SKILL DEVELOPMENT
Project Report

DLithe Consultancy Services Pvt. Ltd.

Project Report Assessment

Student Name: Preeti Patil

Reg. no: 2JR23CS068

Assignment: Java

Organization: DLithe Consultancy Services Pvt. Ltd.

Supervisor's Name: Archana SM

Submitted to

Signature of Training Supervisor

Date:

Signature of Students

Date:

DLithe™
Be Agile..Be in Demand

TABLE OF CONTENTS:

INTRODUCTION:

In an increasingly interconnected world, dealing with multiple currencies has become a part of daily life. Whether it's for travel, international shopping, freelancing, or business, people often need to know the equivalent value of one currency in another. A currency converter simplifies this process by providing a quick and accurate way to convert amounts between different currencies based on the latest exchange rates.

This project focuses on creating a simple yet effective Currency Converter application that helps users convert currencies with ease. It allows users to select the source and target currencies, input an amount, and instantly get the converted value. The app can fetch live exchange rates using external APIs or use manually set rates for offline usage. Built with web technologies like HTML, CSS, and JavaScript, the converter offers a clean and interactive user interface that ensures a smooth user experience.

The goal of this project is to demonstrate the practical application of real-time data in web development and highlight the importance of currency conversion in global financial activities. This tool is particularly useful for travelers, online shoppers, students, and anyone involved in international transactions. By making conversions faster and more accessible, the currency converter serves as a reliable companion in managing financial decisions across borders.

DLithe™
Be Agile..Be in Demand

USE CASE REQUIREMENTS:

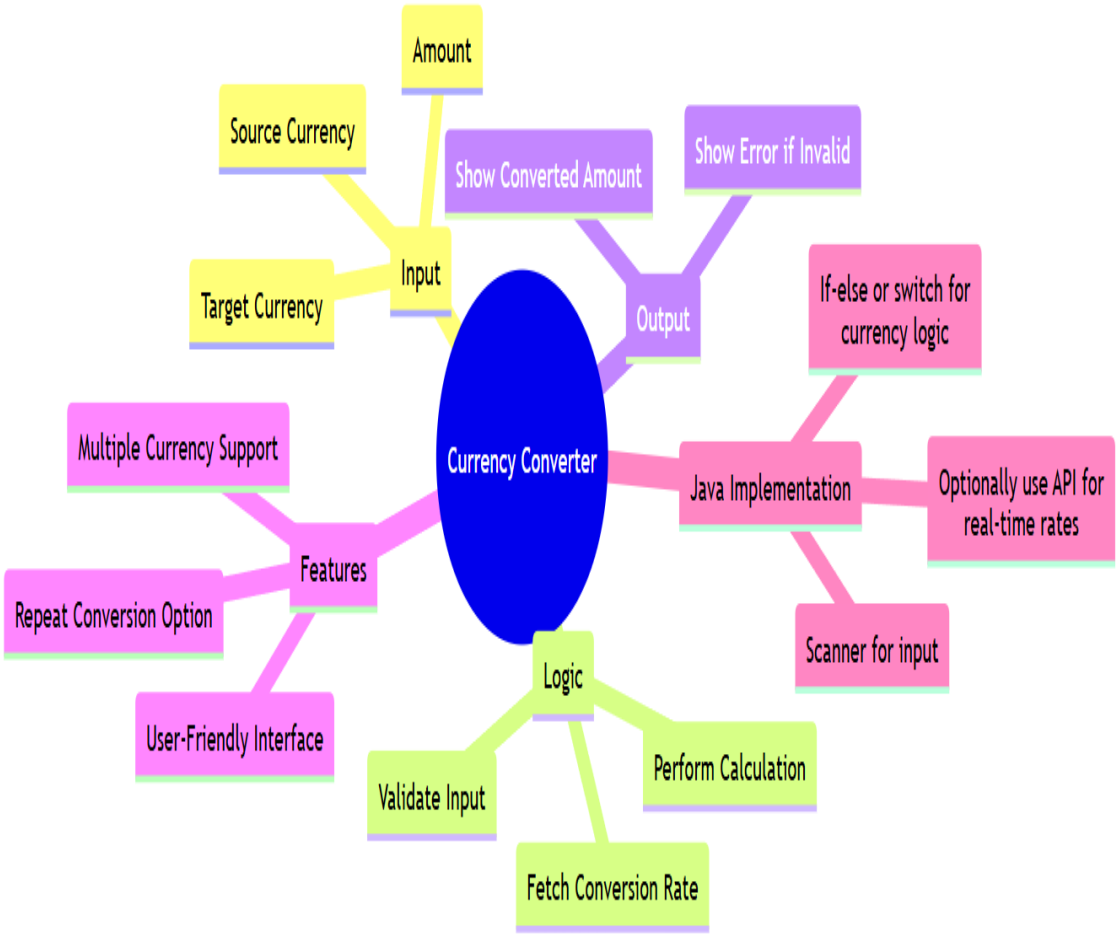
The **Currency Converter** application is designed to allow users to easily convert values from one currency to another using up-to-date exchange rates. The main use cases begin with the user selecting the **source currency**, which is the currency they currently have. The application must display a list of supported currencies for the user to choose from. Once the source currency is selected, the user then chooses the **target currency**, which is the currency they want to convert into. Again, a dropdown or list should be provided for this selection.

After both currencies are selected, the user **enters the amount** they wish to convert. The system should ensure that the input is a valid numeric value. Once the amount is entered, the application will **fetch the latest exchange rate** between the selected currencies. If the app is connected to a real-time exchange rate API, it must ensure a stable internet connection to retrieve the latest data. Alternatively, the app can use manually set rates for offline functionality. With the exchange rate available, the system then **performs the conversion calculation** by multiplying the amount with the rate.

Finally, the **converted amount is displayed** clearly to the user, indicating the equivalent value in the target currency. Additionally, users should be able to **reset the input fields** to perform a new conversion. In the event of any errors—such as invalid input, network issues, or failure to retrieve rates—the system must **handle errors gracefully** and display user-friendly error messages. These use cases together ensure that the currency converter functions accurately, efficiently, and provides a smooth user experience.

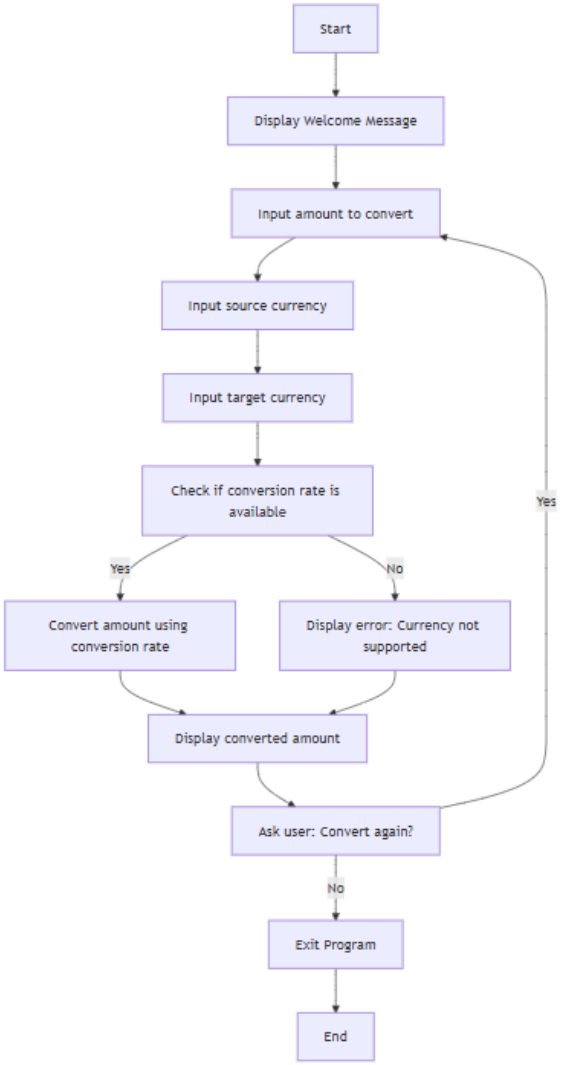
DLithe™
Be Agile..Be in Demand

MIND MAP:



DLithe™
Be Agile..Be in Demand

FLOW CHART:



DLithe™
Be Agile..Be in Demand

SOURCE CODE:

```java
import java.util.Scanner;

public class CurrencyConverter {

    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        boolean repeat = true;

        while (repeat) {
            System.out.println(x:"=== Currency Converter ===");

            // Input amount
            System.out.print(s:"Enter amount: ");
            double amount = scanner.nextDouble();

            // Input source currency
            System.out.print(s:"Enter source currency (USD, EUR, INR): ");
            String source = scanner.next().toUpperCase();

            // Input target currency
            System.out.print(s:"Enter target currency (USD, EUR, INR): ");
            String target = scanner.next().toUpperCase();

            double convertedAmount = convertCurrency(amount, source, target);

            if (convertedAmount != -1) {
                System.out.printf(format:"Converted Amount: %.2f %s\n", convertedAmount, target);
            } else {
                System.out.println(x:"Invalid currency code or conversion not supported.");
            }

            System.out.print(s:"Do you want to convert another amount? (yes/no): ");
            String answer = scanner.next().toLowerCase();
            repeat = answer.equals(anObject:"yes");
```

```
            System.out.print(s:"Do you want to convert another amount? (yes/no): ");
            String answer = scanner.next().toLowerCase();
            repeat = answer.equals(anObject:"yes");
        }

        System.out.println(x:"Thank you for using the Currency Converter!");
        scanner.close();
    }

    public static double convertCurrency(double amount, String source, String target) {
        // Conversion rates are relative to 1 USD
        double usdToInr = 83.0;
        double usdToEur = 0.93;

        if (source.equals(target)) {
            return amount;
        }

        switch (source) {
            case "USD":
                if (target.equals(anObject:"INR"))
                    return amount * usdToInr;
                if (target.equals(anObject:"EUR"))
                    return amount * usdToEur;
                break;
            case "INR":
                if (target.equals(anObject:"USD"))
                    return amount / usdToInr;
                if (target.equals(anObject:"EUR"))
                    return (amount / usdToInr) * usdToEur;
                break;
```

```
        switch (source) {
            case "USD":
                if (target.equals(anObject:"INR"))
                    return amount * usdToInr;
                if (target.equals(anObject:"EUR"))
                    return amount * usdToEur;
                break;
            case "INR":
                if (target.equals(anObject:"USD"))
                    return amount / usdToInr;
                if (target.equals(anObject:"EUR"))
                    return (amount / usdToInr) * usdToEur;
                break;
            case "EUR":
                if (target.equals(anObject:"USD"))
                    return amount / usdToEur;
                if (target.equals(anObject:"INR"))
                    return (amount / usdToEur) * usdToInr;
                break;
        }

        return -1; // invalid or unsupported conversion
    }
}
```

DLithe™
Be Agile..Be in Demand

OUTPUT:

```
=== Currency Converter ===
Enter amount: 100
Enter source currency (USD, EUR, INR): USD
Enter target currency (USD, EUR, INR): EUR
Converted Amount: 93.00 EUR
Do you want to convert another amount? (yes/no): yes
=== Currency Converter ===
Enter amount: 200
Enter source currency (USD, EUR, INR): EUR
Enter target currency (USD, EUR, INR): INR
Converted Amount: 17849.46 INR
Do you want to convert another amount? (yes/no): yes
=== Currency Converter ===
Enter amount: 250
Enter source currency (USD, EUR, INR): INR
Enter target currency (USD, EUR, INR): USD
Converted Amount: 3.01 USD
Do you want to convert another amount? (yes/no): no
Thank you for using the Currency Converter!
PS C:\Users\hp\OneDrive\Desktop\currencyconverter.java>
```

TECHNOLOGIES USED:

CHALLENGES:

1. **Real-time Exchange Rate Accuracy:**

   Ering the converter uses up-to-date and accurate currency exchange rates, which constantly fluctuate.

2. **API Limitations and Reliability:**

   Dependence on external APIs for exchange rates can lead to issues like rate limits, downtime, additional costs.

3. **Handling Different Currency Formats:**

   Managing various currency symbols, decimal places, and formatting differences across countries.

4. **Input Validation and Error Handling:**

   Preventing invalid user inputs (e.g., letters instead of numbers) and providing clear error messages.

5. **Offline Functionality:**

   Maintaining useful functionality without internet access, where exchange rates may become outdated.

6. **Security Concerns:**

   Protecting sensitive information such as API keys and user data from unauthorized access or misuse.

DLithe™
Be Agile..Be in Demand

APPLICATIONS:

- **International Travel:**Helps travelers quickly convert foreign currency to their home currency while abroad.
- **E-commerce and Online Shopping:**Allows buyers and sellers to calculate product prices in different currencies on global shopping platforms.
- **Freelancing and Remote Work:**Enables freelancers to understand payment values in their local currency when working with international clients.
- **Stock and Forex Trading:**Useful for traders to analyze market values and convert earnings across different currencies.
- **Banking and Finance:**Assists in cross-border transactions, international fund transfers, and currency exchange rate tracking.
- **Education Purposes:**Helps students learn about exchange rates, currency values, and financial concepts in economics or computer science projects.
- **Business Operations:**Supports companies dealing with international clients or vendors by converting pricing and invoices across currencies.
- **Mobile Apps and Web Services:**Integrated into apps to offer users quick currency conversion while browsing or making payments.

CONCLUSION:

In today's fast-paced and globally connected world, a currency converter plays a crucial role in helping individuals and businesses manage currency values efficiently. Whether it's for international travel, online shopping, freelancing, or trading, it allows users to convert one currency into another accurately and quickly. By using real-time exchange rates, this tool ensures that users always have the most updated conversion values at their fingertips.

Developing a currency converter not only highlights the practical use of APIs and data handling but also strengthens skills in frontend development and user interface design. While there are challenges like handling exchange rate accuracy, API limitations, and input validation, overcoming them results in a reliable and user-friendly application. Overall, this project is a great example of how technology simplifies real-world financial tasks and provides convenient solutions for daily use.