**Real-time currency converter using python.**

**Overview of Project.**

Real Time Currency converter using Python is a simple project developed using Python. This project is a GUI application which converts Currency from one unit to another (euros and pounds). Also, this app is capable of handling all types of exceptions

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications.

**Prerequisites :**Introduction to tkinter , Get the real time currency exchange rate.

**Modules required:**

1. tkinter - For User Interface (UI).
2. requests - to get url.
3. json

To install the tkinter and requests library, type the following code in your terminal:

pip install tkinter

pip install requests

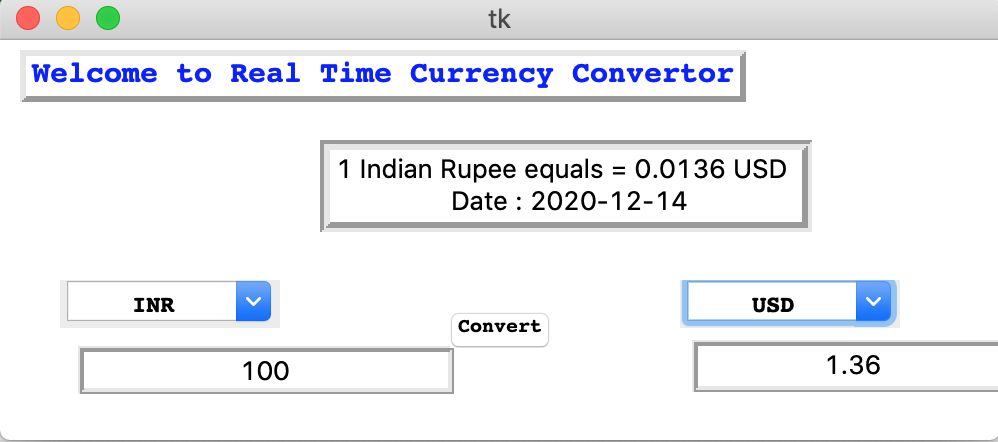
Steps involved in building Python Project on Real time Currency Converter

1. Real-time Exchange rates
2. Import required Libraries
3. CurrencyConverter Class
4. UI for CurrencyConverter
5. Main Function

**For Real-time Exchange rates following URL will be used.**

To get real-time exchange rates, we will use: https://api.exchangerate-api.com/v4/latest/USD

**GUI**



**Implementation (code)**

In [1]:

pip install tkinter

Collecting tkinter

ERROR: Could not find a version that satisfies the requirement tkinter (from versions: non

e)

ERROR: No matching distribution found for tkinter

Note: you may need to restart the kernel to use updated packages.

In [2]:

pip install requests

Requirement already satisfied: requests in ./anaconda3/lib/python3.7/site-packages (2.22.0) Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in ./anaconda3/lib/py thon3.7/site-packages (from requests) (1.24.2)  
Requirement already satisfied: idna<2.9,>=2.5 in ./anaconda3/lib/python3.7/site-packages (fr om requests) (2.8)

Requirement already satisfied: certifi>=2017.4.17 in ./anaconda3/lib/python3.7/site-packages (from requests) (2019.6.16)  
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in ./anaconda3/lib/python3.7/site-packa ges (from requests) (3.0.4)

Note: you may need to restart the kernel to use updated packages.

# # Python Project on Currency Converter

import requests

from tkinter import \*

import tkinter as tk

from tkinter import ttk

class RealTimeCurrencyConverter():

def \_\_init\_\_(self,url):

self.data = requests.get(url).json()

self.currencies = self.data['rates']

def convert(self, from\_currency, to\_currency, amount):

initial\_amount = amount

if from\_currency != 'USD' :

amount = amount / self.currencies[from\_currency]

# limiting the precision to 4 decimal places

amount = round(amount \* self.currencies[to\_currency], 4)

return amount

class App(tk.Tk):

def \_\_init\_\_(self, converter):

tk.Tk.\_\_init\_\_(self)

self.title = 'Currency Converter'

self.currency\_converter = converter

#self.configure(background = 'blue')

self.geometry("500x200")

# Label

self.intro\_label = Label(self, text = 'Welcome to Real Time Currency Convertor', fg = 'blue', relief = tk.RAISED, borderwidth = 3)

self.intro\_label.config(font = ('Courier',15,'bold'))

self.date\_label = Label(self, text = f"1 Indian Rupee equals = {self.currency\_converter.convert('INR','USD',1)} USD \n Date : {self.currency\_converter.data['date']}", relief = tk.GROOVE, borderwidth = 5)

self.intro\_label.place(x = 10 , y = 5)

self.date\_label.place(x = 160, y= 50)

# Entry box

valid = (self.register(self.restrictNumberOnly), '%d', '%P')

self.amount\_field = Entry(self,bd = 3, relief = tk.RIDGE, justify = tk.CENTER,validate='key', validatecommand=valid)

self.converted\_amount\_field\_label = Label(self, text = '', fg = 'black', bg = 'white', relief = tk.RIDGE, justify = tk.CENTER, width = 17, borderwidth = 3)

# dropdown

self.from\_currency\_variable = StringVar(self)

self.from\_currency\_variable.set("INR") # default value

self.to\_currency\_variable = StringVar(self)

self.to\_currency\_variable.set("USD") # default value

font = ("Courier", 12, "bold")

self.option\_add('\*TCombobox\*Listbox.font', font)

self.from\_currency\_dropdown = ttk.Combobox(self, textvariable=self.from\_currency\_variable,values=list(self.currency\_converter.currencies.keys()), font = font, state = 'readonly', width = 12, justify = tk.CENTER)

self.to\_currency\_dropdown = ttk.Combobox(self, textvariable=self.to\_currency\_variable,values=list(self.currency\_converter.currencies.keys()), font = font, state = 'readonly', width = 12, justify = tk.CENTER)

# placing

self.from\_currency\_dropdown.place(x = 30, y= 120)

self.amount\_field.place(x = 36, y = 150)

self.to\_currency\_dropdown.place(x = 340, y= 120)

#self.converted\_amount\_field.place(x = 346, y = 150)

self.converted\_amount\_field\_label.place(x = 346, y = 150)

# Convert button

self.convert\_button = Button(self, text = "Convert", fg = "black", command = self.perform)

self.convert\_button.config(font=('Courier', 10, 'bold'))

self.convert\_button.place(x = 225, y = 135)

def perform(self):

amount = float(self.amount\_field.get())

from\_curr = self.from\_currency\_variable.get()

to\_curr = self.to\_currency\_variable.get()

converted\_amount = self.currency\_converter.convert(from\_curr,to\_curr,amount)

converted\_amount = round(converted\_amount, 2)

self.converted\_amount\_field\_label.config(text = str(converted\_amount))

def restrictNumberOnly(self, action, string):

regex = re.compile(r"[0-9,]\*?(\.)?[0-9,]\*$")

result = regex.match(string)

return (string == "" or (string.count('.') <= 1 and result is not None))

if \_\_name\_\_ == '\_\_main\_\_':

url = 'https://api.exchangerate-api.com/v4/latest/USD'

converter = RealTimeCurrencyConverter(url)

App(converter)

mainloop()