**TRANSACTION SYSTEMS**

**DOCUMENTATION**

**Introduction**

* Overview: Introduction to the monetary transaction system and its key features.
* Purpose: Explanation of the objectives and benefits of the system for users and businesses.
* Audience: The target audience comprises individuals who possess a mobile device and have a bank account.

**Getting Started**

* User Guide: Step-by-step instructions for setting up and configuring the monetary transaction system.

**Features**

1. Money Transfer
2. Autopay for Subscriptions
3. Budget Management
4. Bank Balance Maintenance
5. Payment Notification Email
6. Data Synchronisation(SQLite to Excel)
7. Security and compliance
8. View transaction history
9. View user data

**Use Cases**

1. **Money Transfer:**

Allows the users to electronically transfer or receive money from one's bank account to another. It consists of transactions like paying bills, transferring funds to friends or family or making purchases online

1. **Autopay for Subscriptions:**

Automates the process of paying recurring bills and subscriptions on a predetermined schedule. Once set up, the system automatically deducts the required amount from your account at regular intervals(eg. monthly) to cover subscription services like streaming platforms, utility bills or memberships.

1. **Budget Management:**

Helps manage finances by setting spending limits across different categories(eg. groceries,entertainment,utilities) and tracking the user's expenses against those limits. It helps the user to stay within their financial goals and make informed spending decisions.

1. **Bank Balance Maintenance:**

Keeps track of the balances in the user's bank accounts, providing real-time updates whenever there are transactions. It ensures that the user has an accurate and up-to-date view of their availability of funds ,helping them to manage their finances more effectively.

1. **Payment Notification Email:**

Sends email notifications to alert the user whenever there is activity on their account, such as making payments or receiving funds.These notifications provide confirmation and transparency regarding financial transactions, helping the user stay informed about their account activity even when the user is not logged into the system.

1. **Data Synchronisation (SQLite to Excel):**

Facilitates the transfer of financial data between the system's internal database (stored in SQLite format) and external Excel spreadsheets. It allows users to export their financial data from the system into Excel for further analysis, reporting or record-keeping purposes, ensuring data synchronization between the two platforms.

1. **Security and Compliance:**

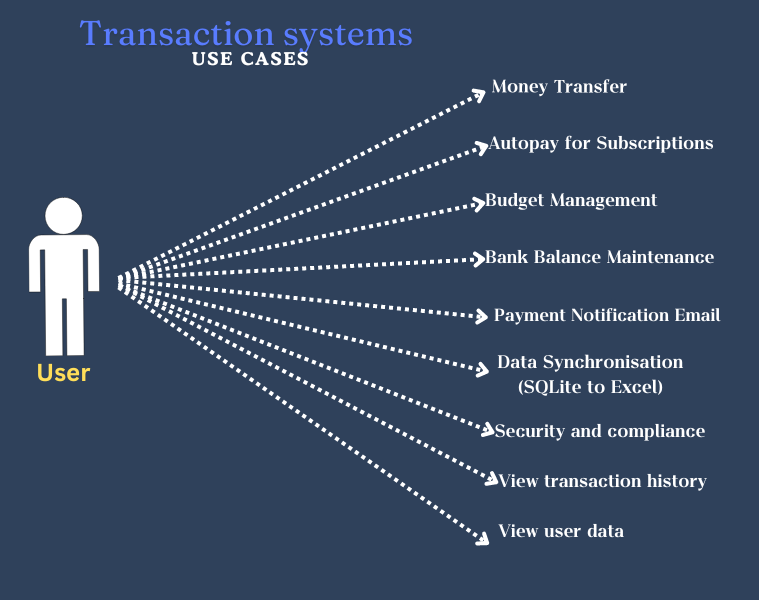
Implements security measures to protect sensitive financial information from unauthorized access, data breaches and fraud. It also ensures compliance with regulatory standards and industry best practices for data protection, privacy and security.

1. **View Transaction History:**

Provides users with a detailed record of all their past financial transactions within the system.It allows the users to review and track their transaction history over time, helping them monitor their spending, identifying patterns and reconcile their accounts.

1. **View User Data:**

Enables users to access and view their personal financial information stored within the system, such as their profile details, account balances,transaction history and other relevant data. It provides a centralized platform for users to manage and monitor their financial accounts, empowering them with insights into their financial health and activity.



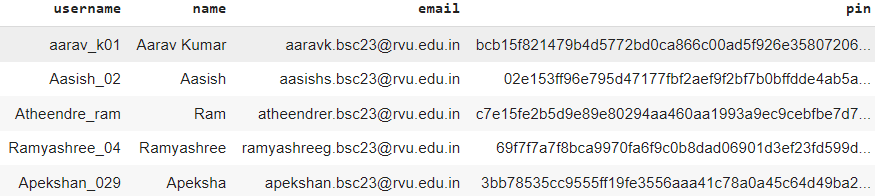
**Database Schema**

The database consists of 5 tables:

1. **Users table:**

* Username - Unique identifier for each user. (primary key)
* Name - Name of the user.
* Email - Email address associated with the user.
* Pin - 6 digit password needed for logging in by the user.

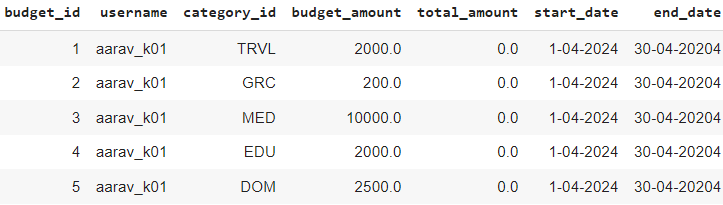
The pin is hashed for security purposes.



1. **Category table:**

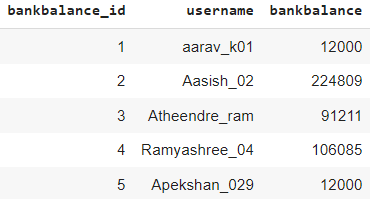
* Category ID - Unique identifier for each category. (primary key)
* Category name - The name of the category.

1. **Budget table:**

* Budget ID - An integer column serving as the primary key with auto-increment.
* Username - Username associated with the budget.(Foreign key)
* Category ID - Category ID associated with the budget. (Foreign key)
* Budget amount - Budget amount set by the user.
* Total amount - Total amount spent by the user within the budget.
* Start date - Start date of the budget period.
* End date - End date of the budget period.

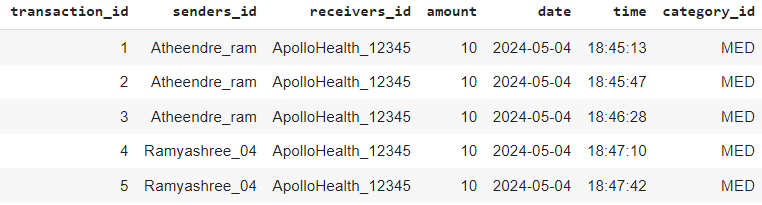
1. **Balance table:**

* Bank balance ID - Unique identifier for the bank balance of the user. (Primary key)
* Username - username associated with the bank balance. (Foreign key)
* Bank balance - Current bank balance associated with the user. Bank balance cannot be 0.



1. **Transactions table:**

* Transaction ID - Unique identifier of transaction associated with a user. (Primary key)
* Sender's ID - Username of the user who is sending the amount.(Foreign key)
* Receiver's ID - Username of the user who is receiving the money. (Foreign key)
* Amount - Transaction amount being transferred and received by the sender and the receiver respectively.
* Date - Date of the transaction being done.
* Time - Time at which the transaction is being done.
* Category ID - Category ID associated with the transaction amount ie, spending in particular categories. (Foreign key)



**Database Implementation**

Project relevant files:

* dbmsproj.ipynb - Contains the implemented code for the project.
* dbmsproj.sqlite - SQLite file used for the implementation of the project.
* transaction\_data.xlsx - Contains data of people initially generated and organized into respective sheets.
* Testing.ipynb - Contains the code for test cases for the implemented code in dbmsproj.ipynb
* Testing.sqlite - Database used for checking test cases in testing.ipynb
* Transaction Systems Documentation - It is a document where project relevant details are present.
* Records.xlsx - An excel file used to store the data after the implementation is done and also to reflect frequent updates done to the database.

Steps followed in the implementation:

1. Collection of data:

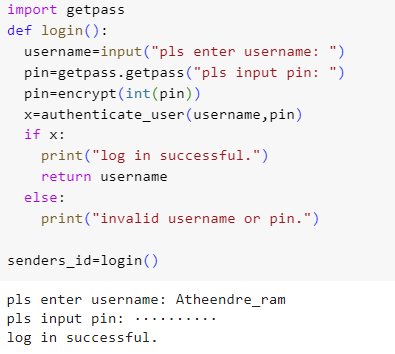
* Objective: Define what kind of data is needed and from where to collect.
* Activities:
* Discussions held on what all data is required and how to define the collected data.
* Collected data from various people who had made transactions through UPI.
* Used the collected data to define the database.

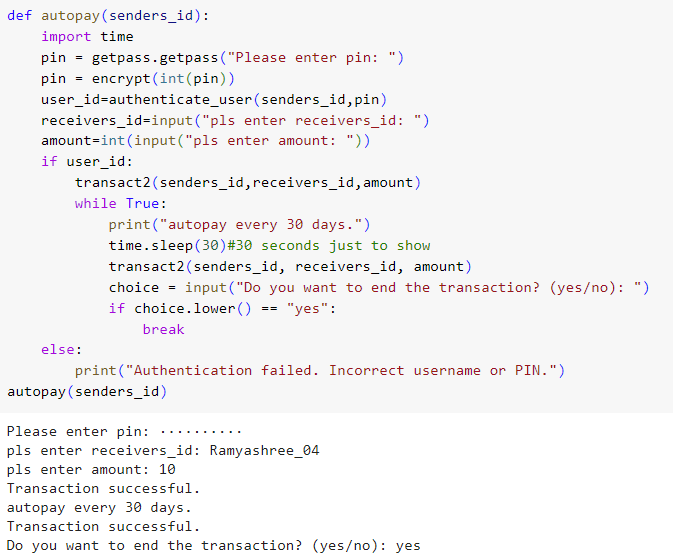
1. Preparation of data:

* Objective: To have ready, clean and transformed data for implementation purposes.
* Activities:
* The collected data was manually cleaned in the excel it was collected.
* Rest of the data was automated and made.

1. Implemented ideas:

* Objective: To implement different features related to transaction systems
* Activities:
* Using SQLite and python we implemented many features we came up with
* We have used various triggers to maintain the referential integrity of data.
* We have used python libraries like smtplib, getpass and openpyxl to implement features like sending mails after a transaction, for secure login using confidential pins and to update the changes in the database to excel file to access as reports later.



1. Updates:

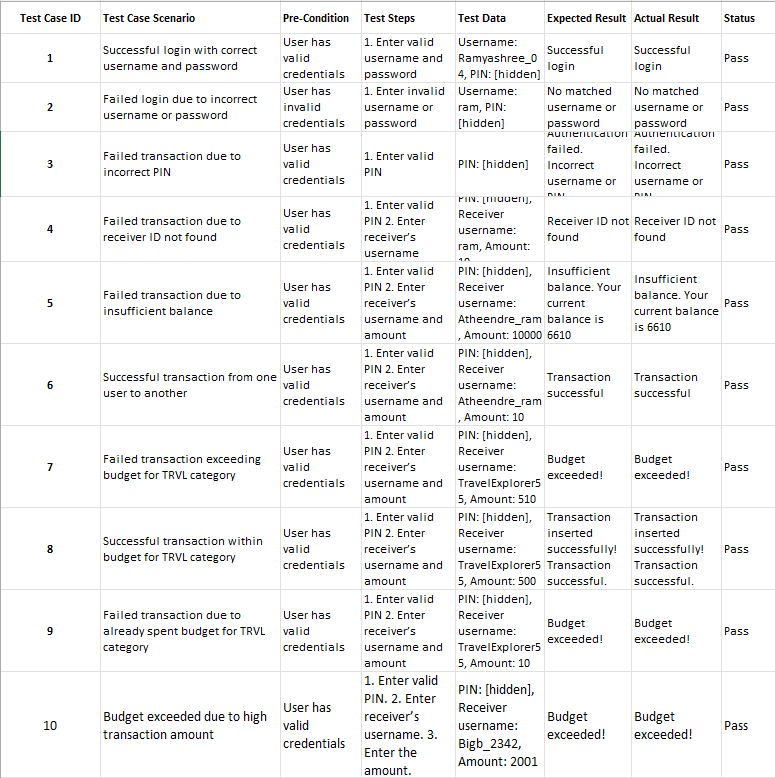
* Objective: To implement minor updates like updating the transactions table to even show the time at which the transaction is being done.
* Activities:
* Implemented a feature that shows the timestamp and date at which the transaction was made.
* We have implemented a feature where after each transaction, a mail is sent that the transaction was successful.
* Implemented a feature that automatically updates bank balance of the sender and receiver when a transaction is being made (using triggers).





1. Testing:

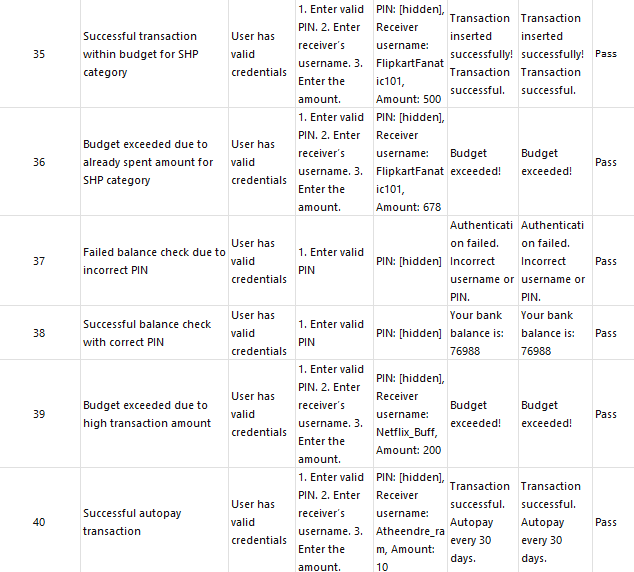
* Objective: To test success and failure cases for all the implemented features.
* Activities:
* Made a separate file for checking success and failure cases for features like login, correct username and pin, budget categories etc.
* Testing.ipynb consists of all the test cases we have tried.
* Below is a table of what all test cases we have done.











**Team Members:**

1. Apoorva Ramesh
2. Atheendre Ramesh
3. Apeksha N
4. Purab Singh Rajpurohit
5. Ramyashree G
6. Keerthi G

—------------------------------------------------------------------------------------------