BUDGET TRACKING SYSTEM

LAB REPORT

Submitted by

NITHISH R [RA2111027010125]

SAI KISHORE S [RA2111027010139]

ARYAN DEV [RA2111027010137]

Under the Guidance of

Dr SHOBANADEVI A

Assistant Professor , Department of Computing Technologies

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE ENGINEERING

with specialization in Big Data Analytics

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

COLLEGE OF ENGINEERING & TECHNOLOGY
SRM INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUE – 603 203
Chengalpattu District

BONAFIDE CERTIFICATE

Register No. **RA2111027010125** Certified to be the bonafide work done by **NITHISH R** of II Year/IV Sem B.Tech Degree Course in the Practical Software Software Engineering and Project Management 18CSC206J in SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, Kattankulathur during the academic year 2022 – 2023.

LAB INCHARGE **Head of the Department**

Dr A.ShobanaDevi

Assistant Professor

Department of Computing Technologies

SRMIST – KTR.

Date :

ABSTRACT :

Budget is essential in every walk of our life – national, domestic and business.A Budget is prepared to have effective utilization of funds and for the realization of objectives as efficiently as possible. Budget is a widely practiced technique and most of us use budgets in some way or the other.

Budget is one of the emphasized terms used in efficient methods of planning and control. It is employed, no doubt, in large business houses, but even the small businesses are using it, in some informal manner. Budget in common parlance is understood as planning for expenditure.

A budget is defined as a comprehensive and Coordinated plan expressed in financial terms, for the operations and resources of an enterprise for some specified period in the future.

A budget tracking system is a software application designed to help individuals, households, and businesses monitor and manage their finances. The system tracks expenses and income, creates reports, and provides insights into spending patterns. By using a budget tracking system, users can set financial goals, identify areas for cost savings, and make informed decisions about their financial future. This abstract provides an overview of the benefits and functionality of a budget tracking system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

1.	WBC	-	WORK BREAKDOWN STRUCTURE
2.	SWOT	-	STRENGTH , WEAKNESS , OPPORTUNITIES , THREATS
3.	ER	-	ENTITY RELATIONSHIP
4.	DFD	-	DATA FLOW DIAGRAM
5.	UI	-	USER INTERFACE
6.	RMMM	-	RISK MITIGATION,MANAGEMENT AND MONITORING

PROBLEM STATEMENT :

A problem statement is a concise description of a problem or challenge that a software project aims to solve. A problem statement for a budget tracking software could be:

"Many individuals and households struggle to effectively manage their finances and stick to a budget. They have a hard time keeping track of their income and expenses, setting financial goals, and creating a personalized budget plan.

They also find it difficult to understand where their money is going and how to make the most of it. As a result, they may end up living paycheck to paycheck, accumulating debt, or not being able to save for the future.

Our budget tracking software aims to solve this problem by providing users with a user-friendly tool that allows them to easily track their income and expenses, set financial goals, and create a personalized budget plan.

The software will also provide users with insights and visualization of their financial data to make informed decisions."This problem statement highlights the main challenge that budget tracking software aims to solve:

helping individuals and households manage their finances more effectively. By providing a user-friendly tool for tracking income and expenses, setting financial goals, and creating a personalized budget plan, the software aims to empower users to take control of their finances and make the most of their money.

STAKEHOLDERS & PROCESS MODEL:

Selection of Methodology:

WHY AGILE MODEL IS BETTER THAN WATERFALL MODEL?

1. The Agile Model is based on iterative development and hence it divides the entire project into smaller parts which reduces the risk factor which is not the case in the waterfall model.

2. The Waterfall model cannot accept the changes in requirements but in an agile model it is easy to change the system requirements.

3. In an agile model, the entire project is divided into smaller parts which helps to minimize the project risk and to reduce the overall project delivery time requirements.

4. In the waterfall model, since the risk factor is high, it is not suitable for complex Projects.

5. In the waterfall model the testing is done in a later stage; it does not allow identifying the challenges and risks in the earlier phase, so the risk reduction strategy is difficult to prepare, which is not the case in agile models.

6. In the waterfall model, it follows a sequential approach whereas in the agile model it explains the process in order of incremental approach.

7. In agile it performs the testing concurrently with software development whereas in the waterfall model the testing comes after the build phase only.

8. In an agile model the distance between the customer and developer is short whereas in a waterfall model it is long.

9. In agile there can be any change in the project but in the waterfall model there are no changes throughout the project work.

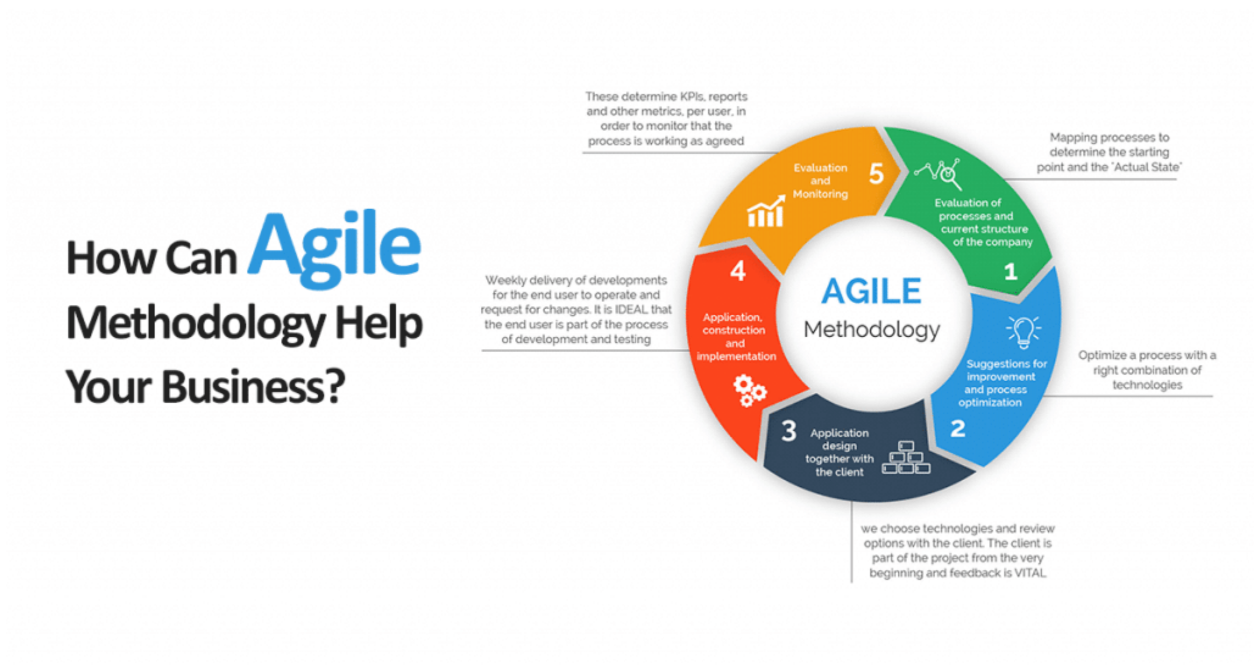
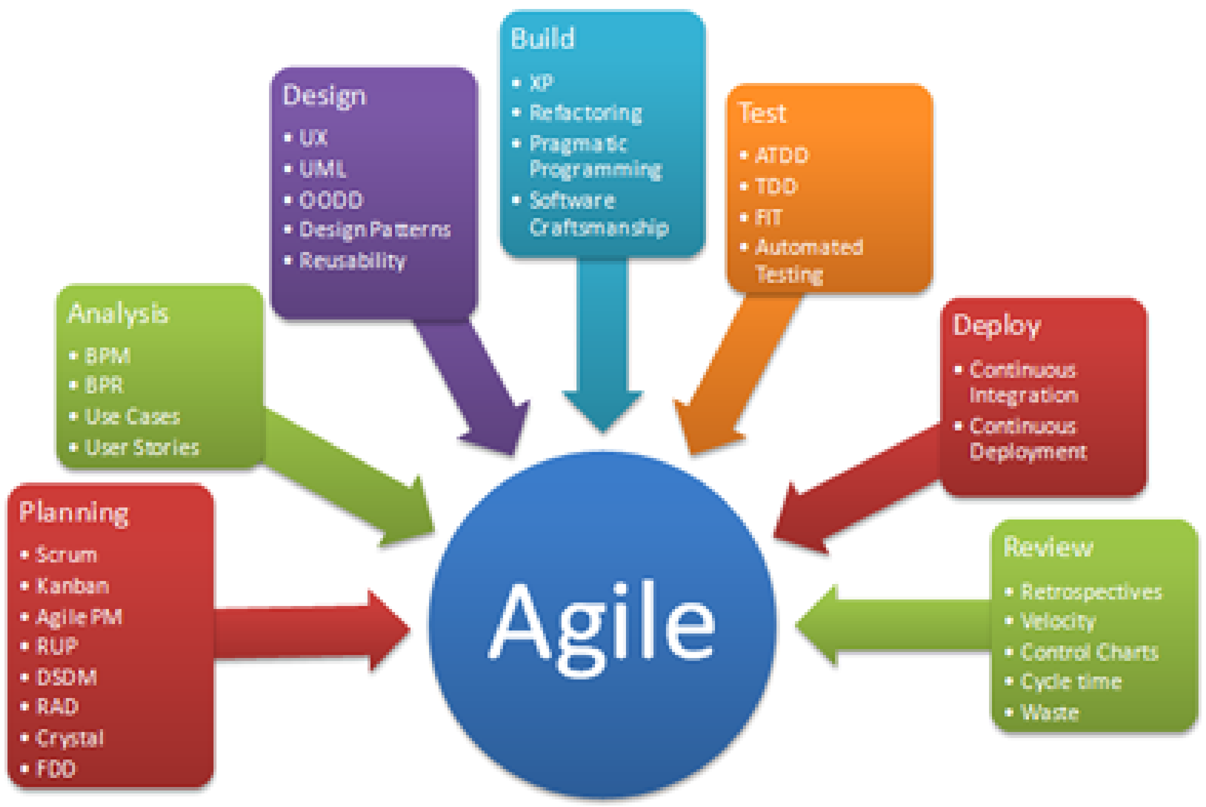

How Can Agile Methodology Help Your Business?
AGILE Methodology
1 Evaluation of processes and current structure of the company
Mapping processes to determine the starting point and the 'Actual State'
2 Suggestions for improvement and process optimization
Optimize a process with a right combination of technologies
3 Application design together with the client
we choose technologies and review options with the client. The client is part of the project from the very beginning and feedback is VITAL
4 Application, construction and implementation
Weekly delivery of developments for the end user to operate and request for changes. It is IDEAL that the end user is part of the process of development and testing
5 Evaluation and Monitoring
These determine KPIs, reports and other metrics, per user, in order to monitor that the process is working as agreed

Fig : Why Agile is better than Waterfall model



Stakeholder Name	Activity/ Area /Phase	Interest	Influence	Priority (High/ Medium/ Low)
User	The users are the persons, who will use the project's service or result. They may be internally or externally helping the community.	High	High	1
Program manager(NITHISH)	They are responsible for managing related projects in a coordinated way to obtain benefits and control.	High	High	1
Project management office	PMO is an organization body or entity assigned various responsibilities related to the centralized and coordinated management of projects under the domain.	High	Low	4
Project manager (SAI KISHORE)	They are assigned by the performing	High	High	2

	organization to achieve the project objective.			
Project team	A project team is composed of the project manager, project management team and other team members who carry out the work.	High	High	2
Function manager (ARYAN DEV)	They are key individuals who play a management role within an administrative or functional area of the business such as human resources, finance,accounting or procurement.	High	Low	3

IDENTIFYING REQUIREMENTS:

Requirements:

Requirements are defined during the early stages of the system development as a specification of what should be implemented. A collection of requirements is a requirements document. They may be user level facility description, detailed specification of system behavior , general system property, a specific constraint on the system or information on how to carry on computation. The three types of requirements are explained below.

System Requirements:

1. Operating System: Windows, MacOS, or Linux
2. Processor: 1 GHz or higher
3. RAM: 2 GB or more
4. Hard Drive: 100 MB or more of free space
5. Web browser: Google Chrome, Mozilla Firefox, or Microsoft Edge (latest version)
6. Remainder Alert System
7. Client feedback system

Functional Requirements:

1. Create an account
2. Managing the account
3. Login to the system
4. Navigate through Expenses Menu
5. Select the category from Menu
6. Customize the options for the selected Menu
7. Add expenses to the current Menu
8. Confirm the expenses

Non-Functional Requirements:

1. Portability:

Since it is performed in a web browser it is portable.

2. Readability:

Readability refers to how easy or difficult a piece of writing is to understand. It is an important aspect of effective communication.

3. Availability:

Availability refers to the state or condition of being present and accessible when needed. In other words, availability is the extent to which someone or something is ready and able to respond to a request or a demand.

4. Maintainability:

In software development, maintainability is a crucial factor that impacts the cost and effort required to keep a system running smoothly.

5. Security:

Security in project management refers to the measures and strategies employed to protect the project and its stakeholders from potential threats and risks.

6. User Friendly:

User-friendly means designing the project management process, tools, and software in a way that is easy to use, understand and navigate for project managers.

7. Performance:

Performance refers to the ability of a project manager and their team to meet the goals, objectives, timelines, and budget of a project.

8. Efficiency:

Efficiency refers to the ability to use resources effectively and productively to achieve project goals and objectives.

9. Safety:

Safety is an essential aspect of ensuring successful project completion. It involves the identification, assessment, and management of potential risks or hazards that could compromise the health, safety, or well-being of project team members, stakeholders.

10. Privacy:

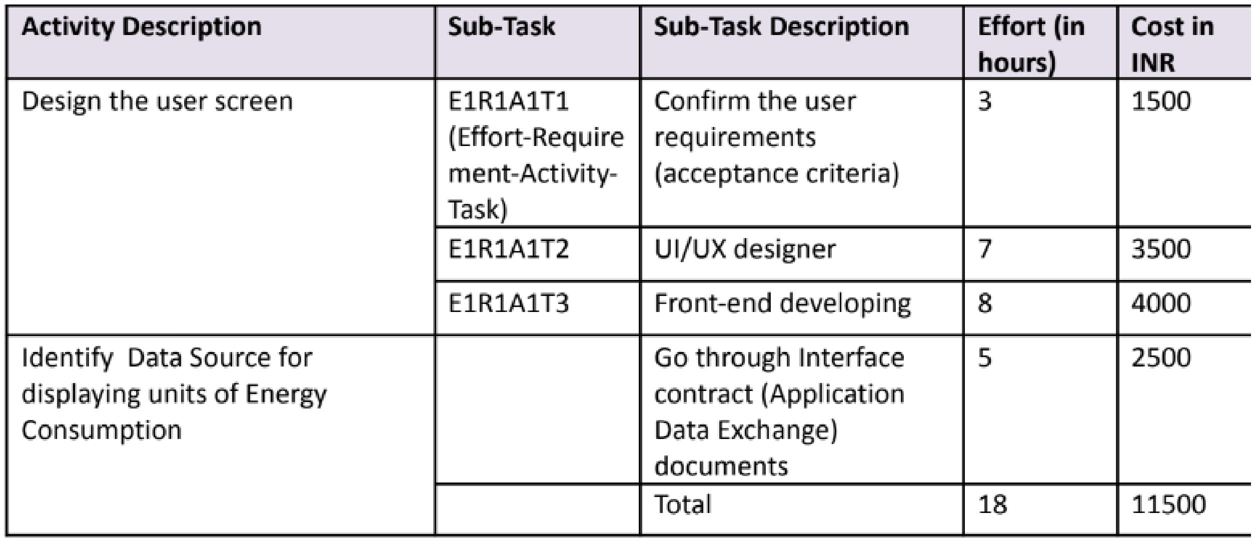
Privacy refers to the protection of sensitive or confidential information related to a project.

PROJECT PLAN & EFFORT

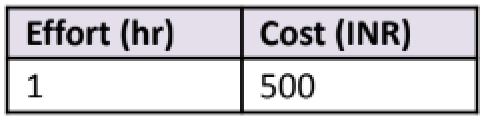
User requirement:

1. Easy-to-understand interface for contributing pay and costs.
2. Capacity to arrange payment and costs for simple following and investigation
3. Adaptable financial plan objectives and cutoff points
4. Constant following of expenditure and financial plan progress
5. Warnings for overspending or arriving at financial plan limits
6. The secure capacity of monetary data
7. Simple admittance to verifiable information and reports for investigation and arranging.

EFFORT AND COST ESTIMATION :



Activity Description	Sub-Task	Sub-Task Description	Effort (in hours)	Cost in INR
Design the user screen	E1R1A1T1 (Effort-Requirement-Activity-Task)	Confirm the user requirements (acceptance criteria)	3	1500
	E1R1A1T2	UI/UX designer	7	3500
	E1R1A1T3	Front-end developing	8	4000
Identify Data Source for displaying units of Energy Consumption		Go through Interface contract (Application Data Exchange) documents	5	2500
		Total	18	11500



Effort (hr)	Cost (INR)
1	500

Infrastructure/Resource Cost [CapEx]:

Infrastructure Requirement	Qty	Cost per qty	Cost per item
Hardware(Processor: 1 GHz or higher RAM: 2 GB or more Hard Drive: 100 MB or more of free space)	3	1 processor - 40000\- RAM - 2000\- Hard Drive - 1000\-	43000\-
Software(Python IDLE,MYSQL)	2	Free	Free

Maintenance and Support Cost [OpEx] :

Category	Details	Qty	Cost per qty per annum	Cost per item
People	Network, System, Middleware and DB admin Developer , Support Consultant	1	10000	10000
License	Operating System Database Middleware IDE	2	5000	10000
Infrastructures	Server, Storage and Network	5	20000	100,000

Project Team Formation:

Identification Team members :

Name	Role	Responsibilities
NITHISH R	Key Business User (Product Owner)	Provide clear business and user requirements
NITHISH R	Project Manager	Manage the project
ARYAN DEV	Business Analyst	Discuss and Document Requirements
SAI KISHORE S	Technical Lead	Design the end-to-end architecture
ARYAN DEV	UX Designer	Design the user experience
SAI KISHORE S	Frontend Developer	Develop user interface
NITHISH R	Backend Developer	Design, Develop and Unit Test Services/API/DB
ARYAN DEV	Cloud Architect	Design the cost effective, highly available and scalable architecture
ARYAN DEV	Cloud Operations	Provision required Services
SAI KISHORE S	Tester	Define Test Cases and Perform Testing

Responsibility Assignment Matrix :

RACI Matrix	Team Members			
Activity	**Name (BA)**	**Name (Developer)**	**Name (Project Manager)**	**Key Business User**
User Requirement Documentation	(ARYAN DEV)A	(SAI KISHORE S)C/I	(NITHISH R)I	(NITHISH)R
Identification of Methodology	(ARYAN DEV)A	(SAI KISHORE S)C	(NITHISH R)I	(SAI KISHORE S)C
Project Management Plan	(SAI KISHORE S)C	(ARYAN DEV)A	(SAI KISHORE S)C	(ARYAN DEV)A
Front-End	(NITHISH R)I	(ARYAN DEV)A	(ARYAN DEV)A	(NITHISH R)I
Back-End	(ARYAN DEV)A	(ARYAN DEV)A	(SAI KISHORE S)	(SAI KISHORE S)C
Testing and Review	(NITHISH R)I	(ARYAN DEV)A	(SAI KISHORE S)C	(NITHISH R)I

A	Accountable
R	Responsible
C	Consult
I	Inform

WORK BREAKDOWN STRUCTURE & RISK ANALYSIS :

WBS :

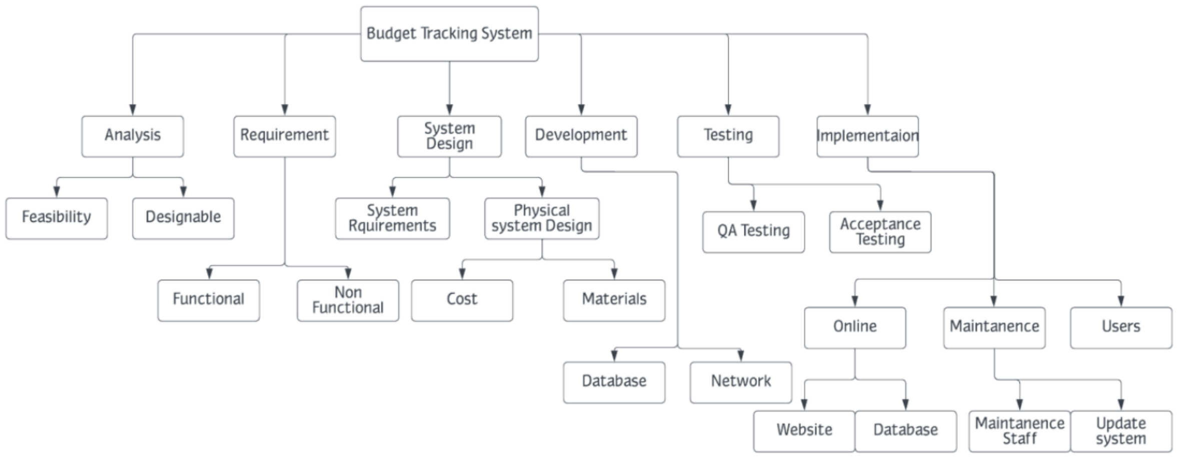


Fig:Work Breakdown Structure

Budget Tracking System:

1. Analysis
   - 1.1 Feasibility
   - 1.2 Designable

2. Requirement
   - 2.1 Functional
   - 2.2 Non Functional

3. System Design
   - 3.1 System Requirements
   - 3.2 Physical system Design
     - 3.2.1 Cost
     - 3.2.2 Materials

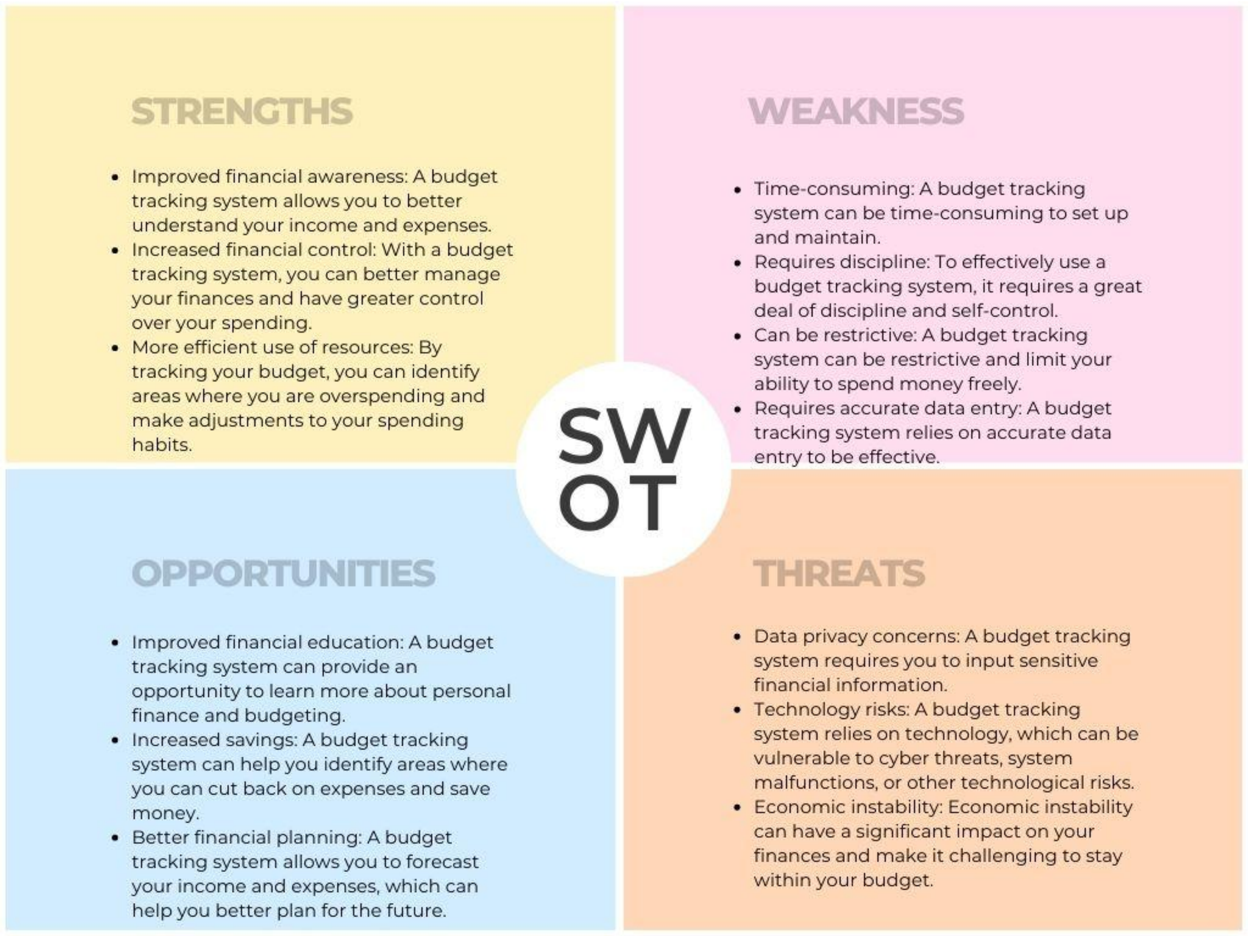
4. Development
   - 4.1 Database

4.2 Network

5. Testing
    5.1 QA Testing
    5.2 Acceptance Testing

6. Implementation
    6.1 Online
        6.1.1 Website
        6.1.2 Database
    6.2 Maintenance
        6.2.1 Maintenance Staff
        6.2.2 Update system
    6.3 Users

RISK ANALYSIS – SWOT :



STRENGTHS

- Improved financial awareness: A budget tracking system allows you to better understand your income and expenses.
- Increased financial control: With a budget tracking system, you can better manage your finances and have greater control over your spending.
- More efficient use of resources: By tracking your budget, you can identify areas where you are overspending and make adjustments to your spending habits.

WEAKNESS

- Time-consuming: A budget tracking system can be time-consuming to set up and maintain.
- Requires discipline: To effectively use a budget tracking system, it requires a great deal of discipline and self-control.
- Can be restrictive: A budget tracking system can be restrictive and limit your ability to spend money freely.
- Requires accurate data entry: A budget tracking system relies on accurate data entry to be effective.

OPPORTUNITIES

- Improved financial education: A budget tracking system can provide an opportunity to learn more about personal finance and budgeting.
- Increased savings: A budget tracking system can help you identify areas where you can cut back on expenses and save money.
- Better financial planning: A budget tracking system allows you to forecast your income and expenses, which can help you better plan for the future.

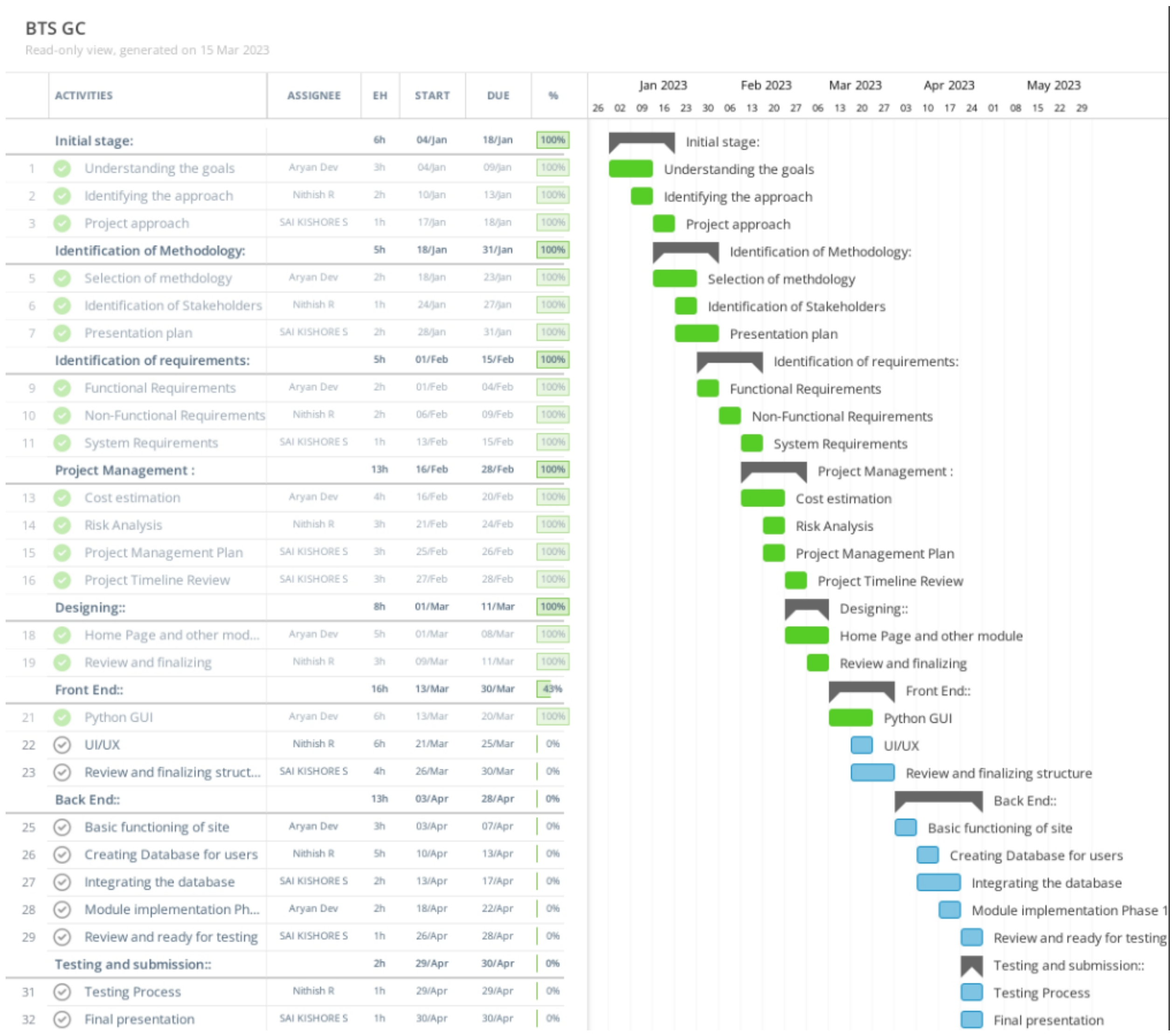
THREATS

- Data privacy concerns: A budget tracking system requires you to input sensitive financial information.
- Technology risks: A budget tracking system relies on technology, which can be vulnerable to cyber threats, system malfunctions, or other technological risks.
- Economic instability: Economic instability can have a significant impact on your finances and make it challenging to stay within your budget.

The ultimate goal when integrating risk management into budget planning is to understand the assumptions your budget is based on. Here are some steps you can take to come to that understanding:

1. Identify the major line items of your budget and the personnel who contributed to them.
2. Ask key personnel to provide insight on major line items.
3. Engage subject matter experts to adjust low confidence line items.
4. Mitigate the risks in your budgeting.
5. Continuously monitor risks and efficacy of controls.

TIMELINE – GANTT CHART:



BTS GC

Read-only view, generated on 15 Mar 2023

	ACTIVITIES	ASSIGNEE	EH	START	DUE	%
	Initial stage:		6h	04/Jan	18/Jan	100%
1	✓ Understanding the goals	Aryan Dev	3h	04/Jan	09/Jan	100%
2	✓ Identifying the approach	Nithish R	2h	10/Jan	13/Jan	100%
3	✓ Project approach	SAI KISHORE S	1h	17/Jan	18/Jan	100%
	Identification of Methodology:		5h	18/Jan	31/Jan	100%
5	✓ Selection of methdology	Aryan Dev	2h	18/Jan	23/Jan	100%
6	✓ Identification of Stakeholders	Nithish R	1h	24/Jan	27/Jan	100%
7	✓ Presentation plan	SAI KISHORE S	2h	28/Jan	31/Jan	100%
	Identification of requirements:		5h	01/Feb	15/Feb	100%
9	✓ Functional Requirements	Aryan Dev	2h	01/Feb	04/Feb	100%
10	✓ Non-Functional Requirements	Nithish R	2h	06/Feb	09/Feb	100%
11	✓ System Requirements	SAI KISHORE S	1h	13/Feb	15/Feb	100%
	Project Management :		13h	16/Feb	28/Feb	100%
13	✓ Cost estimation	Aryan Dev	4h	16/Feb	20/Feb	100%
14	✓ Risk Analysis	Nithish R	3h	21/Feb	24/Feb	100%
15	✓ Project Management Plan	SAI KISHORE S	3h	25/Feb	26/Feb	100%
16	✓ Project Timeline Review	SAI KISHORE S	3h	27/Feb	28/Feb	100%
	Designing::		8h	01/Mar	11/Mar	100%
18	✓ Home Page and other mod...	Aryan Dev	5h	01/Mar	08/Mar	100%
19	✓ Review and finalizing	Nithish R	3h	09/Mar	11/Mar	100%
	Front End::		16h	13/Mar	30/Mar	43%
21	✓ Python GUI	Aryan Dev	6h	13/Mar	20/Mar	100%
22	UI/UX	Nithish R	6h	21/Mar	25/Mar	0%
23	Review and finalizing struct...	SAI KISHORE S	4h	26/Mar	30/Mar	0%
	Back End::		13h	03/Apr	28/Apr	0%
25	Basic functioning of site	Aryan Dev	3h	03/Apr	07/Apr	0%
26	Creating Database for users	Nithish R	5h	10/Apr	13/Apr	0%
27	Integrating the database	SAI KISHORE S	2h	13/Apr	17/Apr	0%
28	Module implementation Ph...	Aryan Dev	2h	18/Apr	22/Apr	0%
29	Review and ready for testing	SAI KISHORE S	1h	26/Apr	28/Apr	0%
	Testing and submission::		2h	29/Apr	30/Apr	0%
31	Testing Process	Nithish R	1h	29/Apr	29/Apr	0%
32	Final presentation	SAI KISHORE S	1h	30/Apr	30/Apr	0%

Risk Management Framework :

Risk identification:

Identify all possible risks that can occur in the budget tracking system. This can include risks related to data security, system failure, or human error.

Risk assessment:

Evaluate the likelihood and potential impact of each identified risk. This can be done using a risk matrix, which assigns a likelihood and impact score to each risk.

Risk mitigation:

Develop strategies to mitigate or reduce the likelihood and impact of identified risks. This can include implementing security measures, backup and recovery plans, and training programs for system users.

Risk monitoring:

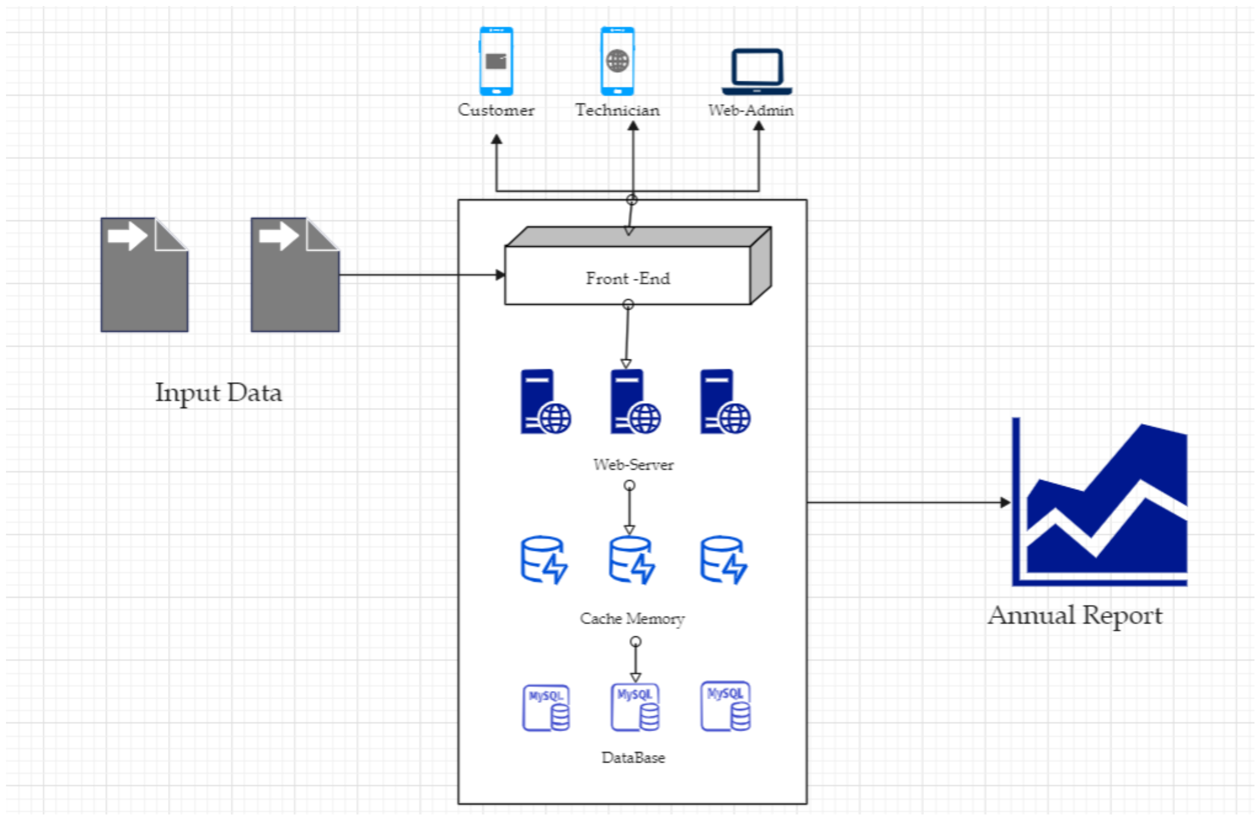
Continuously monitor the system for potential risks and evaluate the effectiveness of risk mitigation strategies. This can include regular system audits, security testing, and user feedback.

Risk response:

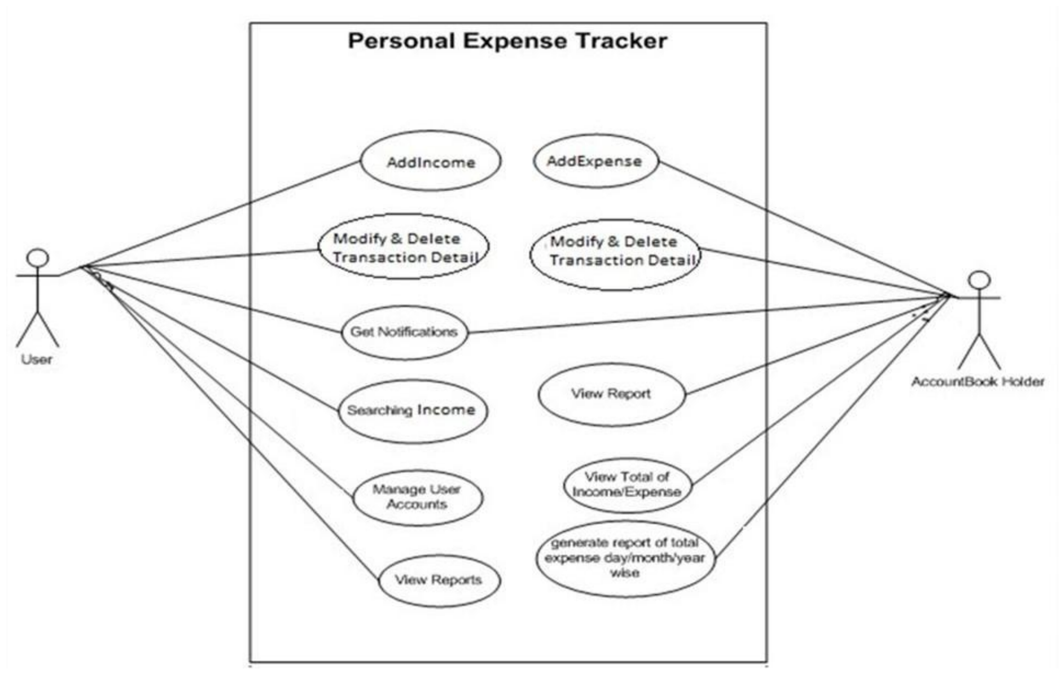
Develop a plan of action in case a risk does occur. This should include steps to contain and resolve the risk, as well as communication protocols to inform system users and stakeholders.

SYSTEM ARCHITECTURE, USE CASE & CLASS DIAGRAM :

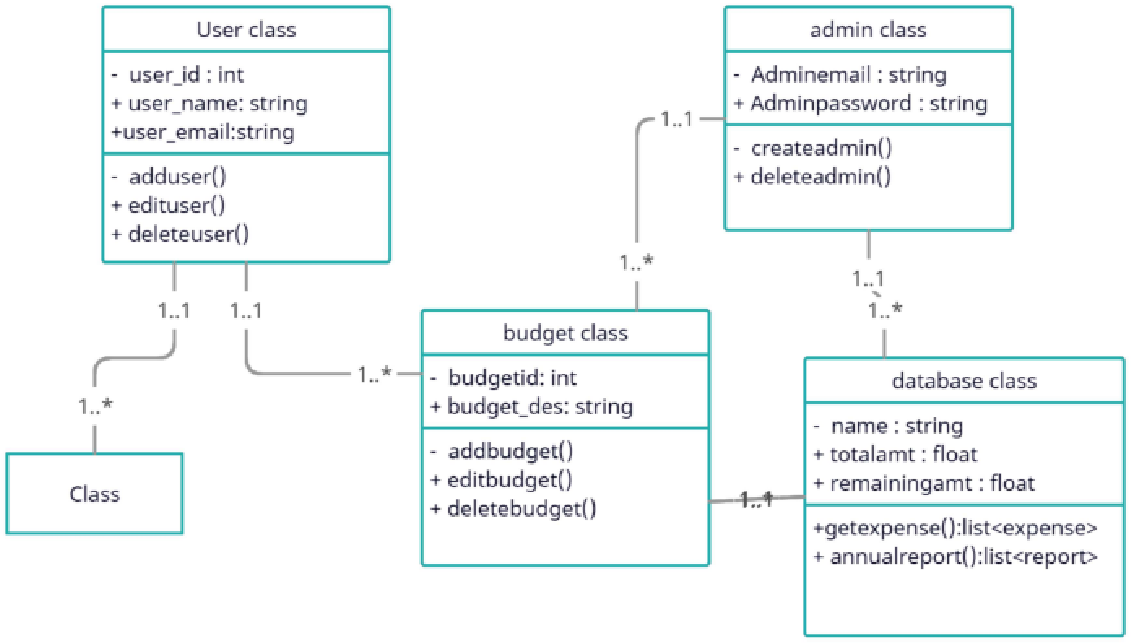
SYSTEM ARCHITECTURE FOR BUDGET TRACKING SYSTEM :



USE CASE DIAGRAM :


Personal Expense Tracker
AddIncome
AddExpense
Modify & Delete Transaction Detail
Modify & Delete Transaction Detail
Get Notifications
Searching Income
View Report
Manage User Accounts
View Total of Income/Expense
generate report of total expense day/month/year wise
View Reports
User
AccountBook Holder

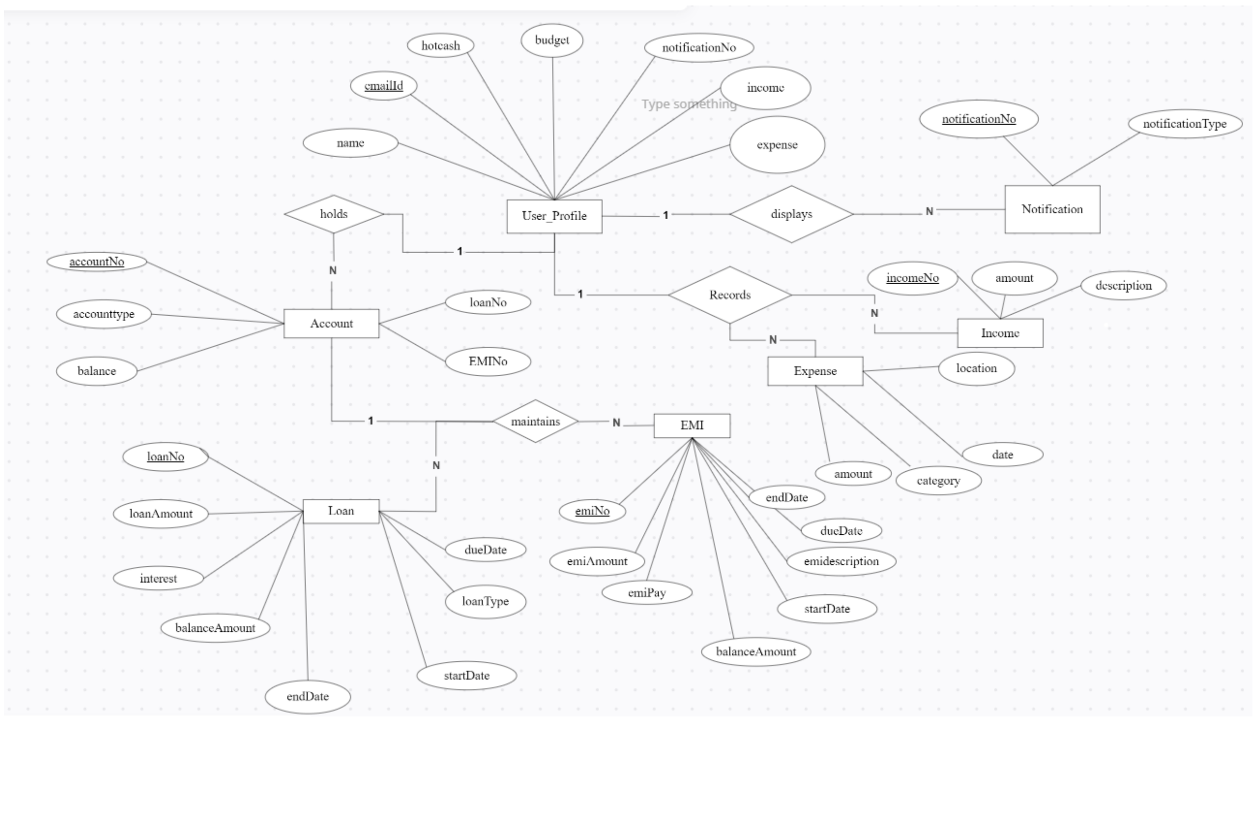
CLASS DIAGRAM :



ENTITY RELATIONSHIP DIAGRAM :

Components of the ER Diagram
This model is based on three basic concepts: Entities, Attributes, Relationships

ER Diagram – Notations
- Rectangles represent entity sets.
- Diamonds represent relationship sets.
- Lines link attributes to entity sets and entity sets to relationship sets.
- Ellipses represent attributes
- Double ellipses represent multivalued attributes.
- Dashed ellipses denote derived attributes.
- Underline indicates primary key attributes



DATA FLOW DIAGRAM :

The DFD takes an input-process-output view of a system. That is, data objects flow into the software, are transformed by processing elements, and resultant data objects flow out of the software. Data objects are represented by labeled arrows, and transformations are represented by circles (also called bubbles). The DFD is presented in a hierarchical fashion. That is, the first data flow model (sometimes called a level 0 DFD or context diagram) represents the system as a whole. Subsequent data flow diagrams refine the context diagram, providing increasing detail with each subsequent level.

The data flow diagram enables you to develop models of the information domain and functional domain. As the DFD is refined into greater levels of detail, you perform an implicit functional decomposition of the system. At the same time, the DFD refinement results in a corresponding refinement of data as it moves through the processes that embody the application.

A few simple guidelines can aid immeasurably during the derivation of a data flow diagram:

(1) Level 0 data flow diagram should depict the software/system as a single bubble;

(2) Primary input and output should be carefully noted.

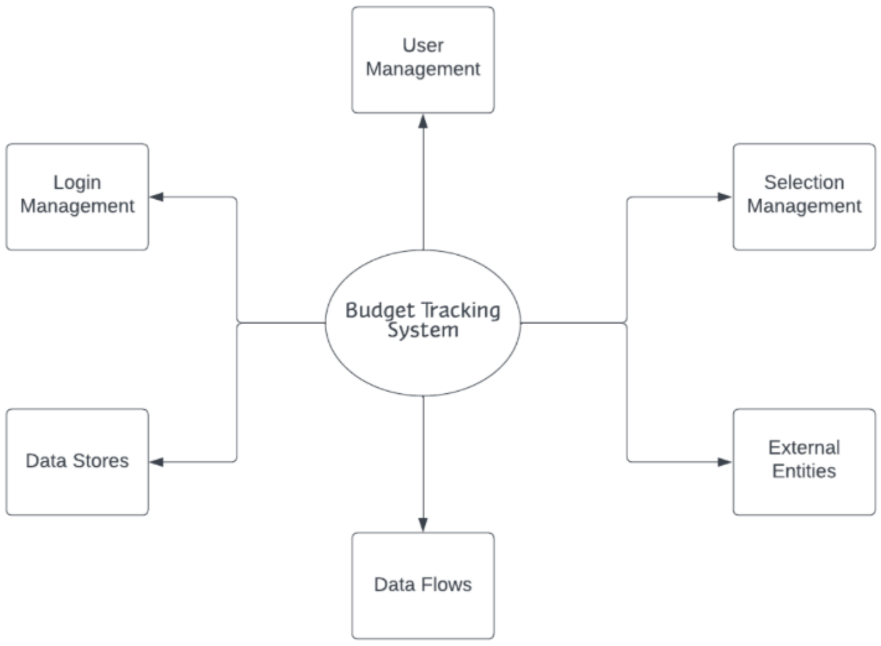
(3) Refinement should begin by isolating candidate processes, data objects, and data stores to be represented at the next level.

(4) All arrows and bubbles should be labeled with meaningful names;

(5) Information flow continuity must be maintained from level to level and

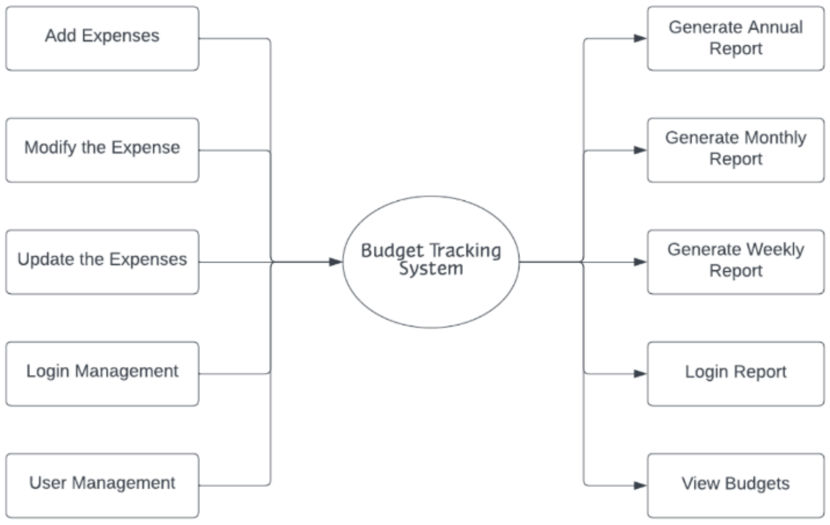
(6) One bubble at a time should be refined. There is a natural tendency to overcomplicate the data flow diagram. This occurs when you attempt to show too much detail too early or represent procedural aspects of the software in lieu of information flow.

DFD LEVEL 0 :



Zero Level Data Flow Diagram for Budget Tracking System

DFD LEVEL 1 :



First Level Data Flow Diagram for Budget Tracking System

SEQUENCE & COLLABORATION DIAGRAM :

SEQUENCE DIAGRAM :

A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process

Benefits of sequence diagrams:

1. Represent the details of a UML use case.
2. Model the logic of a sophisticated procedure, function, or operation.
3. See how objects and components interact with each other to complete a process.
4. Plan and understand the detailed functionality of an existing or future scenario.

The drawback of a Sequence Diagram:

1. In the case of too many lifelines, the sequence diagram can get more complex.
2. The incorrect result may be produced, if the order of the flow of messages changes.
3. Since each sequence needs distinct notations for its representation, it may make the diagram more complex

Collaboration Diagram:

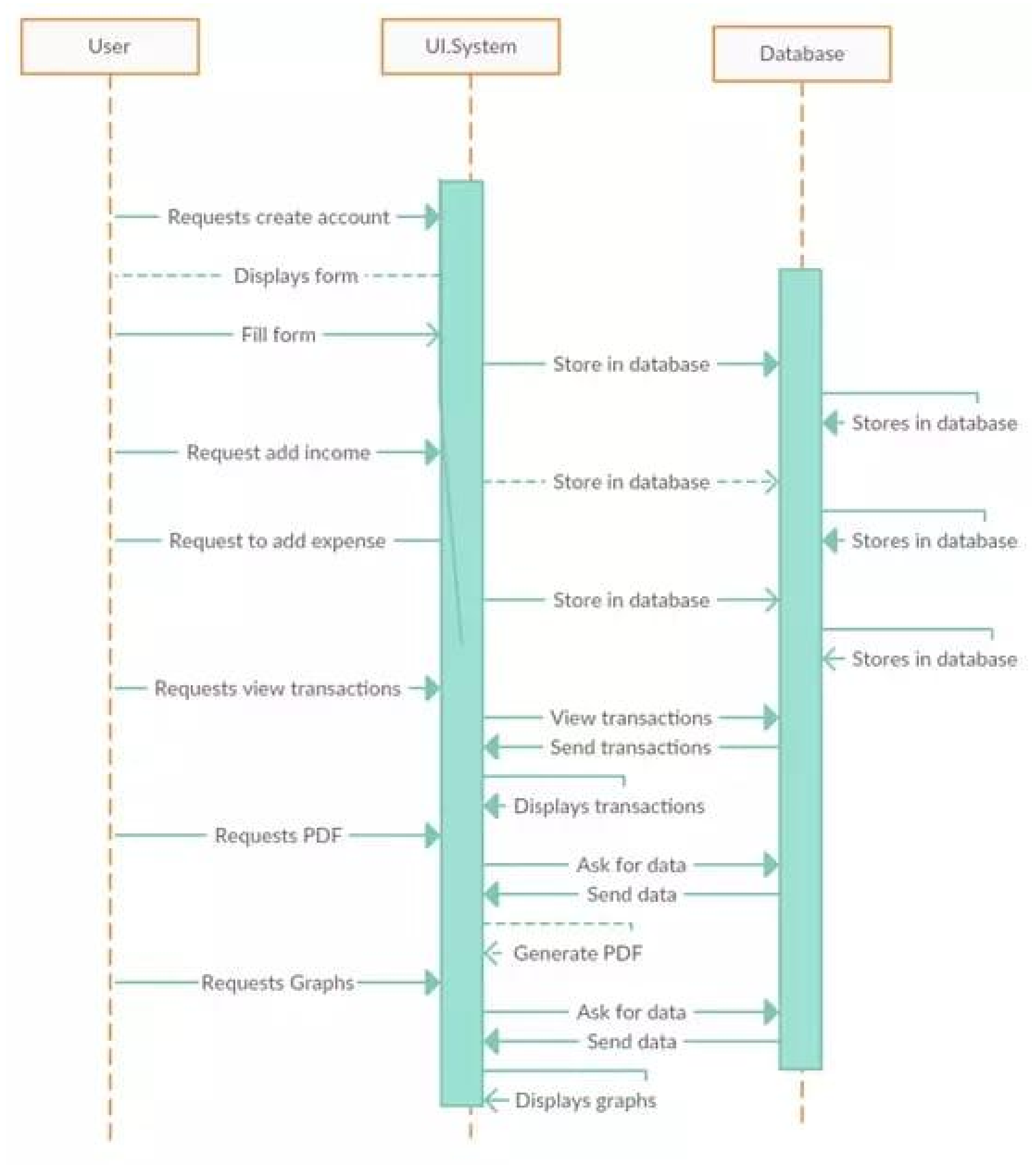
A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). Developers can use these diagrams to portray the dynamic behavior of a particular use case and define the role of each object.

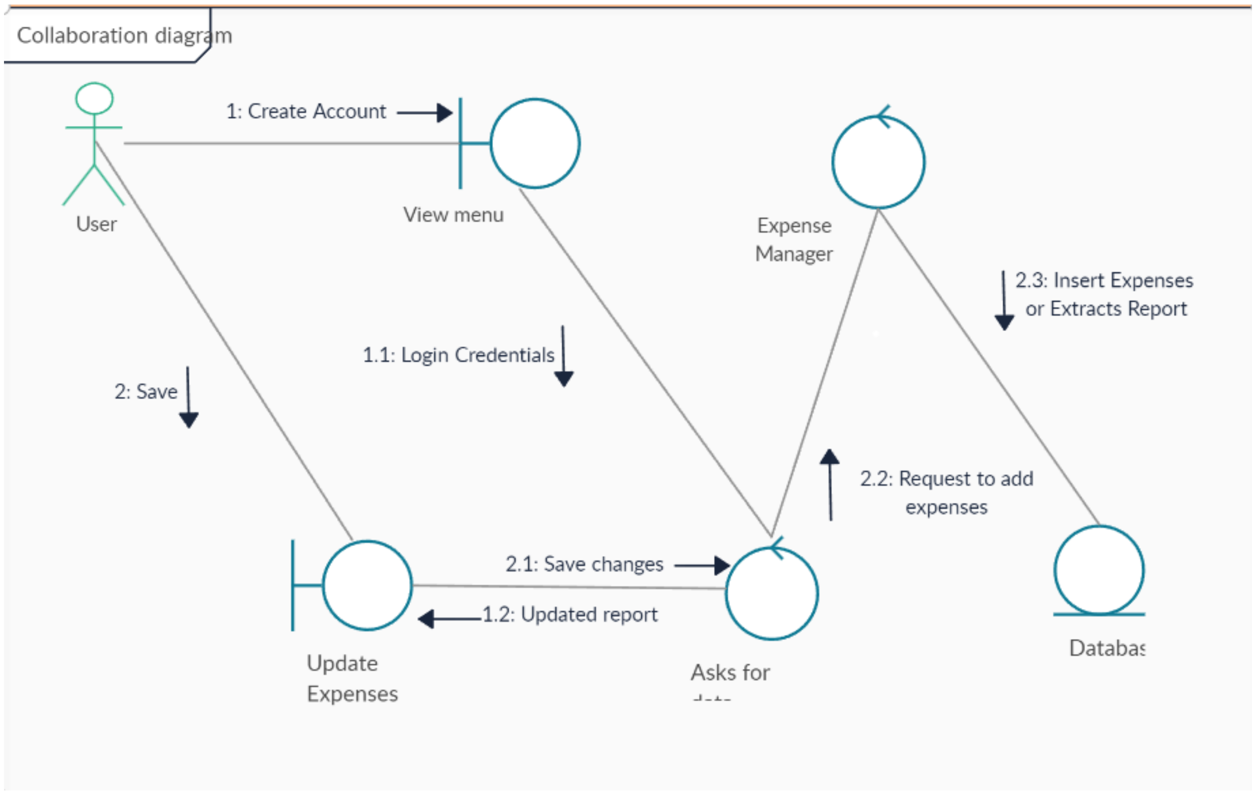
Advantages of collaboration Diagram :

Collaboration diagrams allow you to see both the dynamic aspects and static relationships of business objects in a single diagram. Often, business objects need to call on the services of other business objects to accomplish a particular task.

The drawback of a Collaboration Diagram:

It is a time-consuming diagram. After the program terminates, the object is destroyed. As the object state changes momentarily, it becomes difficult to keep an eye on every single thing that has occurred inside the object of a system.




Collaboration diagram
1: Create Account
User
View menu
Expense Manager
2.3: Insert Expenses or Extracts Report
1.1: Login Credentials
2: Save
2.2: Request to add expenses
2.1: Save changes
1.2: Updated report
Update Expenses
Asks for
Databas

DEVELOPMENT OF TESTING FRAMEWORK/USER INTERFACE :

Scope:

The scope of testing the software application for a budget tracking system would involve verifying that the system is functioning as per the requirements and meeting the end-user expectations. The testing process would include the verification of features such as user interface, data entry, data processing, and generating reports. It would also involve testing the application's compatibility with different operating systems, browsers, and devices.

Objective:

The primary objective of testing the software application for a budget tracking system is to ensure that the system is bug-free and performs as per the desired specifications. The testing process will help identify defects, issues, and risks that need to be addressed before the application goes live. Additionally, it will ensure that the system meets the end-user's needs and requirements.

Approach:

The approach to testing the software application for a budget tracking system would include the following steps:

Requirements Analysis: Analyzing and understanding the functional and non-functional requirements of the system.

Test Planning: Creating a test plan that outlines the testing strategy, scope, objectives, test cases, and timelines.

Test Environment Setup: Setting up the test environment with the required hardware, software, and data.

Test Case Design: Developing test cases that cover all the functional and non-functional requirements of the system.

Test Execution: Executing test cases and recording the results.

Defect Management: Documenting and tracking defects found during testing and retesting the fixed defects.

Reporting: Preparing reports that summarize the test results, including the number of defects found, defect severity, and the status of the testing process.

Sign-off: Obtaining sign-off from stakeholders that the software application for the budget tracking system is ready for release.

By following the above approach, we can ensure the software application for the budget tracking system is thoroughly tested, and any issues are identified and addressed before the system is deployed to end-users.

Scope of testing:

The scope of testing for a budget tracking system would typically include the following areas:

1. **Functionality Testing:** Testing the software application's functionality to ensure that it meets the specified requirements, performs as expected, and delivers accurate results. This would include verifying data entry, data processing, and report generation features.

2. **Performance Testing:** Testing the software application's performance to determine its speed, scalability, and responsiveness. This would involve measuring response times, load testing, and stress testing to identify and resolve performance bottlenecks.

3 **Compatibility Testing:** Testing the software application's compatibility with different operating systems, browsers, and devices to ensure that it works seamlessly across all platforms.

4. **Security Testing:** Testing the software application's security features to ensure that it is safe and secure from unauthorized access, data breaches, and cyber-attacks.

5. **Usability Testing:** Testing the software application's user interface, user experience, and ease of use to ensure that it is intuitive, user-friendly, and meets the end-user's needs and expectations.

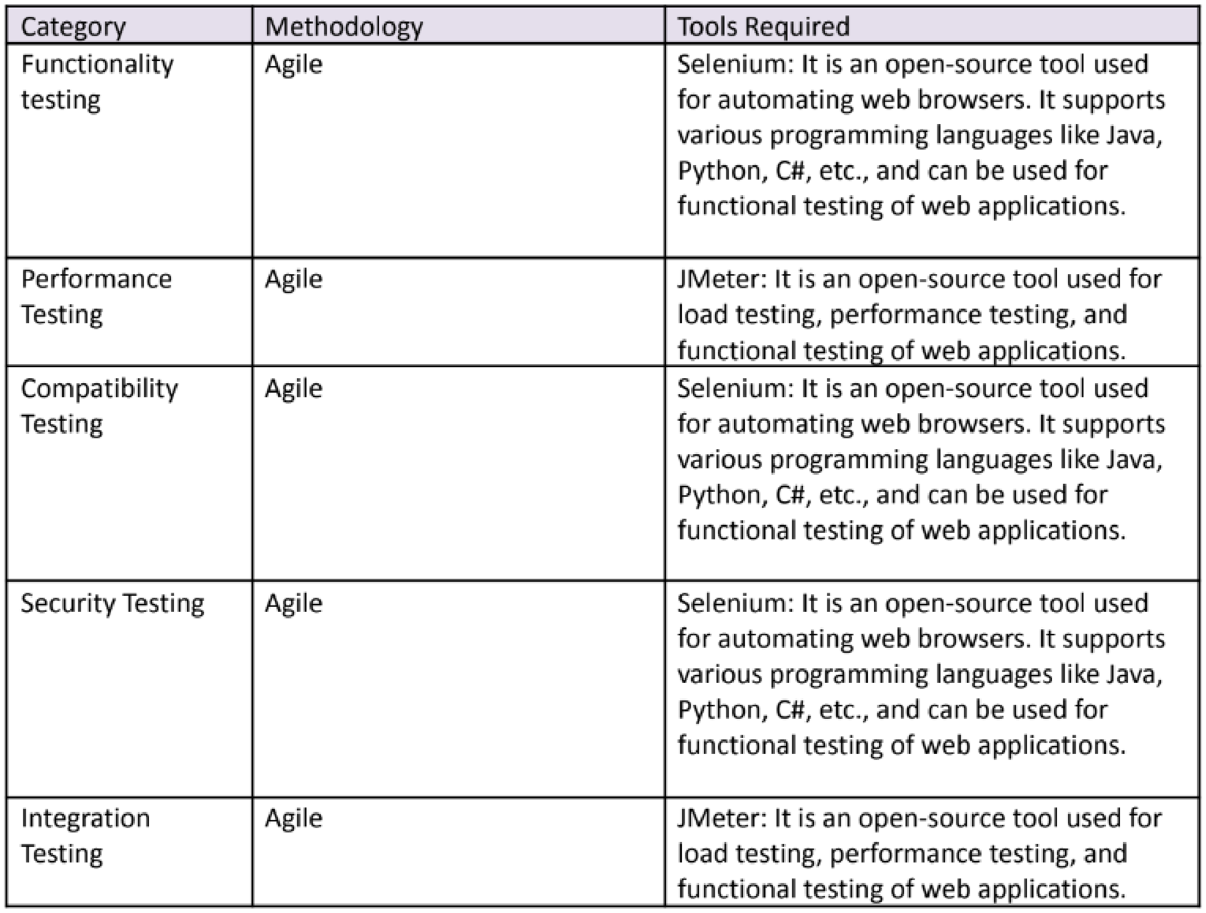
6. **Integration Testing**: Testing the software application's integration with other systems and third-party applications to ensure that it works seamlessly with other software and services.

7. **Regression Testing:** Testing the software application to ensure that new features, updates, or fixes do not adversely affect existing functionality.

8. **User Acceptance Testing:** Testing the software application with end-users to ensure that it meets their needs, requirements, and expectations.

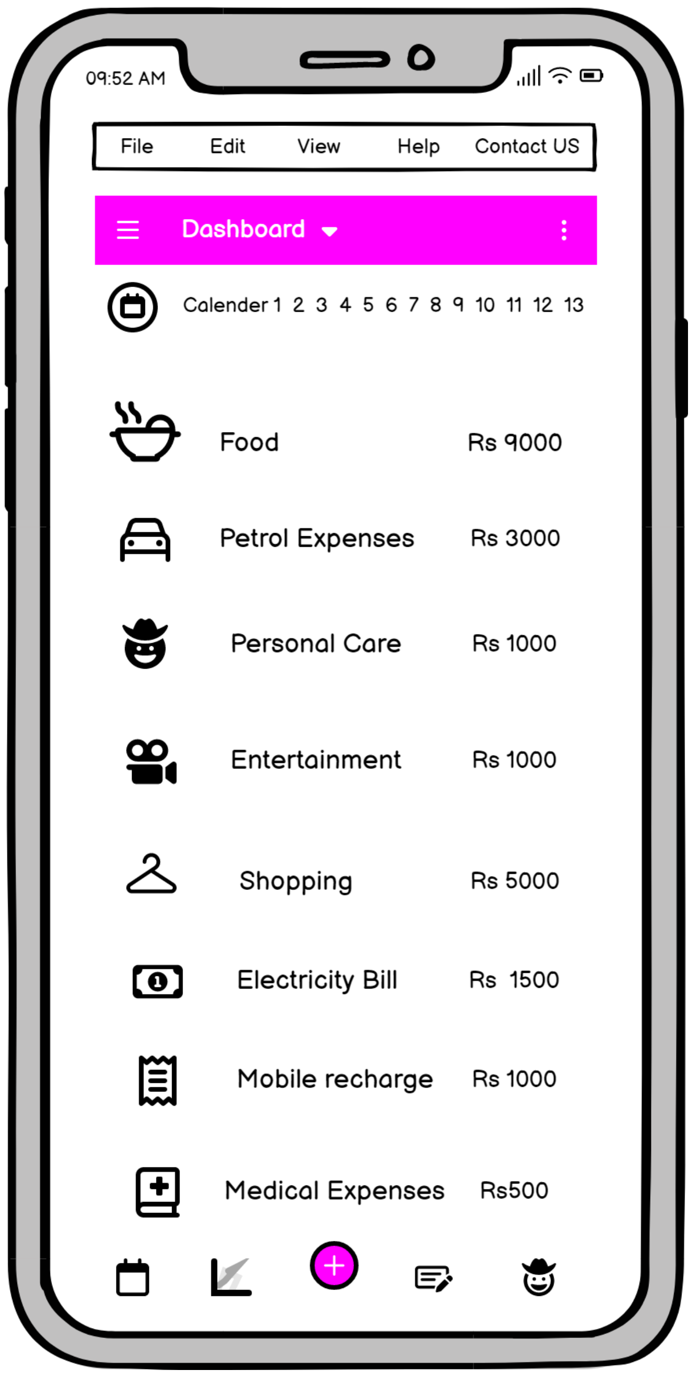
By testing the budget tracking system in these areas, we can ensure that the software application is thoroughly tested, meets all the requirements, and is ready for deployment to end-users.

Types of Testing, Methodology, Tools :



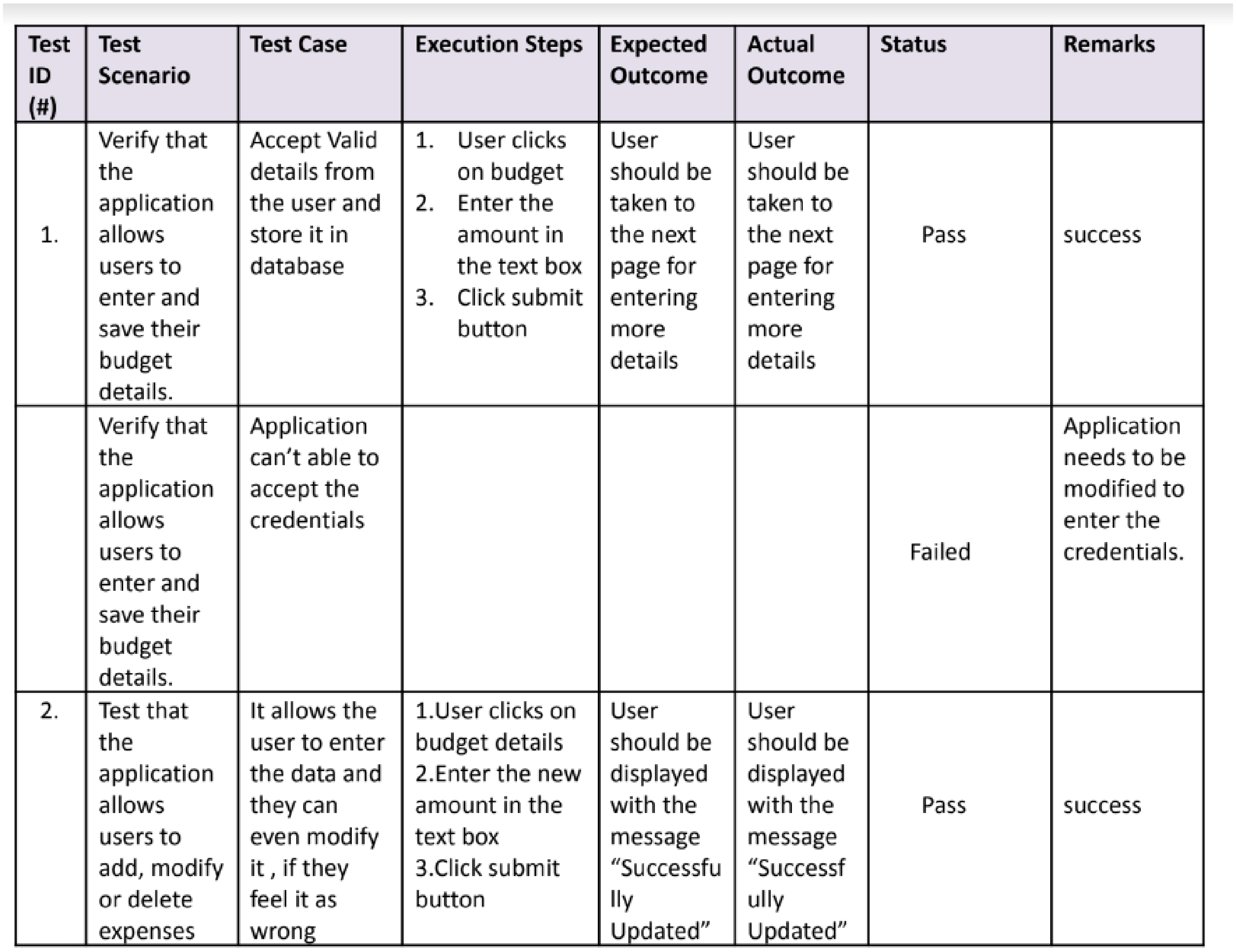
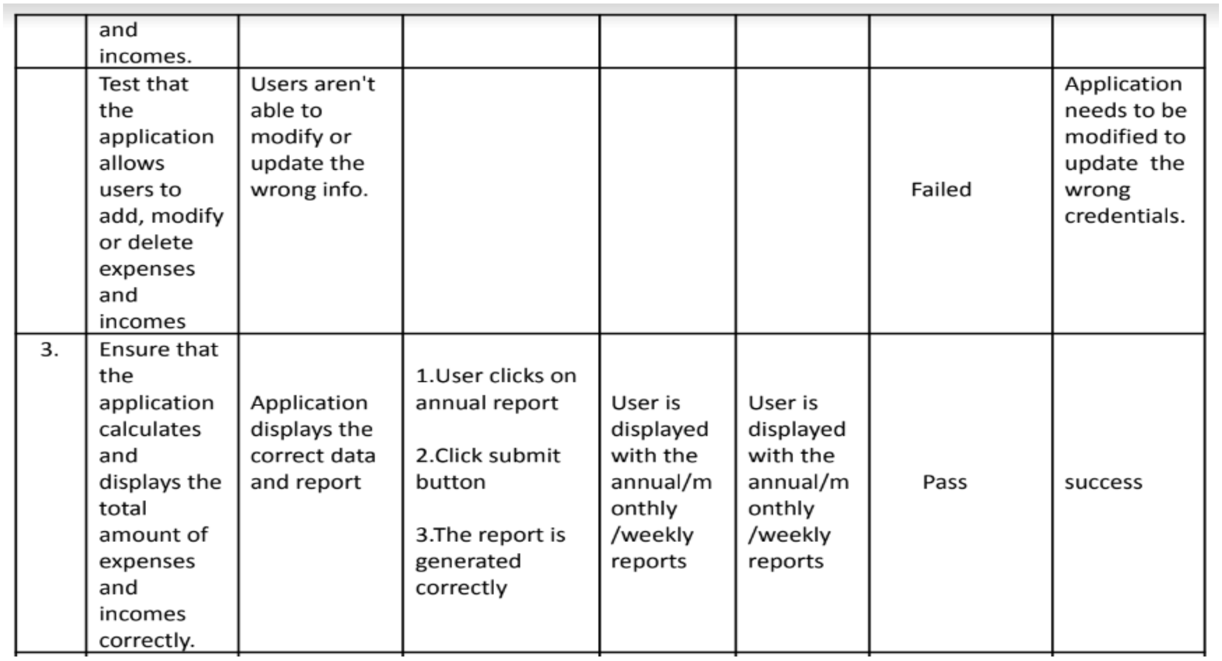
Category	Methodology	Tools Required
Functionality testing	Agile	Selenium: It is an open-source tool used for automating web browsers. It supports various programming languages like Java, Python, C#, etc., and can be used for functional testing of web applications.
Performance Testing	Agile	JMeter: It is an open-source tool used for load testing, performance testing, and functional testing of web applications.
Compatibility Testing	Agile	Selenium: It is an open-source tool used for automating web browsers. It supports various programming languages like Java, Python, C#, etc., and can be used for functional testing of web applications.
Security Testing	Agile	Selenium: It is an open-source tool used for automating web browsers. It supports various programming languages like Java, Python, C#, etc., and can be used for functional testing of web applications.
Integration Testing	Agile	JMeter: It is an open-source tool used for load testing, performance testing, and functional testing of web applications.

USER INTERFACE :



Food	Rs 9000
Petrol Expenses	Rs 3000
Personal Care	Rs 1000
Entertainment	Rs 1000
Shopping	Rs 5000
Electricity Bill	Rs 1500
Mobile recharge	Rs 1000
Medical Expenses	Rs500

TEST CASES & REPORTING :

Test ID (#)	Test Scenario	Test Case	Execution Steps	Expected Outcome	Actual Outcome	Status	Remarks
1.	Verify that the application allows users to enter and save their budget details.	Accept Valid details from the user and store it in database	1. User clicks on budget 2. Enter the amount in the text box 3. Click submit button	User should be taken to the next page for entering more details	User should be taken to the next page for entering more details	Pass	success
	Verify that the application allows users to enter and save their budget details.	Application can't able to accept the credentials				Failed	Application needs to be modified to enter the credentials.
2.	Test that the application allows users to add, modify or delete expenses and incomes.	It allows the user to enter the data and they can even modify it , if they feel it as wrong	1.User clicks on budget details 2.Enter the new amount in the text box 3.Click submit button	User should be displayed with the message "Successfu lly Updated"	User should be displayed with the message "Successf ully Updated"	Pass	success
	Test that the application allows users to add, modify or delete expenses and incomes	Users aren't able to modify or update the wrong info.				Failed	Application needs to be modified to update the wrong credentials.
3.	Ensure that the application calculates and displays the total amount of expenses and incomes correctly.	Application displays the correct data and report	1.User clicks on annual report 2.Click submit button 3.The report is generated correctly	User is displayed with the annual/m onthly /weekly reports	User is displayed with the annual/m onthly /weekly reports	Pass	success

	Ensure that the application calculates and displays the total amount of expenses and incomes correctly.	Application does not display the correct data and report				Failed	Application needs to be modified to generate the correct report.
4.	Test that the application has a feature to set reminders for upcoming expenses and to notify the user when the budget is exceeded.	Allow access to manage notifications	1.User allows to notify 2.Click on save changes 3.The notification will be received.	Received the notificatio n on-time	Received the notificatio n on-time	Pass	success
	Test that the application has a feature to set reminders for upcoming expenses and to notify the user when the budget is exceeded.	The application does not have a feature to pop notifications				Failed	Application needs to be modified to send the notification.

Non-Functional Test Cases:

Test ID (#)	Test Scenario	Test Case	Expected Outcome	Status	Remarks
1.	Verify that the application is responsive and performs well under different user load scenarios.	The application is responsive and performs well	The application is responsive and performs well	Pass	success
2.	Test that the application is secure, and user data is protected from unauthorized access.	The data's are secure and safe	The data's are secure and safe	Pass	success
3.	Test that the application is compatible with different web browsers, mobile devices, and operating systems.	This application can be run on any web browser in any devices	This application can be run on any web browser in any devices	Pass	success
4.	Ensure that the application is easy to use, has intuitive user interface and clear error messages.	The application has smooth UI/UX	The application has smooth UI/UX	Pass	success

ARCHITECTURE/DESIGN/FRAMEWORK/IMPLE-MENTATION

In general, the status of testing for a budget tracking system should be regularly monitored to ensure that the system is functioning as intended and that any defects or issues are identified and addressed promptly. Testing should cover both functional and non-functional requirements, including user acceptance testing, regression testing, security testing, and performance testing. Test results should be documented and tracked to ensure that all defects are resolved, and the system meets the desired quality standards.

There can be several obstacles that may arise while developing a budget tracking system, and some ways to rectify them are:

1. Complex Business Logic: Developing a budget tracking system requires understanding complex business logic, including financial calculations, budget allocation, and tracking expenses. One way to rectify this is by breaking down complex tasks into smaller ones and using modular code that can be tested and reused.

2. Data Security: Budget tracking systems may contain sensitive financial data that requires a high level of security. To rectify this, developers should ensure that the system has secure authentication and authorization mechanisms, encryption of data at rest and in transit, and regular security testing to identify and fix any vulnerabilities.

3. Integration with External Systems: Budget tracking systems may need to integrate with other systems, such as accounting software or payment gateways. Integration can be challenging, as it requires ensuring compatibility between different systems, data mapping, and handling errors. To rectify this, developers should use standardized APIs, perform extensive testing, and have a well-defined integration plan.

4. User Experience: A budget tracking system needs to be user-friendly, easy to navigate, and provide a positive user experience. To rectify this, developers should conduct user testing and feedback sessions, use consistent UI design patterns, and ensure that the system is responsive across different devices.

5. Performance: A budget tracking system needs to perform well, even under heavy loads. To rectify this, developers should optimize the code, database queries, and server configurations, use caching and load balancing techniques, and perform regular load testing to identify and resolve performance bottlenecks.

In summary, the key to rectifying obstacles during the development of a budget tracking system is to have a well-planned development process, test rigorously, and iterate frequently based on user feedback.

Category	Progress Against Plan	Status
Functional Testing	Green(As per the plan)	Completed
Non-Functional Testing	Amber	In Progress

Functional	Test Case Coverage (%)	Status
HTML	90	Completed
CSS	90	Completed
JS	90	In Progress

Here is a general architectural design for a budget tracking system:

Presentation layer: This layer is responsible for handling the user interface and user experience of the system. It includes web pages, forms, and other visual elements that users interact with. This layer can be built using a web framework likeHTML,CSS,JS

Application layer: This layer contains the core business logic and data processing components of the system. It is responsible for handling user requests, processing data, and communicating with the data access layer. This layer can be built using a server-side framework like Django, MySQL

Data access layer: This layer is responsible for accessing and manipulating data from the database. It includes an object-relational mapping (ORM) tool like Hibernate or SQLAlchemy to interact with the database. The database can be a relational database like MySQL or a NoSQL database like MongoDB.

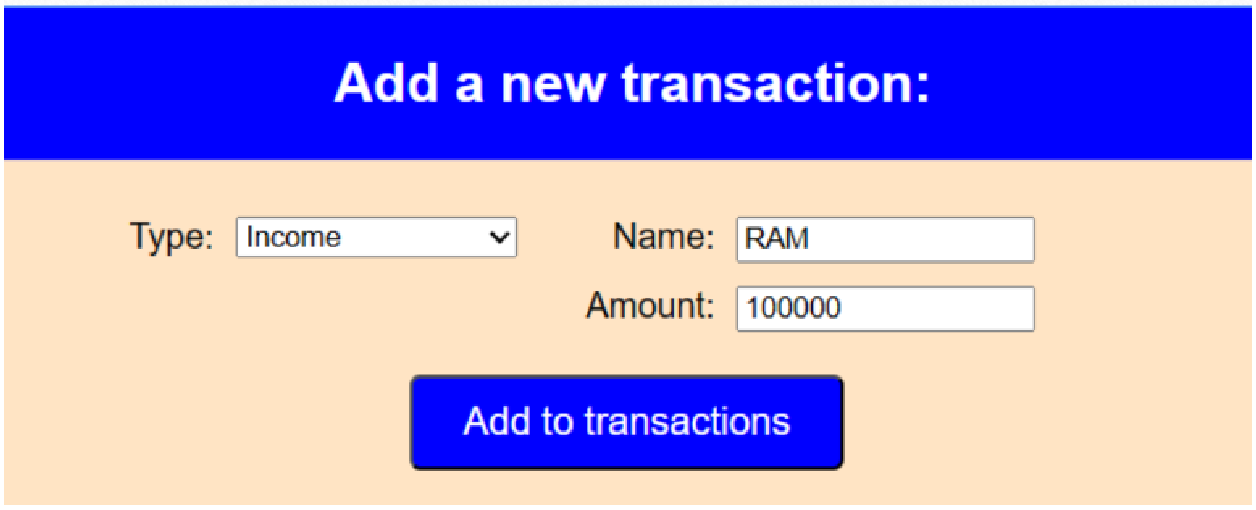
Integration layer: This layer connects the budget tracking system with other third-party systems or services, such as payment gateways, accounting systems, or email services. It can use APIs, webhooks, or other integration methods.

Infrastructure layer: This layer includes the hardware and software infrastructure required to run the system, including servers, network infrastructure, and cloud-based services like AWS or Azure. It can also include monitoring and logging tools to track system performance and identify issues.

Security layer: This layer includes various security measures to protect the system from unauthorized access or attacks, including encryption, user authentication and authorization, and secure coding practices. Overall, the design should be scalable, flexible, and modular, with clear separation of concerns between different layers. It should also be well-documented and tested to ensure it meets the functional and non-functional requirements.

Overall, the design should be scalable, flexible, and modular, with clear separation of concerns between different layers. It should also be well-documented and tested to ensure it meets the functional and non-functional requirements.

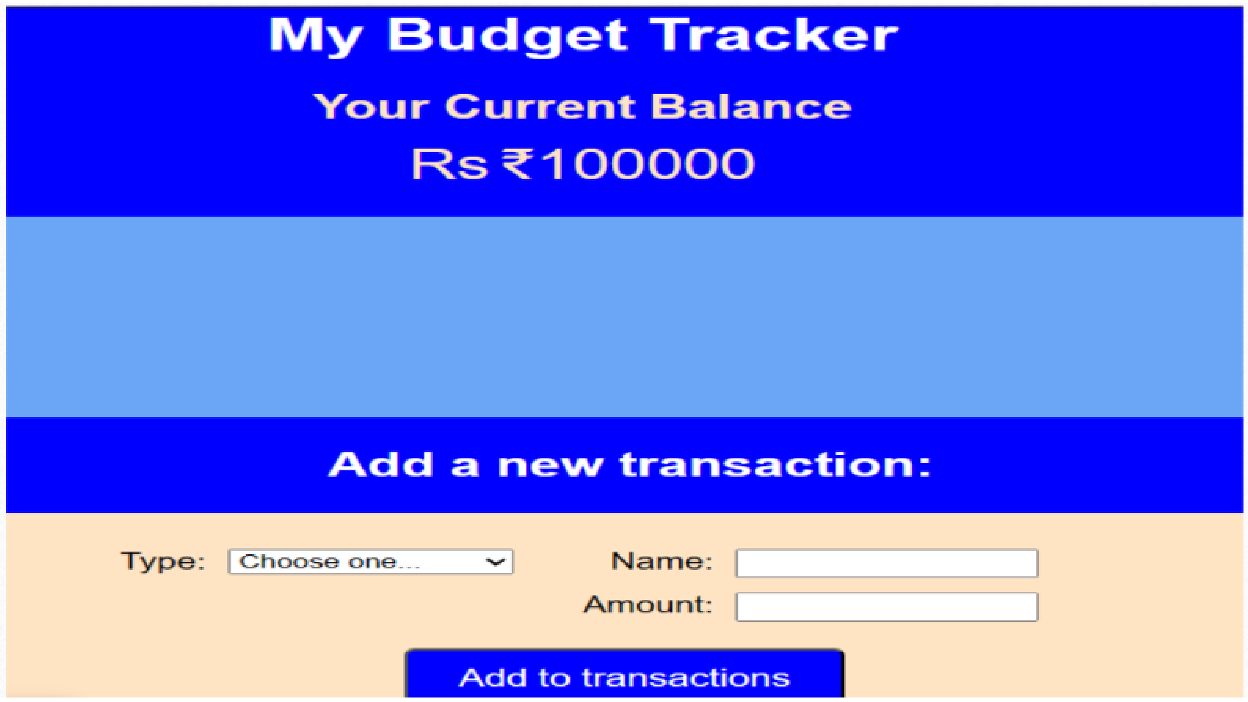
IMPLEMENTATION :



Type: Income

Name: RAM

Amount: 100000



My Budget Tracker

Your Current Balance

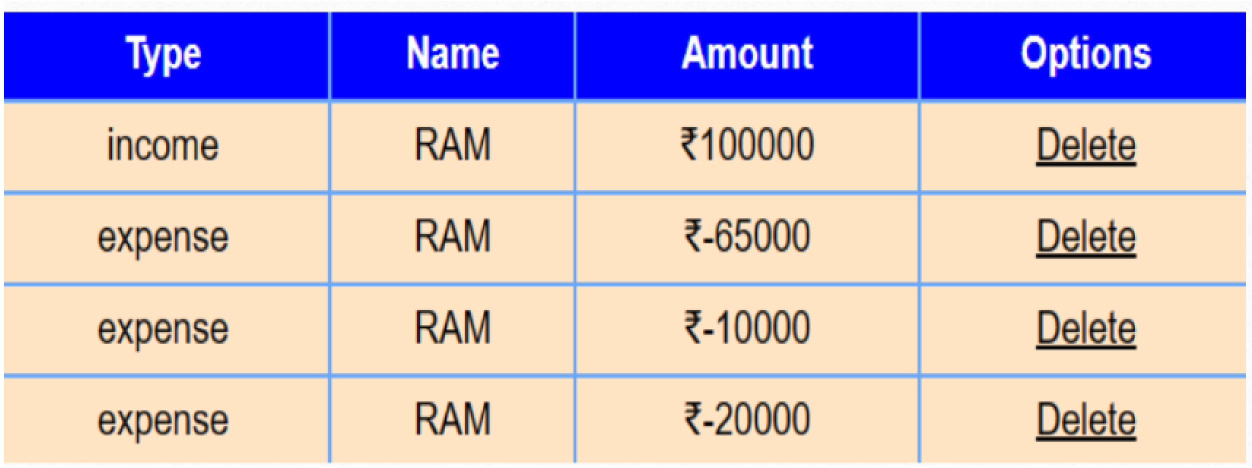
Rs ₹100000

Add a new transaction:

Type: Choose one...   Name:

Amount:

Add to transactions



Type	Name	Amount	Options
income	RAM	₹100000	Delete
expense	RAM	₹-65000	Delete
expense	RAM	₹-10000	Delete
expense	RAM	₹-20000	Delete

SOURCE CODE :

HTML:
```
<!DOCTYPE html>
<html>
<head>
   <title>Budget Tracker</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <div class="headerBar">
      <header>
         <h1 class="title">My Budget Tracker</h1>
         <h2 class="topbar">Your Current Balance</h2>
         <p>
            <span class="currency">Rs</span>
            <span class="balance"></span>
         </p>
         <header>
   </div>
   <div class="content">
      <h3 class="secondTitle">Add a new transaction: </h3>
      <div class="form">
         <form id="expForm">
            <div class="formLine left">
               <span for="type">Type:</span>
               <select id="type">
                  <option value="chooseOne">Choose one...</option>
                  <option value="income">Income</option>
                  <option value="expense">Expense</option>
               </select>
            </div>
            <div class="formLine right">
               <span for="name">Name:</span>
               <input type="text" id="name">
            </div>

            <div class="formLine right">
               <span for="amount">Amount:</span>
               <input type="text" id="amount">
            </div>
            <button type="submit" class="buttonSave">Add to transactions</button>
         </form>
      </div>
   </div>
   <div class="content">
```

```
        <table class="table">
            <thead>
                <tr>
                    <th>Type</th>
                    <th>Name</th>
                    <th>Amount</th>
                    <th>Options</th>
                </tr>
            </thead>
            <tbody id="transactionTable"></tbody>
        </table>
    </div>

    <script src="script.js"></script>
</body>
</html>
```

CSS :

```
 * {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
}

body {
  min-height: 1000px;
  display: flex;
  flex-direction: column;
  background-color: rgb(106, 166, 245);
  color: black;
}

.headerBar {
  background-color: blue;
  color: bisque;
  text-align: center;
  padding: 20px;
}

.title {
  margin-bottom: 20px;
  color: white;
}

.topbar {
  margin-bottom: 10px;
```

```
}

.currency {
 font-size: 30px;
 font-weight: 300;
}

.balance {
 font-size: 30px;
 font-weight: 300;
}

.content {
 width: 580px;
 margin: 0 auto;
 padding: 3%;
 padding-left: 6%;
}

.secondTitle {
 background-color: blue;
 color: white;
 text-align: center;
 margin-top: 100px;
 padding: 20px;
 font-size: 25px;
}

.form {
 padding: 5px;
 padding-top: 20px;
 padding-left: 10%;
 justify-content: center;
background-color: bisque; }

.formLine {
 display: inline-flex;
 padding: 5px 0px;
}

.left {
 float: left;
}

.right {
 float: right;
 margin-right: 100px;
```

```
}

input,
select {
  width: 130px;
  margin-left: 10px;
}

/* table style */
table {
  width: 100%;
}

thead {
  background-color: blue;
  color: white;
  line-height: 30px;
}

tbody {
  background-color: bisque;
  line-height: 30px;
  text-align: center;
}

/* Button */

button {
  width: 200px;
  color: #fff;
  padding: 10px;
  text-align: center;
  font-size: 1.1em;
  line-height: 20px;
  background-color: blue;
  border-radius: 5px;
  margin: 14px 25%;
  cursor: pointer;
}

button:hover {
  box-shadow: 0 0 0 2px grey;
  transition: 0.5s;
}

a {
  text-decoration: underline;
  cursor: pointer;
}
```

JS :

```
document.getElementById('expForm').addEventListener('submit', addTransaction);
// initial array of transactions, reading from localStorage
const transactions = JSON.parse(localStorage.getItem('transactions')) || [];

function addTransaction(e) {
   e.preventDefault();

   // get type, name, and amount
   let type = document.getElementById('type').value;
   let name = document.getElementById('name').value;
   let amount = document.getElementById('amount').value;
   if (type != 'chooseOne'
      && name.length > 0
      && amount > 0) {
      const transaction = {
         type,
         name,
         amount,
         id: transactions.length > 0 ? transactions[transactions.length - 1].id + 1 : 1,
      }

      transactions.push(transaction);
      // localStorage
      localStorage.setItem('transactions', JSON.stringify(transactions));
   }

   document.getElementById('expForm').reset();
```

```
    showTransactions();
    updateBalance();
}

const showTransactions = () => {

    const transactionTable = document.getElementById('transactionTable');

    transactionTable.innerHTML = '';

    for (let i = 0; i < transactions.length; i++) {
        transactionTable.innerHTML += `
            <tr>
                <td>${transactions[i].type}</td>
                <td>${transactions[i].name}</td>
                <td>$${transactions[i].amount}</td>
                <td><a class="deleteButton" onclick="deleteTransaction(${transactions[i].id})">
                    Delete</td>
            </tr>
        `;
    }
}


const deleteTransaction = (id) => {
    for (let i = 0; i < transactions.length; i++) {
        if (transactions[i].id == id) {
            transactions.splice(i, 1);
        }
    }
    // localStorage
    localStorage.setItem('transactions', JSON.stringify(transactions));
    showTransactions();
    updateBalance();
}

const updateBalance = () => {
    let balance = 0;

    transactions.forEach((transaction) => {
        if (transaction.type === "income") {
            balance += Number(transaction.amount);
        } else if (transaction.type === "expense") {
            balance -= transaction.amount;
        }
    });

    document.querySelector(".balance").textContent = balance;
}
```

CONCLUSION :

In conclusion, a budget tracking system is a useful tool for managing personal or business finances. By keeping track of income and expenses, it allows individuals and organizations to better understand their financial situation and make informed decisions about spending and saving. With a budget tracking system, it is easier to identify areas where spending can be reduced or optimized, and to set financial goals for the future. Overall, implementing a budget tracking system can help improve financial stability and reduce stress related to money management

In this paper, After making this application we assure that this application will help its users to manage the cost of their daily expenditure. It will guide them and make them aware about their daily expenses. It will prove to be helpful for the people who are frustrated with their daily budget management, irritated because of the amount of expenses and wish to manage money and to preserve the record of their daily cost which may be useful to change their way of spending money. In short, this application will help its users to overcome the wastage of money.

REFERENCES :

- **Google for problem solving**
- https://www.w3schools.com/html/
- https://www.w3schools.com/css/
- https://www.w3.org/Style/CSS/Overview.en.html
- https://www.w3schools.com/js/
- https://developer.mozilla.org/en-US/docs/Web/CSS
- https://web.dev/learn/css/
- https://codepen.io/