Database Design Document

Business problem being addressed:

- A centralized system that consists of all the financial data related to investments across various instruments like stocks and mutual funds
- This will enable users to track income, expenses, loans, assets, insurance coverages and liabilities comprehensively
- Goal Setting: Enable users to set financial goals, such as saving for retirement, buying a home, or paying off a debt
- Integrate credit Score tracking and reporting to understand users to improve their creditworthiness
- Calculate the user's current net worth based on the information available related to the assets and liabilities and help them in assessing the overall financial stability
- Generate alerts for important financial events to the users, like credit card bill payment.

Changes in this Version:

- Created **Address** table to store multiple addresses related to a profile.
- Created **MFA_Method** table to store the multi factor authentication options set up for an user.
- Updated **Sip_Details** table to be a subtype of the stock_book entity.
- Removed Additional_Policy_Details table and Created **Life_Insurance**, **Health_Insurance** and **Auto_Insurance** entities to capture specific details for the respective insurance types.
- Added **Bank_Infromation** to store information of the Bank in which users open bank and credit card accounts.
- Updated **User** entity ro store the additional details related to the login for the user.
- Added **data types** for the attributes in all the entities.
- Renamed the Credit_Bureau table to **Credit_Agency** and added attributes to store contact information for the agencies.
- Renamed the entity names to more meaningful names.(Purchase_Details -> Transaction_History, Assets->Stock_Book)
- Have added new attributes in entities like Credit_Card, Policy_Details, Claim_History, Notification, Trading_account, Transaction_History

Entities

User:

Purpose: The User entity represents the individuals who have created accounts in the application to use the services. This typically stores user-specific information such as their username ,password hash(for authentication) ,user id as the unique identifier, number of failed login attempts ,date when the password is last updated, last logged in date and boolean indicating if the mfa is enabled.

Importance: Users is one of the primary and most important tables in the database, as it is essential for managing user access and personalization.

Relation to Other Entities:

Related to Profile: Each logged in user can create multiple profiles, which can be further used to track the profile specific financial information. This establishes a one-to-many relation, where each user can create and manage multiple profiles.

Credit_Info:

Purpose: This entity is used to store the information related to an individual's credit profile. It typically includes data like current credit score, last updated date , unique identifier which identifies credit bureau ,unique identifier credit id which uniquely identifies each row and

Importance: Credit information is essential in financial applications for assessing an individual's creditworthiness and managing credit-related aspects

Relation to Other Entities:

Related to Profile: Each set of credit information is associated with a specific user profile. The "Profile_ID" field in the "Credit_Info" table is a foreign key referencing the "Profile_ID" primary key in the "Profile" table, establishing a one-to-many relationship. This can be used to track credit information for a user from different credit bureau

Related to Credit_Agency: Credit details can be collected from various credit bureaus, so each record will be associated with the credit bureau from which this data has been collected by using the foreign key agency_id which is referencing the primary key agency_id in the credit_agency table.

Credit_History:

Purpose: The "Credit_History" entity stores historical data related to a user's credit activities. It includes historical records of credit scores and other credit-related events over time.

Importance: Credit history is crucial for assessing an individual's creditworthiness and making informed lending decisions.

Relation to other Entities:

Related to Credit_info: The relationship between these tables is established through the credit_id (FK) field in the "Credit_History" table, which references the credit_id field in the "Credit_Info" table. This relationship allows to associate each credit history entry in the "Credit_History" table with a specific credit record in the "Credit_Info" table. In other words, we can track the history of a user's credit information over time using the "Credit_History" table

Credit_Agency:

Purpose: The "Credit_Agency" entity represents credit reporting agencies or bureaus that provide credit information. It includes information about different credit bureaus, such as their names, contact details, address, and scale or maximum value of the credit score.

Importance: Credit Agencies are primary sources of credit data, and applications use data from various agencies to compile credit profiles.

Loans:

Purpose: The "Loans" entity stores information about loans, including details like the loan's unique identifier, the profile or user associated with the loan (via a foreign key), the loan's name, type, the lender's unique identifier and some other details related to the loan.

Importance: Loans are a fundamental aspect of personal finance application, allowing users to manage their borrowing activities, such as mortgages, personal loans, or other types of loans.

Relation to Other Entities:

Related to "Installments": The "Loan_ID" field in the "Installments" table is a foreign key that references the "Loan ID" primary key in the "Loans" table. This establishes a one-to-many relationship, as one loan can have multiple installments.

Related to "Loan_Documents": The "Loan_ID" field in the "Loan_Documents" table is a foreign key that references the "Loan ID" primary key in the "Loans" table, allowing to associate multiple documents with specific loans.

Related to "Lender": The "Lender ID" field in the "Loans" table is a reference to a lender, which can be used to provide information about the loan's origin or source.

Installments:

Purpose: The "Installments" entity records information about individual loan installments, including the installment's unique identifier, the associated loan (via a foreign key), the due amount, due date, payment method, the date the installment was paid and amount paid.

Importance: Installments are essential for tracking the repayment of loans and ensuring borrowers meet their payment obligations.

Relation to other entities:

Related to "Loans"**:** The "Loan_ID" field in the "Installments" table is a foreign key that references the "Loan ID" primary key in the "Loans" table, establishing a one-to-many relationship, as one loan can have multiple installments.

Loan_Documents:

Purpose: The "Loan_Documents" entity allows to store documents related to loans. It includes information about the document's unique identifier, the associated loan (via a foreign key), the document's name, URL, and a description.

Importance: Storing loan-related documents is essential for record-keeping and document management in loan applications.

Relation to Other Entities:
Related to "Loans": The "Loan_ID" field in the "Loan_Documents" table is a foreign key that references the "Loan ID" primary key in the "Loans" table, enabling you to link documents to specific loans.

Lender

Purpose: The "Lender" entity represents information about lenders, including their unique identifier, name, type, address, contact information, website, logo, and licensing details.

Importance: Lender information is crucial for providing context and transparency to users regarding the sources of loans and their associated lenders.

Relation to Other Entities:

Related to "Loans": The "Lender ID" field in the "Loans" table is a reference to lenders, allowing us associate loans with specific lenders.

Bank_Information:

Purpose: The "Bank_Information" entity stores information related to banks, including details like the bank's unique identifier (Bank_ID), the name(Name), address (Address), and contact details (Contact_Number, Contact_Email) of the bank.

Importance: Bank Information gives the details about the bank where the user uses the accounts.

Relation to Other Entities:

Related to "Accounts"**:** The "Bank_ID" field in the "Accounts" table is a foreign key that references the "Bank_ID" primary key in the "Bank_Information" table. This establishes a one-to-many relationship, as one bank can have bank accounts.

Related to "Credit_Card"**:** The "Bank_ID" field in the "Credit_Card" table is a foreign key that references the "Bank_ID" primary key in the "Bank_Information" table. This establishes a one-to-many relationship, as one bank can have multiple credit cards.

Accounts:

Purpose: The "Accounts" entity stores information related to bank accounts, including details like the account's unique identifier, the profile of the user associated with the account (via a foreign key), the account type, account number, current balance and bank information(via foreign key).

Importance: Bank accounts are a core part of personal finance applications, allowing users to manage their accounts, track balances, and perform financial transactions.

Relation to Other Entities:

Related to "Bank_Information"**:** The "Bank_ID" field in the "Bank_Information" table is a foreign key that references the "Bank_ID" primary key in the "Bank_Information" table. This establishes a one-to-many relationship, as one bank can have multiple accounts.

Related to "Bank_Transaction"**:** The "Account_ID" field in the "Bank_Transaction" table is a foreign key that references the "Account_ID" primary key in the "Accounts" table. This establishes a one-to-many relationship, as one bank account can have multiple transactions.

Bank_Transaction:

Purpose: The "Bank_Transaction" entity records information about individual bank transactions, including details like the transaction's unique identifier, the associated bank account (via a foreign key), the transaction amount, date, type, and an expense category (via a foreign key).
Importance: Tracking bank transactions is vital for managing personal finances and monitoring income and expenses

Relation to Other Entities:

Related to "Bank_Information"**:** The "Account_ID" field in the "Bank_Transaction" table is a foreign key that references the "Account_ID" primary key in the "Bank_Information" table, establishing a one-to-many relationship, as one bank account can have multiple transactions.

Related to "Expense_Category": The "Expense_Category_ID" field in the "Bank_Transaction" table is a foreign key that references the "Expense_Category_ID" primary key in the "Expense_Category" table, allowing it to categorize transactions.

Expense_Category

Purpose: The "Expense_Category" entity represents expense categories, including details like the category's unique identifier, name, description, and URL.

Importance: Expense categories help users classify and organize their spending, providing insights into their financial habits.

Relation to Other Entities:

Related to "Bank_Transaction": The "Expense_Category_ID" field in the "Bank_Transaction" table is a foreign key that references the "Expense_Category_ID" primary key in the "Expense_Category" table. This relationship allows us to categorize transactions based on expense categories.

Related to "Credit_Card_Transaction": The "Expense_Category_ID" field in the "Credit_Card_Transaction" table is a foreign key that references the "Expense_Category_ID" primary key in the "Expense_Category" table. This relationship allows us to categorize transactions based on expense categories.

Credit_Card

Purpose: The "Credit_Card" entity stores information about credit cards, including the card's unique identifier, the associated profile or user (via a foreign key), the card issuer(via foreign key), due date, current due balance,payment due amount and credit limit.

Importance: Credit card information is vital for tracking credit card accounts, their balances, and payment due dates.

Relation to Other Entities:
Related to "Credit_Card_Transaction": The "Credit_Card_Id" field in the "Credit_Card_Transaction" table is a foreign key that references the "Credit_Card_Id" primary key in the "Credit_Card" table, establishing a one-to-many relationship, as one credit card can have multiple transactions.

Related to "Profile": The "Profile_ID" field in the "Credit_Card" table is a reference to a user's profile, indicating which user owns the credit card.

Related to “Bank_Information”: The “Bank_ID” field in the “Credit_Card” table is a reference to the Bank information table , providing information about the bank that has issued this credit card.

Credit_Card_Transactions:

Purpose: The "Credit_Card_Transaction" entity records individual credit card transactions, including transaction ID, the associated credit card (via a foreign key), transaction amount, date, transaction type(D/C), and an expense category identifier (via a foreign key).

Importance: Tracking credit card transactions helps users monitor their credit card spending and payment obligations.

Relation to Other Entities:

Related to "Credit_Card": The "Credit_Card_Id" field in the "Credit_Card_Transaction" table is a foreign key that references the "Credit_Card_Id" primary key in the "Credit_Card" table, establishing a one-to-many relationship, as one credit card can have multiple transactions.

Related to "Expense_Category": The "Expense_category_id" field in the "Credit_Card_Transaction" table is a foreign key that links credit card transactions to specific expense categories.

Policy_Details

Purpose: The "Policy_Details" entity stores information about insurance policies, including the policy's unique identifier, the associated profile or user (via a foreign key), the provider (via a foreign key), the type of insurance (via a foreign key), policy number, name, start date, end date, payment frequency, premium amount and upcoming payment date.

Importance: Policy details are fundamental for managing insurance coverage, tracking policy information, and associating policies with specific users.

Relation to other entities:

Related to "Insurance_Type": The "Type_id" field in the "Policy_Details" table is a foreign key that references the "Type_ID" primary key in the "Insurance_Type" table, indicating the type of insurance for each policy.

Related to "Claim_History": The "Policy_ID" field in the "Claim_History" table is a foreign key that associates claims with specific insurance policies.

Related to "Insurance_Providers": The "Provider_id" field in the "Policy_Details" table is a foreign key that references the "Provider_Id" primary key in the "Insurance_Providers" table, indicating the insurance provider for each policy.

Health_Insurance:

Purpose: The "Health_Insurance" entity allows to monitor specific details related to each health insurance policy, such as coverage type, and deductible amount.

Importance: These details provide a comprehensive view of the health insurance type of policies.

Relation to Other Entities:

Related to "Policy_Details": The "Health_Policy_ID" field in the "Health_Insurance" table is a foreign key that references the "Policy_ID" primary key in the "Policy_Details" table, allowing to link additional policy details to specific insurance policies.

Auto_Insurance:

Purpose: The "Auto_Insurance" entity allows to monitor specific details related to each automobile insurance policy, such as vehicle make, vehicle model, and purchase year of the vehicle.

Importance: These details provide a comprehensive view of the automobile insurance policies.

Relation to Other Entities:

Related to "Policy_Details": The "Auto_Policy_ID" field in the "Auto_Insurance" table is a foreign key that references the "Policy_ID" primary key in the "Policy_Details" table, allowing to link additional policy details to specific insurance policies.

Life_Insurance:

Purpose: The "Life_Insurance" entity allows to monitor specific details related to each life insurance policy, such as coverage amount, and beneficiary of the policy.

Importance: These details provide a comprehensive view of the life insurance policies.

Relation to Other Entities:

Related to "Policy_Details": The "Life_Policy_ID" field in the "Life_Insurance" table is a foreign key that references the "Policy_ID" primary key in the "Policy_Details" table, allowing to link additional policy details to specific insurance policies.

Insurance_Type:

Purpose: The "Insurance_Type" entity defines the types of insurance available in the application, including a unique identifier, name, and description for each insurance type.

Importance: Categorizing and describing different insurance types helps users understand their coverage and select the appropriate type of insurance.

Relation to Other Entities:

Related to "Policy_Details": The "Type_id" field in the "Policy_Details" table is a foreign key that references the "Type_ID" primary key in the "Insurance_Type" table, indicating the type of insurance associated with each policy.

Claim_History:

Purpose: The "Claim_History" entity records information about insurance claims, including a unique identifier, the associated insurance policy (via a foreign key), claim amount, claim date and description to store any additional details related to this claim.

Importance: Claim history is essential for tracking and managing insurance claims, providing a record of when and how claims were made.

Relation to Other Entities:
Related to "Policy_Details": The "Policy_ID" field in the "Claim_History" table is a foreign key that associates claims with specific insurance policies.

Insurance_Providers:

Purpose: The "Insurance_Providers" entity contains information about insurance providers or companies, including a unique identifier, name, address, contact details and website.

Importance: Knowing the providers of insurance policies helps users understand their insurance sources and providers' characteristics.

Relation to Other Entities:

Related to "Policy_Details": The "Provider_id" field in the "Policy_Details" table is a foreign key that references the "Provider_ID" primary key in the "Insurance_Providers" table, indicating the provider for each insurance policy.

Financial_Goals

Purpose: The "Financial_Goals" entity stores information about financial goals set by users. It includes details like the goal's unique identifier (Goal_ID), the associated profile or user (Profile_ID), the goal name, description, target amount, target date, and start date.

Importance: Financial goals are an important feature of personal finance applications, enabling users to set, track, and achieve specific financial objectives.

Relation to Other Entities:

Related to "Goal_Progress": The "Goal_ID" field in the "Goal_Progress" table is a foreign key that references the "Goal_ID" primary key in the "Financial_Goals" table. This establishes a one-to-many relationship, as each financial goal can have multiple progress entries tracking its achievement.

Goal_Progress:

Purpose: The "Goal_Progress" entity is used to track the progress made toward financial goals. It includes details like the progress entry's unique identifier (Progress_ID), the associated goal (via a foreign key), the date of the progress entry, the amount achieved, and the current amount toward the goal.

Importance: Tracking progress helps users monitor their journey toward financial goals, stay motivated, and make necessary adjustments to their financial plans.

Relation to Other Entities:

Related to "Financial_Goals": The "Goal_ID" field in the "Goal_Progress" table is a foreign key that links each progress entry to a specific financial goal.

Notification:

Purpose: The "Notification" entity is designed to store information about notifications generated by application. This information may include a unique notification identifier (notification_id), the associated profile (profile_id), the date when the notification was created, the type of notification (e.g., payment reminder, transaction alert), the due date related to the notification, and potentially an associated financial amount, message and a check to indicate if the user has read the notification(isRead)

Importance: Notifications are a vital component of financial applications, providing users with reminders, alerts, and important information regarding their financial activities. This can include reminders for bill payments, upcoming financial events, account balance alerts, and more.

Relation to Other Entities:

Related to "Profile": The "profile_id" field in the "Notification" table is a foreign key that references the user's profile, indicating which user the notification is associated with. This allows notifications to be linked to specific users.

Trading_Account:

Purpose: The "Trading_Account" entity stores information related to a user's trading account, including a unique trading account identifier (Trading_Account_Id), the associated user profile (via a foreign key), the user's account number for the trading account, the last updated date, account type, current_balance and the broker's unique identifier.

Importance: Trading accounts are central to financial applications that involve investing and trading in assets like stocks and securities. This entity allows users to manage and track their trading accounts.

Relation to Other Entities:

Related to "Stock_Book": The "Trading_Account_Id" field in the "Stock_Book" table is a foreign key that associates holdings with specific trading accounts.

Related to "Profile": The "profile_id" field in the "Trading_Account" table is a foreign key that references the user's profile, indicating which user profile the Account is associated with.

Stock_Book:

Purpose: The "Stock_Book" entity stores information about assets held in a trading account, such as stocks or securities. It includes details like a unique holding identifier (Holding_ID), the associated trading account (via a foreign key), asset symbol(Symbol), quantity held, average value,investment type and name of the asset(Name).

Importance: Stock Books are the core holdings within a trading account. This entity allows users to track users' stock portfolios.

Relation to Other Entities:
Related to "Trading_Account": The "Account_id" field in the "Stock_Book" table is a foreign key that links holdings to specific trading accounts.

Related to "Transaction_History": The "Transaction_ID" field in the "Transaction_History" table is a foreign key that associates transaction history with specific holding, this is used to track transactions associated with a holding.

Related to "Sip_Details": The "Sip_Holding_ID" field in the "Sip_Details" table is a foreign key that links systematic investment plan(SIP) to specific holding.

Transaction_History:

Purpose: The "Transaction_History" entity is used to record transaction history for stocks, including a unique transaction identifier (Transaction_ID), the associated stock book (via a foreign key), purchase date, buy/sell type, quantity, and price and symbol.

Importance: Transaction History is essential for tracking individual transactions, such as buying and selling stocks.

Relation to Other Entities:

Related to "Stock_Book": The "Holding_ID" field in the "Transaction_History" table is a foreign key that associates purchase details with specific stocks.

Sip_Details:

Purpose: The "Sip_Details" entity is designed to store information about systematic investment plans (SIPs), is a sub type for the Stock_book entity as it is used to store additional details that are specific to the SIP investments. It includes a unique SIP identifier (Sip_Holding_ID), start date, payment frequency, payment amount, quantity and the next payment date.

Importance: SIPs are a common investment strategy in which regular payments are made into assets over time. This entity allows users to set up and manage SIPs.

Relation to Other Entities:
Related to "Stock_Book": The "Sip_Holiding_Id" field in the "Sip_Details" table is a foreign key that associates SIPs with specific holding and maintains a one-to-one relationship with the Stock_Book entity.

Profile:

Purpose: The "Profile" entity serves as the primary entity for storing user profiles, including critical user details such as a unique profile identifier (Profile_Id), user ID (User_id), first name (First_Name), last name (Last_Name), email (Email), date of birth (Dob), parent profile identifier (Parent_Profile_ID),Social Security Number (SSN_No), Mobile number(Mobile_Number) and Gender

Importance: User profiles are at the core of the application, and this entity holds user-specific information, allowing users to interact with applications and access personalized features.

Address:

Purpose: The "Address" entity is designed to store the multiple addresses of the user, including critical user details such as a unique address identifier (Address_Id), user ID (User_id), address type (Address_Type), is primary (IS_Primary), address line 1 (Address_Line1), address line 2 (Address_Line2), city (City), zip code (Zip_Code), state (State), country (Country), notes (NOTES).

Importance: Address is used to store different types of addresses of the user.

Relation to Other Entities:
Related to "Profile": The "Profile_ID" field in the "Address" table is a foreign key that associates the Profile of the user whose address is there.

MFA_Method:

Purpose: The "MFA_Method" entity is designed to store the multiple Multi-Factor Authentication options of the user, including critical details such as a unique method identifier (Method_Id), user ID (User_id), type of method (Method_Type), additional information (Additional_Information), and recovery code (Recovery_Code).

Importance: MFA_Method is used to store different types of Multi-Factor Authentication options associated with the user.

Relation to Other Entities:
Related to "User": The "User_ID" field in the "MFA_Method" table is a foreign key that associates the MFA method with the User_ID of an user.