NAME:- ANUBHAV CHAUHAN REGISTRATION NO. 25BSA10160

TEXT FILE

To tackle your request systematically, I will identify a real-world problem related to currency conversion, define clear objectives, and outline how the concepts, tools, and structured processes apply to designing a solution.

Real-World Problem

Problem: Global travelers and international businesses face difficulties in accurately and quickly converting currencies due to fluctuating exchange rates, multiple currencies, and the need for up-to-date information. Manual calculation or reliance on static data often leads to inaccuracies and inefficiencies in financial transactions, budgeting, and currency exchange decisions.

Objectives and Expected Outcomes

• Objectives: Develop a real-time currency converter application that supports all world currencies, providing accurate and current exchange rates.

• Expected outcomes: An efficient, user-friendly tool capable of converting any currency to any other, integrated with a reliable data source, with minimal latency, and capable of handling a high volume of conversions accurately.

Application of Concepts & Design Approach

• Knowledge application: Use concepts of API integration, data structures (dictionaries for storing rates), and modular programming principles learned in course (such as function abstraction, error handling, and I/O).

• Tools & libraries: Utilize Python with the requests library to fetch real-time data, and consider additional libraries for GUI if needed (e.g., Tkinter).

Structured Development Process

1. Problem Definition:
Address the need for real-time, accurate currency conversion across all supported currencies worldwide.

2. Requirement Analysis:

• Data source must be reliable and up-to-date (live exchange rates).

• Support for international currency codes (ISO 4217).

• User inputs: source currency, target currency, amount.

• Outputs: converted amount, exchange rate.

3. Top-Down Design / Modularization:
Break down into modules: data fetching, conversion computation, user interaction.

4. Algorithm Development:

• Fetch current exchange rates via API.

• Lookup rates based on user input.

• Calculate converted value.

• Handle errors (invalid currency codes, API failure).

5. Implementation:
Write Python code in functions for fetching data, converting, and user I/O.

6. Testing & Refinement:

• Test with different currency pairs and amounts.

• Handle API failures or invalid inputs gracefully.

• Optimize performance and user interface.

This structured approach ensures a reliable, scalable, and maintainable currency conversion application that fulfills a real-world need efficiently. If you want, I can help create a project plan, code, or detailed design for this solution!