Currency Converter

Import Useful Libraries

In [1]:
```
import requests
from tkinter import *
import tkinter as tk
from tkinter import ttk
```

Class for Real Time Currency Converter with base currency of USD

For Example:

1 USD = 154.81 PKR

1 USD = 1.31 AUD

If we want to convert 254 PKR into AUD then,

1st step: Convert PKR to USD

amount = 254/154.81 = 1.64

2nd step: Convert USD to AUD

amount = 1.64*1.31 = 2.15

In [2]:
```
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):
        amount = amount / self.currencies[from_currency]
        amount = round(amount * self.currencies[to_currency], 4)
        return amount
```

Class for GUI

In [3]:
```
class App(tk.Tk):
    def __init__(self, converter):
        tk.Tk.__init__(self)
        self.title = 'Currency Converter'
        self.currency_converter = converter
        self.geometry("500x200")

        self.from_currency_variable = StringVar(self)
        self.from_currency_variable.set("PKR")   # default value
        self.to_currency_variable = StringVar(self)
        self.to_currency_variable.set("USD")     # default value

        # ================================================================================================================
        # ================================================ Labels ========================================================
        # ================================================================================================================
        cc = self.currency_converter.convert('PKR','USD',1)
        cd = self.currency_converter.data['date']

        self.intro_label = Label(self, text='Welcome to Real Time Currency Convertor', fg='blue')
        self.intro_label.config(font = ('Courier',15,'bold'), relief = tk.RAISED, borderwidth=3)
        self.intro_label.place(x=10 , y=5)

        self.date_label = Label(self, text = f"1 PKR Rupee equals = {cc} USD \n Date : {cd}")
        self.date_label.config(relief = tk.GROOVE, borderwidth=5)
        self.date_label.place(x=160, y=50)

        # ================================================================================================================
        # ============================================== Entry boxes =====================================================
        # ================================================================================================================
        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')
        self.converted_amount_field_label.config(relief = tk.RIDGE, justify = tk.CENTER, width=17, borderwidth=3)

        # ================================================================================================================
        # ============================================== Dropdowns =======================================================
        # ================================================================================================================
        self.from_currency_dropdown = ttk.Combobox(self, textvariable = self.from_currency_variable)
        self.from_currency_dropdown.config(values = list(self.currency_converter.currencies.keys()))
        self.from_currency_dropdown.config(font=("Courier", 12, "bold"), state='readonly', width=12, justify = tk.CENTER)

        self.to_currency_dropdown = ttk.Combobox(self, textvariable = self.to_currency_variable)
        self.to_currency_dropdown.config(values = list(self.currency_converter.currencies.keys()))
        self.to_currency_dropdown.config(font=("Courier", 12, "bold"), 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_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)

    # To perform Currency conversion and display Converted amount
    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))

    # To restrict Numbers only
    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))
```

Main Function

In [4]:
```
if __name__ == '__main__':
    url = 'https://api.exchangerate-api.com/v4/latest/USD'
    converter = RealTimeCurrencyConverter(url)

    App(converter)
    mainloop()
```