SOURCE CODE

```
from flask import Flask, render_template, request, session, flash, send_file
import mysql.connector
import os
import pandas as pd
import numpy as np
import math
import datetime as dt
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, mean_absolute_error,
explained_variance_score, r2_score
from sklearn.metrics import mean_poisson_deviance, mean_gamma_deviance,
accuracy_score
from sklearn.preprocessing import MinMaxScaler
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras.layers import LSTM
import matplotlib.pyplot as plt
from itertools import cycle
import plotly.graph_objects as goimport plotly.express as px
import plotly.io as pio
pio.renderers.default = "browser"
```

```
# pio.renderers.default = 'png'
from plotly.subplots import make_subplots
import io
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import plotly.graph_objects as go
import plotly.io as pio
import pytz
import yfinance as yf
import plotly.express as px
# Set the start and end date for the historical data
from datetime import datetime as dt
import pytz
from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer
from requests import get
from bs4 import BeautifulSoup
import os
from flask import Flask, render_template, request, jsonify
english_bot = ChatBot('Bot',
storage_adapter='chatterbot.storage.SQLStorageAdapter',
logic_adapters=[
{'import_path': 'chatterbot.logic.BestMatch'
},
```

```
],
trainer='chatterbot.trainers.ListTrainer')
english_bot.set_trainer(ListTrainer)
app = Flask( name )
app.config['SECRET_KEY'] = 'aaa'
@app.route('/')
def home():
return render_template('index.html')
@app.route('/AdminLogin')
def AdminLogin():
return render_template('AdminLogin.html')
@app.route('/Chat')
def Chat():
return render_template('chat.html')
@app.route('/NewUser')
def NewUser():
return render_template('NewUser.html')future_predictions = []
while True:
next_pred = model.predict([last_features])[0]
# Append prediction
current_date += datetime.timedelta(days=1)
future_predictions.append({
"Date": current_date,
"Predicted_Amount": next_pred
```

```
})
# Stop when target is reached
if next_pred >= target_amount:
break
# Update input with new predicted amount (if single feature)
last_features['Amount'] = next_pred
# Result DataFrame
future_df = pd.DataFrame(future_predictions)
print("\nFuture predictions:")
print(future_df)
# Show target hit
target_row = future_df[future_df['Predicted_Amount'] >= target_amount].iloc[0]
print(
f"\n□ Target of {target_amount} predicted to be reached on:
{target_row['Date'].date()} with amount: {round(target_row['Predicted_Amount'], 2)}")#
Plot prediction path
plt.plot(future_df['Date'], future_df['Predicted_Amount'], label="Predicted")
plt.axhline(y=target_amount, color='red', linestyle='
'
--
, label='Target Amount')
plt.title("Forecast Until Target is Reached")
plt.xlabel("Date")
plt.ylabel("Predicted Amount")
plt.xticks(rotation=45)
```

```
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
return render_template('Target.html')
@app.route('/Query')
def Query():
uname = session['uname']
conn = mysql.connector.connect(user='root', password=''
, host='localhost',
database='2stocknew')
cur = conn.cursor()
cur.execute("SELECT * FROM Querytb where UserName='" + uname + "' ")
data = cur.fetchall()
return render_template('NewQuery.html', data=data)
@app.route('/Recommend')def Recommend():
conn = mysql.connector.connect(user='root', password=''
, host='localhost',
database='2stocknew')
cur = conn.cursor()
cur.execute("SELECT UserName,StockName,date, sum(coun) as count FROM stocktb
group by UserName,StockName,date ")
data = cur.fetchall()
return render_template('Recommend.html', data=data)
```

```
@app.route("/newquery", methods=['GET', 'POST'])
def newquery():
if request.method == 'POST':
name = session['uname']
Query = request.form['Query']
conn = mysql.connector.connect(user='root', password="
, host='localhost',
database='2stocknew')
cursor = conn.cursor()
cursor.execute(
"INSERT INTO Querytb VALUES ('" + name + "','" + Query + "','')")
conn.commit()
conn.close()
flash('New Query Register successfully')
uname = session['uname']
conn = mysql.connector.connect(user='root', password="
, host='localhost',
database='2stocknew')
cur = conn.cursor()cur.execute("SELECT * FROM Querytb where UserName='" +
uname + "' ")
data = cur.fetchall()
return render_template('NewQuery.html', data=data)
def sendmsg(targetno, message):
import requests
requests.post(
```

```
"http://smsserver9.creativepoint.in/api.php?username=fantasy&password=596692&to="
+ targetno + "&from=FSSMSS&message=Dear user your msg is " + message + " Sent
By FSMSG
FSSMSS&PEID=1501563800000030506&templateid=1507162882948811640")
if name == ' main ':
app.run(debug=True, use_reloader=True)
```