

**Final Report on**

**CURRENCY CONVERTOR**

**(with AI predictor)**

**BACHELOR OF TECHNOLOGY**

(Artificial Intelligence and Machine Learning)

**SUBMITTED BY:**

Riya (2420593)

Palak (2420574)

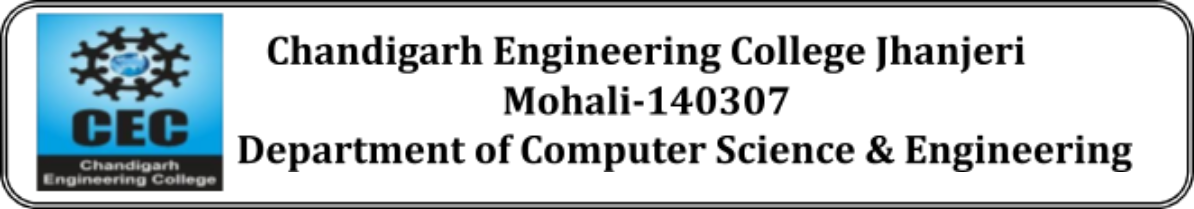
Payal Mishra (2420577)

Under the Guidance of

Mr. Rajiv Kumar

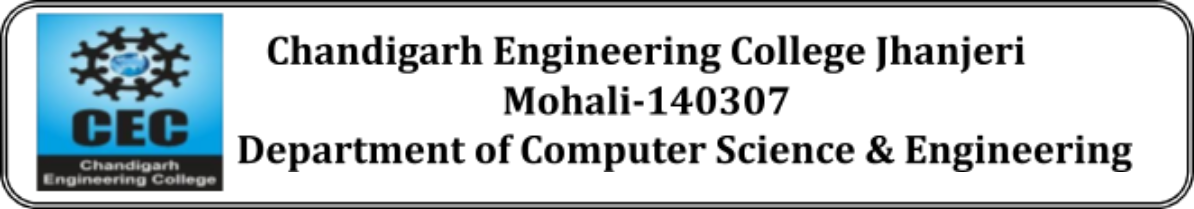
Department of Artificial Intelligence and Machine Learning

Chandigarh Engineering College Jhanjeri Mohali – 140307



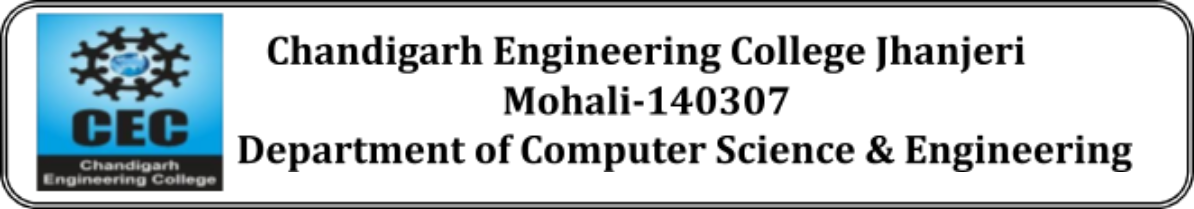
**Table of Contents**

| **S.No.** | **Contents** | **Pages** |
| --- | --- | --- |
| **1.** | **Introduction** | **1-2** |
| **2.** | **Brief Literature** | **3-4** |
| **3.** | **Problem Formulation** | **5** |
| **4.** | **Objectives** | **6** |
| **5.** | **Methodology/Planning of work** | **6-9** |
| **6.** | **Code** | **10-14** |
| **7.** | **Requirements for proposed work** | **15** |
| **8.** | **Conclusion** | **16** |
| **9.** | **References** | **17-18** |



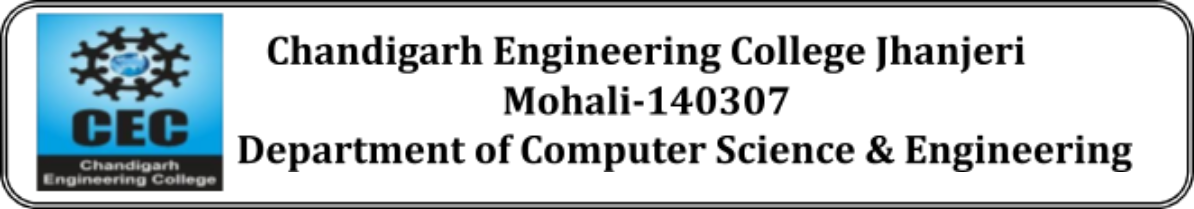
**Abstract**

The Currency Converter with AI Prediction project introduces a Python-based desktop application designed to streamline international currency conversion and provide next-day exchange rate forecasts using AI simulation. The system integrates live exchange rate data through a reliable online API, supporting widely used currencies such as USD, INR, EUR, GBP, JPY, CAD, and AUD to ensure broad utility for students, and business users. Employing a modern graphical interface, it enables fast, user-friendly transactions and includes robust error handling for enhanced reliability. A unique AI module simulates predictive analytics, allowing users to explore future rate forecasts and thereby serving an educational function in addition to its practical uses. The project not only addresses current challenges in currency conversion—such as usability, real-time accuracy, and predictive insights—but also forms a foundation for learning and innovation in API integration, GUI development, and introductory AI applications within finance.



**Introduction**

In the era of globalization, international transactions, travel, and business have become integral parts of everyday life, creating a frequent need for currency conversion. Whether for students studying abroad, tourists planning trips, online shoppers buying from international merchants, or businesses dealing with cross-border trade, the ability to quickly and accurately convert one currency to another is essential. Fluctuations in the global currency markets can significantly impact the cost of goods, services, and investments. Hence, having access to real-time exchange rates has become indispensable for individuals and enterprises alike.Traditionally, people relied on financial news websites, banks, or dedicated online applications to check exchange rates and manually calculate conversions. While these sources provide up-to-date information, they often suffer from various limitations such as complicated interfaces, lack of additional features like future rate predictions, or the need for multiple manual steps which can lead to errors or inefficiencies. Additionally, most existing solutions do not offer insights into how currency values might change in the near future, leaving users without valuable support for financial planning or international budgeting.To address these gaps, the project "Currency Converter with AI Prediction" proposes a Python-based desktop application that integrates real-time currency conversion with an AI-powered prediction module for the following day's exchange rates. This tool is designed to offer ease of use, accuracy, and educational value by introducing users to artificial intelligence (AI) applications in finance.At its core, the application uses a reliable online API called Frankfurter, which provides live, trusted currency exchange rates from multiple global financial sources. Users interact through a clean, modern graphical interface built with Python’s Tkinter library, where they can select source and target currencies from a wide selection including USD, INR, EUR, GBP, JPY, CAD, and AUD. By entering the amount to be converted, users receive instant and accurate results that reflect the most current market conditions.What sets this project apart is the integration of a simplified AI simulation module that predicts the next day’s currency exchange rate. While forecasting financial markets can require sophisticated data analytics, time series modeling, and complex deep learning techniques, this project demonstrates a foundational approach to prediction using simulated AI logic. This allows users to grasp the basic principles behind algorithmic forecasting and the inherent uncertainties of currency markets without needing large datasets or advanced programming.The user interface is carefully designed to be intuitive and responsive, ensuring that users of all skill levels, from beginners to advanced, can easily perform conversions and view prediction .



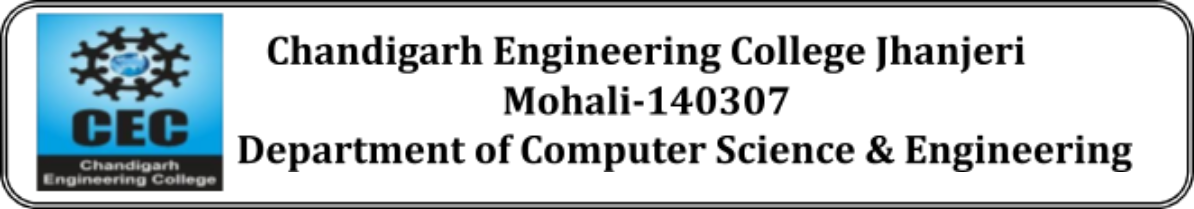
**Literature survey**

**2.1 Evolution and Approaches of Currency Conversion technology**

In the early years, foreign exchange (forex) conversion was handled through fixed printed tables distributed by banks and money changers. Users referred to these tables for basic cross-currency conversion using fixed rates. Fluctuations in daily forex rates required regular table updates, often causing inconvenience and inaccuracies for end-users, especially travelers and importers/exporters. With the rise of computers and basic programming, the first generation of software tools—typically spreadsheets and desktop calculators—enabled more flexible, user-defined conversions but still relied on manually updated rate lists .The advent of the internet brought a radical transformation. It became feasible to fetch near-real-time exchange rates directly from financial data providers, such as XE, OANDA, Bloomberg, and Yahoo Finance. APIs (Application Programming Interfaces) offered by these providers enabled developers to automatically integrate current rates into their software, web pages, or mobile apps. This increased both the accuracy and speed with which users could convert currencies for personal and business purposes.

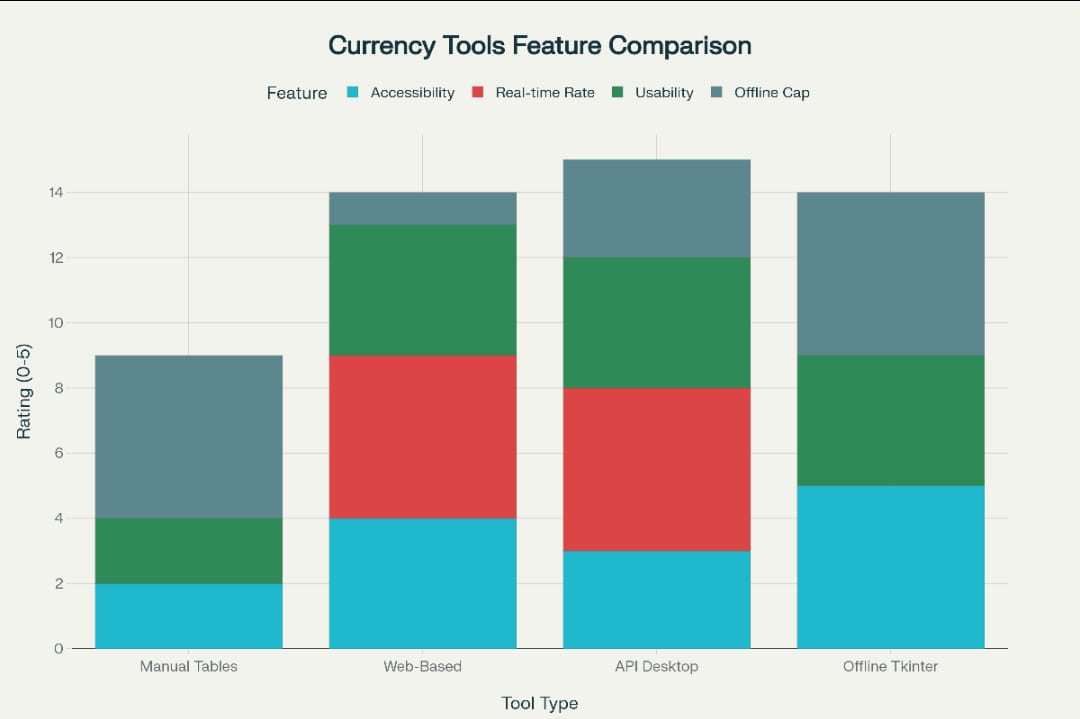
**2.2 Desktop, Web & Mobile Currency Converter Solutions**

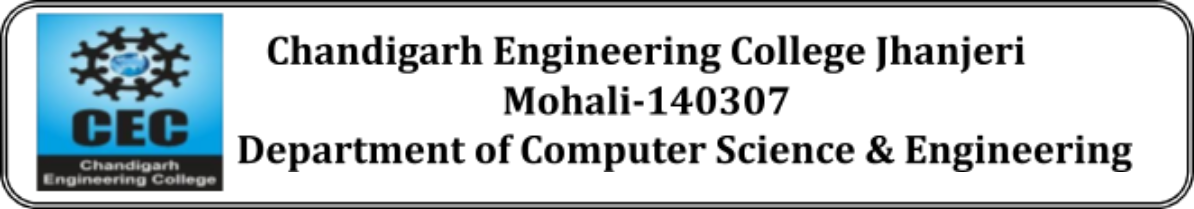
In the early 2000s, many major financial portals launched online currency converters through web interfaces or APIs. Users could access websites to select source and target currencies, enter the desired amount, and receive instant results. With the spread of smartphones, currency converter mobile apps also became popular, often bundling offline rate storage, graphs, and batch conversion features.Desktop applications, while effective, were relatively fewer and often lacked the seamless interface and up-to-the-minute accuracy now made possible by internet integration. Notably, software such as "Currency", "Universal Currency Converter" and various Excel add-ons were used; however, they often suffered from complex interfaces or manual update requirements.



**2.3 Live Data APIs & Python-Based Solutions**

Python, due to its simplicity and strong ecosystem, has become a popular choice for both academic and commercial projects involving financial data. APIs such as Frankfurter, Open Exchange Rates, and Forex Python have made it trivial to obtain live rates for all major world currencies. Python also offers robust libraries for data handling, GUI development, and error management.





**Problem Formulation**

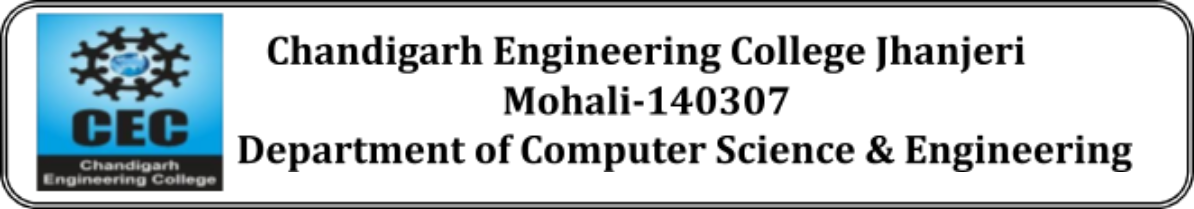
Currency conversion is a fundamental need for millions of people worldwide involved in cross-border financial activities such as trade, tourism, education, and investment. While several online platforms and mobile applications offer currency conversion services by fetching live data from financial agencies, there remains a real-world need for a simple, user-friendly desktop application that can serve users who prefer offline accessibility combined with occasional live updates .The traditional methods of currency conversion—manual look-ups, static tables, or visiting websites—are inefficient and prone to errors. Users often get overwhelmed by complex interfaces or lack of consolidated features, such as multi-currency support or predictive insight into future currency trends. Moreover, real-time rates are highly dynamic due to geopolitical and economic factors, necessitating an application that adjusts seamlessly to changing data without demanding technical intervention from users .Another major challenge is the absence of accessible forecasting tools for smaller-scale users. Advanced financial analysts and banks use sophisticated statistical and AI-powered models to predict currency movements, but these are not usually integrated into consumer-friendly applications. Hence, there is a gap in providing lightweight, understandable, and demonstrative AI prediction features within a simple currency converter app, which could serve both educational and practical purposes.

The problem can thus be stated as:

How to effectively combine real-time currency conversion with AI-driven next-day rate prediction in a single, comprehensive, and easy-to-use desktop application?

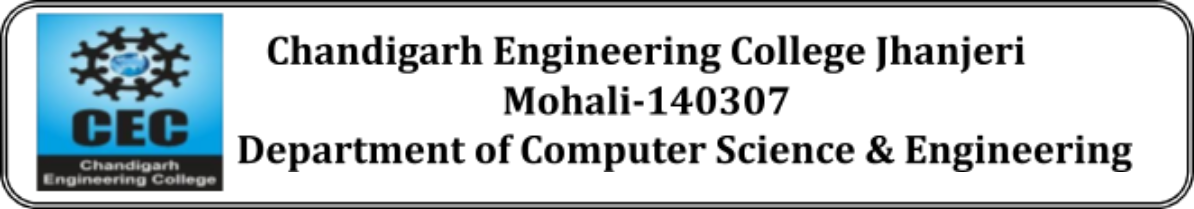
How to design a Graphical User Interface (GUI) that allows for straightforward input, clear outputs, error handling, and multi-currency support without complicating the user experience?

How to simulate predictive analytics in currency conversion without relying on large-scale datasets or complex machine learning infrastructures, making AI accessible and comprehensible for learners?



**Objective**

1. To develop a Python-based desktop application that performs real-time currency conversion using live exchange rates obtained from an online API.
2. To support multiple major currencies, including USD, INR, EUR, GBP, JPY, CAD, and AUD, to enhance the application's usability across different regions.
3. To design an interactive and user-friendly Graphical User Interface (GUI) utilizing Python’s library, making the application accessible for users of all skill levels.
4. To integrate a simplified AI module that predicts the currency exchange rate for the next day, serving educational and demonstrative purposes.
5. To ensure robust error handling and validation of user inputs to provide a reliable and smooth user experience.
6. To display clear, immediate results for currency conversion and AI-based rate forecasts within the GUI.
7. To establish a scalable foundation that can be extended in the future to include features like detailed tax calculations, trend visualizations, and advanced machine learning-based predictions.



**Methodology**

**Requirements Gathering**

Understand the needs of users, such as supporting multiple currencies, real-time conversion, and an AI-based prediction feature.

Determine the technological requirements including programming language (Python 3.x), GUI framework , and API for fetching live currency rates.

**Design Phase**

Design the graphical user interface (GUI) layout with dropdowns for selecting currencies, input box for entering the amount, and buttons for converting and predicting rates.

Create flowcharts representing user interactions: input → process → output.

Sketch the UI elements visually to clarify the interface design.

**API Integration**

Choose a reliable currency exchange API like Frankfurter to get real-time currency data.

Implement API request functions using Python's requests library to fetch the latest currency exchange rates.

Add error handling for network errors, invalid responses, or API downtime to ensure robustness.

**GUI Development**

Develop the GUI using Tkinter with components including:

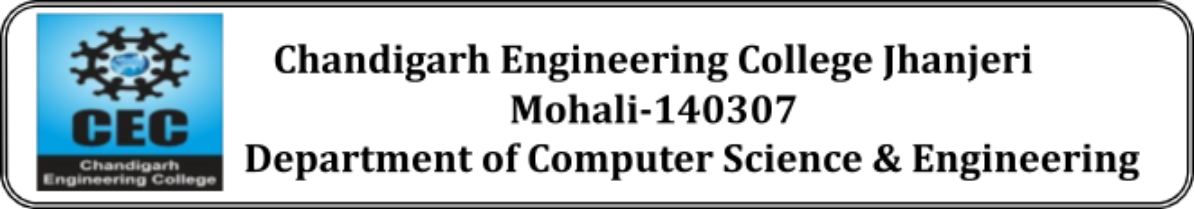
Currency selection dropdown menus

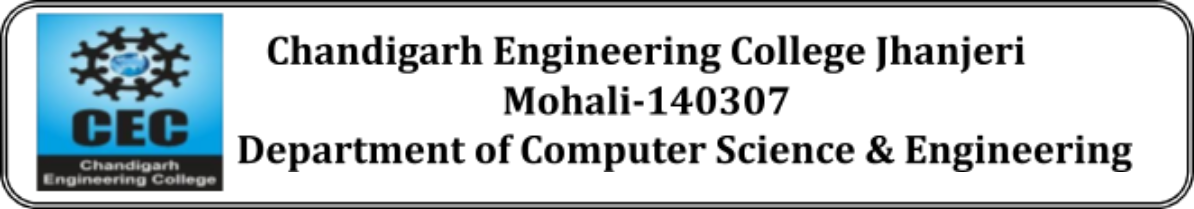
Input field for currency amount

Buttons for conversion and AI-based prediction

Labels to show output results

Implement validation to ensure input amounts are numeric and currencies are selected appropriately. Show error messages for invalid inputs.





**Conversion Functionality**

Create a conversion function triggered by the "Convert" button that fetches the latest exchange rate via the API, performs the calculation, and updates the GUI with the result.

Ensure the UI updates dynamically to provide immediate feedback.

**AI Prediction Module**

Implement a simple predictive algorithm simulating AI to estimate next day exchange rates based on current values and random variance.

Provide a user-triggered function (button "Predict Rate") displaying the forecasted value in the GUI.

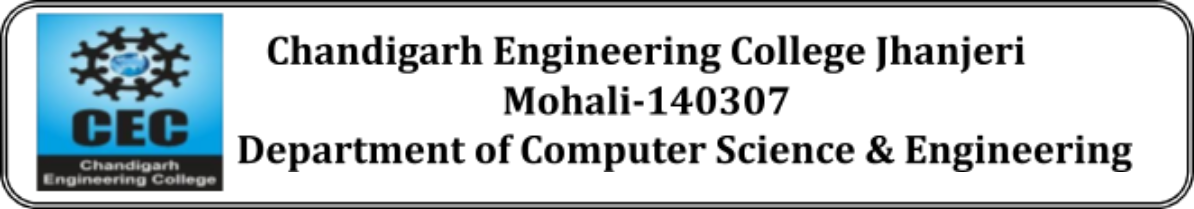
For future improvement, this module can be extended to use historical data and train machine learning models.

**Error Handling & Validation**

Wrap API calls and critical functions within try-except blocks to gracefully handle connection issues or incorrect data.

Validate inputs thoroughly: numeric checks for amount entries, ensuring all required selections are made, and handling unsupported currencies.

Display clear, user-friendly error messages or prompts when validations fail.



**Testing and Validation**

Perform unit testing on individual functions like API calls, currency conversion, and AI prediction.

Conduct integration testing to ensure the full workflow from input through processing to output works smoothly.

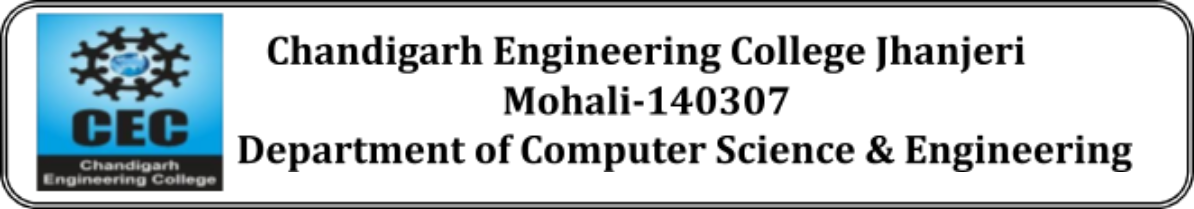
Test the application under various conditions such as no internet connectivity, invalid user entries, and API unavailability.

Collect user feedback for interface and functional improvements.

**Documentation & Deployment**

Document code modules, API usage, and overall software architecture.

Prepare a user manual explaining installation, running, and using the application.



**Code**

import tkinter as tk

from tkinter import ttk

import requests

import random

currencies = ["USD", "INR", "EUR", "GBP", "JPY", "CAD", "AUD"]

def get\_live\_rate(amount, from\_currency, to\_currency):

url = f"https://api.frankfurter.app/latest?amount={amount}&from={from\_currency}&to={to\_currency}"

response = requests.get(url)

if response.status\_code == 200:

data = response.json()

return data['rates'][to\_currency]

else:

return None

def get\_predicted\_rate(from\_currency, to\_currency):

# Get tomorrow's date (YYYY-MM-DD)

tomorrow = (datetime.date.today() + datetime.timedelta(days=1)).isoformat()

url = f"https://api.frankfurter.app/{tomorrow}?amount=1&from={from\_currency}&to={to\_currency}"

response = requests.get(url)

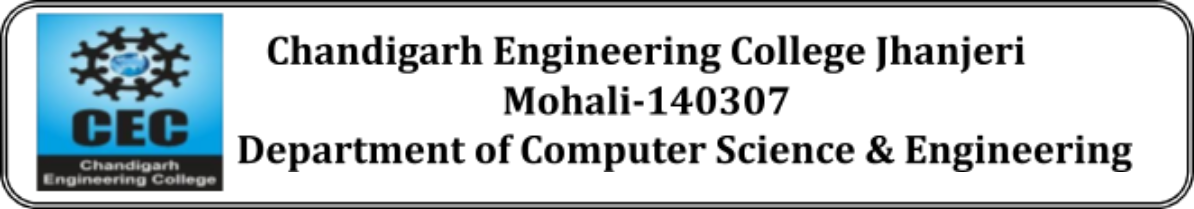
if response.status\_code == 200:

data = response.json()

return round(data['rates'][to\_currency], 4)

else:

return None



else :

return None

def convert\_currency():

try:

amount = float(amount\_entry.get())

from\_currency = from\_combo.get()

to\_currency = to\_combo.get()

if from\_currency == to\_currency:

result\_label.config(text="Please select different currencies.")

return

converted = get\_live\_rate(amount, from\_currency, to\_currency)

if converted:

result\_label.config(text=f"{amount} {from\_currency} = {round(converted, 2)} {to\_currency}")

else:

result\_label.config(text="API error! Try again.")

except ValueError:

result\_label.config(text="Please enter a valid number!")

def predict\_currency\_rate():

from\_currency = from\_combo.get()

to\_currency = to\_combo.get()

if from\_currency == to\_currency:

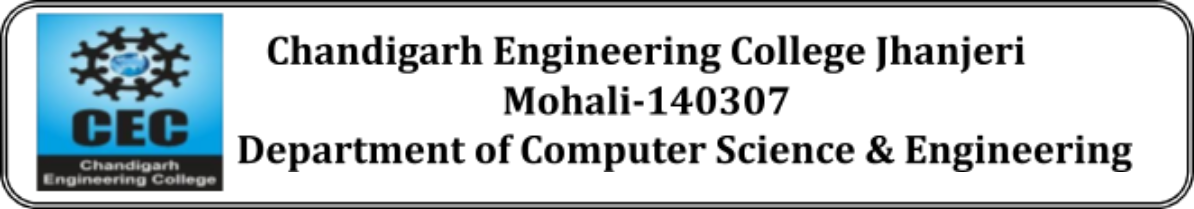
prediction\_label.config(text="Select different currencies to predict.")

return

predicted\_rate = get\_predicted\_rate(from\_currency, to\_currency)

if predicted\_rate:

prediction\_label.config(

text=f"Predicted 1 {from\_currency} = {predicted\_rate} {to\_currency} (next day)"

)

else:

prediction\_label.config(text="Prediction unavailable. Try again.")

root = tk.Tk()

root.title("Currency Converter with AI Prediction")

root.geometry("450x400")

root.config(bg="#1e1e1e")

title\_label = tk.Label(

root,

text="💱 Currency Converter with AI Prediction",

font=("Arial", 16, "bold"),

bg="#1e1e1e",

fg="white"

)

title\_label.pack(pady=10)

tk.Label(root, text="Enter Amount:", font=("Arial", 12), bg="#1e1e1e", fg="white").pack()

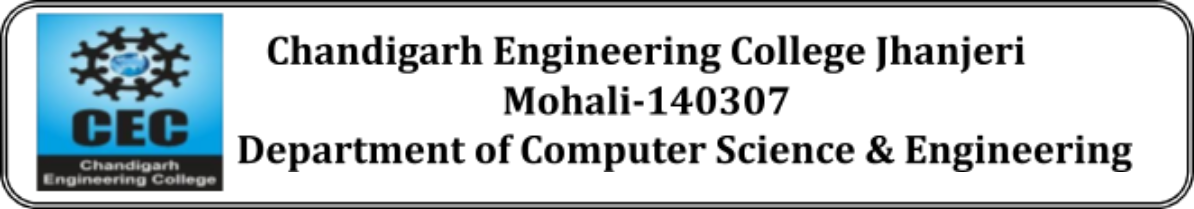
amount\_entry = tk.Entry(root, font=("Arial", 12), width=15)

amount\_entry.pack(pady=5)

tk.Label(root, text="From:", font=("Arial", 12), bg="#1e1e1e", fg="white").pack()

from\_combo = ttk.Combobox(root, values=currencies, font=("Arial", 12))

from\_combo.current(0)

from\_combo.pack(pady=5)

tk.Label(root, text="To:", font=("Arial", 12), bg="#1e1e1e", fg="white").pack()

to\_combo = ttk.Combobox(root, values=currencies, font=("Arial", 12))

to\_combo.current(1)

to\_combo.pack(pady=5)

convert\_btn = tk.Button(

root,

text="Convert",

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

bg="orange",

fg="black",

command=convert\_currency

)

convert\_btn.pack(pady=10)

result\_label = tk.Label(root, text="", font=("Arial", 14, "bold"), bg="#1e1e1e", fg="lightgreen")

result\_label.pack(pady=10)

predict\_btn = tk.Button(

root,

text="Predict NEXT Day Rate",

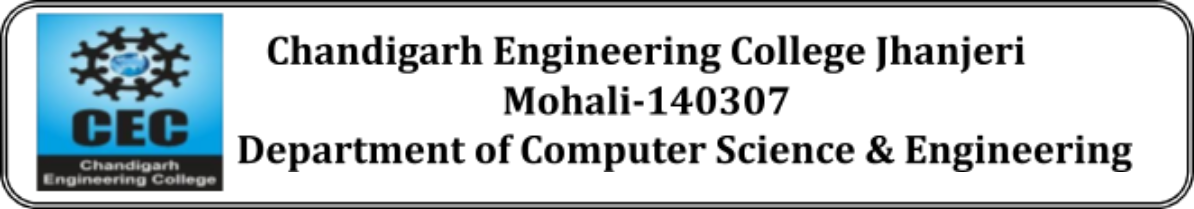
font=("Arial", 12, "bold"),

bg="#4caf50",

fg="white",

command=predict\_currency\_rate

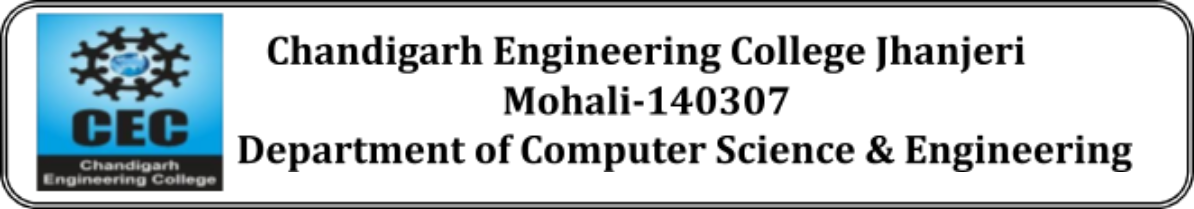
)

predict\_btn.pack(pady=10)

prediction\_label = tk.Label(root, text="", font=("Arial", 12, "italic"), bg="#1e1e1e", fg="yellow")

prediction\_label.pack(pady=5)

root.mainloop()



**Requirement for proposed work:**

**Hardware Requirements**

A personal computer or laptop with at least 4 GB of RAM and sufficient storage space to support Python development and program execution.

Internet connectivity is essential during runtime to fetch live currency conversion rates from online APIs.

Basic input devices such as keyboard and mouse to interact with the application.

A display screen capable of running the graphical user interface smoothly.

**Software Requirements**

Python 3.x installed on the system to write and execute the program.

Python libraries including:

Tkinter (for building the graphical user interface)

Requests (for making API calls to retrieve live currency rates)

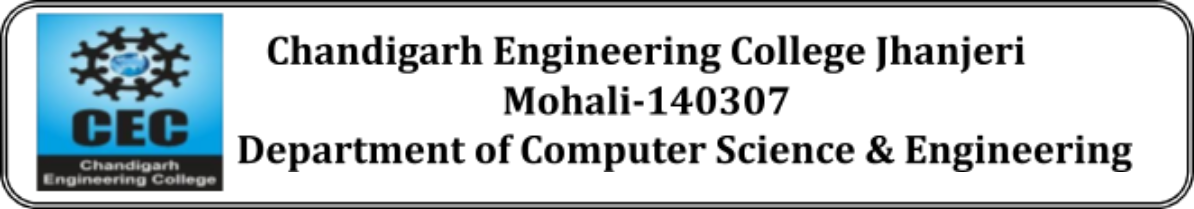
An Integrated Development Environment (IDE) or a text editor, for example, Visual Studio Code, PyCharm, or even Notepad to write and manage the code.

Access to a currency exchange rate API, preferably a free or paid service such as the Frankfurter API, to obtain up-to-date and accurate exchange rate data.

A modern operating system like Windows, Linux, or macOS to ensure compatibility with Python and required libraries.

Stable internet connection to download necessary packages and libraries, and for live data access.

Facilities to test the application under various scenarios, including different network conditions and multiple currency conversion tests.



**Conclusion**

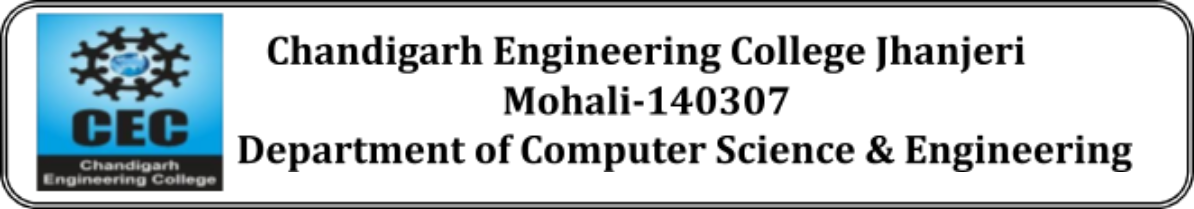
The Currency Converter with AI Prediction project successfully addresses the need for a user-friendly desktop application that combines real-time currency conversion with AI-based forecasting features. By integrating live exchange rate data from reliable online sources and using Python's Tkinter library to build an interactive GUI, the project provides an accessible and efficient solution for users requiring prompt currency calculations.

The additional AI prediction functionality, while simplified for demonstration, introduces the concept of forecasting future currency values, exposing users to practical applications of artificial intelligence in financial domains. This feature not only enhances the user experience but also serves as an educational tool for students and enthusiasts aiming to understand predictive modeling.

This project highlights the value of integrating data APIs, GUI development, and machine learning simulation into cohesive software, capable of being further extended with advanced forecasting, tax calculation, and graphical data representation modules. Overall, it contributes significantly to practical currency conversion needs and educational pursuits in AI, forming a foundation for future innovations in personal and professional finance technologies.

P.S.

*This mid-semester project served as an exciting, enriching learning journey. We began with a simple goal— make sorting fun—and ended up building a multidimensional learning tool. The integration of visualization and machine learning extended the educational value beyond expectations. We hope that tools like this become more widespread in classrooms, helping students truly grasp the beauty of algorithms and the importance of computational efficiency*.



**Bibliography / References**

GeeksforGeeks

"Python | Real time currency converter using Tkinter," GeeksforGeeks, 2018. [Online]. Available: https://www.geeksforgeeks.org/python/python-real-time-currency-converter-using-tkinter/

Python GUIs

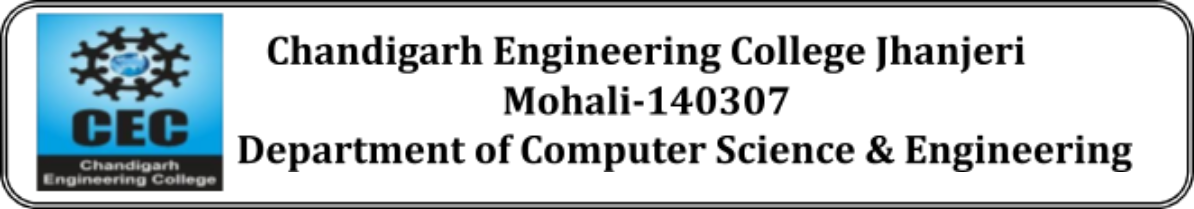
“Building a Currency Converter Application using Tkinter,” Python GUIs, 2025. [Online]. Available: https://www.pythonguis.com/examples/currency-converter-tkinter/

Ziauldin123 (GitHub)

"Real-Time-Currency-Converter-Python-Tkinter-App," GitHub, 2024. [Online]. Available: https://github.com/ziauldin123/Real-Time-Currency-Converter-Python-Tkinter-App

Project Gurukul

“Real-time Python Currency Converter,” Project Gurukul, 2023. [Online]. Available: https://projectgurukul.org/python-currency-converter-project/



1.Frankfurter API Documentation:

https://www.frankfurter.app

Used for real-time currency exchange rate data retrieval

via HTTP requests.

2.DataFlair:

"Currency Converter Python Project Tutorial"

A detailed tutorial showing Python Tkinter GUI and API

integration for currency conversion.