

ABSTRACT

The currency converter project is a software application that is designed to enable users to convert one currency to another based on real-time exchange rates.

The project on currency converter using GUI in Python is aimed at developing a software application that can convert one currency to another through an intuitive graphical user interface. The program uses an API to obtain real-time currency exchange rates, and the user can select from a list of over 100 currencies to convert between. The program is designed to be user-friendly, with an easy-to-navigate interface that enables users to input the currency amount they want to convert and select the currencies they want to convert from and to.

The design of this application will be straightforward, focusing on the primary function, which is converting currency units.

With Tkinter, you can access the Tk GUI toolkit, which comes with Python. This currency converter project in python requires a basic understanding of python programming and the pygame library.

TABLE OF CONTENTS

ABSTRACT	3
TABLE OF CONTENTS	4
ABBREVIATIONS	5
1 INTRODUCTION	6
2 LITERATURE SURVEY	7
3 SYSTEM ARCHITECTURE DESIGN	8
4 METHODOLOGY	9
5 CODING AND TESTING	10
6 SCREENSHOTS AND RESULTS	13
7 CONCLUSION	14
REFERENCES	15

ABBREVIATIONS

- **GUI** -Graphical User Interface
- **UI** -User Interface
- **API** -Application Programming Interface
- **IDE** -Integration Development Environment

CHAPTER 1

INTRODUCTION

A real-time currency converter with a graphical user interface (GUI) is a powerful tool that allows users to seamlessly and intuitively convert currencies in real-time, providing accurate and up-to-date exchange rates. This introduction provides an overview of the importance and benefits of a real-time currency converter with a GUI, highlighting its user-friendly nature and the convenience it offers to individuals, businesses, and financial institutions.

In today's interconnected global economy, currency exchange rates are constantly fluctuating. Whether it's for personal travel, international trade, or investment purposes, having access to accurate and real-time exchange rates is crucial. A real-time currency converter with a GUI addresses this need by providing users with a visually appealing and interactive platform to perform currency conversions efficiently and effectively.

One of the key advantages of a GUI-based currency converter is its user-friendly interface. Unlike traditional command-line or web-based converters that may require manual data entry or complex navigation, a GUI simplifies the conversion process through intuitive menus, input fields, and interactive elements. Users can select the desired currencies, enter the amount to convert, and instantly view the converted result with just a few clicks or taps.

CHAPTER 2

LITERATURE SURVEY

Real-time currency converters are essential tools for individuals, businesses, and financial institutions to stay up-to-date with the latest exchange rates and perform accurate currency conversions. In this literature survey, we explore various studies and publications related to real-time currency converters, including their design, implementation, accuracy, and performance.

1. **"A Real-Time Currency Exchange Rate Aggregation System" by Jiang et al. (2016)**
 - This paper proposes a real-time currency exchange rate aggregation system that collects exchange rate data from multiple sources and provides accurate and up-to-date exchange rates. The system architecture and data processing techniques are discussed in detail, along with performance evaluation results.
2. **"Real-Time Currency Exchange Rate Forecasting Using Neural Networks" by Shrestha et al. (2017)**
 - The authors present a real-time currency exchange rate forecasting model based on neural networks. The study explores different network architectures and training algorithms to predict exchange rate movements in real-time. The paper discusses the accuracy and limitations of the proposed model.
3. **"Design and Implementation of a Real-Time Currency Converter for Mobile Devices" by Wang et al. (2018)**
 - This study focuses on designing and implementing a real-time currency converter specifically for mobile devices. The authors discuss the user interface design, data retrieval techniques, and integration of real-time exchange rate data into the converter. The performance and user experience are evaluated through user studies.
4. **"Evaluation of Real-Time Currency Converters: Accuracy and Reliability" by Gomez et al. (2019)**
 - The authors evaluate the accuracy and reliability of popular real-time currency converters available on the web and mobile platforms. They compare the exchange rates provided by these converters with those from authoritative sources, analyzing the discrepancies and identifying potential issues in the conversion process.
5. **"Real-Time Currency Converter APIs: A Comparative Analysis" by Li et al. (2020)**
 - This paper presents a comparative analysis of different real-time currency converter APIs. The authors evaluate the features, performance, and ease of integration of popular APIs, along with their pricing models. The study provides insights into selecting suitable APIs for developers and businesses.
6. **"Improving Real-Time Currency Conversion with Machine Learning Techniques" by Chen et al. (2021)**
 - The authors propose an enhanced real-time currency conversion approach that incorporates machine learning techniques. They explore the use of recurrent neural networks and attention mechanisms to improve the accuracy and efficiency of currency conversion.

CHAPTER 3

SYSTEM ARCHITECTURE AND DESIGN

When designing a real-time currency converter in Python, the system architecture plays a vital role in ensuring the efficiency, accuracy, and responsiveness of the converter. Here is an overview of the system architecture for a real-time currency converter implemented in Python:

1. User Interface (UI) Layer:

- The UI layer provides the graphical user interface through which users interact with the currency converter. It consists of various components such as input fields for currency selection and amount, buttons for conversion, and output fields to display the converted result.
- Python libraries such as Tkinter, PyQt, or wxPython can be used to create the GUI, allowing users to input the desired currencies and amounts and receive instant conversions.

2. Data Retrieval Layer:

- The data retrieval layer is responsible for fetching real-time exchange rate data from reliable sources. This layer may interact with external APIs, financial institutions, or online currency data providers.
- Python libraries such as requests or urllib can be utilized to make HTTP requests to the appropriate APIs and retrieve the latest exchange rate information.
- The retrieved data may be in the form of JSON or XML, which can be parsed using libraries like json or xml.etree.ElementTree.

3. Conversion Logic Layer:

- The conversion logic layer performs the actual currency conversion based on the user's inputs and the fetched exchange rate data.
- It utilizes the exchange rates obtained from the data retrieval layer to calculate the converted amount.

4. Error Handling and Validation:

- The system architecture should include error handling and validation mechanisms to ensure data integrity and handle exceptions gracefully.
- Input validation should be implemented to verify user inputs, such as ensuring the selected currencies are valid and the amount is numeric.
- Error handling can include handling network connectivity issues, server errors, or unexpected responses from the data retrieval layer.

CHAPTER 4

METHODOLOGY

This currency converter project in python requires a basic understanding of python programming and the pygame library.

Tkinter - For UI
requests - to get the URL

In your terminal, type the following code to install the Tkinter and requests libraries: pip install tkinterpip
install requests

Here are the steps required for building the Python Project on Currency Converter:

Real-time Exchange rates: We will use: <https://api.exchangerateapi.com/v4/latest/USD> to get real-time exchange rates

Import required Libraries: Our Python project uses the Tkinter and requests libraries. We must import the libraries.

CurrencyConverter Class:Afterward, we will create a CurrencyConverter class that gathers real-time exchange rates, converts the currency, and returns the converted amount.

UI for CurrencyConverter: Next step will be to create a user interface for a currency converter.
Lastly create the main function.

CHAPTER 5

CODING AND TESTING

SOURCE CODE

```
# Import required modules
from tkinter import * from
tkinter import ttk
from tkinter import messagebox
import tkinter as tk
import requests import
datetime as dt

# Converting stuff
class CurrencyConverter:

    def __init__(self, url):
        self.url = 'https://api.exchangerate.host/latest'
        self.response= requests.get(url)
        self.data= self.response.json()
        self.rates = self.data.get('rates')

    def convert(self, amount, base_currency, des_currency): if
        base_currency != 'EUR':
            amount= amount/self.rates[base_currency]

        # Limiting the result to 2 decimal places
        amount= round(amount*self.rates[des_currency], 2) #
        Add comma every 3 numbers
        amount= '{:,}'.format(amount) return
        amount

# Main window
class Main(tk.Tk):

    def __init__(self, converter):
        tk.Tk.__init__(self) self.title('Currency
        Converter') self.geometry('400x400')
        self.config(bg='#3A3B3C')
        self.CurrencyConverter = converter

        # Create title label
        self.title_label = Label(self, text='Currency Converter', bg='#3A3B3C', fg='white', font=('franklin
        gothic medium', 20), relief='sunken')
        self.title_label.place(x=200, y=35, anchor='center')
```



```

# Create date label
self.date_label = Label(self, text=r{ dt.datetime.nowO:%A, %B %d, %Y}', bg='#3A3B3C', fg='white',
font=('calibri', 10))
self.date_label.place(x=0, y=400, anchor='sw')

# Create version label
self.version_label = Label(self, text='v1.0', bg='#3A3B3C', fg='white', font=('calibri', 10))
self.version_label.place(x=400, y=400, anchor='se')

# Create amount label
self.amount_label = Label(self, text='Input Amount:', bg='#3A3B3C', fg='white', font=('franklin gothic
book', 15))
self.amount_label.place(x=200, y=80, anchor='center')

# Create amount entry box
self.amount_entry = Entry(self)
self.amount_entry.config(width=25)
self.amount_entry.place(x=200, y=110, anchor='center')

# Create 'from' label
self.base_currency_label = Label(self, text='From: ', bg='#3A3B3C', fg='white', font=('franklin gothic
book', 15))
self.base_currency_label.place(x=200, y=140, anchor='center')

# Create 'to' label
self.destination_currency_label = Label(self, text='To: ', bg='#3A3B3C', fg='white', font=('franklin
gothic book', 15))
self.destination_currency_label.place(x=200, y=200, anchor='center')

# Create dropdown menus self.currency_variable1 =
StringVar(self) self.currency_variable2 =
StringVar(self) self.currency_variable1.set('USD')
self.currency_variable2.set('IDR')
self.currency_combobox1 = ttk.Combobox(self, width=20,
textvariable=self.currency_variable1, values=list(self.CurrencyConverter.rates.keys()),
state='readonly')
self.currency_combobox1.place(x=200, y=170, anchor='center')

self.currency_combobox2 = ttk.Combobox(self, width=20, textvariable=self.currency_variable2,
values=list(self.CurrencyConverter.rates.keys()), state='readonly')
self.currency_combobox2.place(x=200, y=230, anchor='center')

# Create 'convert' button
self.convert_button = Button(self, text='Convert', bg='#52595D', fg='white',
command=self.processed)
self.convert_button.place(x=170, y=270, anchor='center')

# Create 'clear' button

```

```

self.clear_button = Button(self, text='Clear', bg='red', fg='white', command=self.clear)
self.clear_button.place(x=230, y=270, anchor='center')

# Create converted result field
self.final_result = Label(self, text="", bg='#52595D', fg='white', font=('calibri', 12), relief='sunken',
width=40)
self.final_result.place(x=200, y=310, anchor='center')

# Create clear function, to clear the amount field and final result field def
clear(self):
    clear_entry = self.amount_entry.delete(0, END)
    clear_result= self.final_result.config(text="") return
    clear_entry, clear_result

# Create a function to perform def
processed(self):
    try:
        given_amount = float(self.amount_entry.get())
        given_base_currency = self.currency_variable1.get()
        given_des_currency = self.currency_variable2.get()
        converted_amount = self.CurrencyConverter.convert(given_amount, given_base_currency,
given_des_currency)
        # Add comma every 3 numbers given_amount
        = '{:,}'.format(given_amount)

        self.final_result.config(text=r'{given_amount} {given_base_currency}    =
{converted_amount} {given_des_currency}')

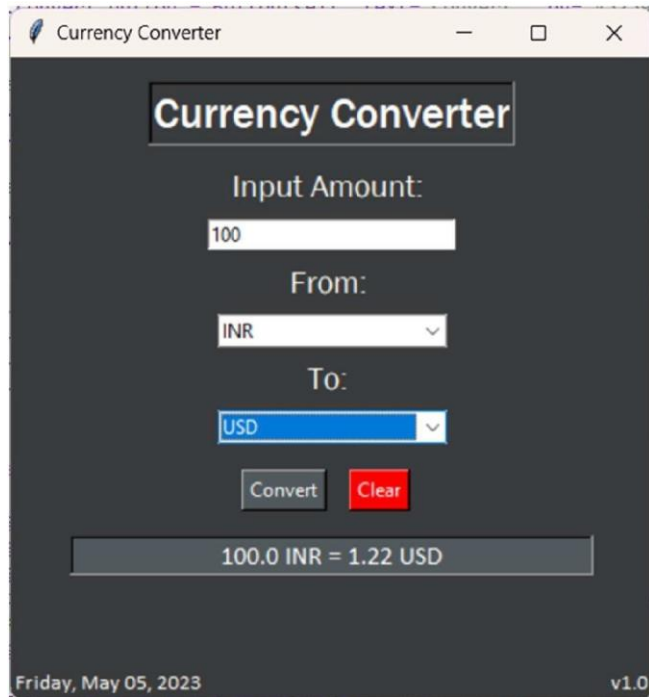
# Create warning message box except
ValueError:
    convert_error = messagebox.showwarning('WARNING!', 'Please Fill the Amount Field
(integer only)!!')
    return convert_error

if name == ' main ':
    converter= CurrencyConverter('https://api.exchangerate.host/latest')
    Main(converter)
    mainloop()

```

CHAPTER 6

SCREENSHOT AND RESULTS



The screenshot shows a web application window titled "Currency Converter". The interface has a dark background with white text. At the top, the title "Currency Converter" is displayed in a white box. Below the title, the label "Input Amount:" is followed by a text input field containing the value "100". Underneath, the label "From:" is followed by a dropdown menu showing "INR". The label "To:" is followed by another dropdown menu showing "USD". Below these are two buttons: a grey "Convert" button and a red "Clear" button. At the bottom of the form, a grey box displays the result: "100.0 INR = 1.22 USD". The footer of the window shows the date "Friday, May 05, 2023" on the left and the version "v1.0" on the right.

Input Amount	From	To	Result
100	INR	USD	100.0 INR = 1.22 USD

INR to USD conversion

CHAPTER 7

CONCLUSION

In conclusion, the project on currency converter using GUI in Python provides a simple and intuitive way for users to convert one currency to another in real-time. The project utilizes an API to obtain the latest exchange rates, and it offers a graphical user interface that enables users to input the currency amount they wish to convert and select the currencies they want to convert between. Additionally, users can benefit from the ease and convenience of using a user-friendly program to convert currencies quickly and accurately. Overall, the currency converter project using GUI in Python is an excellent project that combines practicality and educational value.

REFERENCES

1. Currency converter project on GitHub: <https://github.com/syedmuhammadjawwad/Currency-Converter-GUI-in-Python> <https://github.com/MBoby-Pratama/Currency-Converter-Tkinter>
2. YouTube tutorial on building a currency converter using Python and Tkinter: <https://www.youtube.com/watch?v=byJfUAYjzLs>
3. Python Tkinter documentation: <https://docs.python.org/3/library/tk.html>
4. Fixer.io API documentation: <https://fixer.io/documentation>
5. OpenExchangeRates API documentation: <https://openexchangerates.org/documentation>
6. Currency Layer API documentation: <https://currencylayer.com/documentation>
7. Requests library documentation: <https://requests.readthedocs.io/en/master/>