**टप्पा १: पूर्वतयारी (Prerequisites)**

सर्वात आधी, आपल्याला Flask आणि इतर आवश्यक लायब्ररी इन्स्टॉल कराव्या लागतील. तुमच्या कंम्प्युटरच्या कमांड प्रॉम्प्ट (Terminal) मध्ये खालील कमांड चालवा:

Bash

pip install Flask pandas matplotlib seaborn

**टप्पा २: फोल्डरची रचना (Folder Structure)**

एका नवीन फोल्डरमध्ये खालीलप्रमाणे फाईल्स आणि फोल्डर्स तयार करा. **ही रचना अत्यंत महत्त्वाची आहे.**

OptionChainApp/

├── app.py <-- आपली मुख्य Python फाईल

└── templates/

└── index.html <-- आपली HTML फाईल

* app.py नावाची एक Python फाईल तयार करा.
* templates नावाचा एक फोल्डर तयार करा आणि त्याच्या आत index.html नावाची एक HTML फाईल बनवा.

**टप्पा ३: Python कोड (app.py)**

तुमच्या app.py फाईलमध्ये खालील संपूर्ण कोड कॉपी-पेस्ट करा. या कोडमध्ये डेटा वाचणे, चार्ट तयार करणे आणि वेब पेजला पाठवणे, या सर्व गोष्टी हाताळल्या आहेत.

Python

import pandas as pd

import numpy as np

import matplotlib

matplotlib.use('Agg') # हे सर्व्हरवर चालवण्यासाठी आवश्यक आहे

import matplotlib.pyplot as plt

import seaborn as sns

import io

import base64

from flask import Flask, render\_template, request

# Flask ॲप सुरू करणे

app = Flask(\_\_name\_\_)

# मुख्य फंक्शन जे चार्ट तयार करेल

def create\_plots(df):

try:

# कॉल आणि पुट डेटा वेगळा करणे

calls\_df = df[['STRIKE'] + [col for col in df.columns if 'CALLS' in col]]

puts\_df = df[['STRIKE'] + [col for col in df.columns if 'PUTS' in col]]

# चार्टसाठी एक डिक्शनरी तयार करणे

charts = {}

sns.set\_style("darkgrid")

# --- चार्ट १: LTP vs. Strike Price ---

plt.figure(figsize=(12, 6))

plt.plot(calls\_df['STRIKE'], calls\_df['CALLS\_LTP'], marker='o', linestyle='-', color='orangered', label='Call LTP')

plt.plot(puts\_df['STRIKE'], puts\_df['PUTS\_LTP'], marker='o', linestyle='-', color='limegreen', label='Put LTP')

plt.title('LTP vs. Strike Price', fontsize=16)

plt.xlabel('Strike Price', fontsize=12)

plt.ylabel('Last Traded Price (LTP)', fontsize=12)

plt.legend()

# चार्टला इमेजमध्ये रूपांतरित करणे

img = io.BytesIO()

plt.savefig(img, format='png', bbox\_inches='tight')

img.seek(0)

charts['ltp\_chart'] = base64.b64encode(img.getvalue()).decode('utf8')

plt.close()

# --- चार्ट २: Open Interest vs. Strike Price ---

plt.figure(figsize=(12, 6))

plt.bar(calls\_df['STRIKE'], calls\_df['CALLS\_OI'], width=20, color='orangered', label='Call OI (Resistance)')

plt.bar(puts\_df['STRIKE'], puts\_df['PUTS\_OI'], width=20, color='limegreen', label='Put OI (Support)', alpha=0.8)

plt.title('Open Interest vs. Strike Price', fontsize=16)

plt.xlabel('Strike Price', fontsize=12)

plt.ylabel('Open Interest (OI)', fontsize=12)

plt.legend()

img = io.BytesIO()

plt.savefig(img, format='png', bbox\_inches='tight')

img.seek(0)

charts['oi\_chart'] = base64.b64encode(img.getvalue()).decode('utf8')

plt.close()

# --- चार्ट ३: Implied Volatility (IV) vs. Strike Price ---

valid\_calls\_iv = calls\_df[calls\_df['CALLS\_IV'] > 0]

valid\_puts\_iv = puts\_df[puts\_df['PUTS\_IV'] > 0]

plt.figure(figsize=(12, 6))

plt.plot(valid\_calls\_iv['STRIKE'], valid\_calls\_iv['CALLS\_IV'], marker='.', linestyle='-', color='orangered', label='Call IV')

plt.plot(valid\_puts\_iv['STRIKE'], valid\_puts\_iv['PUTS\_IV'], marker='.', linestyle='-', color='limegreen', label='Put IV')

plt.title('Implied Volatility (IV) vs. Strike Price', fontsize=16)

plt.xlabel('Strike Price', fontsize=12)

plt.ylabel('Implied Volatility (%)', fontsize=12)

plt.legend()

img = io.BytesIO()

plt.savefig(img, format='png', bbox\_inches='tight')

img.seek(0)

charts['iv\_chart'] = base64.b64encode(img.getvalue()).decode('utf8')

plt.close()

return charts

except Exception as e:

print(f"Error creating plots: {e}")

return None

# वेबपेजसाठी रूट (Route)

@app.route('/', methods=['GET', 'POST'])

def index():

if request.method == 'POST':

# फाईल अपलोड झाली आहे का ते तपासणे

if 'file' not in request.files:

return "No file part"

file = request.files['file']

if file.filename == '':

return "No selected file"

if file and file.filename.endswith('.csv'):

try:

# CSV फाईल वाचणे आणि साफ करणे

df = pd.read\_csv(file, skiprows=1)

df = df.iloc[:, 1:-1]

new\_columns = [

'CALLS\_OI', 'CALLS\_CHNG\_IN\_OI', 'CALLS\_VOLUME', 'CALLS\_IV', 'CALLS\_LTP', 'CALLS\_CHNG',

'CALLS\_BID\_QTY', 'CALLS\_BID\_PRICE', 'CALLS\_ASK\_PRICE', 'CALLS\_ASK\_QTY', 'STRIKE',

'PUTS\_BID\_QTY', 'PUTS\_BID\_PRICE', 'PUTS\_ASK\_PRICE', 'PUTS\_ASK\_QTY', 'PUTS\_CHNG',

'PUTS\_LTP', 'PUTS\_IV', 'PUTS\_VOLUME', 'PUTS\_CHNG\_IN\_OI', 'PUTS\_OI'

]

df.columns = new\_columns

df.replace(['-', ' -'], np.nan, inplace=True)

for col in df.columns:

if df[col].dtype == 'object':

df[col] = pd.to\_numeric(df[col].str.replace(',', '', regex=False), errors='coerce')

df.dropna(subset=['STRIKE'], inplace=True)

df.fillna(0, inplace=True)

# चार्ट्स तयार करणे

charts = create\_plots(df)

return render\_template('index.html', charts=charts)

except Exception as e:

return f"An error occurred: {e}"

# GET request आल्यास फक्त अपलोड पेज दाखवणे

return render\_template('index.html', charts=None)

# ॲप चालवणे

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

app.run(debug=True)

**टप्पा ४: HTML कोड (index.html)**

तुमच्या templates/index.html फाईलमध्ये खालील कोड कॉपी-पेस्ट करा. हे एक सोपे वेबपेज आहे जे फाईल अपलोडचा फॉर्म आणि चार्ट्स दाखवेल.

HTML

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Option Chain Chart Generator</title>

<style>

body { font-family: Arial, sans-serif; background-color: #f4f4f9; color: #333; margin: 0; padding: 20px; text-align: center; }

.container { max-width: 900px; margin: auto; background: white; padding: 30px; border-radius: 8px; box-shadow: 0 4px 8px rgba(0,0,0,0.1); }

h1 { color: #444; }

form { margin-bottom: 40px; }

input[type="file"] { border: 1px solid #ddd; padding: 10px; border-radius: 4px; }

button { background-color: #007bff; color: white; padding: 12px 20px; border: none; border-radius: 4px; cursor: pointer; font-size: 16px; }

button:hover { background-color: #0056b3; }

.charts { margin-top: 30px; }

.chart-container { margin-bottom: 40px; }

img { max-width: 100%; height: auto; border: 1px solid #ddd; border-radius: 4px; }

</style>

</head>

<body>

<div class="container">

<h1>📈 Option Chain Analysis Chart Generator</h1>

<p>तुमची ऑप्शन चेन CSV फाईल अपलोड करा आणि चार्ट्स मिळवा.</p>

<form method="post" enctype="multipart/form-data">

<input type="file" name="file" accept=".csv" required>

<button type="submit">Generate Charts</button>

</form>

{% if charts %}

<div class="charts">

<div class="chart-container">

<h2>LTP vs. Strike Price</h2>

<img src="data:image/png;base64,{{ charts.ltp\_chart }}">

</div>

<div class="chart-container">

<h2>Open Interest vs. Strike Price</h2>

<img src="data:image/png;base64,{{ charts.oi\_chart }}">

</div>

<div class="chart-container">

<h2>Implied Volatility (IV) vs. Strike Price</h2>

<img src="data:image/png;base64,{{ charts.iv\_chart }}">

</div>

</div>

{% endif %}

</div>

</body>

</html>

**टप्पा ५: ॲप्लिकेशन चालवणे (Run the Application)**

1. तुमच्या कमांड प्रॉम्प्ट (Terminal) मध्ये OptionChainApp फोल्डरमध्ये जा.
2. खालील कमांड चालवा:

Bash

python app.py

1. तुम्हाला \* Running on http://127.0.0.1:5000 असा मेसेज दिसेल.

**टप्पा ६: वापर कसा करायचा (How to Use)**

1. तुमचा वेब ब्राउझर (उदा. Chrome, Firefox) उघडा.
2. ॲड्रेस बारमध्ये http://127.0.0.1:5000 टाईप करा आणि Enter दाबा.
3. तुम्हाला वेबपेज दिसेल. 'Choose File' वर क्लिक करून तुमची CSV फाईल निवडा.
4. 'Generate Charts' बटन दाबा. थोड्याच वेळात तुम्हाला त्याच पेजवर खाली तिन्ही चार्ट्स दिसतील! 🎉