Team Members: Zayn Zaidi (615), Feilian Huang (615)
zzaidi3@jhu.edu, fhuang13@jhu.edu

The Personal Finance Tracker is a financial management application designed to help users monitor their income, expenses, budgets, and savings goals. It enables tracking of transactions across multiple accounts, categorization of expenses, and visualization of progress toward financial targets. The system aims to provide actionable insights for better financial planning and decision-making.

https://github.com/EDGAhab/Database_Project

Data Extraction:

Daily transaction data was imported from a csv file. Users, accounts, budgets, investments, and savings goals data were inserted into their respective tables.

Daily transaction data was found on Kaggle (https://www.kaggle.com/datasets/prasad22/daily-transactions-dataset/data) and preprocessed using the Python script preprocessing.py.

Tables, views, stored procedures, and other data were inserted into the database using the script pfsql.sql.

Platform Details

- **Database:** MySQL
- **Server:** Node.js with Express.js
- **Frontend:** React.js

User Guide

1. **Setup Instructions:**
   - Run npm install to install dependencies.
   - Start the backend server: node index.js.
   - Start the React frontend: npm start.
2. **Specific Instructions:**
   - If running on local MySQL, remember to change this field at /backend/index.js
   - Replace with your openai key at src/components/NaturalLanguageSearch.js

```
const db = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "Feilian990725",
  database: "finance_track",
});
```

3. **Main Features:**
   - View accounts, transactions, budgets, and savings goals.

- ○ Search data using **natural language queries**.
- ○ Generate SQL commands dynamically.
4. For more information, check readme.md at the Github:
   https://github.com/EDGAhab/Database_Project/blob/main/README.md

Areas of Specialization

1. **Natural Language Interface**:
   - ○ Integration with GPT-4 API for translating user queries into SQL commands.
2. **Particularly advanced GUI form interface** :
   - ○ Direct UI Interface as a web application

Project Strengths

1. **Comprehensive Data Organization:**
   The database schema covers essential aspects of personal finance, including users, accounts, transactions, budgets, savings goals, and investments.
2. **Extensibility:**
   The use of structured foreign key relationships and modular design allows easy extension of functionality, such as adding new financial instruments or tracking currencies.
3. **User-Focused Features:**
   - ○ Tracks budgets and provides insights into category-wise spending limits.
   - ○ Supports savings goals with progress monitoring to motivate users toward financial targets.
4. **Automation-Ready:**
   The database structure and stored procedures facilitate automation of common financial calculations, such as monthly budget rollovers, recurring transactions, or savings goal projections.
5. **Performance Optimization:**
   Indexing on critical columns like TransactionDate ensures faster retrieval of transaction history for reporting and analytics.
6. **Financial Transparency:**
   Tracks both credit and debit transactions, providing a clear distinction between income and expenses, while supporting multi-currency accounts.
7. **Investment Tracking:**
   The inclusion of investment-specific data (e.g., stock symbols, current values) enhances its utility for users managing diversified portfolios.
8. **User friendly Front End Design:**
   Implemented a comprehensive, user friendly front-end website that interacts with our SQL database.

9. **Natural Language Search Function:**
   Implemented a Natural Language search function which allows users to input a statement in english and receive results by mapping the statement to a SQL query.

Project Limitations

1. **Multi-User Roles or Access Levels:**
   The system assumes all users have the same level of access, with no differentiation for shared accounts or restricted views.
2. **No Currency Conversion Support:**
   Although multi-currency accounts are supported, the system lacks integration with live exchange rates for conversion or comparison.
3. **Lack of Notifications or Alerts:**
   Users are not notified when they approach budget limits or deadlines for savings goals, reducing its usability for proactive financial planning.
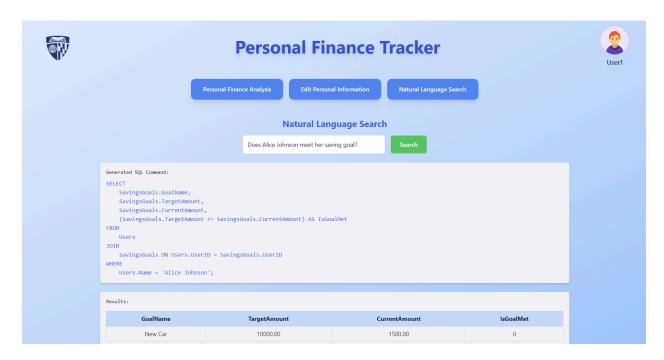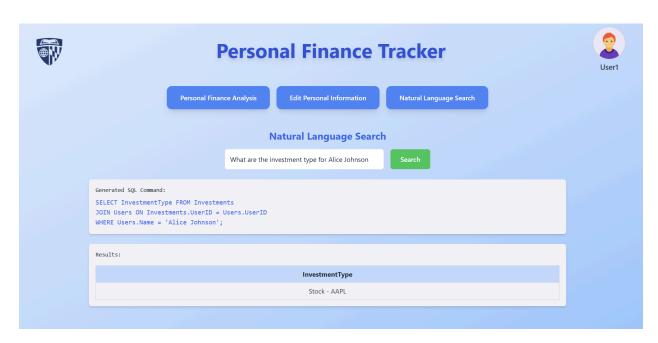4. **Static Budgets:**
   Budgets are fixed and cannot automatically adjust based on actual spending trends or income variations over time.
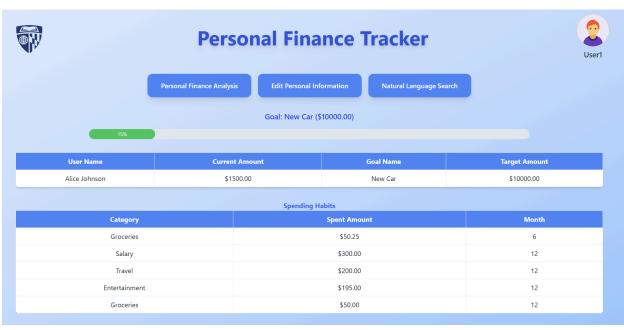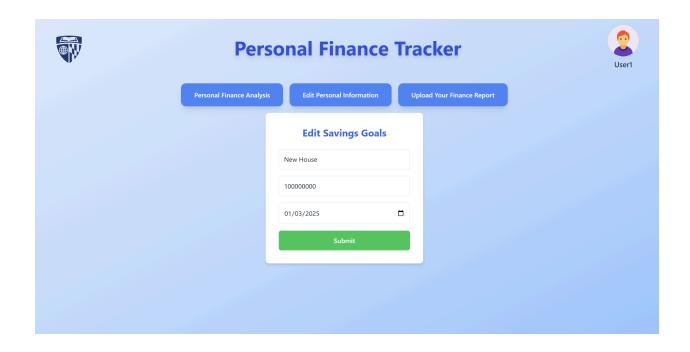5. **No Fraud Detection or Security Features:**
   There are no mechanisms to flag unusual transactions or enforce multi-factor authentication for enhanced security.

Screenshots:

Personal Finance Tracker

Personal Finance Analysis   Edit Personal Information   Natural Language Search

User1

Natural Language Search

What are the investment type for Alice Johnson   Search

Generated SQL Command:
```
SELECT InvestmentType FROM Investments
JOIN Users ON Investments.UserID = Users.UserID
WHERE Users.Name = 'Alice Johnson';
```

Results:

InvestmentType
Stock - AAPL

Personal Finance Tracker

Personal Finance Analysis   Edit Personal Information   Natural Language Search

User1

Goal: New Car ($10000.00)

15%

User Name	Current Amount	Goal Name	Target Amount
Alice Johnson	$1500.00	New Car	$10000.00

Spending Habits

Category	Spent Amount	Month
Groceries	$50.25	6
Salary	$300.00	12
Travel	$200.00	12
Entertainment	$195.00	12
Groceries	$50.00	12

Relational data model

- **Entities and Attributes**
  - Users:
    - UserID: Primary key.
    - Name: User's full name.
    - Email: User's email address.
    - PasswordHash: Encrypted password for secure login.
    - CreatedAt: Timestamp when the user account was created.
  - Accounts:
    - AccountID: Primary key.
    - UserID: Foreign key linking to the Users entity.
    - AccountName: Descriptive name of the account (e.g., "Savings Account").
    - AccountType: Type of account (e.g., Bank, Credit Card, Wallet, Investment).
    - Balance: Current balance in the account.
    - Currency: Currency type.
    - CreatedAt: Timestamp when the account was added.
  - Categories:
    - CategoryID: Primary key.
    - Name: Name of the category.
    - Description: Optional description of the category.
  - Transactions:
    - TransactionID: Primary key.

- AccountID: Foreign key linking to the Accounts entity.
- CategoryID: Foreign key linking to the Categories entity.
- Amount: Transaction amount.
- TransactionType: Type of transaction.
- TransactionDate: Date of the transaction.
- Description: Optional description of the transaction.
- CreatedAt: Timestamp when the transaction was added.
  - Budgets:
    - BudgetID: Primary key.
    - UserID: Foreign key linking to the Users entity.
    - CategoryID: Foreign key linking to the Categories entity.
    - BudgetAmount: Budgeted amount for a specific category.
    - StartDate: Start date for the budget period.
    - EndDate: End date for the budget period.
    - CreatedAt: Timestamp when the budget was created.
  - Savings Goals:
    - GoalID: Primary key.
    - UserID: Foreign key linking to the Users entity.
    - GoalName: Name of the savings goal.
    - TargetAmount: Total amount the user aims to save.
    - CurrentAmount: Amount saved so far.
    - Deadline: Target date for achieving the savings goal.
    - CreatedAt: Timestamp when the savings goal was added.
  - Investments:
    - InvestmentID: Primary key.
    - UserID: Foreign key linking to the Users entity.
    - InvestmentType: Type of investment.
    - InitialValue: Value of the investment at the time of purchase.
    - CurrentValue: Current value of the investment.
    - PurchaseDate: Date the investment was purchased.
    - Notes: Optional notes about the investment.
    - CreatedAt: Timestamp when the investment was added.
- **Relationship**
  - Users ↔ Accounts: One-to-many.
  - Accounts ↔ Transactions: One-to-many.
  - Categories ↔ Transactions: One-to-many.
  - Users ↔ Budgets: One-to-many.
  - Users ↔ SavingsGoals: One-to-many.
  - Users ↔ Investments: One-to-many.

```sql
CREATE TABLE Users (
    UserID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL UNIQUE,
    PasswordHash VARCHAR(255) NOT NULL,
    CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE Accounts (
    AccountID INT AUTO_INCREMENT PRIMARY KEY,
    UserID INT NOT NULL,
    AccountName VARCHAR(255) NOT NULL,
    AccountType ENUM('Bank', 'Credit Card', 'Wallet', 'Investment') NOT NULL,
    Balance DECIMAL(15, 2) DEFAULT 0.00,
    Currency CHAR(3) NOT NULL,
    CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (UserID) REFERENCES Users(UserID) ON DELETE CASCADE
);

CREATE TABLE Categories (
    CategoryID INT AUTO_INCREMENT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    Description TEXT
);

CREATE TABLE Transactions (
    TransactionID INT AUTO_INCREMENT PRIMARY KEY,
    AccountID INT NOT NULL,
    CategoryID INT NOT NULL,
    Amount DECIMAL(15, 2) NOT NULL,
    TransactionType ENUM('Credit', 'Debit') NOT NULL,
    TransactionDate DATE NOT NULL,
    Description TEXT,
    CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID) ON DELETE CASCADE,
    FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID) ON DELETE SET NULL
);
```

```sql
CREATE TABLE Budgets (
    BudgetID INT AUTO_INCREMENT PRIMARY KEY,
    UserID INT NOT NULL,
    CategoryID INT NOT NULL,
    BudgetAmount DECIMAL(15, 2) NOT NULL,
    StartDate DATE NOT NULL,
    EndDate DATE NOT NULL,
    CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (UserID) REFERENCES Users(UserID) ON DELETE CASCADE,
    FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID) ON DELETE SET NULL
);

CREATE TABLE SavingsGoals (
    GoalID INT AUTO_INCREMENT PRIMARY KEY,
    UserID INT NOT NULL,
    GoalName VARCHAR(255) NOT NULL,
    TargetAmount DECIMAL(15, 2) NOT NULL,
    CurrentAmount DECIMAL(15, 2) DEFAULT 0.00,
    Deadline DATE NOT NULL,
    CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (UserID) REFERENCES Users(UserID) ON DELETE CASCADE
);

CREATE TABLE Investments (
    InvestmentID INT AUTO_INCREMENT PRIMARY KEY,
    UserID INT NOT NULL,
    InvestmentType VARCHAR(255) NOT NULL,
    InitialValue DECIMAL(15, 2) NOT NULL,
    CurrentValue DECIMAL(15, 2),
    PurchaseDate DATE NOT NULL,
    Notes TEXT,
    CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (UserID) REFERENCES Users(UserID) ON DELETE CASCADE
);
```