CURRENCY CONVERTER

**Obectives:**

The objective of this project is to provide a simple and efficient solution for converting currency between INR and USD while demonstrating the use of functional programming features in Java. It focuses on handling user inputs with proper validation, implementing a modular design using functional interfaces and lambda expressions, and exploring the Java Stream API for sequential task execution.

**Featrues:**

1. Convert Amounts from INR to USD:
   * Accepts an amount in Indian Rupees (INR) from the user.
   * Converts the given amount into US Dollars (USD) using a predefined conversion rate.
   * Displays the converted amount to the user in a formatted output.
2. Convert Amounts from USD to INR:
   * Accepts an amount in US Dollars (USD) from the user.
   * Converts the given amount into Indian Rupees (INR) using a predefined conversion rate.
   * Displays the converted amount to the user in a formatted output.
3. Handle Invalid Inputs:
   * Validates user input to ensure it is numeric.
   * Prompts the user with an error message if the input is invalid.
   * Allows the user to re-enter a valid numeric input.
4. Functional Programming Implementation:
   * Demonstrates the use of functional interfaces and lambda expressions for modular and concise code.
   * Uses the @FunctionalInterface annotation to enforce a single-method contract.
5. Stream API for Sequential Execution:
   * Utilizes the Stream API to execute currency conversion tasks in a sequential manner.
   * Encapsulates each task as a Runnable for clean and organized execution.

**Step-by-Step Program Implementation :**

**Step 1 : Functional Interface**

@FunctionalInterface

interface CurrencyConverterFunction {

double convert(double amount);

}

Explanation:

* Defines a functional interface with a single abstract method convert.
* This is used to implement the currency conversion logic.

**Step 2 : Conversion Rates**

// Conversion rates

double usdToInrRate = 82.5;

double inrToUsdRate = 0.012;

Explanation :

* The exchange rates are hardcoded in the application for simplicity.
* usdToInrRate converts 1 USD to INR.
* inrToUsdRate converts 1 INR to USD.

**Step 3 : Currency Conversion Logic**

CurrencyConverterFunction inrToUsd = amount -> amount \* inrToUsdRate;

CurrencyConverterFunction usdToInr = amount -> amount \* usdToInrRate;

Explanation :

* Lambda expressions are used to define the conversion logic.
* These lambdas implement the CurrencyConverterFunction interface.

**Step 4 : Stream API for Sequential Execution**

Stream.of(

(Runnable) () -> {

System.out.println("Enter the amount in Indian Rupees (INR):");

if (scanner.hasNextDouble()) {

double amountInInr = scanner.nextDouble();

double convertedToUsd = inrToUsd.convert(amountInInr);

System.out.printf("%.2f INR = %.2f USD\n", amountInInr, convertedToUsd);

} else {

System.out.println("Invalid input! Please enter a valid numeric amount in INR.");

scanner.next(); // Clear invalid input

}

},

(Runnable) () -> {

System.out.println("Enter the amount in US Dollars (USD):");

if (scanner.hasNextDouble()) {

double amountInUsd = scanner.nextDouble();

double convertedToInr = usdToInr.convert(amountInUsd);

System.out.printf("%.2f USD = %.2f INR\n", amountInUsd, convertedToInr);

} else {

System.out.println("Invalid input! Please enter a valid numeric amount in USD.");

}

}

).forEach(Runnable::run);

Explanation :

* A Stream is used to execute two tasks sequentially:
  + Convert INR to USD.
  + Convert USD to INR.
* Each task is represented as a Runnable.
* The forEach method executes each Runnable in sequence.

**Step 5 : Input Validation**

* The program validates user input using scanner.hasNextDouble() to ensure only numeric values are accepted.
* For invalid input, an error message is displayed, and the invalid input is cleared using scanner.next().

**Program:**

**package** java\_project;

**import** java.util.Scanner;

**import** java.util.stream.Stream;

@FunctionalInterface

**interface** CurrencyConverterFunction {

**double** convert(**double** amount);

}

**public** **class** CurrencyConverter {

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

// Conversion rates

**double** usdToInrRate = 82.5;

**double** inrToUsdRate = 0.012;

// Method references

CurrencyConverterFunction inrToUsd = amount -> amount \* inrToUsdRate;

CurrencyConverterFunction usdToInr = amount -> amount \* usdToInrRate;

System.***out***.println("Welcome to Currency Converter!");

// Using Stream API

Stream.*of*(

(Runnable) () -> {

System.***out***.println("Enter the amount in Indian Rupees (INR):");

**if** (scanner.hasNextDouble()) {

**double** amountInInr = scanner.nextDouble();

**double** convertedToUsd = inrToUsd.convert(amountInInr);

System.***out***.printf("%.2f INR = %.2f USD\n", amountInInr, convertedToUsd);

} **else** {

System.***out***.println("Invalid input! Please enter a valid numeric amount in INR.");

scanner.next(); // Clear invalid input

}

},

(Runnable) () -> {

System.***out***.println("Enter the amount in US Dollars (USD):");

**if** (scanner.hasNextDouble()) {

**double** amountInUsd = scanner.nextDouble();

**double** convertedToInr = usdToInr.convert(amountInUsd);

System.***out***.printf("%.2f USD = %.2f INR\n", amountInUsd, convertedToInr);

} **else** {

System.***out***.println("Invalid input! Please enter a valid numeric amount in USD.");

}

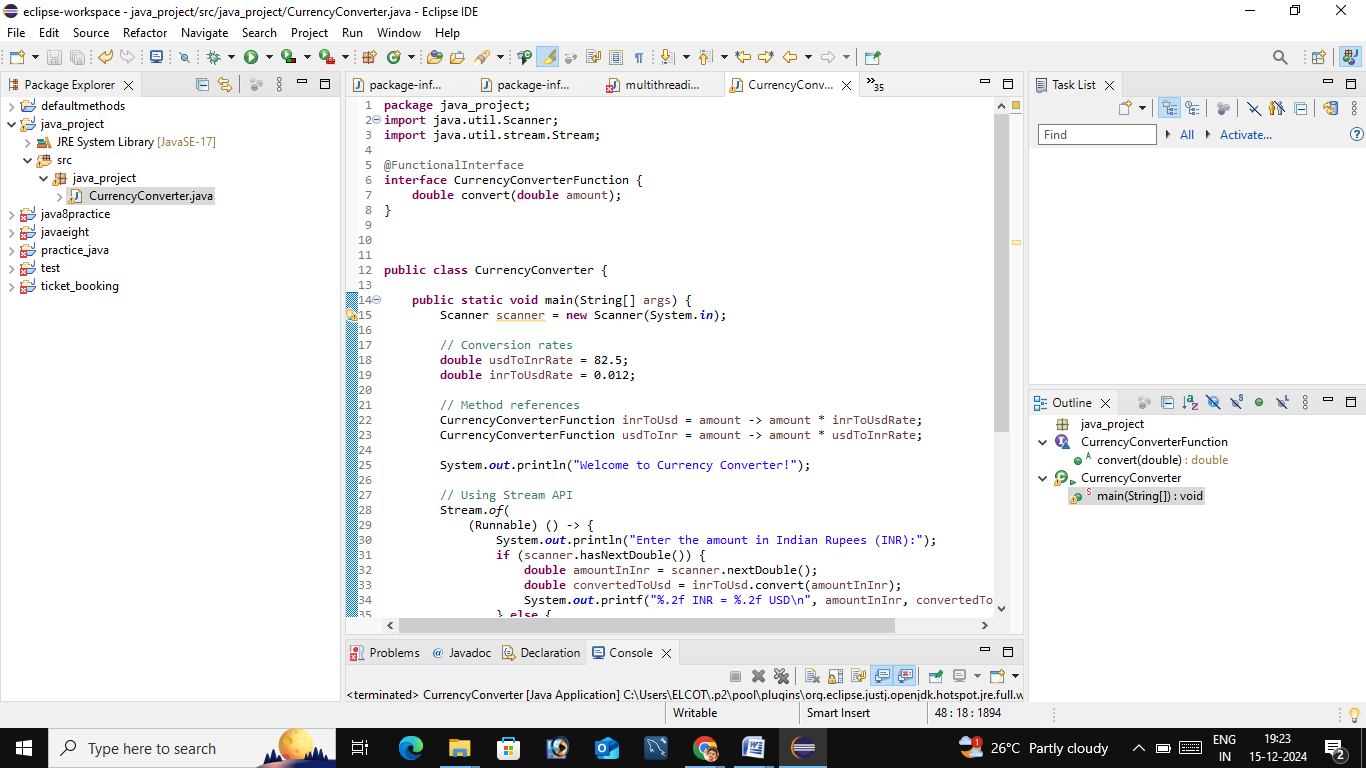
}

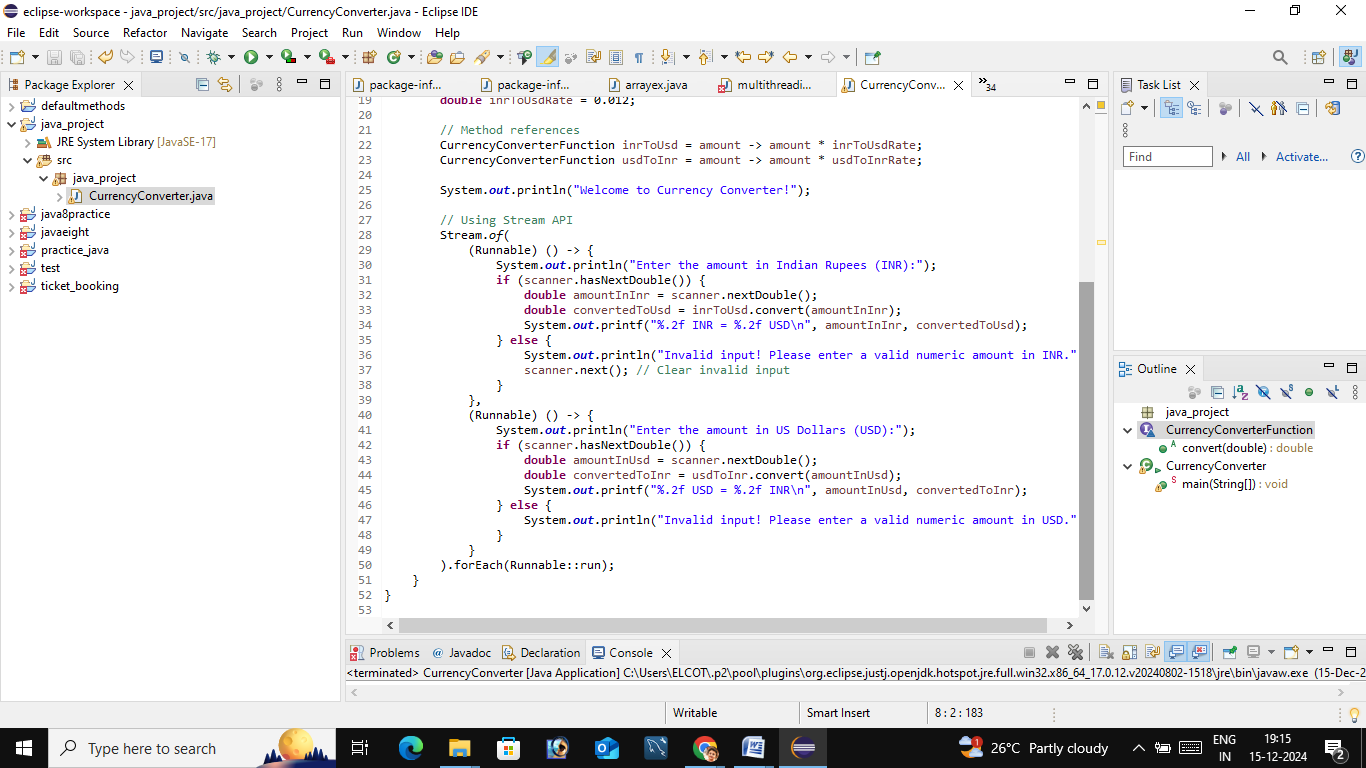
).forEach(Runnable::run);

}

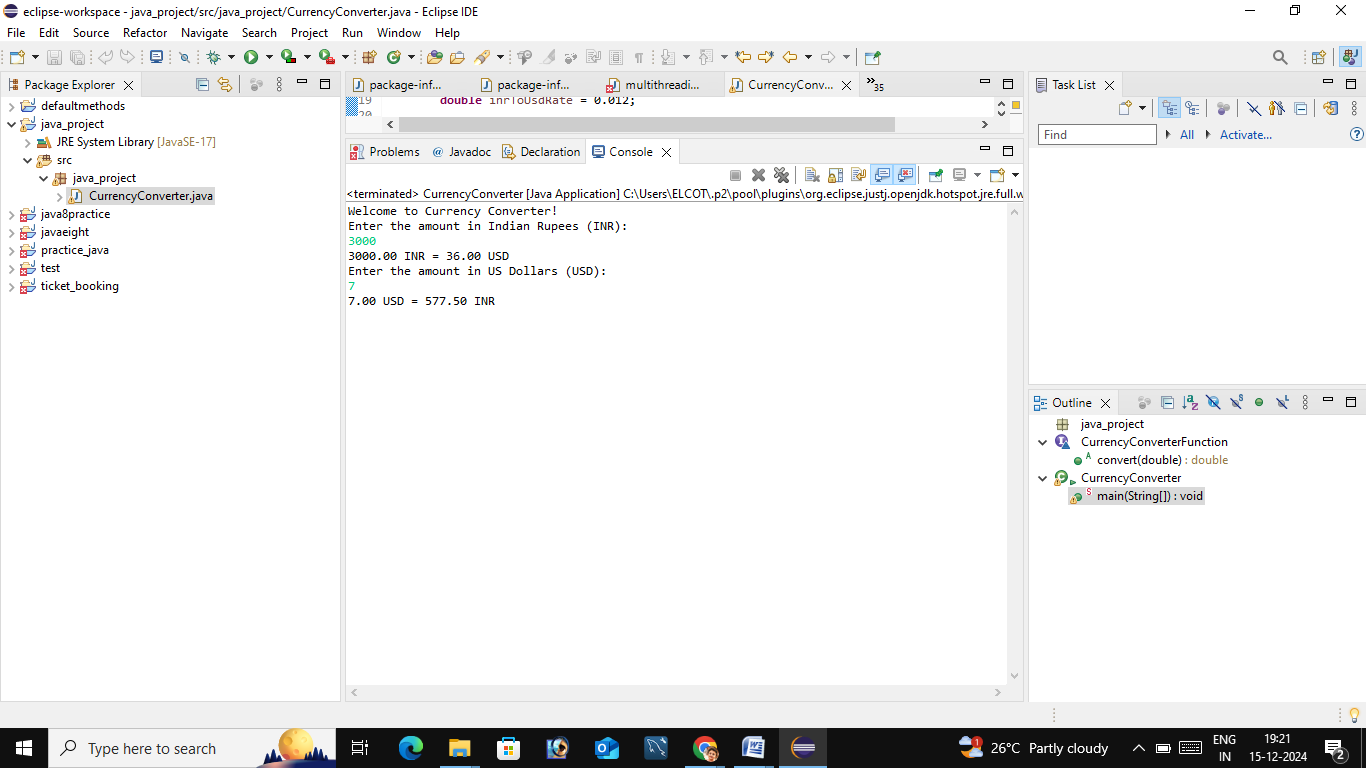
}

**Program screenshot :**

****



**Output :**



## Conclusion :

The **Currency Converter** project highlights the effective use of functional programming and stream-based sequential execution in Java. It showcases the seamless handling of currency conversion between INR and USD through a modular and scalable approach. By integrating functional interfaces, lambda expressions, and the Stream API, the project lays a strong foundation for developing more advanced and robust currency conversion tools in the future.

***Presented by – bhavani . M***