| **Team Name:** SmartExpense Tracker | |
| --- | --- |
| **Team Members** | Zuhair Khan |
| Joe Viola |
| Israail Ghazzal |
| Omer Khaleel |
| **Project Pitch:**   * SmartExpense Tracker is a web-based application designed to help individuals easily manage their expenses and gain insights into their spending habits. Many people struggle to keep track of their finances, leading to overspending and financial stress. What sets SmartExpense Tracker apart is its intuitive interface that makes it incredibly easy for users to categorize their transactions, allowing for better visualization of where their money is going through engaging charts and graphs, making budgeting simple and accessible. | |
| **Front End:**   * The SmartExpense Tracker will be accessible through a user-friendly website, allowing users to manage their expenses from any device with an internet connection. Initially hosted using Python Flask for testing, the application will be deployed on Azure Cloud to ensure reliable performance and scalability as user demand grows. | |
| **Back End:**   * The SmartExpense Tracker will use Python Flask to handle backend logic and act as the intermediary between the website and the database. For data storage, SQLite will be used during development due to its lightweight and simple setup, with plans to transition to a more robust solution like PostgreSQL if the application scales. This setup will securely store user data, transaction records, and categorized expenses while ensuring efficient retrieval for analysis and visualization. | |

**Functional Requirements:**

1. **User Registration and Login**:  
   Users must be able to create an account with a username and password, log in securely, and manage their profile information.
2. **Expense Tracking**:  
   Users can add, edit, and delete expense records, categorizing them (e.g., food, travel, utilities) and adding relevant details like date, amount, and notes.
3. **Data Visualization**:  
   The application will provide visual reports such as bar charts, pie charts, and line graphs to show spending patterns and trends over time.
4. **Budget Setting and Notifications**:  
   Users can set monthly budgets for specific categories and receive alerts or notifications when they approach or exceed their budget limits.

**Non-Functional Requirements:**

1. **Performance**:  
   The application must load pages and perform transactions within 2 seconds under normal conditions.
2. **Scalability**:  
   The system should be designed to handle an increasing number of users and transactions, with the ability to transition from SQLite to PostgreSQL as needed.
3. **Security**:  
   All user data must be securely stored and transmitted, using encryption protocols such as HTTPS and password hashing.
4. **Availability**:  
   The application must have a minimum uptime of 99.9% and be hosted on Azure Cloud to ensure reliability and accessibility from anywhere.