on groceries yesterday."

**User Inputs**

- The User initiates a conversation with the chatbot and provides an expense-related statement or query.

**NLP Parsing**

- The chatbot utilizes Natural Language Processing (NLP) techniques to parse the user's input. NLP algorithms analyze the text to understand the context, intent, and entities within the sentence. In our example, the chatbot identifies that the user is talking about an expense, extracts the amount ($50), and recognizes the expense category (groceries) and the date (yesterday).
- This is the main feature of this module and helps the users to increase their efficiency a lot.

**Data Validation**

- The chatbot validates the extracted information to ensure accuracy and completeness. It checks if the amount is in a valid format, the category exists in the database, and the date is within an acceptable range.

**Recording the Expenses**

- Once the data is validated, the chatbot records the expense in the database. It creates a new entry that includes the amount, description, category, and date. This entry becomes a part of the user's financial records.

**User Confirmation**

- The chatbot responds to the user, confirming that the expense has been successfully recorded. It may also provide a summary of the expense for the user's reference.

## 3.1.3 User Management Modules

In our User Management Modules, we've adopted a user-first approach, simplifying the user experience. Instead of navigating complex interfaces, users can manage their accounts effortlessly by typing natural language commands. This chatbot-like interaction allows users to perform tasks like resetting passwords, updating information, or checking balances in a conversational manner. Our system interprets these commands and executes actions promptly, streamlining user management and enhancing the overall experience.

#### 3.1.3.1 Login Module

Users have the option to register new accounts by providing basic details such as their email, chosen username, and password. Once registered, users can securely log in using their credentials, gaining access to their personalized financial data and a range of features tailored to their needs.

#### 3.1.3.2 Setting up new Users

- Users will input their monthly income, providing the system with a clear understanding of their financial resources.
- Users will enter their fixed expenses, including items like rent and Wi-Fi bills, ensuring accurate financial tracking.
- Users can specify their monthly subscriptions, such as Netflix and Spotify, helping the system manage recurring expenses effectively.

#### 3.1.3.3 User Settings

The user management module offers a comprehensive set of settings to enhance the user experience:

- Currency Preferences: Users can customize their currency settings to match their financial context.
- Fixed Expenses: Users have the flexibility to update their fixed expenses, accommodating changes in their financial commitments.
- Password Management: Users can securely manage their passwords and make changes as needed.
- Income Updates: Users can conveniently update their income information whenever there is a change in their financial situation.

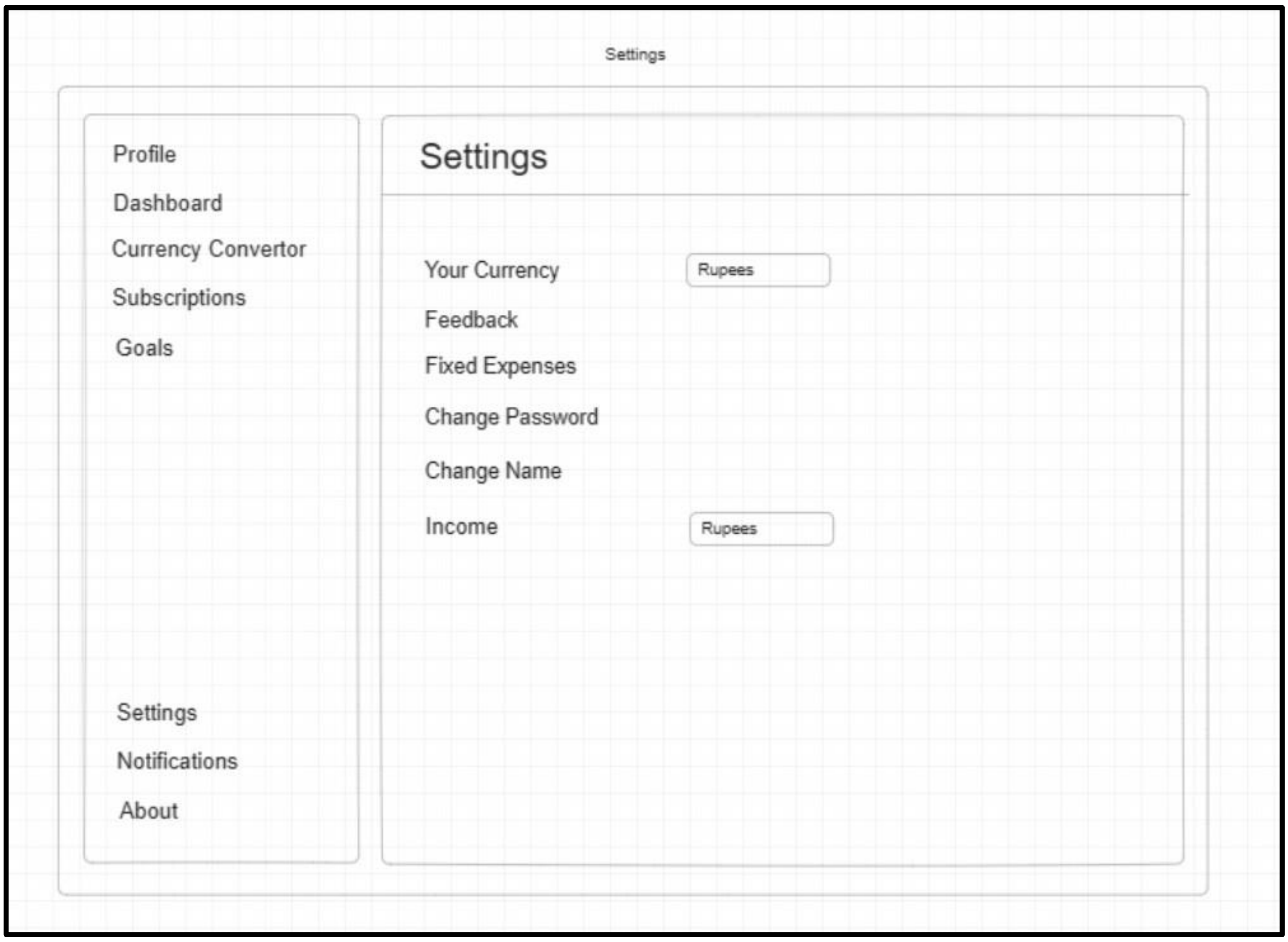


*Figure 5 :* ***Dummy design for Settings Page***

### 3.1.3.4 Profile Tab

- Name
- Occupation
- Income
- Age
- Financial Goals (only top 2)
- Spending Habits
- Fixed Expense

## 3.1.4 Feature Modules

Once the foundational elements of the application have been established and the user management module ensures a seamless and secure experience, we can delve into the myriad of features that this comprehensive financial tool brings to the table.

Our application isn't just a simple expense tracker; it's a financial companion that empowers users to take control of their monetary affairs. From budgeting and expense tracking to goal setting and smart analytics, each feature has been thoughtfully designed to provide users with valuable insights, improve their financial well-being, and simplify their financial management journey.

In the chapters that follow, we will explore these features in detail, unveiling how they work, why they matter, and how they can revolutionize the way individuals and organizations interact with their financial resources. So, let's embark on this journey through the world of financial empowerment and discover how our application can make managing your finances not just efficient, but also rewarding.

### 3.1.4.1 Main Dashboard

The main dashboard serves as the financial control center for users, offering a comprehensive snapshot of their monetary activities and financial health. It goes beyond merely displaying numbers; it paints a vivid picture of their financial landscape. At a glance, users can access vital information such as their total expenses, showcasing where their money has gone. It provides insights into income, shedding light on the inflow of funds and financial stability. Moreover, it doesn't stop at displaying numbers; it visualizes these data points, making it easier for users to comprehend complex financial information.

One of the standout features of the dashboard is its ability to summarize budget adherence. Users can see whether they are meeting their budgetary goals, and if not, where they might be overspending. This level of transparency empowers users to make informed decisions about their spending habits.

In essence, the main dashboard is not just a collection of financial data; it's a financial advisor, helping users understand their financial status, enabling them to make strategic financial choices, and ultimately paving the way towards a more secure and prosperous financial future.

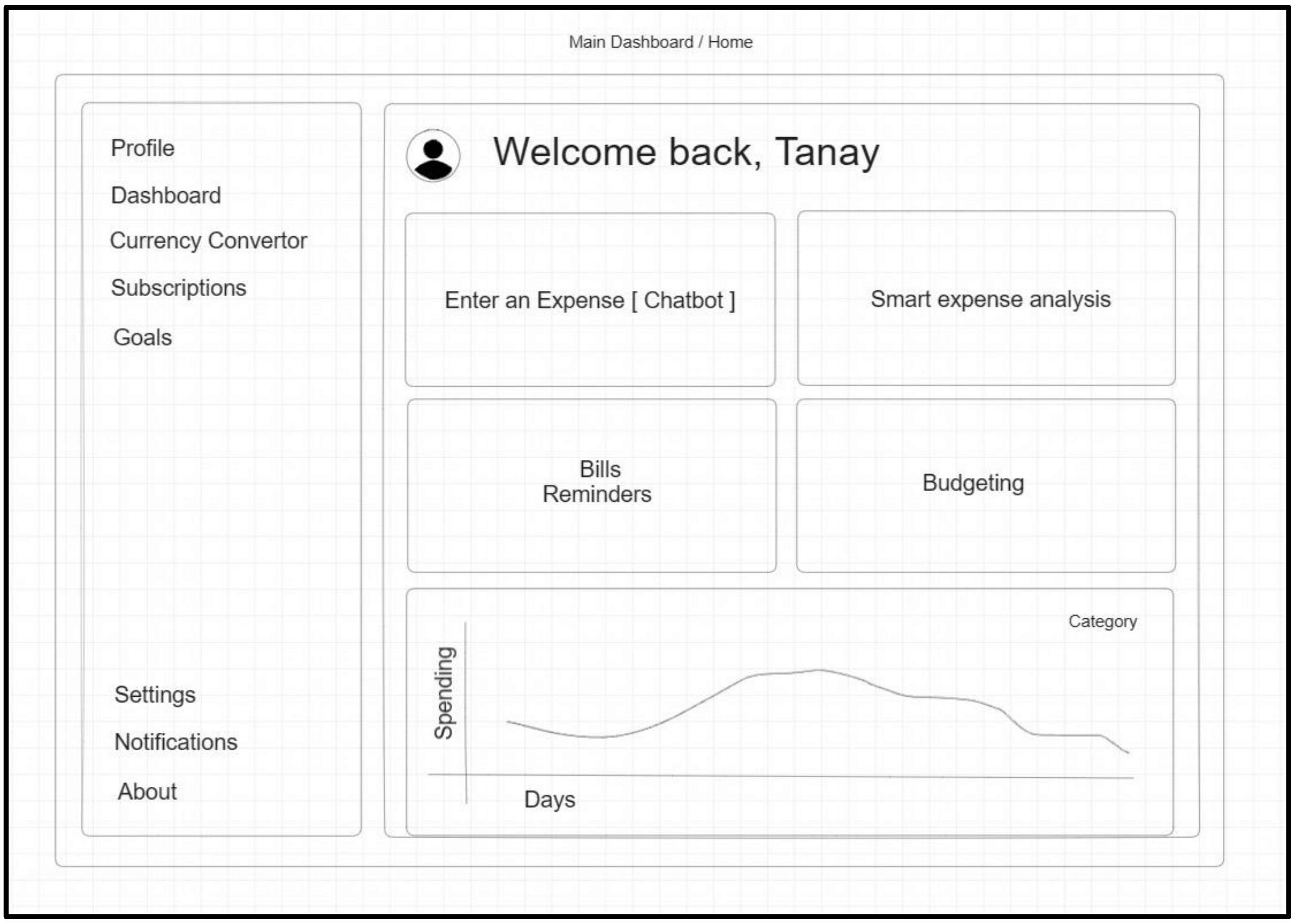


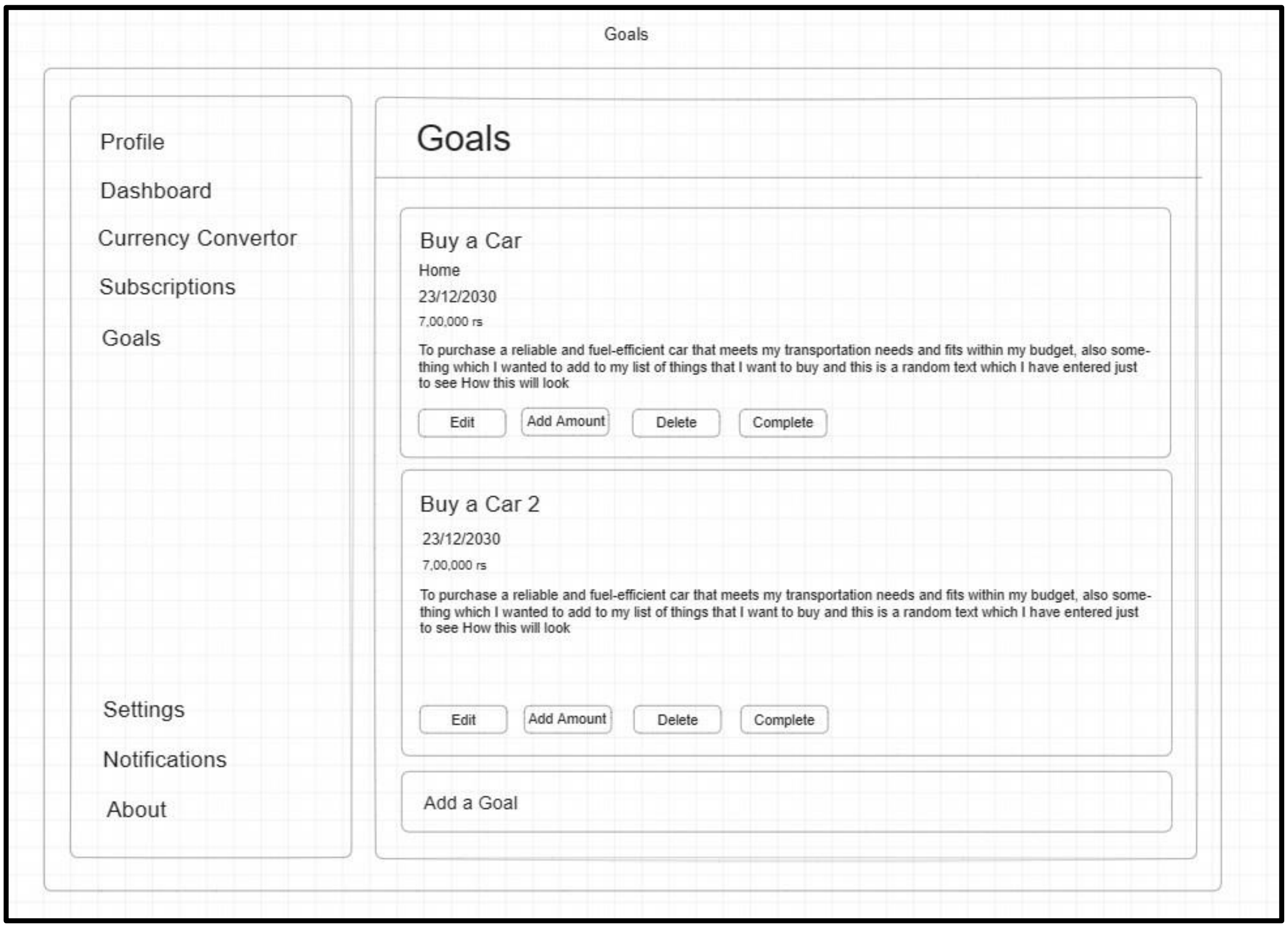
*Figure 6 : **Card Based design for Dash Board***

### 3.1.4.2 Goal Tracking Module

Within the application, users are not only equipped to manage their day-to-day expenses but are also empowered to plan for their future with precision and clarity. The goal-setting feature is a pivotal aspect of this financial journey.

Users can set a variety of financial goals, from short-term dreams like vacations or buying a car to long-term plans like retirement. This feature is versatile and accommodates various financial ambitions.

The process is user-friendly and intuitive. Users define their goal, set a target amount, and establish a timeline to reach it.



Buy a Car
Home
23/12/2030
7,00,000 rs
To purchase a reliable and fuel-efficient car that meets my transportation needs and fits within my budget, also something which I wanted to add to my list of things that I want to buy and this is a random text which I have entered just to see How this will look

Buy a Car 2
23/12/2030
7,00,000 rs
To purchase a reliable and fuel-efficient car that meets my transportation needs and fits within my budget, also something which I wanted to add to my list of things that I want to buy and this is a random text which I have entered just to see How this will look

*Figure 7 : **Dummy Design for Goals window***

The process is user-friendly and intuitive. Users begin by defining the specific goal they wish to achieve, whether it's a dream vacation, a down payment for a home, or securing their retirement. They set a target amount, determining the financial milestone they aim to reach. achieving these

### 3.1.4.3 Remaining Payment / Bills

#### Automated Bill Reminders

- Users can effortlessly input their various bills into the application, ranging from utilities and rent to credit card payments and subscriptions. They can specify the due dates for each bill, creating a comprehensive bill payment schedule within the app.

**Customizable Notifications**

- The system allows users to tailor their notification preferences. They can choose how many days in advance they'd like to receive reminders before a bill's due date, ensuring they have ample time to prepare and make payments promptly.

**Preventing Late Fees and Penalties**

- By receiving timely reminders and staying organized with their bills, users can avoid the costly pitfalls of late payments.

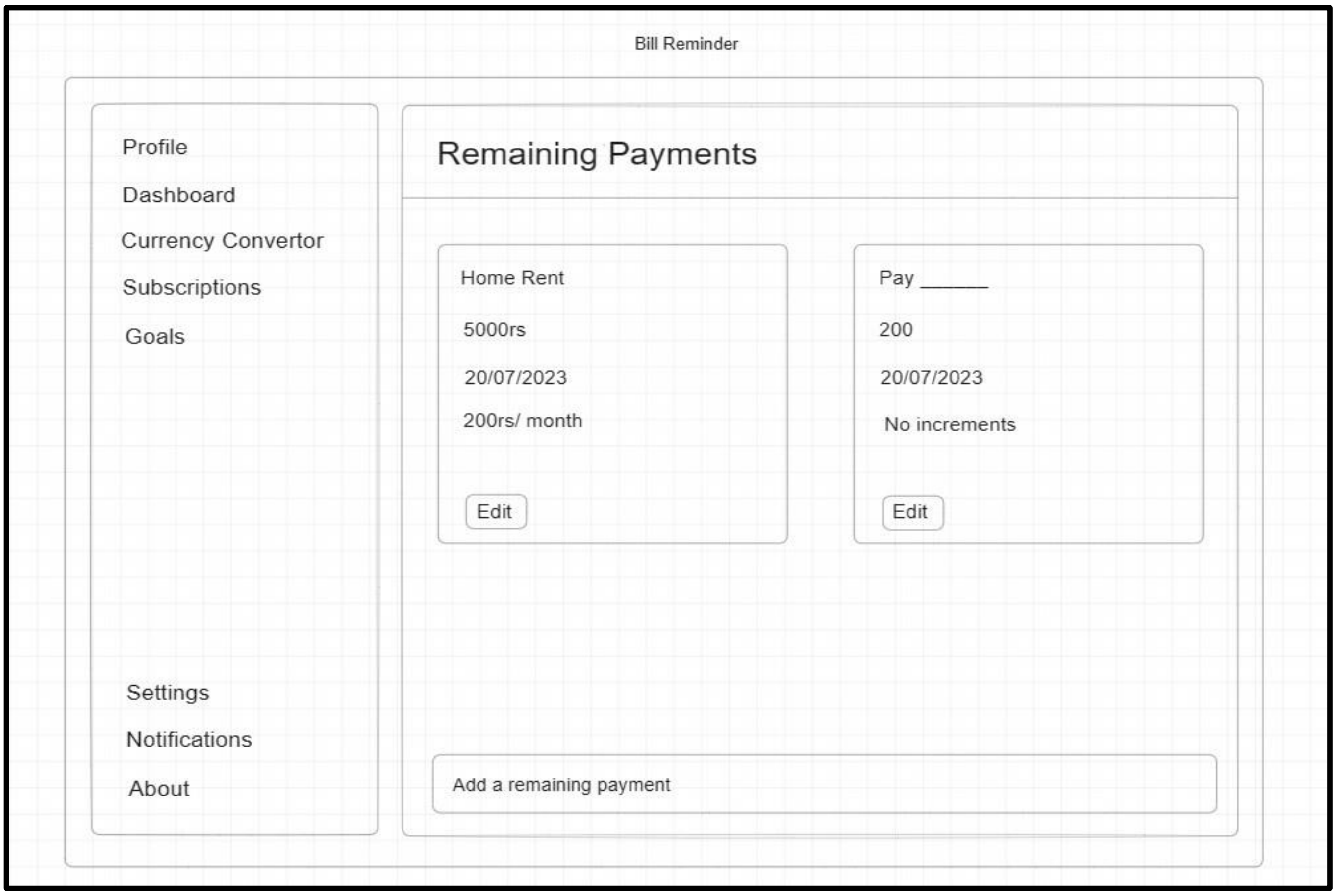


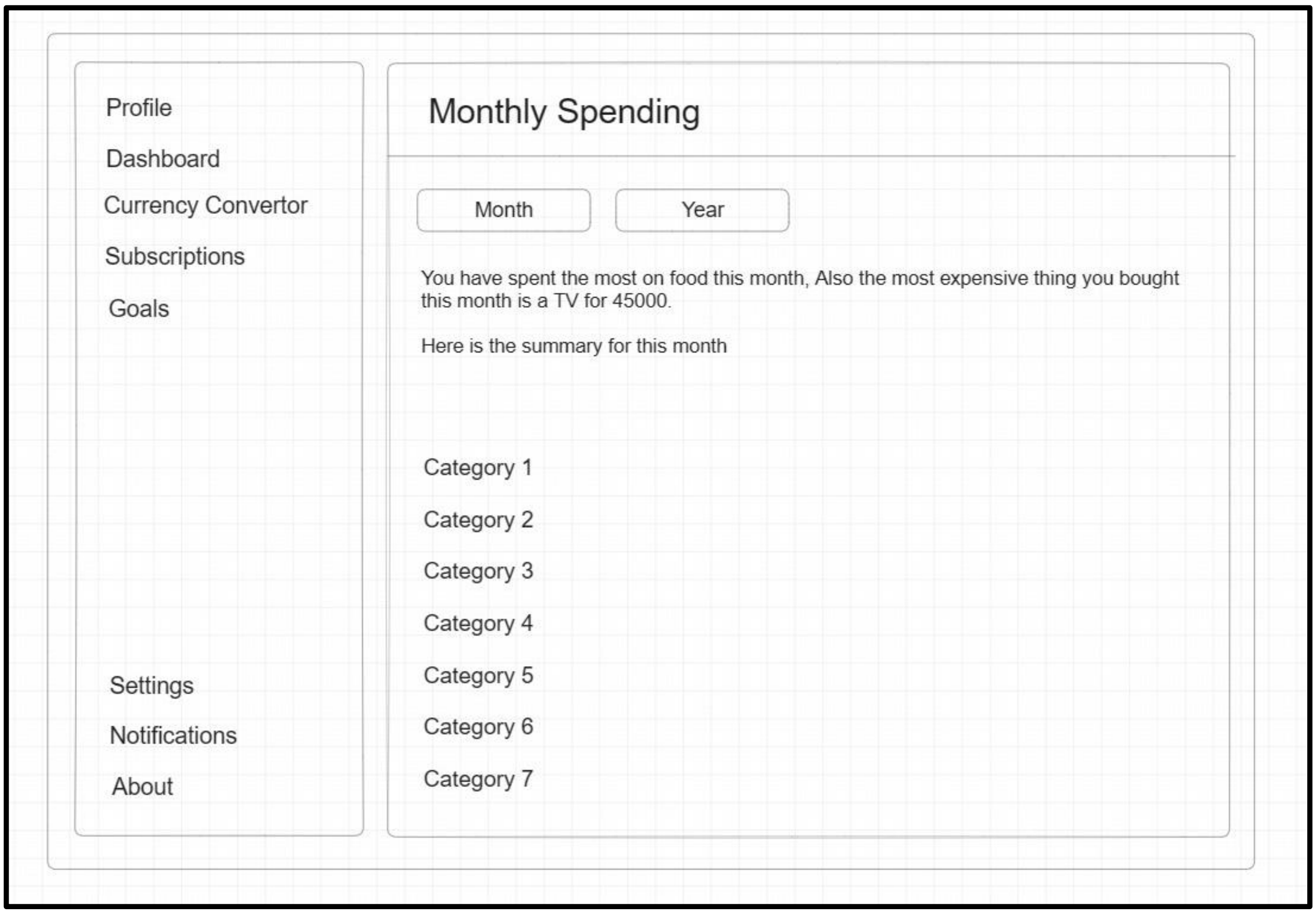
*Figure 8 : **Design idea for Bills / Remaining Payments***

**3.1.4.4 Monthly Expense Chart**

A Simple Chart which is present on the Dashboard which will show the expense trends for that particular month.

### 3.1.4.5 Overall Expense Tracking | By Month and Year

This module offers users an extensive overview of their expenses, providing the capability to sort expenses both by category and by month. Additionally, it offers valuable insights into spending habits, such as highlighting the highest expenditure for a specific category in a given month. Furthermore, it presents a summarized breakdown of expenses by category for the month, enhancing users' understanding of their financial activities.



You have spent the most on food this month, Also the most expensive thing you bought this month is a TV for 45000.

Here is the summary for this month

*Figure 9 : **Monthly spending page Design***

### 3.1.4.6 Subscription Tracking Module

Within the application, users can effortlessly manage their monthly subscriptions, which encompass a wide range of services like streaming platforms, online memberships, and digital subscriptions. By adding these subscriptions to the system, users gain a comprehensive view of their ongoing financial commitments. The application keeps a watchful eye on these recurring expenses, ensuring that users stay well-informed about upcoming payments.

One of the standout aspects of this feature is its ability to send timely reminders. As subscription due dates approach, the system proactively notifies users, helping them avoid accidental overdrafts or missed payments. This level of automation provides peace of mind and financial control.

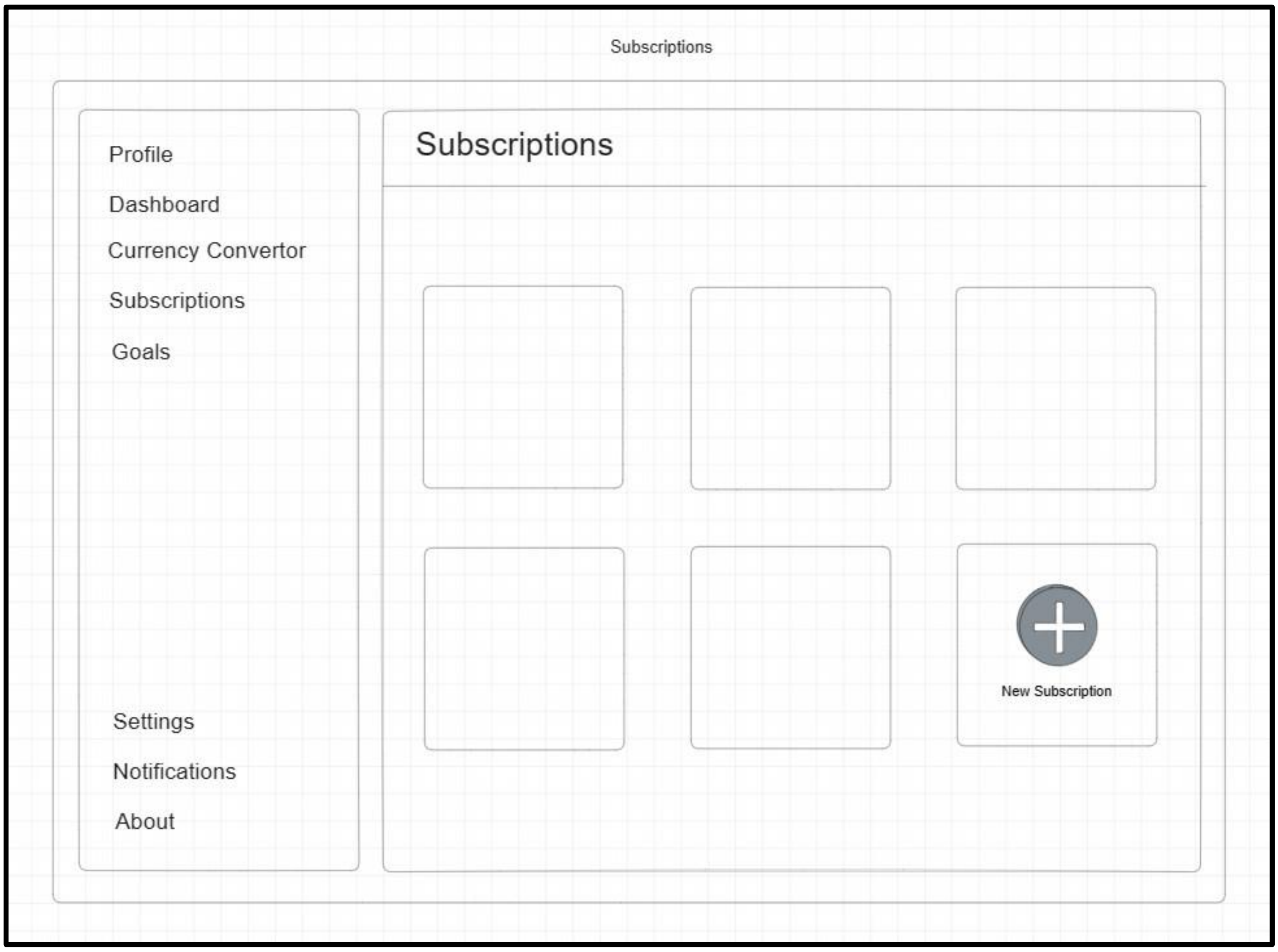


*Figure 10 :* ***Design Idea for Subscriptions***

### 3.1.4.7 Currency Conversion Module

This is a simple add-on to the app and no it is not groundbreaking but provides simple capabilities to the user and makes the app one stop shop. The currency conversion feature within the application offers users the convenience of handling finances involving different currencies effortlessly. Whether you're managing international expenses or traveling frequently, this feature ensures you always have accurate and up-to-date currency conversion at your fingertips.

Here's how it works: users can input an expense or income in one currency, and the application will swiftly and accurately convert it into the user's preferred or default currency. This functionality extends to the entire application, allowing for seamless and real-time currency conversion throughout.

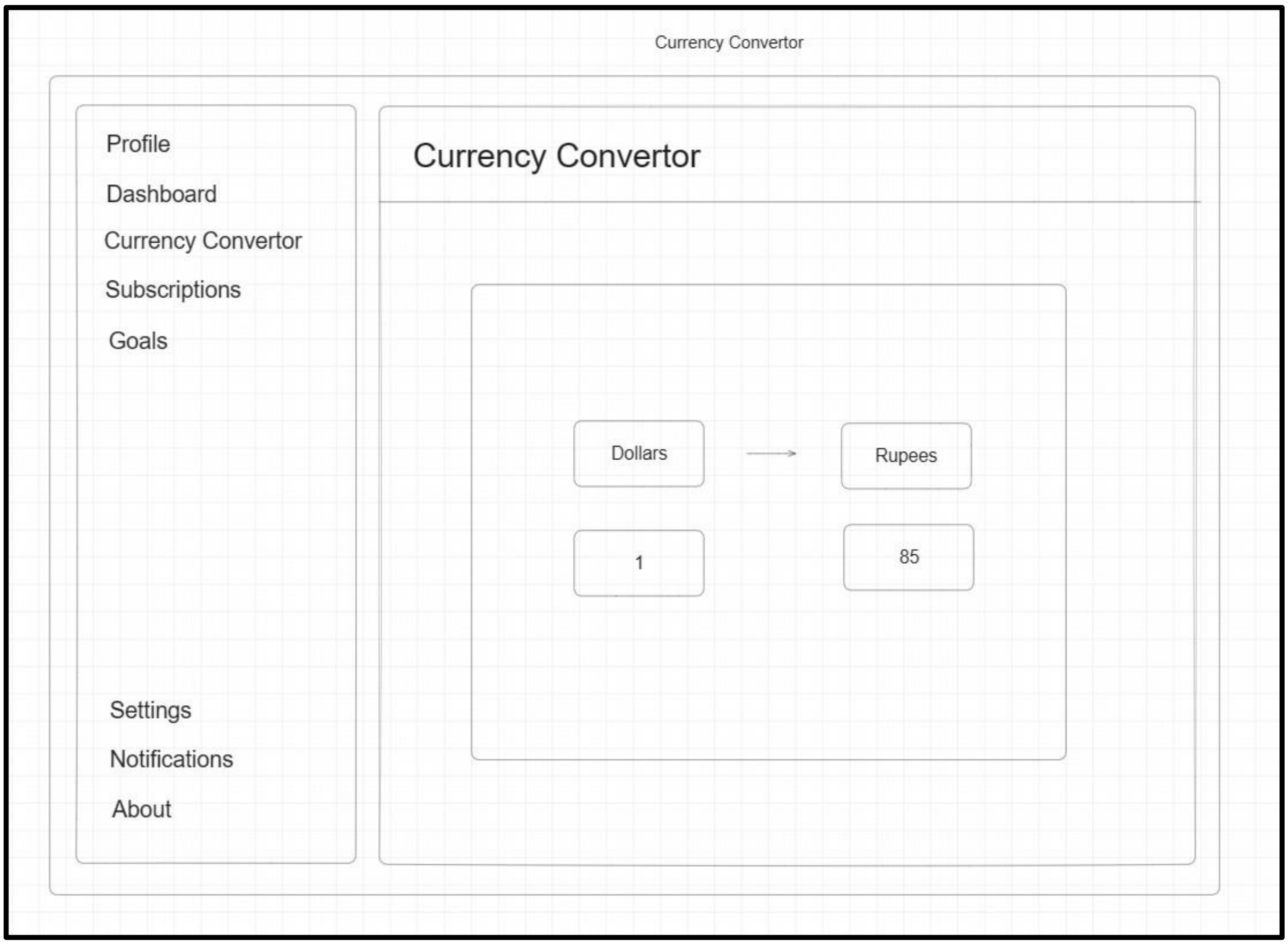


*Figure 11 : **Design idea for Currency Converter Module***

### 3.1.4.8 Notifications Module

Responsible for letting the user know the coming deadlines may it be for ***Subscription*** or ***Remaining Payments*** that the user is still to do.

## 3.2 Data Dictionary

The Data Dictionary serves as a comprehensive reference guide outlining the various data elements, their characteristics, relationships, and constraints within the DimeSage application. This critical resource ensures clarity and consistency in data management, facilitates system understanding, and aids in the seamless integration of data across modules.

### 3.2.1 User Information

- **Table Name :** User
- **Description :** Contains user-specific data , including login credentials, personal information and financial profile.
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| User_id | Unique identifier for each user | primary key |
| Username | User's chosen username for login. | varchar |
| Email | User's email address for communication. | varchar |
| Password | Securely stored user password. | varchar |
| Full Name | User's full name. | varchar |
| Age | User's age for demographic analysis. | int |
| Occupation | User's occupation or employment status. | varchar |
| Income | Monthly income of the user. | int |
| Financial Goals | User's defined financial goals. | varchar |
| Fixed Expenses | Recurring expenses like rent, utilities, etc. | varchar |
| Monthly Subscription | User's subscribed services (e.g., Netflix) | varchar |

*Table 2 : **User Database***

### 3.2.2 Expense Data

- **Table Name :** expenses
- **Description :** Stores details of users expenses , including amount, category, date and description.
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| Expense_Id | Unique Identifier for each expense | primary key |
| User_Id | Links expense to respective users | foreign key |
| Amount | Expense amount in the user's chosen currency | int |
| Category | Expense category | varchar |
| Description | Additional details about the expense | varchar |
| Date | Date of the expense transaction | date |

*Table 3 : **Expense Database***

### 3.2.3 Financial Goals

- **Table Name :** financial_goals
- **Description :** Records user-defined financial goals, including target amounts and completion dates.
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| Goal_Id | Unique identifier for each financial goal | Primary Key |
| User_Id | Associates goals with respective users | Foreign Key |
| Goal_Name | Descriptive name for the financial goal | varchar |
| Target_Amount | The amount the users aims to save | int |
| Completion_Date | The Target date for achieving the goal | date |
| Monthly_Saving | User-defined monthly contribution toward the goal | int |

*Table 4 : **Goals Database***

### 3.2.4 Subscription Management

- **Table Name :** subscriptions
- **Description :** This table will allow users to add their monthly subscriptions, specify the subscription name, cost, and due date. It ensures that users can track and receive reminders for subscription payments.
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| subscription_id | Unique Identifier for each subscription | primary key |
| user_id | Foreign key referencing the user who own the subscription | foreign key |
| subscription_name | Name or description of the subscription | varchar |
| subscription_amount | Monthly Cost of the subscription | int |
| subscription_due_date | Due date for subscription payment | date |

*Table 5 : **Subscription Database***

### 3.2.5 Currency Conversion rates

- **Table Name :** currency_conversion
- **Description :**
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| Currency_id | A unique identifier for each currency | Primary key |
| Currency_name | The name or code of the currency | varchar |
| Exchange_rate | The currency exchange rate relative to the systems default | int |

*Table 6 : **Currency conversion database***

This structure is straightforward and suitable for storing currency conversion rates. It's not a percentage; instead, it represents the exchange rate of one unit of the currency relative to the default currency (e.g., 1 USD = 0.85 EUR).

### 3.2.6 Remaining Expenses

- **Table Name :** remaining_expense
- **Description :** This table enables users to manage and track their remaining monthly expenses, such as utility bills, rent, or any other non-recurring expenses.
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| expense_id | Unique identifier for each remaining expense. | foreign key |
| user_id | Foreign key referencing the user who has the remaining expense. | foreign key |
| expense_name | Name or description of the remaining expense (e.g., "Electricity Bill"). | varchar |
| expense_amount | Amount of the remaining expense. | int |
| expense_due_date | Due date for the remaining expense payment. | date |

*Table 7 : **Remaining Payments and Bills***

### 3.2.7 Monthly Expenditure

- **Table Name : month_exp**
- **Description :** This table will contain the monthly total expenditure for a particular category for that particular month
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| User_id | Unique Identifier for the user (foreign key) | foreign key |
| record_id | primary key | primary key |
| month | Contains the month | varchar |
| year | Contains the year | varchar |
| category | Contains the category | varchar |
| expenditure | Total spent on that Category | int |
| ai_ budget | Contains the budget created by the AI | int |

*Table 8 : **Monthly Spending Database***

### 3.2.8 Fixed Expenses

- **Table Name : fixed_exp**
- **Description :** The table contains the fixed expenses on the user
- **Attributes**

| Column Name | Description | Type |
|---|---|---|
| User_id | Unique Identifier for the User | foreign key |
| expense_id | The Id for the expense | foreign key |
| expense | Description of the Expense | varchar |
| amount | The amount of expense | int |

*Table 9 : **Fixed Expenses Database***

## 3.3 Entity Relationship Diagram

An Entity Relationship (ER) Diagram is a visual representation used to depict how various "entities," which can be people, objects, or abstract concepts, are interconnected within a system. ER Diagrams find extensive application in designing and troubleshooting relational databases across domains like software engineering, business information systems, education, and research. These diagrams, also referred to as ERDs or ER Models, employ a set of standardized symbols such as rectangles, diamonds, ovals, and connecting lines to illustrate the relationships between entities, their attributes, and the connections between them. ER Diagrams often mirror grammatical structure, portraying entities as nouns and relationships as verbs. They serve a similar purpose to data structure diagrams (DSDs), which emphasize relationships within entities rather than between them. Additionally, ER Diagrams are frequently used alongside data flow diagrams (DFDs) to delineate information flow within processes or systems.

At its core, an ER diagram showcases the associations among sets of entities. Each entity set represents a grouping of similar entities, each with its own attributes. In the context of Database Management Systems (DBMS), an entity can correspond to a table or an attribute within a database table. By illustrating the connections between tables and their attributes, ER diagrams provide a comprehensive overview of a database's logical structure. They offer a visual blueprint for designing databases that can be subsequently implemented as functional database systems. The fundamental elements of an E-R model encompass entity sets and relationship sets.

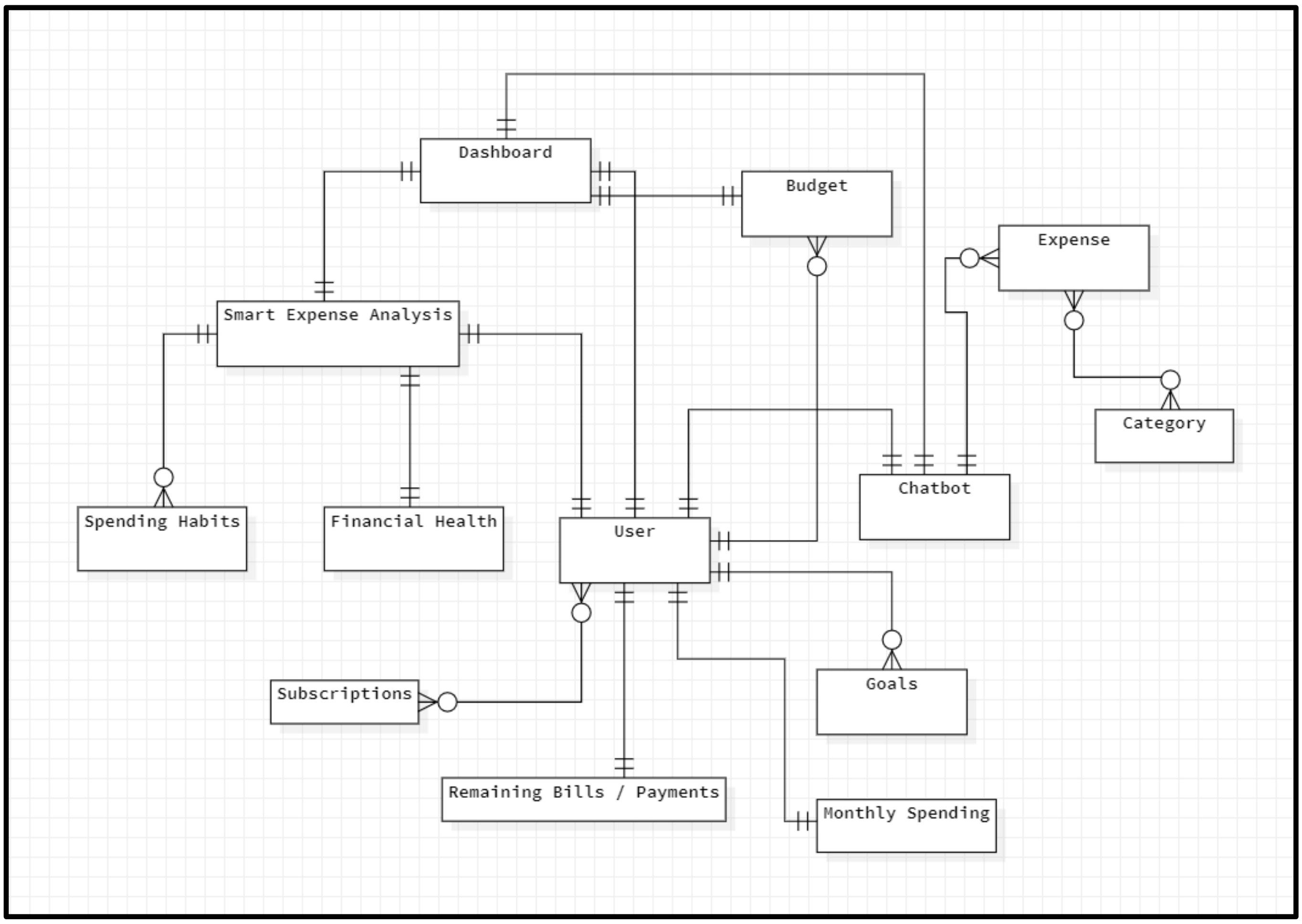


*Figure 12 : **Entity Relationship Diagram***

## 3.4 Logic Design

Logic design is a crucial component of a project dissertation in the field of software engineering and system development for several significant reasons:

**System Understanding and Specification**

- Logic design helps in defining and specifying the logical structure and behavior of a software system. This is essential in the early stages of a project to ensure that all stakeholders, including the researchers, project team, and users, have a clear understanding of the system's functionality and requirements. A well-defined logic design serves as a blueprint for the entire project.

**Error Detection and Debugging**

- Logic design enables researchers to identify potential errors and inconsistencies in the system's behavior before the implementation phase. This early detection of issues can significantly reduce the time and cost associated with debugging and maintenance during and after the project.

**Optimization and Efficiency**

- Logic design allows for careful consideration of system efficiency and performance. Researchers can optimize the logical structure, data flow, and algorithms to achieve the desired functionality while minimizing resource utilization, response times, and computational complexity. This optimization is crucial for delivering a system that meets performance expectations.

**System Scalability**

- Logic design helps in planning for system scalability and extensibility. By designing the logic with scalability in mind, researchers can ensure that the system can accommodate future growth and increasing data volumes or user loads. This is particularly important in today's fast-evolving technological landscape.

**Documentation and Communication**

- Logic design provides a means to document the system's architecture, data flow, and behavior in a visual and structured manner. This documentation serves as a valuable resource for both the research team and stakeholders, facilitating effective communication, collaboration, and knowledge transfer throughout the project's lifecycle.

**Requirements Verification**

- Logic design allows researchers to validate that the system's logical structure and behavior align with the specified requirements and objectives of the project. It ensures that the system's design is a faithful representation of the intended functionality, reducing the risk of costly deviations during implementation.

**Foundation for Implementation**

- Once the logic design is complete, it serves as a solid foundation for the implementation phase of the project. Developers can use the design as a reference to build the actual software system, ensuring that it faithfully reflects the envisioned functionality.

## 3.4.1 Activity Diagram

An activity diagram serves as a tool for illustrating the control flow within a system, focusing on the process flow rather than its implementation details. This type of diagram effectively models both concurrent and sequential activities. The primary purpose of an activity diagram is to provide a visual representation of how activities progress from one to another within a system. It places a significant emphasis on the order and conditions governing this flow. The flow itself can take on various forms, such as sequential, branched, or concurrent. To handle these diverse flow patterns, activity diagrams incorporate elements like forks and joins.

In essence, an activity diagram can be thought of as an object-oriented flowchart, capturing a system's activities as a series of actions or operations. These actions collectively form the basis for modeling behavioral aspects of the system.

### 3.4.1.1 Key Components of a Activity Diagram

- **Initial State**: It depicts the initial stage or beginning of the set of actions.
- **Final State**: It is the stage where all the control flows and object flows end.
- **Decision Box**: It makes sure that the control flow or object flow will follow only one path.
- **Action Box**: It represents the set of actions that are to be performed

### 3.4.1.2 Activity Diagram for Registration

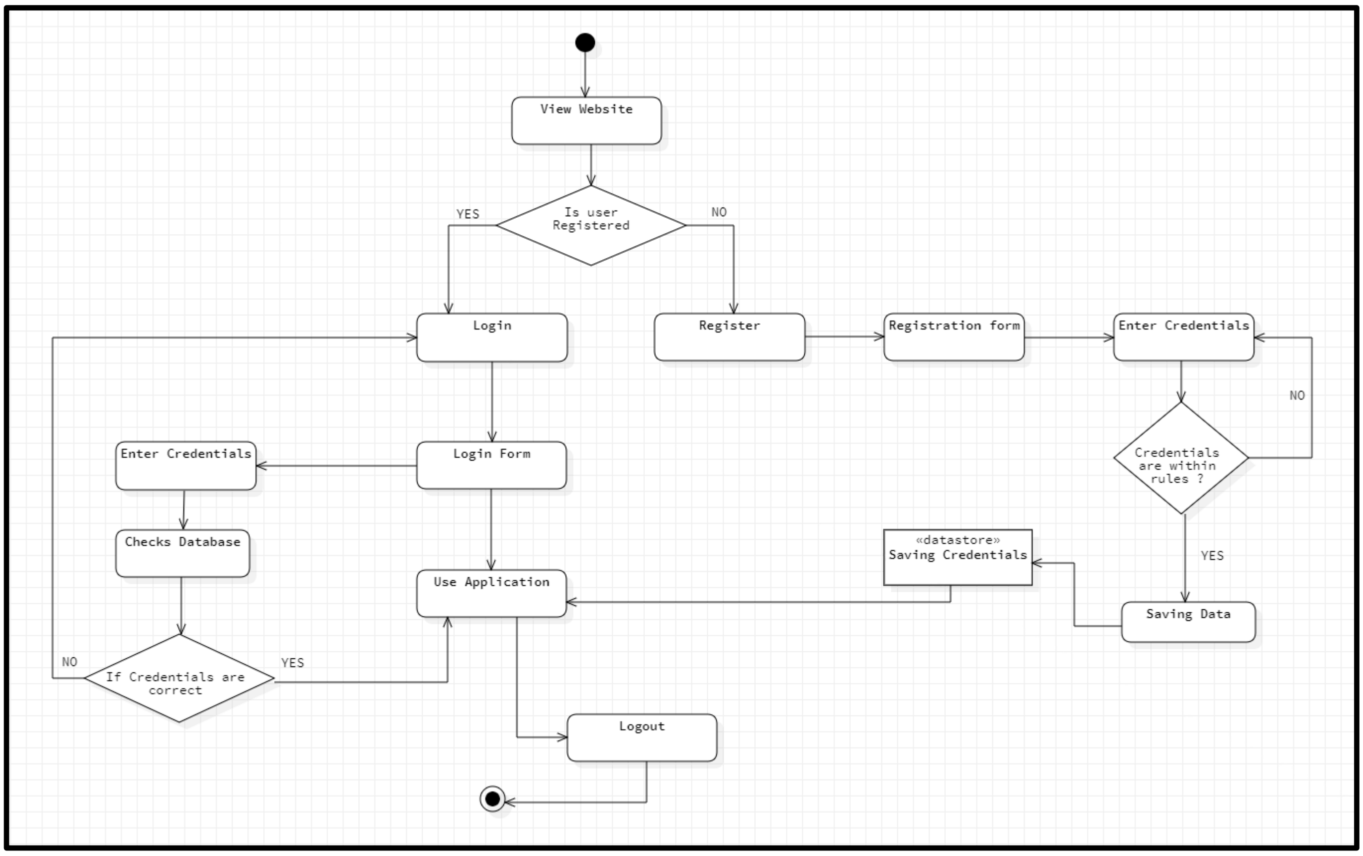


*Figure 13 : **Activity Diagram for Registration***

### 3.4.1.3 Activity Diagram for the Application

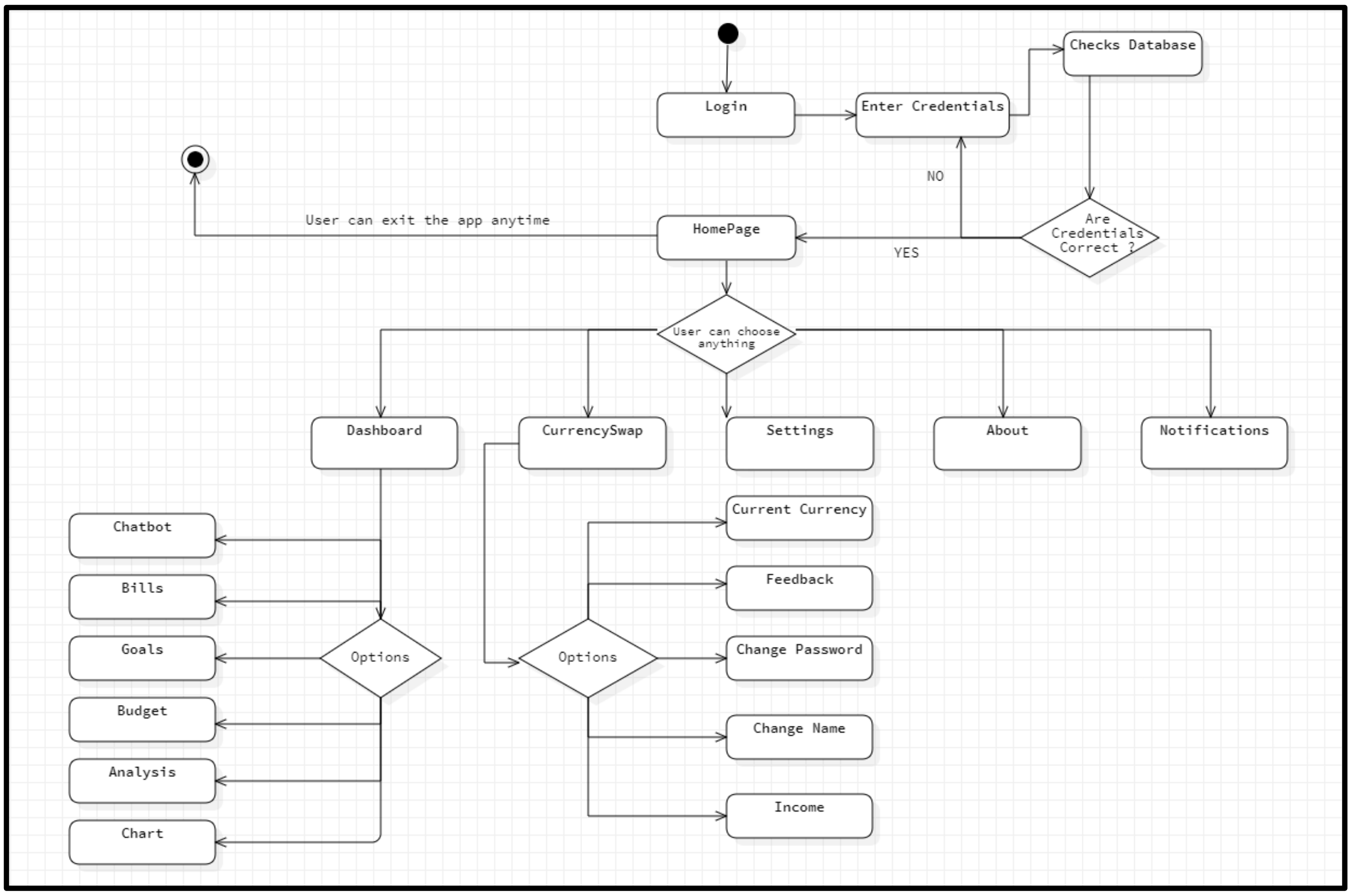


*Figure 14 : **Activity Diagram for the Application***

## 3.4.2 Class Diagram

A Class Diagram is a fundamental component of the Unified Modeling Language (UML), a standardized visual language used in software engineering for modeling and designing software systems. A Class Diagram provides a graphical representation of the classes, their attributes, methods, and the relationships between them within a system. It is an essential tool for software architects, designers, and developers to document and communicate the structural aspects of a software application.

Here's an explanation of the key components and the importance of Class Diagrams:

### 3.4.2.1 Key Components of a Class Diagram:

- **Class:** A class is a blueprint or template for creating objects. It represents a concept, entity, or component within a system and contains attributes (properties) and methods (functions) that define its behavior.
- **Attributes:** Attributes represent the properties or characteristics of a class. They describe the data that an object of the class will hold. Attributes are typically depicted as name : data type pairs.
- **Methods:** Methods define the operations or behaviors that a class can perform. They represent the functions that can be called on objects of the class and are typically depicted as name(parameters) : return type pairs.
- **Association:** Associations represent relationships between classes. They indicate how classes are connected or interact with each other. Associations can be one-to-one, one-to-many, or many-to-many, and they can have roles and multiplicities.
- **Inheritance:** Inheritance relationships represent the "is-a" relationship between classes. It indicates that one class (subclass or child) inherits attributes and methods from another class (superclass or parent).
- **Dependency:** Dependency relationships represent a weaker form of association where one class relies on another class but is not tightly coupled. Changes in the independent class can impact the dependent class.

### 3.4.2.2 Importance of Class Diagrams:

- **Visualization and Understanding:** Class Diagrams provide a visual representation of the structure of a software system, making it easier for stakeholders, including developers and non-technical users, to understand the system's architecture, components, and relationships.

- **Design and Planning:** Class Diagrams play a crucial role in the design phase of software development. They help in designing the class hierarchy, defining attributes and methods, and planning the overall system structure.
- **Communication:** Class Diagrams serve as a common language for communication among development teams, stakeholders, and clients. They facilitate discussions about system requirements, features, and design decisions.
- **Documentation:** Class Diagrams are valuable documentation artifacts that can be used for reference throughout the software development lifecycle. They provide a comprehensive overview of the system's design.
- **Code Generation:** In some development environments, Class Diagrams can be used to generate code automatically. This streamlines the development process and ensures consistency between the design and implementation.
- **Maintenance and Refactoring:** When changes are required in the system, Class Diagrams help in identifying the impact of those changes on other classes and components. This is crucial for maintenance and refactoring efforts.
- **Testing and Quality Assurance:** Testers can use Class Diagrams to understand the relationships between classes and plan test scenarios. This helps ensure that the system functions correctly and that test coverage is thorough.

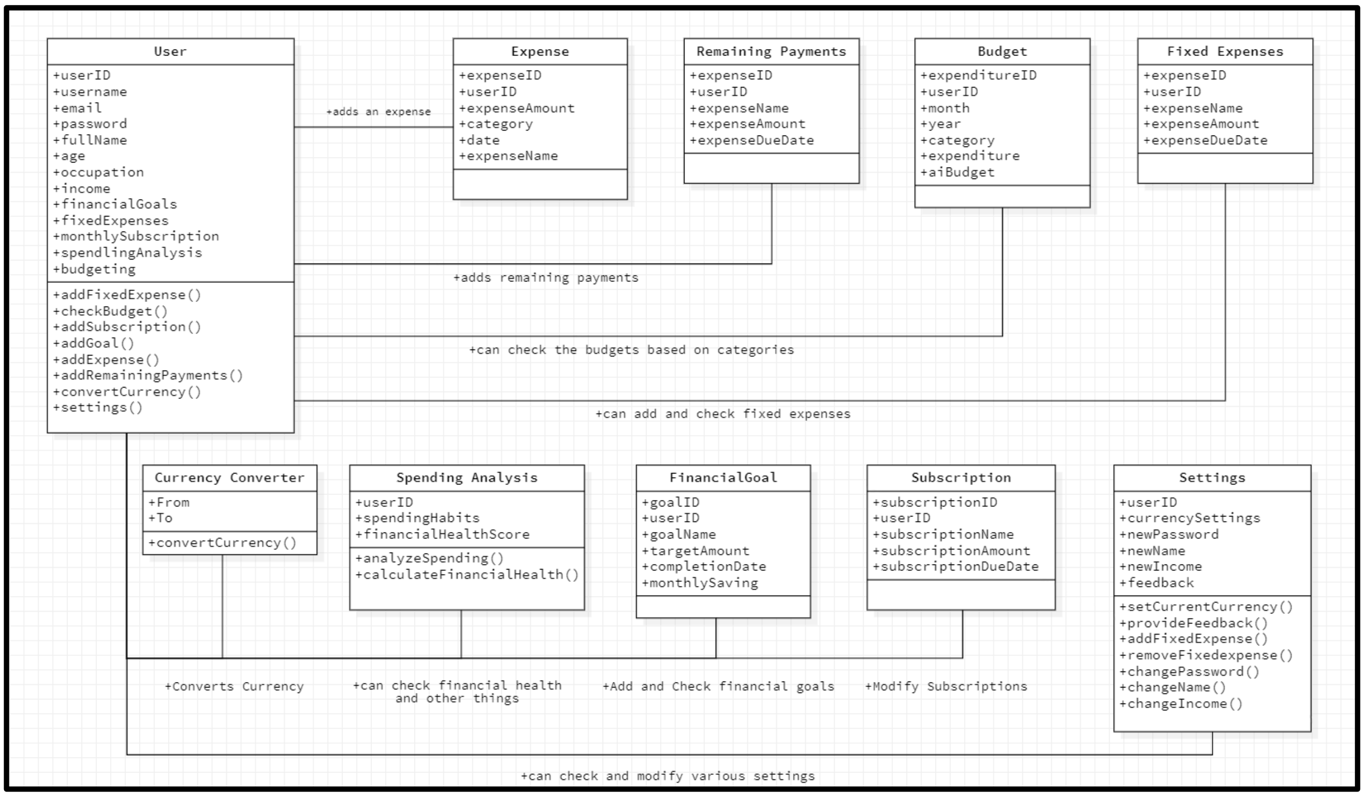


*Figure 15 : **Class Diagram***

## 3.4.3 Sequence Diagram

The sequence diagram is used primarily to show the interactions between objects in the sequential order that those interactions occur. Much like the class diagram, developers typically think sequence diagrams were meant exclusively for them. However, an organization's business staff can find sequence diagrams useful to communicate how the business currently works by showing how various business objects interact. Besides documenting an organization's current affairs, a business-level sequence diagram can be used as a requirements document to communicate requirements for a future system implementation. During the requirements phase of a project, analysts can take use cases to the next level by providing a more formal level of refinement. When that occurs, use cases are often refined into one or more sequence diagrams.

An organization's technical staff can find sequence diagrams useful in documenting how a future system should behave. During the design phase, architects and developers can use the diagram to force out the system's object interactions, thus fleshing out overall system design. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction. A sequence diagram shows the sequence of messages passed between objects

### 3.4.3.1 Benefits of a Sequence Diagram

- It explores the real-time application.
- It depicts the message flow between the different objects.
- It has easy maintenance.
- It is easy to generate.
- Implement both forward and reverse engineering.
- It can easily update as per the new change in the system.

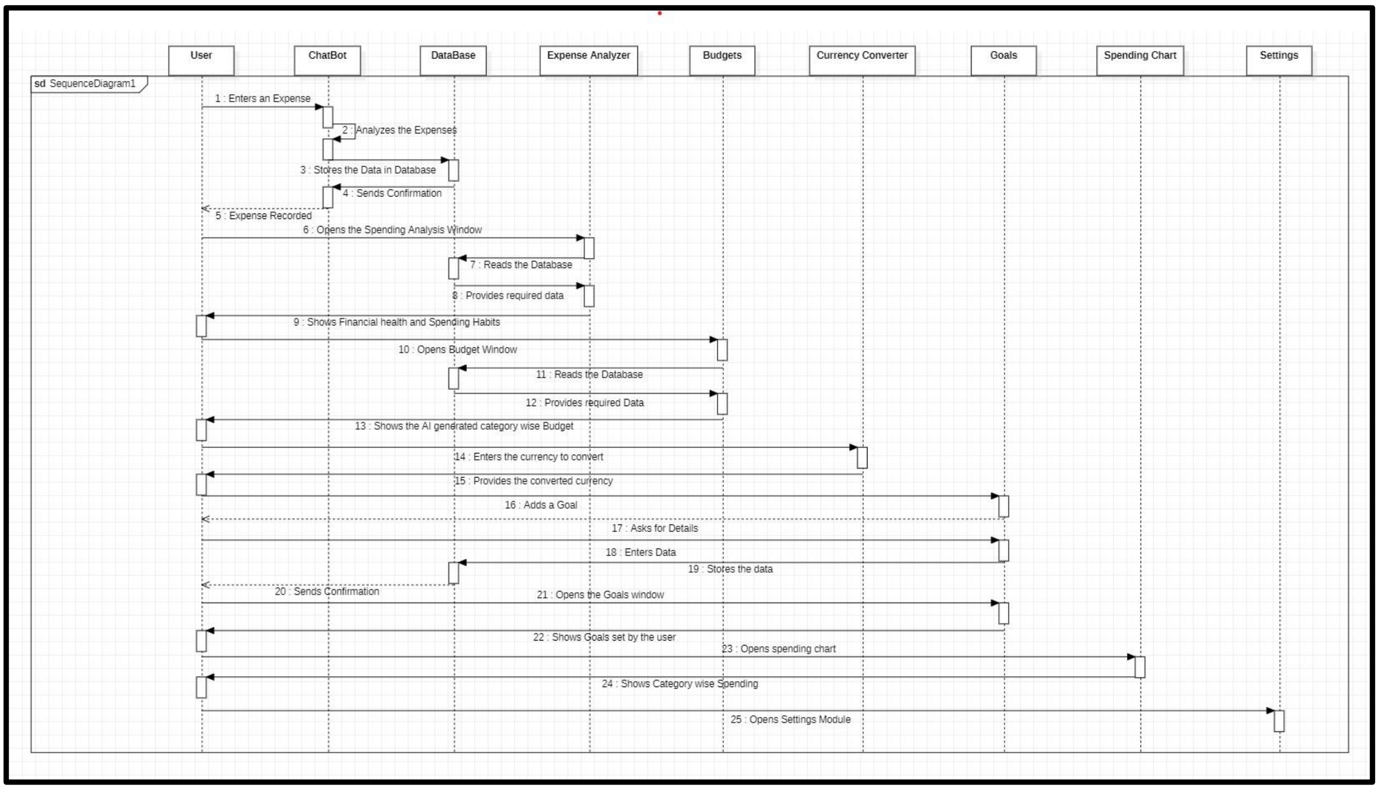
### 3.4.3.2 Drawbacks of Sequence Diagrams

- In the case of too many lifelines, the sequence diagram can get more complex.
- The incorrect result may be produced, if the order of the flow of messages changes.

- Since each sequence needs distinct notations for its representation, it may make the diagram more complex.
- The type of sequence is decided by the type of message

### 3.4.3.3 Purpose of Sequence Diagram

- To model high-level interaction among active objects within a system.
- To model interaction among objects inside a collaboration realizing a use case.
- It either models generic interactions or some certain instances of interaction



User
ChatBot
DataBase
Expense Analyzer
Budgets
Currency Converter
Goals
Spending Chart
Settings
sd SequenceDiagram1
1 : Enters an Expense
2 : Analyzes the Expenses
3 : Stores the Data in Database
4 : Sends Confirmation
5 : Expense Recorded
6 : Opens the Spending Analysis Window
7 : Reads the Database
8 : Provides required data
9 : Shows Financial health and Spending Habits
10 : Opens Budget Window
11 : Reads the Database
12 : Provides required Data
13 : Shows the AI generated category wise Budget
14 : Enters the currency to convert
15 : Provides the converted currency
16 : Adds a Goal
17 : Asks for Details
18 : Enters Data
19 : Stores the data
20 : Sends Confirmation
21 : Opens the Goals window
22 : Shows Goals set by the user
23 : Opens spending chart
24 : Shows Category wise Spending
25 : Opens Settings Module

*Figure 16 : **Sequence Diagram***

# IMPLEMENTATION & TESTING

In this pivotal chapter, we delve into the heart of our project, where the theoretical concepts and meticulous planning take shape in the form of tangible implementations, Here we will explore the intricacies of our codebase, discuss the architectural decisions that guided our development process, and detail the comprehensive testing methods employed to ensure the robustness and reliability of our solution.

## 4.1 Testing Approach

In a project's test approach, a strategic implementation is crucial for defining the methodology of testing execution. Two primary techniques exist within this approach. The proactive technique involves initiating the test design process early on to identify and rectify defects before the build is created. On the other hand, the reactive technique commences testing only after the completion of design and coding phases.

The overarching aim of software testing is to evaluate software quality in terms of identified defects, the number of test runs, and the system coverage. These assessments extend to both the functional and non-functional attributes of the software in development. Upon the discovery of bugs or defects during testing, a meticulous process follows. Bugs are logged, and the development team addresses and resolves them. Subsequent testing ensures the bug's successful resolution, preventing the introduction of new defects and contributing to an overall enhancement of software quality.

Verification and validation testing are integral components performed before handing over software to the customer. These tests serve to ensure adherence to client requirements, initiating the software testing life cycle at an early stage. Validation testing, in particular, plays a critical role in software quality assurance procedures and standards. Its purpose is to guarantee that the software product aligns with client requirements, effectively fulfilling its intended purpose.

## 4.2 Types of Testing

### 4.2.1 Component Testing

Component testing, a type of software testing, involves the independent testing of each individual component without integration with other components. From an architectural perspective, it is also referred to as Module Testing. The terms Unit Testing, Program Testing, or Module Testing are synonymous with Component Testing. Typically, software comprises multiple components, and Component Level Testing is focused on testing these components in isolation.

Also known as Unit Testing, the primary objective of this testing type is to identify defects within the software component. Simultaneously, it validates the functionality of distinct software components such as modules, objects, and classes, allowing for separate testing of these elements.

### 4.2.2 Integration Testing

This constitutes a crucial segment of the software validation model, emphasizing the examination of interactions among various components of the system. This phase involves testing not only the interaction within the system but also the interface of the system with the computer operating system, file system, hardware, and any other software systems it may interact with. The goal is to ensure seamless integration and compatibility across different facets of the software environment.

### 4.2.3 Unit Testing

Unit Testing stands as the initial stage in the test life cycle of any software build. As the name suggests, individual units undergo functionality and efficiency tests to ensure they align with the specified requirements before integration.

Also known as component testing, the primary objective of unit testing is to identify defects in the software component while simultaneously verifying the functioning of different software elements such as modules, objects, and classes. It is a level of software testing where individual units/components of software are tested to validate that each unit performs as designed. A unit, in this context, represents the smallest testable part of any software with one or a few inputs and usually a single output.

The main goal of unit testing is to isolate written code, test it, and determine if it functions as intended. This step is crucial in the development process as it helps detect early flaws in code that might be more challenging to identify in later testing stages. A typical unit test comprises three stages: planning, creating test cases and scripts, and executing the unit test. In the planning stage, the unit test is prepared and reviewed. The subsequent step involves creating test cases and scripts, followed by the actual testing of the code.

Each test case is independently tested in an isolated environment to ensure a lack of dependencies in the code. Developers should establish criteria to verify each test case, and a testing framework can be employed to report any failed tests. It is important to note that developers need not create a test for every line of code, as this may be time-consuming. Instead, the focus should be on creating tests that address code likely to impact the behavior of the software being developed.

#### 4.2.3.1 Benefits of Unit Testing

**Make the process agile**

Unit testing goes hand in hand with the Agile methodology and different flavors of it and thus allows to make changes more easily as compared to other Testing and Project management methodologies

**Early Bug Detection**

By conducting unit testing early in the development cycle, issues are identified at their inception. Since developers test individual code components before integration, problems are detected and resolved promptly without adversely impacting other parts of the code. This encompasses both bugs in the code implementation and discrepancies in the specification for the unit.

**Facilitates Changes and Integration**

Unit testing empowers developers to refactor code or update system libraries with confidence in the module's continued functionality. It identifies changes that may break a design contract, streamlining the process of maintaining and modifying code. This practice significantly reduces defects in newly developed features and minimizes bugs during alterations to existing functionality. The verification of individual units during unit testing makes subsequent integration testing more straightforward.

**Documentation Through Testing**

Unit testing serves as documentation for the system, offering insights into the functionality provided by a unit and how to utilize it. Developers seeking information on a unit's interface can refer to the unit tests for a foundational understanding of its API.

**Simplified Debugging Process**

Unit testing simplifies the debugging process. If a test fails, developers need only focus on the latest changes made to the code, streamlining the identification and resolution of issues.

**Cost Reduction**

Early detection of bugs during unit testing contributes to cost reduction in bug fixes. Bugs identified early in the development process are inherently easier and less costly to fix compared to those detected later, which often result from numerous changes, making it challenging to pinpoint the exact cause.

### 4.2.4 System Testing

System Testing represents a comprehensive testing level focused on validating the fully integrated software product. Its primary objective is to assess the end-to-end system specifications, recognizing that the software is just one component within a larger computer-based system. Typically, the software interfaces with other software and hardware systems. Conducted when the entire software system is prepared, System Testing, also known as functional and system testing, scrutinizes the overall system behavior within the project's defined scope.

Distinct from the internal workings, the primary concern during System Testing is to ensure that the system behaves in accordance with specified requirements. Testers do not delve into the system's internals but rather verify its external behavior against expectations.

In the broader context of software testing, there are two main categories: Black Box Testing and White Box Testing. System Testing falls under the category of Black Box Testing, where the focus is on examining the external workings of the software from the user's perspective. Unlike White Box Testing, which involves scrutinizing the internal code and workings of a software application, System Testing takes a holistic approach by evaluating the system's functionalities externally.

### 4.2.5 Acceptance Testing

Acceptance testing, a crucial phase in software testing, evaluates whether a software application aligns with its intended business requirements and user needs. The overarching goal is to ensure that the software system performs in accordance with stakeholders' expectations and meets the specified acceptance criteria.

This form of testing is primarily executed by end-users or stakeholders who are not directly involved in the software development process. Acceptance testing operates as a black-box testing technique, wherein the focus is solely on the software's inputs and outputs, without delving into the internal workings of the software.

The acceptance testing process involves the creation of test scenarios, encompassing the software's functionality, and verifying adherence to specified criteria. Any issues or defects identified during the acceptance testing phase are reported back to the development team for thorough investigation and resolution. Upon successful completion of acceptance testing, the software is deemed ready for release to production.