CURRENCY CONVERTER

A PROJECT REPORT

Submitted by

M.SIVARAJ (2303811710421152)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

DECEMBER- 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **“CURRENCY CONVERTER”** is the bonafide work of **SIVARAJ M (2303811710421152)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

[illegible]

SIGNATURE

Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

CGB1201-JAVA PROGRAMMING
Mr.MALARMANNAN A, M.E.,
[illegible]
ASSISTANTP ROFESSOR

[illegible]

SIGNATURE

Mr. A. Malarmannan, M.E.,

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology

(Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on – 06/12/2024

CGB1201-JAVA PROGRAMMING
Mr. M. SARAVANAN, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

[illegible]

INTERNAL EXAMINER

CGB1201-JAVA PROGRAMMING
Mr.R. KARTHIK, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8138-SCE, TRICHY.

[illegible] 06/12/2024

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **“CURRENCY CONVERTER”**is the result of original work done by us and best of our knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

Signature

Sivaraj.M

SIVARAJ M

Place: Samayapuram
Date: 06.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **MR. A. MALARMANNAN, M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a leading force in leveraging cutting-edge technology to drive students with innovation, efficiency and Strategic growth.

MISSION OF DEPARTMENT

M1: Academic Excellence: To provide high-quality education that equips students with the theoretical knowledge and practical skills necessary to excel in their careers and adapt to the evolving technological landscape.

M2: Innovative Research: To foster a research-oriented environment that encourages innovation, interdisciplinary collaboration, and the pursuit of cutting-edge solutions to complex problems in technology and related fields.

M3: Ethical and Inclusive Education: To promote ethical practices, diversity, and inclusion within the IT field, preparing students to be responsible and culturally aware professionals.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

A currency converter application is a digital tool designed to facilitate the conversion of one currency into another in real-time. This application caters to global travelers, businesses, and financial analysts, offering accurate and up-to-date exchange rates sourced from reliable financial institutions. It aims to simplify complex currency calculations through an intuitive user interface, enabling users to perform quick and efficient conversions. Key features include multi-currency support, historical rate analysis, offline functionality, and customizable settings for preferred currencies. By integrating advanced algorithms and responsive design, the application ensures seamless performance across devices, making it an indispensable resource for anyone dealing with foreign exchange transactions.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
A currency converter application is a digital tool designed to facilitate the conversion of one currency into another in real-time. This application caters to global travelers, businesses, and financial analysts, offering accurate and up-to-date exchange rates sourced from reliable financial institutions. It aims to simplify complex currency calculations through an intuitive user interface, enabling users to perform quick and efficient conversions. Key features include multi-currency support, historical rate analysis, offline functionality, and customizable settings for preferred currencies. By integrating advanced algorithms and responsive design, the application ensures seamless performance across devices, making it an indispensable resource for anyone dealing with foreign exchange transactions.t	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1. Objective:

A **Currency Converter** application is a program that allows users to convert an amount from one currency to another based on real-time or fixed exchange rates. This can be especially useful for travelers, businesses, or financial institutions that need to perform currency exchange tasks. In Java, a Currency Converter application can be built using basic programming concepts, including functions, user input, control flow, and data structures. The application typically requires an interface for the user to input the amount, select currencies, and display the converted result.

2. Overview:

The Currency Converter application in Java is a simple tool that allows users to convert an amount from one currency to another using predefined exchange rates. The program provides a user-friendly interface where users can input the amount, select the source and target currencies, and get the converted result. Using basic Java programming concepts such as variables, methods, control flow statements, and data structures like HashMap, the application efficiently performs currency conversions. The program can be easily extended by integrating real-time exchange rates from external APIs or expanding the set of supported currencies. This application is ideal for demonstrating core Java skills, including handling user input, performing arithmetic operations, and managing data using collections.

3. Java Programming Concepts:

1. Object-Oriented Programming (OOP):

- **Encapsulation:** Encapsulation refers to bundling the data (variables) and the methods (functions) that operate on the data into a single unit or class. It also involves restricting access to certain components of an object to protect its internal state.
- **Inheritance:** Inheritance allows one class to inherit the properties and methods of another class. For example, if you have different types of currencies (USD, EUR, GBP), you might create a base Currency class and then extend it with specific currency classes (e.g., USD Currency, Euro Currency).
- **Currency**: This class can represent a currency, including its name (e.g., USD, EUR),

symbol, and possibly a conversion rate.

- **Converter:** This class handles the logic for converting amounts between different currencies.
- **Exchange RateAPI:** A class to fetch exchange rates either from an external API or a hardcoded source.

2. **Graphical User Interface (GUI) Programming:**
   - **AWT (Abstract Window Toolkit):** It offers a set of basic components like buttons, labels, text fields, and windows, but lacks advanced features like customizable components and modern UI designs.
   - **Swing:** It provides a richer set of GUI components compared to AWT, including buttons, text fields, tables, trees, and more, with customizable features.
   - **Event Handling:** we need to manage user interactions such as selecting currencies, entering an amount, and clicking a button to perform the conversion.

2. **Data Structures:**
   - **ArrayList:** A simple **array** or **ArrayList** can be used to store a collection of available currencies.
   - This can be useful when you want to list all the currencies the user can select from.

4. **Control Flow Statements:**
   - **Conditional Statements:** Conditional statements allow you to execute certain blocks of code based on specific conditions. The most common ones are if, if-else, else-if, and switch.
   - **Loops:** Looping statements allow you to repeat a block of code multiple times based on a condition.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work

The **Proposed Work** section outlines the approach and methodology that will be followed to develop a Currency Converter application. The development process is broken down into distinct phases, starting with **requirements gathering** and concluding with deployment and testing. Here's an overview of each phase:

1. Requirements Gathering and Analysis:

The currency conversion application should meet the following key requirements: it must support multiple currencies and allow users to select a source and target currency. Users should be able to input an amount for conversion, and the system must display the converted value clearly. The application must provide fast and accurate conversion results, ideally within a few seconds. It should cater to a diverse user base, including individuals, businesses, and financial institutions. The system should ensure the conversion process is reliable, with exchange rates either hardcoded or dynamically fetched from an API. The focus should be on delivering a smooth, user-friendly experience.

2. System Design:

- ➢ Design the overall the application will run on the client (user's computer) and may connect to an external server for real-time exchange rates (if needed).
- ➢ Built using Java Swing or JavaFX for desktop applications. It includes elements like combo boxes (for currency selection), text fields (for amount input), and labels (for displaying results).
- ➢ Integrates with a real-time exchange rate API (e.g., Open Exchange Rates or Currency Layer) for up-to-date rates.

3. Implementation:

Prerequisites:

- Java Development Kit (JDK) installed on your machine.
- Basic understanding of Java Swing for GUI development.

Currency Converter:

- We'll use a JFrame for the window, JComboBox for currency selection, JTextField for entering the amount, and JLabel for showing the result.
- For simplicity, let's assume we have some static exchange rates.
- Add action listeners to the convert button to perform the conversion.

User Interface (UI):

- Display text (e.g., "Enter Amount in USD", "From Currency", "To Currency", "Converted Amount").
- For entering the amount to convert (e.g., USD amount).
- Dropdown lists to select source and target currencies (USD, EUR, INR, GBP).
- Convert" button to trigger the conversion.
- Shows the converted amount.

4. Testing:

- Test individual components (like the conversion logic) in isolation and ensure that each function works correctly.
- Check if the conversion logic correctly integrates with the UI inputs.
- Test the complete functionality of the application (e.g., currency conversion from one currency to another).
- Check if the app is user-friendly, and the UI is intuitive and easy to use.
- Test error messages for invalid inputs or API failures.

5. Documentation:

- Document the system design, including class diagrams, interaction diagrams, and UI wireframes.
- Prepare user manuals and guides to help users understand and use the application.
- Maintain code comments and documentation for future reference and maintenance.

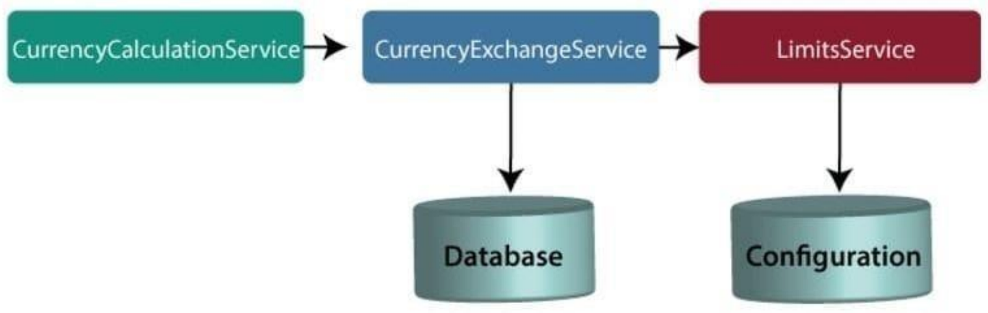
6. Deployment:

- Use your IDE (e.g., Eclipse or IntelliJ IDEA) to compile the Java project into .class files.
- This can be done using an IDE like Eclipse or IntelliJ IDEA, which allows you to compile the Java project into a .jar file containing all necessary code and resources.
- his eliminates the need for users to have Java installed. Alternatively, if you prefer to deploy the application online.
- you could convert it into a web application and host it on platforms like **Heroku** or **AWS**, making it accessible from any browser without requiring a local installation.

It refers to the process of making the **Currency Converter** application available for use by others. The deployment process will depend on the environment in which you want to run the application (e.g., as a standalone desktop app, in a web environment, or in the cloud).

2.2 Block Diagram

Currency convertor application flowchart



CHAPTER 3
MODULE DESCRIPTION

1. Currency Conversion Logic Module:

The Currency Converter module is designed to convert amounts between different currencies based on exchange rates. It includes a conversion function that takes three inputs: the amount, the source currency, and the target currency. The function calculates the converted amount by multiplying the input amount by the appropriate exchange rate. Exchange rates can either be hardcoded into the system or dynamically fetched from an external API. The module is responsible for looking up the exchange rate, performing the conversion, and handling potential errors, such as missing rates or invalid currencies. This approach allows for efficient and flexible currency conversion in various applications, making it useful for financial software, e-commerce platforms, and other systems that require currency management.

2. Validation Module:

This module ensures that the input provided by the user is valid and that no errors occur during the conversion process.

- ➢ Input Validation: Ensures the amount entered is a valid number and is greater than zero.
- ➢ Currency Validation: Checks that the selected source and target currencies are valid and available.
- ➢ Error Handling: Manages situations where the user has input invalid data (e.g., non-numeric values or empty fields).

3. Logging and Error Handling Module:

This module manages application logs and error handling, ensuring that any issues or exceptions are captured and handled properly.

- ➢ Logging System: Records application events such as conversions made and errors encountered.
- ➢ Error Handler: Catches unexpected exceptions and displays user-friendly error messages.

4. User Interface (UI) Module:

This module is responsible for presenting the graphical interface to the user and capturing user input.

- ➢ Input Fields: Where the user enters the amount they wish to convert.
- ➢ Currency Selectors: Dropdown menus or combo boxes for selecting the source and target currencies.

- Convert Button: Triggers the currency conversion when clicked.
- Output Area: Displays the converted amount and any error messages if the input is invalid.

3.5 Unit Testing Module:

This module is responsible for testing individual parts of the application to ensure that they work correctly.

- Unit Tests: Test methods and components of the application, such as the currency conversion logic, input validation, and event handling.
- Mocking: Simulate external dependencies like the exchange rate API to test functionality in isolation.

CHAPTER 4
CONCLUSION

The Currency Converter application is a user-friendly tool designed to quickly and accurately convert currencies based on real-time or static exchange rates. It consists of key modules: the User Interface (UI) for user input and output, Currency Conversion Logic for performing the conversion calculations, and the Data Management Module for retrieving or storing exchange rates. The application handles potential errors, such as invalid inputs or API failures, and can fall back on cached data when necessary. Its modular design makes it easy to extend and update, allowing for the addition of new currencies or features. With real-time data integration, the Currency Converter ensures reliable and accurate conversions, making it a valuable tool for personal or business use, particularlyfor travelers or anyone dealing with international transactions.

FUTURE SCOPE

The future scope of the Currency Converter application includes integrating advanced features such as AI-driven predictive exchange rate analysis, multi-currency conversion, and offline mode with dynamic updates upon reconnection. The application could expand to support cryptocurrency conversions and integrate with digital wallets for seamless transactions. Additionally, incorporating voice recognition for hands-free operation and localized language support would enhance accessibility and user experience. Such developments would make the application more versatile and indispensable for global users in an increasingly interconnected world.

APPENDIX - A

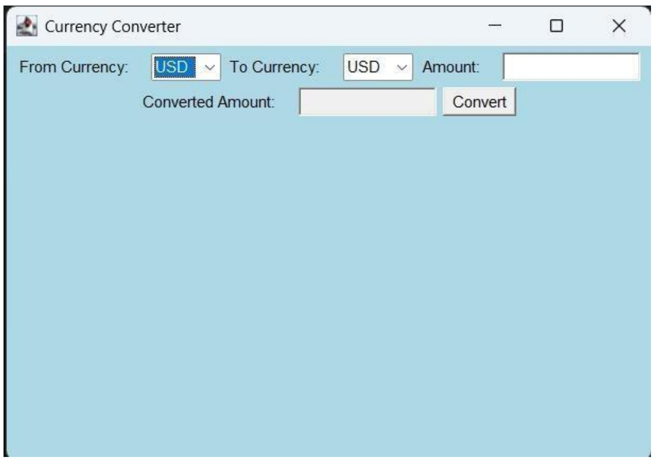
(Project Source Code)

```
import java.awt.*;
import java.awt.event.*;
import java.util.HashMap;
import javax.swing.ImageIcon;

public class CurrencyConverter extends Frame implements ActionListener {

    // Define components
    Label labelFrom, labelTo, labelAmount, labelResult;
    Choice choiceFrom, choiceTo;
    TextField textAmount, textResult;
    Button buttonConvert;

    // Exchange rates (for simplicity, we'll use hardcoded rates)
    HashMap<String, Double> exchangeRates;

    // Declare an Image object for the background image
    Image backgroundImage;

    public CurrencyConverter() {
        // Set up the frame
        setTitle("Currency Converter");
        setSize(500, 350);
        setLayout(new FlowLayout());

        // Set the background color for the frame (it will be replaced by the image)
        setBackground(new Color(173, 216, 230)); // Light blue background

        // Initialize exchange rates
        exchangeRates = new HashMap<>();
        exchangeRates.put("USD", 1.0);
        exchangeRates.put("INR", 82.74);
        exchangeRates.put("EUR", 0.92);  // Example: 1 USD = 0.92 EUR
        exchangeRates.put("GBP", 0.77);  // Example: 1 USD = 0.77 GBP
        exchangeRates.put("JPY", 148.76); // Example: 1 USD = 148.76 JPY

        // Load the background image
        backgroundImage = new ImageIcon("currency_background.jpg").getImage(); // Provide
the path to your image

        // Initialize components
        labelFrom = new Label("From Currency:");
        labelTo = new Label("To Currency:");
        labelAmount = new Label("Amount:");
        labelResult = new Label("Converted Amount:");
        choiceFrom = new Choice();
        choiceFrom.add("USD");
        choiceFrom.add("INR");
        choiceFrom.add("EUR");
        choiceFrom.add("GBP");
        choiceFrom.add("JPY");

        choiceTo = new Choice();
        choiceTo.add("USD");
        choiceTo.add("INR");
        choiceTo.add("EUR");
        choiceTo.add("GBP");
        choiceTo.add("JPY");

        textAmount = new TextField(10);
        textResult = new TextField(10);
        textResult.setEditable(false);

        buttonConvert = new Button("Convert");

        // Add components to the frame
        add(labelFrom);
        add(choiceFrom);
        add(labelTo);
        add(choiceTo);
        add(labelAmount);
        add(textAmount);
        add(labelResult);
        add(textResult);
        add(buttonConvert);

        // Add action listener to the button
        buttonConvert.addActionListener(this);

        // Window close event
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });
    }

    // Override the paint() method to draw the background image
    public void paint(Graphics g) {
        // Draw the background image
        g.drawImage(backgroundImage, 0, 0, getWidth(), getHeight(), this);

        // Call the superclass paint method to ensure components are drawn over the background
        super.paint(g);
    }
```

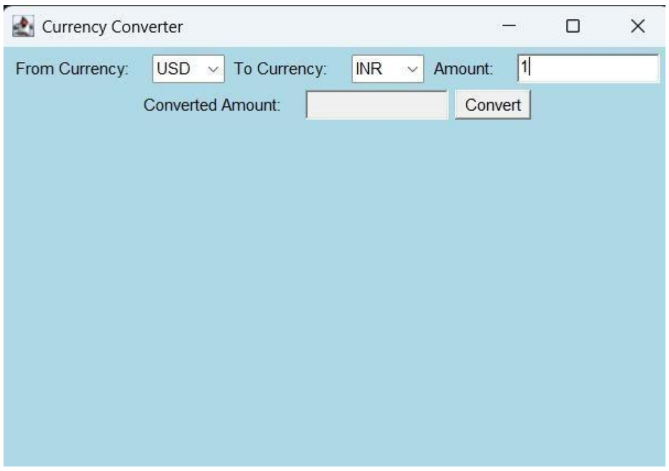
```
    // Action listener method
    public void actionPerformed(ActionEvent e) {
        try {
            // Get the input amount
            double amount = Double.parseDouble(textAmount.getText());

            // Get selected currencies
            String fromCurrency = choiceFrom.getSelectedItem();
            String toCurrency = choiceTo.getSelectedItem();

            double convertedAmount = 0;

            // Conversion logic based on selected currencies
            if (!fromCurrency.equals(toCurrency)) {
                double fromRate = exchangeRates.get(fromCurrency);
                double toRate = exchangeRates.get(toCurrency);

                // Conversion formula: (amount / fromCurrency rate) * toCurrency rate
                convertedAmount = (amount / fromRate) * toRate;
            } else {
                // No conversion needed if both currencies are the same
                convertedAmount = amount;
            }

            // Display the converted amount
            textResult.setText(String.format("%.2f", convertedAmount));
        } catch (NumberFormatException ex) {
            textResult.setText("Invalid input");
        }
    }

    // Main method to launch the application
    public static void main(String[] args) {
        CurrencyConverter app = new CurrencyConverter();
        app.setVisible(true);
    }
}
```

APPENDIX - B

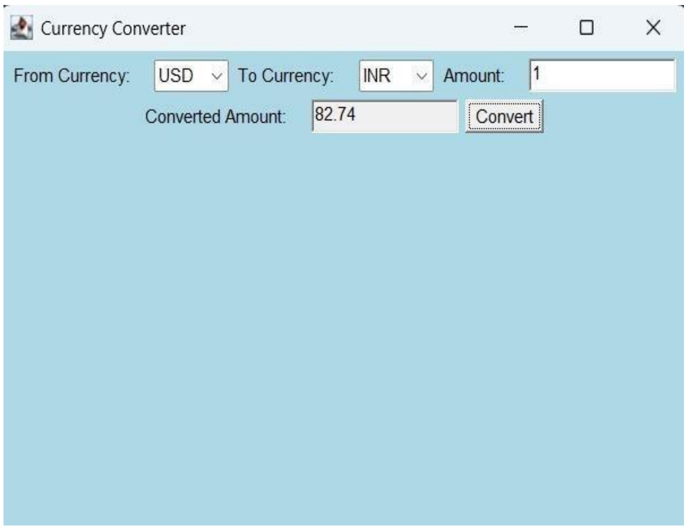
Currency Converter Display



Read The Inputs



Exicution The Output



REFERENCES

1. Ariponnammal, S. and Natarajan, S. (1994) 'Transport Phonomena of Sm Sel - X Asx', Pramana - Journal of Physics Vol.42, No.1, pp.421-425.

2. Barnard, R.W. and Kellogg, C. (1980) 'Applications of Convolution Operators to Problems in Univalent Function Theory', Michigan Mach, J., Vol.27, pp.81-94.

3. Shin, K.G. and Mckay, N.D. (1984) 'Open Loop Minimum Time Control of Mechanical Manipulations and its Applications', Proc. Amer.Contr. Conf., San Diego, CA, pp. 1231-1236.