**JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY**

**BSc COMPUTER SCIENCE**

**DESIGN AND IMPLEMENTATION OF COMPUTER APPLICATION**

**EXPENSE TRACKING APPLICATION**

**GROUP MEMBERS:**

DAVID NZAMBULI: SCT 211-0068/2022

IAN NDOLO: SCT 211-0034/2022

NEEMA OGAO: SCT 211-0086/2022

MAUREEN NYAGA: SCT 211-0052/2022

PHARIS KARIUKI: SCT 211-0033/2022

**Introduction**

An expense tracker management system is a software tool designed to provide users with a user-friendly platform to monitor and manage their expenses effectively in a more coherent and manageable way. The app aims to streamline the process of expense tracking, offering features that simplify budgeting, categorizing expenses, and generating insightful reports. In today’s fast-paced world, managing personal finances efficiently is crucial for individuals and businesses alike because of the multitude of expenses incurred daily.

**Target Audience**

The Expense Tracking App caters to a wide range of users including:

* Individuals looking to track their personal expenses, manage budgets, and improve their financial habits.
* Families looking to manage their household finances, track shared expenses, and save for future goals.
* Small businesses and startups in need of a simple yet effective expense management solution.
* Freelancers and self-employed professionals seeking to manage business expenses.
* Educational institutions wanting to educate students about personal finance and money management.
* Government agencies responsible for managing public funds, budget allocations, and expenditures.
* Nonprofit organizations that need to manage and track expenses for fundraising events, programs, and administrative operations.

**Functional Requirements**

The core objective of the Expense Tracking Application is to deliver a seamless and intuitive platform that empowers users to effortlessly monitor and manage their finances. To achieve this, the app should be able to perform the following key functionalities:

* Bank account and credit Card Integration: Users should have the ability to seamlessly connect their bank accounts and credit cards with the app for not only real-time monitoring of transactions but also for users to have a comprehensive overview of their financial activities.
* Automated transaction detection: The app should be able to automatically detect and import transactions conducted across the linked bank accounts and credit cards.
* Goal Setting and Budgeting: Users should be able to establish personalized financial goals and budgets within the app. The app will offer flexible budgeting features to accommodate varying financial objectives.
* Visual Progress Analysis: The app should leverage visual analytics to provide users with insightful and simple reports on their financial progress. Through interactive summaries, users can get an idea of their spending patterns, savings achievements, and adherence to budgetary goals.
* Customizable Notifications and Alerts: To keep the users informed of their progress and engaged with their financial goals, the app should offer customizable notifications and alerts.

**Non-functional requirements**

Non-functional requirements for the expense tracking app outline the characteristics that describe how the system should behave, rather than the specific features it should have.

They include:

* Performance- the app should respond to user interactions fast and efficiently to provide a seamless user experience. It should be able to handle a large number of concurrent users without fail.
* Security- the app should implement strong authentication mechanisms to ensure that only authorized users can access sensitive financial information. Security patches should be applied promptly to mitigate any potential risks.
* Reliability- The app should be highly reliable with minimal downtime and should have built-in error handling mechanisms to capture and report errors, ensuring timely resolution of issues.
* Scalability- the app should be designed to scale horizontally and vertically to accommodate increases in user traffic and data volume.
* Compatibility- the app should be compatible with a wide range of devices and operating systems, including desktop computers, laptops, tablets, and smartphones.
* Usability- the user interface should be easy to navigate, with clear labels, instructions, and feedback messages.

Software development tools we will need:

* Programming Language suitable for our app for example Python to implement the backend.
* Integrated Development Environment (IDE): Select an IDE for coding and debugging.
* Version Control: Use version control systems like Git for managing code changes.
* Database Management System (DBMS): Choose a DBMS to store and manage expense data for example MySQL

Hardware Development tools we will use:

* Development Machine: A powerful computer capable of running the selected development tools smoothly.
* Mobile Devices: For testing the app on real devices, you may need Android and iOS smartphones or tablets.
* Web Hosting Providers: If you have a web-based component, you'll need a hosting provider to deploy your backend services.
* Simulators: Use built-in simulators provided by development platforms for testing during development.

**Schedule**

***Week 1: Planning and Setup***

Day 1-2: Define project scope, goals, and target audience. Research competitors and identify unique selling points.

Day 3-4: Create wireframes and design prototypes for the app's interface and user flow.

Day 5-7: Set up a development environment, choose a technology stack, and establish a version control system (e.g., Git).

***Week 2: Core Functionality Development***

Day 8-10: Begin development of user authentication system and database architecture.

Day 11-14: Implement bank account and credit card integration using secure APIs. Develop transaction import functionality.

***Week 3: Feature Development***

Day 15-17: Build goal-setting and budgeting features, allowing users to set financial goals and create budgets.

Day 18-20: Develop visual progress analysis tools, including graphs and charts to visualize spending patterns and savings achievements.

Day 21-23: Implement customizable notifications and alerts system to keep users informed of their financial progress.

***Week 4: Testing and Refinement***

Day 24-26: Conduct thorough testing of all app features, including integration testing with bank APIs.

Day 27-28: Gather feedback from beta testers and make necessary refinements to the app's functionality and user experience.

Day 29-30: Finalize documentation, prepare for app store submission, and deploy the app for release.

**STAGES**

**Feasibility study**

***1. Technical Feasibility****:*

\***Technology Stack**: Evaluated the availability of suitable technologies and frameworks for developing the application, considering factors such as platform compatibility, scalability, and ease of integration.

**\*Expertise**: Assessed the technical skills and expertise of the development team or individual responsible for building the application. Determine if they have the necessary knowledge to implement the required features and functionalities.

**\*Infrastructure**: Considered the infrastructure requirements, including hosting, databases, and other resources needed to support the application's development and deployment.

***2. Operational Feasibility****:*

\***Resource Availability**: Evaluated the availability of resources, including human resources, time, and budget, required to develop and maintain the application.

**\*Integration**: Assessed the feasibility of integrating the application with existing systems or services, such as payment gateways or third-party APIs for currency conversion.

**\*Scalability**: Determined if the application can scale effectively to accommodate increasing user demand and data volume over time.

***3. Economic Feasibility****:*

**\*Cost-Benefit Analysis**: Conducted a cost-benefit analysis to determine the potential financial implications of developing the application. Consider factors such as development costs, maintenance costs, potential revenue streams, and expected return on investment.

**\*ROI**: Estimated the expected return on investment (ROI) based on projected revenue generation, cost savings, or other benefits derived from the application.

***4. Schedule Feasibility****:*

**\*Timeline**: Developed a project timeline outlining key milestones and deliverables to determine if the project can be completed within a reasonable timeframe.

**\*Dependencies**: Identified any dependencies or constraints that may impact the project schedule, such as external dependencies, resource availability, or regulatory requirements.

After, the study of all factors considered, the project was ready for commencement

**Design and Implementation**

1. ***Dashboard:***

•The dashboard serves as the home screen and provides a quick overview of the user's financial status.

•Include a summary of total expenses, income, and balance for the selected time period.

•It uses a clear and visually appealing charts or graphs to represent expense categories.

1. ***Navigation:***

• The expense tracker application uses a clean and simple navigation bar with easily recognizable icons for key features like Home, Transactions, Categories, Reports, and Settings.

• It implements a breadcrumb trail to show users their current location within the app. This is very useful when the user tries to find their way around the application.

1. ***Transaction Entry:***

• The application encompasses various entries with the main one being the transaction entry button. Add a prominent "Add Transaction" button on the dashboard or navigation bar.

• Include fields for date, amount, category, and a brief description of the expense.

• Enable users to upload receipts or attach images to transactions which can be optional.

1. ***Expense Categories:***

• Create a categorized list of common expenses, and allow users to customize categories to suit their needs: for example; entertainment, food, shopping, electricity bills

• The application uses color coding to make categories visually distinguishable.

• The user can set budgets to be implemented in each category of the expenses.

1. ***Transaction filtering:***

• Display a chronological list of transactions with relevant details.

• Enable users to filter and search for transactions based on date, category, or keywords.

• Provide options to edit, delete, or categorize transactions directly from the list.

1. ***Reports and Analytics:***

• Generate visually appealing charts and graphs to illustrate spending patterns over time.

• Include a budget tracker that compares actual spending against set budgets.

1. ***Settings:***

• Allow users to customize app settings, such as currency, date format, etc.

• Include security features like passcodes or biometric authentication.

1. ***Help and Support:***

• Include a feedback option to gather user suggestions or report issues.

**DATABASE DESIGN**

The expense tracker application has a relatively simple design interface and implementation making the database design approach easy to manage. It involves a structure that efficiently receives data from the user from their transaction entries, stores them using simple approaches, and retrieves data related to user transactions, categories, budgets, and other user-tailored information on the reminders of their transactions.

The approaches used in the database design and implementation include:

**I. User Table:**

•Creation of a table to store user information, including a unique user ID, username (First Name), and hashed passwords for security

• Include additional fields for the user preferences such as set reminders for certain transactions, currency format, etc.

**ii. Category Table:**

•This table stores expense categories, with a unique category ID and a user ID to associate categories the specific users.

**iii. Transaction table:**

•This table stores individual transactions linking each transaction to a specific user and expense category.

•Transaction ID, User ID, Category ID, amount, brief description, and a receipt image which may be optional.

**iv. Budget Table:**

• This table stores user-specific budgets for different expense categories

• It provides information on whether the user is overspending or exceeding certain budget constraints in specific expense categories

• It also stores the saving goal of specific users within their budget.

**v. Report Table:**

• Optional to have which stores historical data for reports and analytics, create a table to store aggregated information

SQL will be the most prominently used language to develop the relational tables associated with the various groups of tables to design the expense tracker computer application database design.

**PROGRAM DESIGN**

A program design of an expense tracker application involves planning the software architecture, defining modules, and outlining the functionality of each component. Here's a high-level description of a potential program design for the expense tracker:

1. ***User Authentication and Management:***

• Implement a user authentication system to allow users to register, log in, and manage their accounts.

• Utilize secure password hashing techniques to store user credentials.

1. ***Dashboard Module:***

• Develop a dashboard module to display a summary of the user's financial status.

• Retrieve and display total expenses, income, and balance for the selected time period.

• Integrate charts or graphs for a visual representation of expense categories.

1. ***Transaction Management:***

• Create modules for adding, editing, and deleting transactions.

• Implement validation checks to ensure data integrity (e.g., valid date formats, non-negative amounts).

• Allow users to attach receipts or images to transactions.

1. ***Category Management:***

• Develop modules for managing expense categories.

• Allow users to add, edit, and delete categories.

• Implement customization options for user-defined categories.

1. ***Budget Tracking:***

• Include functionality for setting and managing budgets for different expense categories.

• Provide notifications or alerts when users approach or exceed their budget limits.

1. ***Reporting and Analytics:***

• Create modules to generate reports and analytics based on user transactions.

• Include features such as spending trends, top expense categories, and comparisons against budget.

1. ***Search and Filter Functionality:***

• Implement a search functionality to allow users to find specific transactions.

• Include filters based on date range, category, or keywords.

1. ***Settings Module:***

• Develop a settings module for users to customize application preferences.

• Include options for currency format, date format, and notification settings.

1. ***Security Features:***

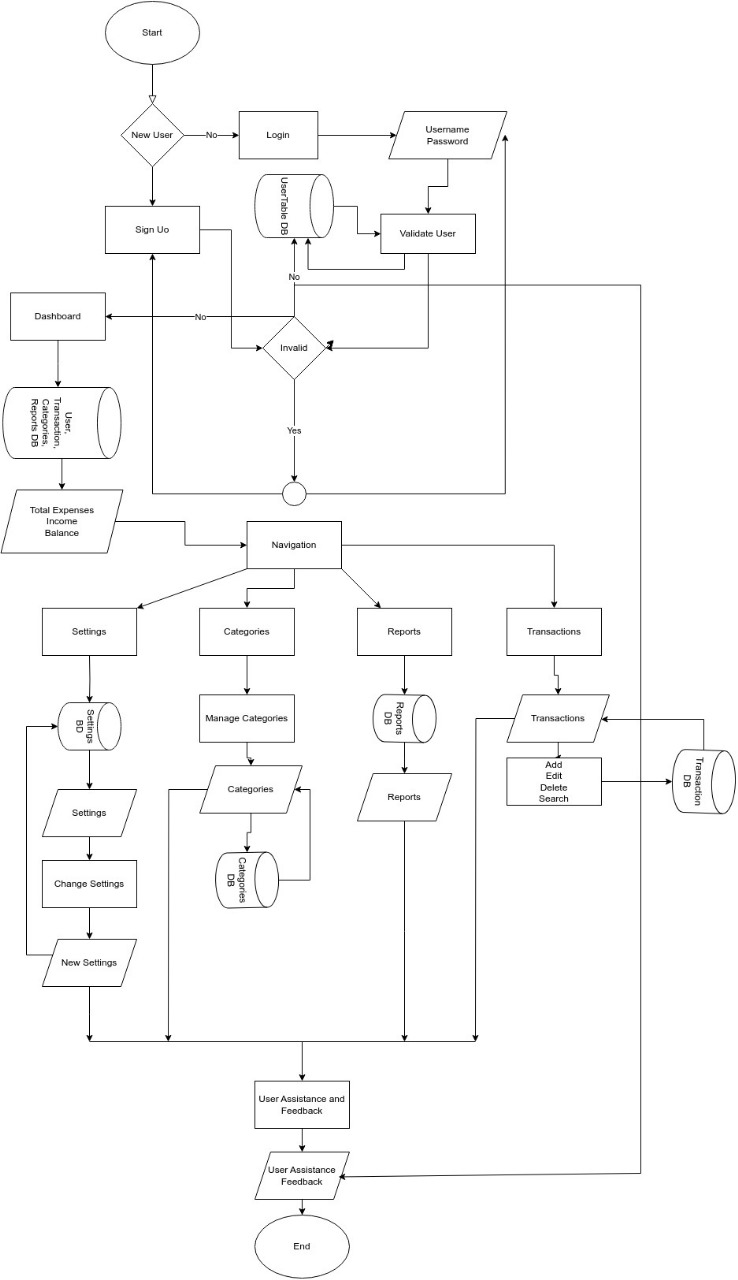
• Implement security measures, such as data encryption and secure connections, to protect user data.

• Include mechanisms to prevent common security threats like SQL injection and cross-site scripting.

1. ***User Assistance and Feedback:***

• Provide a help section or tutorial to guide users through the application.

• Include a feedback mechanism for users to report issues or suggest improvements.



**FLOW DIAGRAM**

**SEQUENCE DIAGRAMS**

The user opens the expense tracker application

The UI Controller presents the login screen

The user enters their credentials

The UI Controller sends the login request to the Expense Tracker System

The Expense Tracker System verifies the credentials against the database

The Expense Tracker System sends back a response indicating success or failure

The UI Controller displays the appropriate screen based on the response

**USER LOGIN SEQUENCE**

The user selects an option to add a new expense

The UI Controller presents a form for entering expense details

The user fills in the details and submits the form

The UI Controller sends the expense data to the Expense Tracker System

The UI Controller displays a confirmation message or error based on the response

The Expense Tracker System sends a response indicating success or failure

The Expense Tracker System validates and stores the expense data in the database

**EXPENSE MANAGEMENT SEQUENCE**

**EXPENSE RETRIVAL SEQUENCE**

**LOGOUT SEQUENCE**

The UI Controller displays the login screen again

The Expense Tracker System logs out the user

The UI Controller sends a logout request to the Expense Tracker System

The user decides to logout from the application

**SETTING BUDJET SEQUENCE**

The user enters the budget details (e.g., total budget amount, time period)

The UI Controller presents options for setting or modifying the budget

The user navigates to the budget management section

The Expense Tracker System sends a response indicating success or failure

The UI Controller displays a confirmation message or error based on the response

The Expense Tracker System validates and stores the budget data in the database.

The UI Controller sends the budget data to the Expense Tracker System

Periodically or upon expense entry, the Expense Tracker System checks the current expenses against the set budget

If the expenses exceed the budget threshold, the system sends a notification to the UI Controller

The UI Controller displays a warning message to the user, indicating that they are nearing or exceeding their budget

**BUDJET MONITORING SEQUENCE**

The UI Controller displays a confirmation message or error based on the response

**BUDJECT ADJUSTMENT SEQUENCE**

The Expense Tracker System sends a response indicating success or failure

The Expense Tracker System validates and updates the budget data in the database

The UI Controller sends the updated budget data to the Expense Tracker System

The UI Controller presents options for modifying the budget

The user makes changes to the budget details

The user decides to adjust the budget

**SETTINGS SEQUENCE**

The UI Controller presents options for updating various settings (e.g., currency, notification preferences).

The user modifies the desired settings.

The user navigates to the settings section.

**HELP SEQUENCE**

The UI Controller displays the help information to the user.

The Expense Tracker System sends the help information back to the UI Controller.

The Expense Tracker System retrieves relevant help information from the help database or resources.

The UI Controller sends the help request to the Expense Tracker System.

The user selects a specific help topic or requests general assistance.

The UI Controller presents options for accessing help resources (e.g., FAQs, user manual).

The user navigates to the help section or clicks on a help icon.

The UI Controller displays a confirmation message or error based on the response.

The Expense Tracker System sends a response indicating success or failure.

The Expense Tracker System validates and updates the settings data in the database.

The UI Controller sends the updated settings data to the Expense Tracker System