

CURRENCY CONVERTER APPLICATION

MINOR PROJECT REPORT

By

PARVATHY V NAIR (RA2211003010295)
ANYESHA BISWAS (RA2211003010298)
ACHANTA BHAVYA SREE(RA2211003010316)

Under the guidance of

Dr. Ajanthaa Lakkshmanan

In partial fulfilment for the Course

of

21CSC203P – ADVANCED PROGRAMMING PRACTICE

in **Department of Computing Technologies**



FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this minor project report for the course **21CSC203P
ADVANCED PROGRAMMING PRACTICE** entitled in “**Currency
Converter Application**” is the bonafide work of **Parvathy V Nair
(RA2211003010295), Anyesha Biswas (RA2211003010298) and Achanta
Bhavya Sree (RA2211003010316)** who carried out the work under my
supervision.

SIGNATURE
DR. AJANTHAA LAKKSHMANAN
ASSISTANT PROFESSOR
DEPARTMENT OF COMPUTING
TECHNOLOGIES

SIGNATURE
DR. M PUSHPALATHA
HEAD OF THE DEPARTMENT
DEPARTMENT OF COMPUTING
TECHNOLOGIES

ABSTRACT

A Currency Converter Application, a Python-based project, offers a user-friendly interface for real-time and offline currency conversions. Leveraging API integration, it ensures precision with up-to-date exchange rates and supports a wide range of currencies. The application's historical exchange rate tracking aids financial analysis, while customizable settings cater to individual preferences. Designed for accessibility, the Currency Converter Application is open-source, encouraging collaborative development and modular extensions for evolving user needs. This versatile tool is poised to streamline currency transactions in our interconnected global economy.

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy**, for his encouragement.

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal**, for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman**, for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professors Dr. Vadivu. G , Professor, Department of Data Science and Business Systems and Dr. Sasikala. E Professor, Department of Data Science and Business Systems and Course Coordinators** for their constant encouragement and support.

We are highly thankful to our my Course project Faculty **Dr. Ajanthaa Lakkshmanan , Assistant Professor , Department of Computing Technologies**, for her assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **HoD Dr. M Pushpalatha, Department of Computing Technologies** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

TABLE OF CONTENTS

CHAPTER NO	CONTENTS	PAGE NO
1	INTRODUCTION	6
	1.1 Motivation	
	1.2 Objective	
	1.3 Problem Statement	
	1.4 Challenges	
2	LITERATURE SURVEY	8
3	REQUIREMENT	10
	ANALYSIS	
4	ARCHITECTURE &	11
	DESIGN	
5	IMPLEMENTATION	12
6	RESULTS	20
7	CONCLUSION	25
8	REFERENCES	26

1. INTRODUCTION

1.1 Motivation

The motivation behind the Currency Converter Application stems from the recognition of a universal need for a reliable, user-friendly currency converter in our increasingly interconnected world. Global transactions demand real-time and accurate currency conversions, yet existing solutions often lack key features or accessibility. The Currency Converter Application is driven by the desire to simplify this process, offering a versatile tool that not only provides real-time conversions but also supports historical tracking and user customization. The project's open-source nature reflects a commitment to collaborative development, inviting contributions to create a robust and adaptable solution that empowers individuals and businesses navigating the complexities of international finance.

1.2 Objective

The primary objective of the Currency Converter Application is to create a user-friendly and efficient currency converter application using Python. The application aims to provide real-time and accurate currency conversions by integrating with external APIs to fetch the latest exchange rates. It seeks to support a wide range of currencies, catering to diverse user needs in the global financial landscape. Additionally, the Currency Converter Application aims to offer features such as historical exchange rate tracking and customizable settings to enhance user experience and utility. As an open-source project, the objective is to foster collaboration, allowing developers to contribute and extend the application's capabilities to meet evolving user requirements.

1.3 Problem Statement

In a world characterized by international transactions and diverse financial landscapes, individuals and businesses often face the challenge of accessing accurate and user-friendly currency conversion tools. Existing solutions may lack real-time data, comprehensive currency support, or user customization options. The need for a reliable, Python-based currency converter that seamlessly integrates with external APIs to fetch up-to-date exchange rates, supports a wide range of currencies, and offers user-friendly features like historical rate tracking and customization is evident. The Currency Converter Application addresses these challenges, aiming to provide a robust solution to simplify and enhance the currency conversion experience for users across different scenarios and requirements.

1.4 Challenges

Developing a Currency Converter Application poses several challenges. Firstly, ensuring real-time accuracy requires effective integration with external APIs, introducing potential complexities in handling data updates and server responses. Supporting a comprehensive range of currencies demands meticulous maintenance to keep pace with dynamic global economic shifts. The implementation of historical exchange rate tracking presents challenges in data storage and retrieval for seamless user experience. Additionally, creating a truly user-friendly interface demands thoughtful design considerations for accessibility across diverse user backgrounds. The offline functionality introduces complexities in synchronizing data without compromising accuracy. Encouraging collaborative contributions and maintaining an open-source community presents challenges in managing diverse coding styles and ensuring version compatibility. Overcoming these challenges is essential to delivering a reliable and versatile currency converter application using Python.

2. LITERATURE SURVEY

1. Currency Conversion Algorithms:

Exploration of algorithms and methodologies used in existing currency converter applications, focusing on their accuracy and efficiency.

2. API Integration Techniques:

Investigation into best practices for integrating external APIs to fetch real-time exchange rate data securely and reliably.

3. User Interface Design:

Review of literature on user interface design principles, with a focus on creating an intuitive and user-friendly experience for diverse user groups.

4. Historical Data Tracking:

Examination of approaches for storing and retrieving historical exchange rate data, including database management and optimization techniques.

5. Offline Functionality:

Analysis of methods employed in applications to provide offline functionality, ensuring a seamless user experience without internet access.

6. Customization in Applications:

Literature on user customization options, exploring how applications allow users to personalize settings such as default currencies and decimal precision.

7. Open-Source Collaboration:

Studies on successful open-source projects, highlighting best practices for fostering collaboration, managing contributions, and maintaining code quality.

8. Security in Financial Applications:

Exploration of security measures employed in financial applications, especially those dealing with sensitive data such as exchange rates and user preferences.

9. Scalability in Financial Tools:

Examination of scalable architectures and techniques for accommodating diverse currencies and features in a growing financial tool.

By conducting a thorough literature survey in these areas, the Currency Converter Application can benefit from insights and best practices established in related research and applications, ensuring a well-informed and robust development process.

3. REQUIREMENT ANALYSIS

1. Functional Requirements:

- Real-time currency conversion using external APIs.
- Support for a diverse range of major and minor currencies.
- Historical exchange rate tracking functionality.
- User customization options for default currencies and decimal precision.
- Offline mode enabling currency conversion without internet access.

2. Non-Functional Requirements:

- Fast and responsive performance for real-time conversions.
- Intuitive and user-friendly interface design.
- Reliable exchange rate data and consistent offline functionality.
- Scalable architecture to accommodate future currency additions.
- Secure API communication and user data handling.

3. Technical Requirements:

- Implementation in the Python programming language.
- Integration with external APIs for real-time exchange rates.
- Utilization of a database for efficient storage and retrieval of historical exchange rate data.
- Version control system for collaborative development.

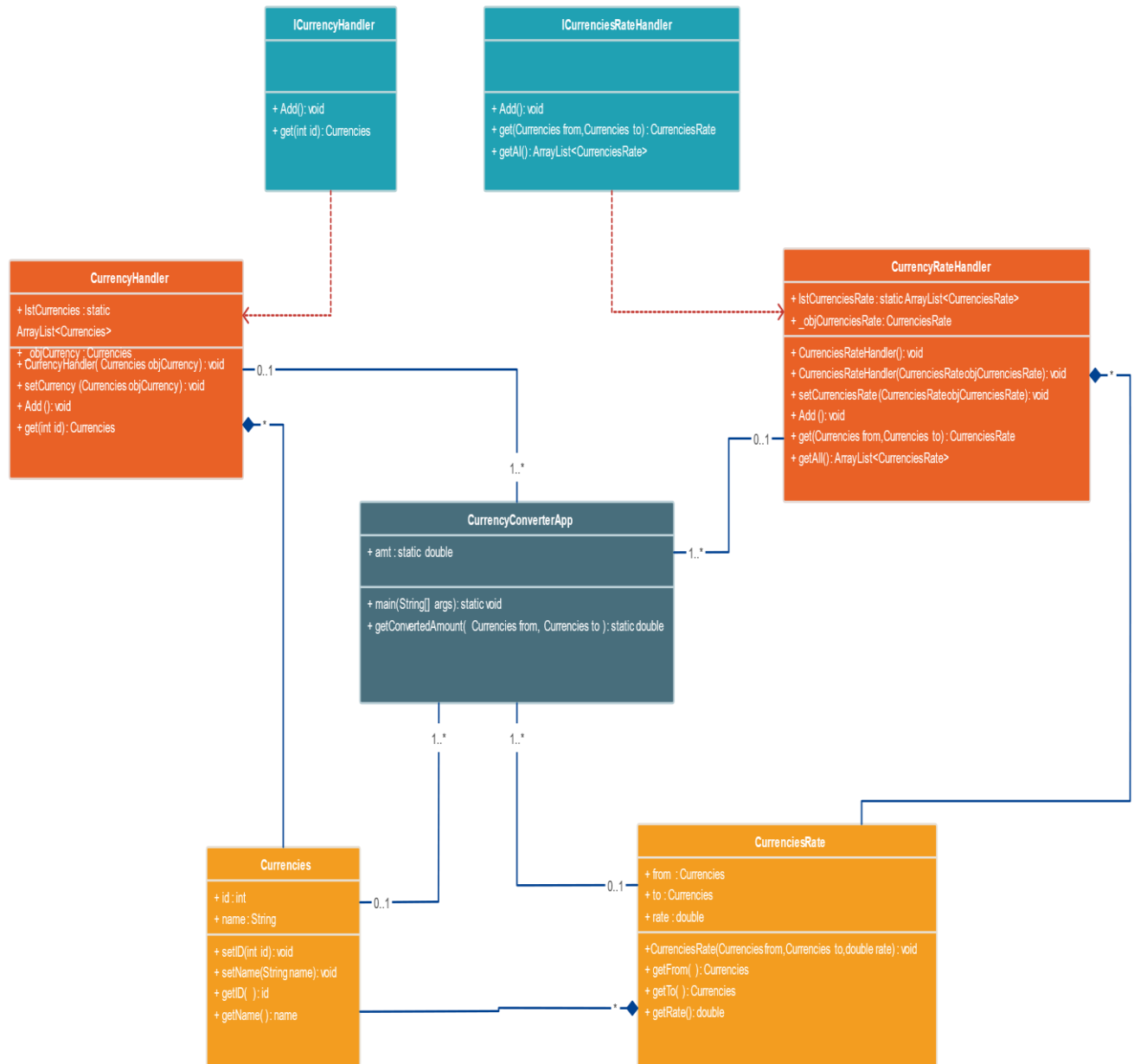
4. Open-Source Collaboration:

- Comprehensive documentation to facilitate open-source contributions.
- Implementation of versioning practices for effective code management.

5. Testing Requirements:

- Rigorous testing for functional correctness, performance, and security.
- Continuous integration and automated testing for code quality assurance.

4. ARCHITECTURE AND DESIGN



5. IMPLEMENTATION

SQL:-

```
Currency converter
Database details
-----
Database name:converter
table name: login
query:
-----
create database converter;
use database converter
create table login
    -> (user int auto_increment key not null,
    -> username varchar(100),
    -> password varchar(100)
    -> );
insert into converter values(1,'Anjali','Anjali@123');
```

Python:-

1.

```
# Import module
from tkinter import *
from tkinter import messagebox
import mysql.connector

# Create object
root = Tk()

# Adjust size
root.geometry("1280x800")
root.title('Login')

# Add image file
bg = PhotoImage(file = "D:\Amritha\currency\img.png")

def login():
    username = username_entry.get()
    password = password_entry.get()
    if username==' ' or password==' ':
        messagebox.showerror(title="Error", message="Invalid login.")
    else:
        try:

mydb=mysql.connector.connect(host='localhost',user='root',password='password',database='converter')
        mycursor =mydb.cursor()
        print("connected")

        except:
            messagebox.showerror("connection","Database connection not established")
            return
        command="use converter"
        mycursor.execute(command)
        command="select * from login where username=%s and password=%s"
        mycursor.execute(command, (username,password))
        myresult= mycursor.fetchone()
        print(myresult)

        if myresult==None:

            messagebox.showinfo("invalid","Invalid username and password")

        else:

            messagebox.showinfo("Login","Successfully Login!!")
            root.destroy()
            import currency
```

```

# Create Canvas
canvas1 = Canvas( root, width = 1000,
                  height = 400)

canvas1.pack(fill = "both", expand = True)

# Display image
canvas1.create_image( 0, 0, image = bg,
                    anchor = "nw")

# Add Text
canvas1.create_text( 996, 220, text = "Login", font=("Times New Roman",
30))
canvas1.create_text( 930, 300, text = "Username", font=("Times New Roman",
20))
canvas1.create_text( 930, 400, text = "Password", font=("Times New Roman",
20))

#text bxes
username_entry = Entry(root, font=("Arial", 16))

password_entry = Entry(root, show="*", font=("Arial", 16))

# Create Buttons
button2 = Button( root, text = "Login" ,width = 10,
                 height = 1, bg="#370466", fg="#FFFFFF",
                 font=("Times New Roman", 16),command=login)

# Display Buttons
username_entry_canvas = canvas1.create_window( 875, 330, anchor = "nw",
                                              window =
username_entry)
password_entry_canvas = canvas1.create_window( 875, 430, anchor = "nw",
                                              window =
password_entry)

button2_canvas = canvas1.create_window( 925, 500,
                                       anchor =
"nw",
                                       window =
button2)

# Execute tkinter
root.mainloop()

```

2.

```
# Import module
import tkinter as tk
from tkinter import Tk, ttk
from tkinter import *
from tkinter import messagebox

from PIL import Image, ImageTk

import requests
import json
import mysql.connector
#colors
cor0 = "#FFFFFF" # white
cor1 = "#333333" # black
cor2 = "#EB5D51" # red
btn = "#250071" #purple
mainc = "#EEF0FD"
subc = "#EDD399"
# Create object
root = Tk()

# Adjust size
root.geometry("1280x800")
root.title('Converter')

# Add image file
bg = PhotoImage(file = "D:\Amritha\currency\img1.png")

def history():
    root.destroy()
    import cnvrt

def convert():
    url = "https://currency-converter18.p.rapidapi.com/api/v1/convert"

    currency_1 = combo1.get()
    currency_2 = combo2.get()
    amount = value.get()

    querystring = {"from":currency_1,"to":currency_2,"amount":amount}
```

```

if currency_2 == 'USD':
    symbol = '$'
elif currency_2 == 'INR':
    symbol = '₹'
elif currency_2 == 'EUR':
    symbol = '€'
elif currency_2 == 'BRL':
    symbol = 'R$'
elif currency_2 == 'CAD':
    symbol = 'CA $'

headers = {
    'x-rapidapi-host': "currency-converter18.p.rapidapi.com",
    'x-rapidapi-key':
"90c59d6c9fmsh4599f814e2ffc92p17fc6djsndaaa0265ac61"
}

response = requests.request("GET", url, headers=headers,
params=querystring)

data = json.loads(response.text)
converted_amount = data["result"]["convertedAmount"]
formatted = symbol + " {:.2f}".format(converted_amount)

result['text'] = formatted

print(converted_amount, formatted)

mydb=mysql.connector.connect(host='localhost',user='root',password='password',database='converter')
mycursor =mydb.cursor()
command="use converter"
mycursor.execute(command)
query = "INSERT INTO `history` (`cfrom` , `cto` , `input` , `output`)
VALUES(%s,%s,%s,%s)"
id = mycursor.execute(query, (currency_1, currency_2, amount,
formatted))

mydb.commit()

# Create Canvas
canvas1 = Canvas( root, width = 1000,
height = 400)

canvas1.pack(fill = "both", expand = True)

# Display image
canvas1.create_image( 0, 0, image = bg,
anchor = "nw")

```



```

#main frame
app_name = Label(root, text = "Currency Converter", font=('Arial 20
bold'),fg=cor1,bg=mainc)
app_name.place(x=500, y=150)

c_name = Label(root, text = "Convert Here !", font=('Arial 20 bold
underline'),bg=subc)
c_name.place(x=300, y=250)

H_name = Label(root, text = "History", font=('Arial 20 bold underline'),
bg=subc)
H_name.place(x=800, y=360)

result = Label(root, text = " ",width=17, height=2, pady=2, relief="flat",
anchor=CENTER, font=('Arial 15 bold'), bg=cor0, fg=cor1)
result.place(x=280, y=500)

currency = ['CAD', 'BRL', 'EUR', 'INR', 'USD']

from_label = Label(root, text = "Convert From", width=11, height=1,
pady=0, padx=0, anchor=NW, font=('Arial 10 bold'), bg=subc, fg=cor1)
from_label.place(x=280, y=320)
combol = ttk.Combobox(root, width=8, justify=CENTER, font=("Ivy 12 bold"))
combol['values'] = (currency)
combol.place(x=280, y=345)

to_label = Label(root, text = "Convert To", width=11, height=1, pady=0,
padx=0, relief="flat", anchor=NW, font=('Arial 10 bold'), bg=subc,
fg=cor1)
to_label.place(x=398, y=320)

combo2 = ttk.Combobox(root, width=8, justify=CENTER, font=("Ivy 12 bold"))
combo2['values'] = (currency)
combo2.place(x=400, y=345)

in_label = Label(root, text = "Enter Amount", width=11, height=1, pady=0,
padx=0, relief="flat", anchor=NW, font=('Ivy 10 bold'), bg=subc, fg=cor1)
in_label.place(x=280, y=380)

value = Entry(root, width=22, justify=CENTER, font=("Ivy 12 bold"),
relief=SOLID)
value.place(x=280, y=410)

button = Button(root, text="Convert", width=18, padx=5, height=1, bg=btn,
fg=cor0,font=("Ivy 12 bold"), command=convert)
button.place(x=280, y=440)

res_label = Label(root, text = "Result", width=11, height=1, pady=0,
padx=0, relief="flat", anchor=NW, font=('Ivy 10 bold'), bg=subc, fg=cor1)
res_label.place(x=350, y=480)

```

```
button = Button(root, text="Click here to view History", width=18, padx=5,  
height=1, bg=btn, fg=cor0,font=("Ivy 12 bold"), command=history)  
button.place(x=750, y=450)  
  
# Execute tkinter  
root.mainloop()
```

3.

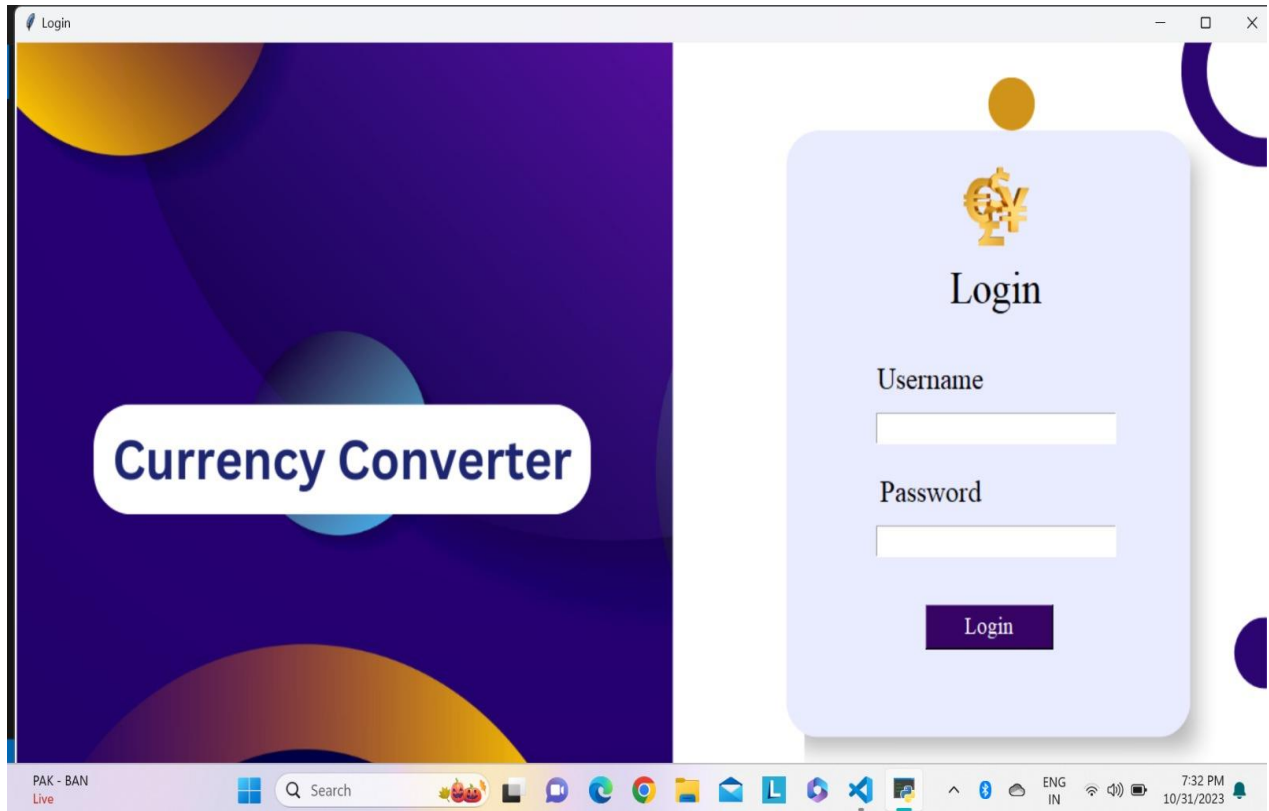
```
import mysql.connector
import tkinter as tk
from tkinter import *
my_w = tk.Tk()
my_w.geometry("400x250")
my_w.title('history')
mydb=mysql.connector.connect(host='localhost',user='root',password='password',database='converter')

my_conn = mydb.cursor()
##### end of connection #####
my_conn.execute("SELECT * FROM history limit 0,10")
e=Label(my_w,width=10,text='From',borderwidth=3,
relief='ridge',anchor='w',bg='beige')
e.grid(row=0,column=0)
e=Label(my_w,width=10,text='To',borderwidth=3,
relief='ridge',anchor='w',bg='beige')
e.grid(row=0,column=1)
e=Label(my_w,width=10,text='Amount',borderwidth=3,
relief='ridge',anchor='w',bg='beige')
e.grid(row=0,column=2)
e=Label(my_w,width=10,text='Result',borderwidth=3,
relief='ridge',anchor='w',bg='beige')
e.grid(row=0,column=3)
i=1
for student in my_conn:
    for j in range(len(student)):
        e=Label(my_w,width=10,
text=student[j],borderwidth=3,relief='ridge', anchor="w")
        e.grid(row=i, column=j)
        i=i+1
my_w.mainloop()

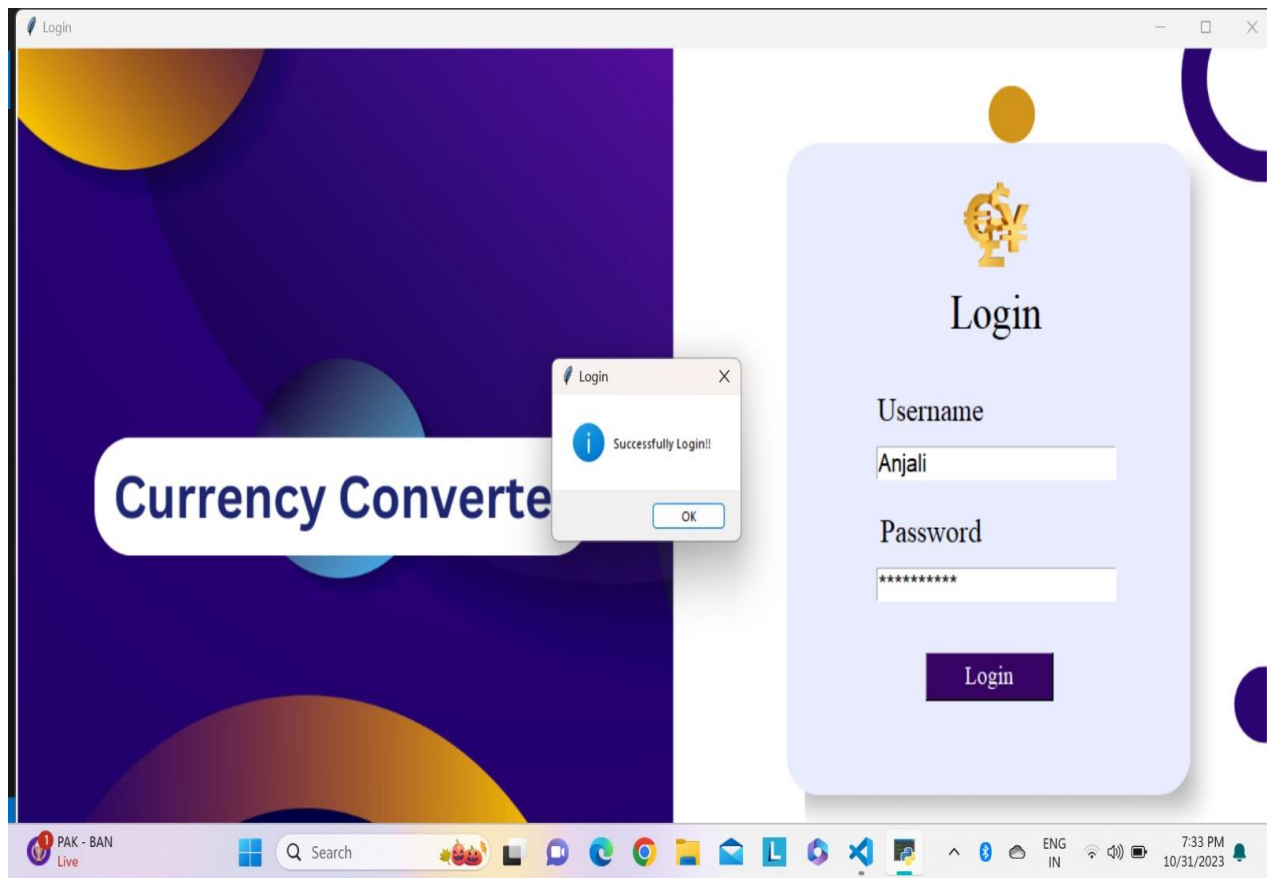
# currencyconverter
```

6. RESULTS

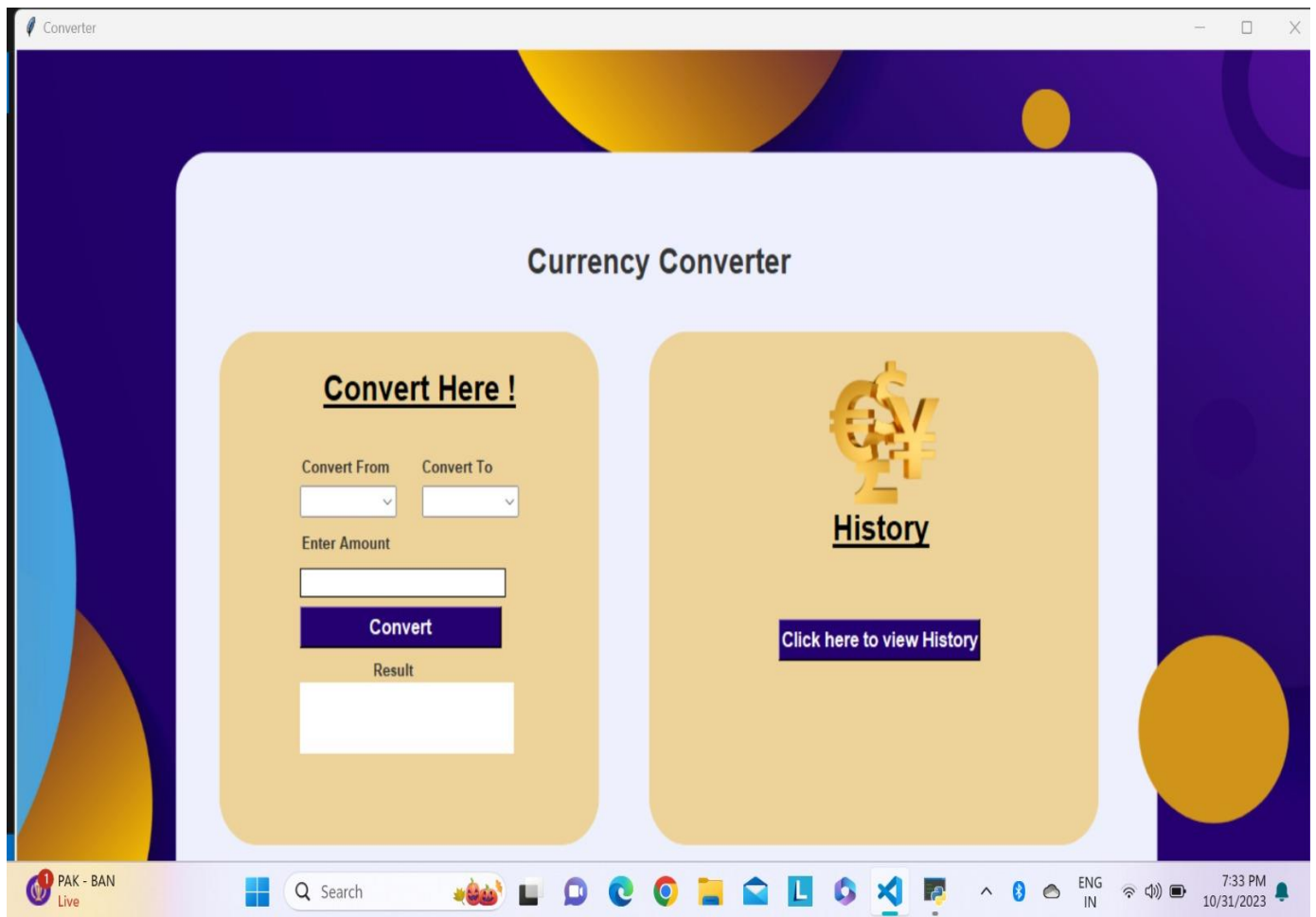
1. Login Page



2. Successful Login Page



3. Currency Converter Page



4. Currency to be converted

Converter

Currency Converter

Convert Here !


Convert From Convert To

Enter Amount

[Convert](#)

Result

R\$ 1,338.66



History

[Click here to view History](#)

Upcoming Earnings

Search

7:33 PM 10/31/2023












5. Currency Converted!

history

From	To	Amount	Result
CAD	USD	342	\$23432
dss	err	22	\$33
dss	err	22	\$33
CAD	EUR	33	€ 22.47
BRL	BRL	22	R\$ 22.00
BRL	INR	22	₹ 368.82
EUR	INR	3465	₹ 305,752.99
CAD	BRL	367	R\$ 1,338.66

1 Upcoming Earnings

Q Search



^

Bluetooth

Cloud

ENG IN

Wi-Fi

Speaker

Battery

7:34 PM

10/31/2023

Notification

24

7. CONCLUSION

In conclusion, the Currency Converter Application represents a concerted effort to address the inherent challenges in currency conversion, drawing inspiration from a thorough literature survey and an understanding of user needs in the global financial landscape. By synthesizing best practices from existing research, the project aims to provide a reliable, efficient, and user-friendly solution for real-time and offline currency conversions. The motivation to create a Currency Converter Application arises from the universal necessity for a versatile tool that not only meets current demands but also evolves with the collaborative input of the developer community. Through its open-source nature and adherence to established principles, the Currency Converter Application aspires to be a valuable resource for individuals and businesses navigating the complexities of international finance.

8. REFERENCES

1. <https://www.geeksforgeeks.org>
2. <https://www.w3schools.in>
3. <https://wiingy.com>
4. <https://www.learningmilestone.com>
5. <https://medium.com>