Title of Work

Crop-Price-Prediction

Language of Work – Python, HTML, CSS, JavaScript

Name of Applicant cum Author	Nationality	Address
Iram Khan	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Khushi Rathore	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Anshuman Singh	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Aditi Mishra	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Adarsh S. Kamde	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Prasoon Chakravati	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Udit Narayan Bera	Indian	House No. 223, Kachhiyana, Jawaharganj Ward, Near Digambar Tailors, Golbazar Road, Jabalpur, 482002, Madhya Pradesh, India
Pragyan Jain	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Silky Pareyani	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
Sonam Sahu	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India
hirsagar	Indian	Gyan Ganga Institute of Technology & Sciences, P.O. Tilwara Ghat, Near Bargi Hills, Jabalpur, Madhya Pradesh Pin:482003, India

Crop-Price-Prediction

app.py

```
from flask import Flask, render_template
from flask_cors import CORS, cross_origin
import numpy as np
import pandas as pd
from datetime import datetime
import crops
import random
```



```
# import matplotlib.pyplot as plt


app = Flask(__name__)
app.config['CORS_HEADERS'] = 'Content-Type'


cors = CORS(app, resources={r"/ticker": {"origins": "http://localhost:port"}})


commodity_dict = {
    "arhar": "static/Arhar.csv",
    "bajra": "static/Bajra.csv",
    "barley": "static/Barley.csv",
    "copra": "static/Copra.csv",
    "cotton": "static/Cotton.csv",
    "sesamum": "static/Sesamum.csv",
    "gram": "static/Gram.csv",
    "groundnut": "static/Groundnut.csv",
    "jowar": "static/Jowar.csv",
    "maize": "static/Maize.csv",
            ": "static/Masoor.csv",
    ': "static/Moong.csv",
```

```
    "niger": "static/Niger.csv",
    "paddy": "static/Paddy.csv",
    "ragi": "static/Ragi.csv",
    "rape": "static/Rape.csv",
    "jute": "static/Jute.csv",
    "safflower": "static/Safflower.csv",
    "soyabean": "static/Soyabean.csv",
    "sugarcane": "static/Sugarcane.csv",
    "sunflower": "static/Sunflower.csv",
    "urad": "static/Urad.csv",
    "wheat": "static/Wheat.csv",
    "Potato": "static/Potato.csv",
    "Onion": "static/Onion.csv"

}

annual_rainfall = [30.1, 22.8, 39.4, 34.7, 56.2, 155.3, 305.0, 265.8, 190.5, 74.1, 42.2, 13.3]
base = {
    "Paddy": 2300,
    "Arhar": 7550,
    "Bajra": 2625,
    "Barley": 1735,
    "Copra": 10860,
    "Cotton": 6620,
    "Sesamum": 8635,
    "Gram": 5440,
    "Groundnut": 6377,
    "Jowar": 3180,
    "Maize": 2090,
            ": 6425,
            ': 8558,
```

```
    "Niger": 7734,
    "Ragi": 3846,
    "Rape": 5650,
    "Jute": 5050,
    "Safflower": 5800,
    "Soyabean": 4600,
    "Sugarcane": 340,
    "Sunflower": 6760,
    "Urad": 6950,
    "Wheat": 2275,
    "Potato":650,
    "Onion":5000
}
commodity_list = []


class Commodity:

    def __init__(self, csv_name):
        self.name = csv_name
        dataset = pd.read_csv(csv_name)
        self.X = dataset.iloc[:, :-1].values
        self.Y = dataset.iloc[:, 3].values

        #from sklearn.model_selection import train_test_split
        #X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.1, random_state=0)

        # Fitting decision tree regression to dataset
            klearn.tree import DecisionTreeRegressor
        = random.randrange(7,18)
```

```
    self.regressor = DecisionTreeRegressor(max_depth=depth)
    self.regressor.fit(self.X, self.Y)
    #y_pred_tree = self.regressor.predict(X_test)
    # fsa=np.array([float(1),2019,45]).reshape(1,3)
    # fask=regressor_tree.predict(fsa)

def getPredictedValue(self, value):
    if value[1]>=2019:
        fsa = np.array(value).reshape(1, 3)
        #print(" ",self.regressor.predict(fsa)[0])
        return self.regressor.predict(fsa)[0]
    else:
        c=self.X[:,0:2]
        x=[]
        for i in c:
            x.append(i.tolist())
        fsa = [value[0], value[1]]
        ind = 0
        for i in range(0,len(x)):
            if x[i]==fsa:
                ind=i
                break
        #print(index, " ",ind)
        #print(x[ind])
        #print(self.Y[i])
        return self.Y[i]

def getCropName(self):
    a = self.name.split('.')
        a[0]
```


```
@app.route('/')
def index():
    context = {
        "top5": TopFiveWinners(),
        "bottom5": TopFiveLosers(),
        "sixmonths": SixMonthsForecast()
    }
    return render_template('index.html', context=context)
```



```
@app.route('/commodity/<name>')
def crop_profile(name):
    max_crop, min_crop, forecast_crop_values = TwelveMonthsForecast(name)
    prev_crop_values = TwelveMonthPrevious(name)
    forecast_x = [i[0] for i in forecast_crop_values]
    forecast_y = [i[1] for i in forecast_crop_values]
    previous_x = [i[0] for i in prev_crop_values]
    previous_y = [i[1] for i in prev_crop_values]
    current_price = CurrentMonth(name)
    #print(max_crop)
    #print(min_crop)
    #print(forecast_crop_values)
    #print(prev_crop_values)
    #print(str(forecast_x))
    crop_data = crops.crop(name)
    context = {
        "name":name,
        "max_crop": max_crop,
            crop": min_crop,
            ast_values": forecast_crop_values,
```

```
        "forecast_x": str(forecast_x),
        "forecast_y":forecast_y,
        "previous_values": prev_crop_values,
        "previous_x":previous_x,
        "previous_y":previous_y,
        "current_price": current_price,
        "image_url":crop_data[0],
        "prime_loc":crop_data[1],
        "type_c":crop_data[2],
        "export":crop_data[3]
    }
    return render_template('commodity.html', context=context)

@app.route('/ticker/<item>/<number>')
@cross_origin(origin='localhost',headers=['Content- Type','Authorization'])
def ticker(item, number):
    n = int(number)
    i = int(item)
    data = SixMonthsForecast()
    context = str(data[n][i])

    if i == 2 or i == 5:
        context = '₹' + context
    elif i == 3 or i == 6:

        context = context + '%'

    #print('context: ', context)
    return context
```

```
def TopFiveWinners():
    current_month = datetime.now().month
    current_year = datetime.now().year
    current_rainfall = annual_rainfall[current_month - 1]
    prev_month = current_month - 1
    prev_rainfall = annual_rainfall[prev_month - 1]
    current_month_prediction = []
    prev_month_prediction = []
    change = []

    for i in commodity_list:
        current_predict = i.getPredictedValue([float(current_month), current_year, current_rainfall])
        current_month_prediction.append(current_predict)
        prev_predict = i.getPredictedValue([float(prev_month), current_year, prev_rainfall])
        prev_month_prediction.append(prev_predict)
        change.append((((current_predict - prev_predict) * 100 / prev_predict),
commodity_list.index(i)))
    sorted_change = change
    sorted_change.sort(reverse=True)
    # print(sorted_change)
    to_send = []
    for j in range(0, 5):
        perc, i = sorted_change[j]
        name = commodity_list[i].getCropName().split('/')[1]
        to_send.append([name, round((current_month_prediction[i] * base[name]) / 100, 2),
round(perc, 2)])
    #print(to_send)
    return to_send
```



```
eLosers():
month = datetime.now().month
```

```
    current_year = datetime.now().year
    current_rainfall = annual_rainfall[current_month - 1]
    prev_month = current_month - 1
    prev_rainfall = annual_rainfall[prev_month - 1]
    current_month_prediction = []
    prev_month_prediction = []
    change = []

    for i in commodity_list:
        current_predict = i.getPredictedValue([float(current_month), current_year, current_rainfall])
        current_month_prediction.append(current_predict)
        prev_predict = i.getPredictedValue([float(prev_month), current_year, prev_rainfall])
        prev_month_prediction.append(prev_predict)
        change.append((((current_predict - prev_predict) * 100 / prev_predict),
commodity_list.index(i)))
    sorted_change = change
    sorted_change.sort()
    to_send = []
    for j in range(0, 5):
        perc, i = sorted_change[j]
        name = commodity_list[i].getCropName().split('/')[1]
        to_send.append([name, round((current_month_prediction[i] * base[name]) / 100, 2),
round(perc, 2)])
    # print(to_send)
    return to_send


def SixMonthsForecast():
    month1=[]
    :[]
    :[]
```

```
month4=[]
month5=[]
month6=[]
for i in commodity_list:
    crop=SixMonthsForecastHelper(i.getCropName())
    k=0
    for j in crop:
        time = j[0]
        price = j[1]
        change = j[2]
        if k==0:
            month1.append((price,change,i.getCropName().split("/")[1],time))
        elif k==1:
            month2.append((price,change,i.getCropName().split("/")[1],time))
        elif k==2:
            month3.append((price,change,i.getCropName().split("/")[1],time))
        elif k==3:
            month4.append((price,change,i.getCropName().split("/")[1],time))
        elif k==4:
            month5.append((price,change,i.getCropName().split("/")[1],time))
        elif k==5:
            month6.append((price,change,i.getCropName().split("/")[1],time))
        k+=1
month1.sort()
month2.sort()
month3.sort()
month4.sort()
month5.sort()
month6.sort()
    nth_wise=[]
```


```
    crop_month_wise.append([month1[0][3],month1[len(month1)-1][2],month1[len(month1)-1][0],month1[len(month1)-1][1],month1[0][2],month1[0][0],month1[0][1]])

    crop_month_wise.append([month2[0][3],month2[len(month2)-1][2],month2[len(month2)-1][0],month2[len(month2)-1][1],month2[0][2],month2[0][0],month2[0][1]])

    crop_month_wise.append([month3[0][3],month3[len(month3)-1][2],month3[len(month3)-1][0],month3[len(month3)-1][1],month3[0][2],month3[0][0],month3[0][1]])

    crop_month_wise.append([month4[0][3],month4[len(month4)-1][2],month4[len(month4)-1][0],month4[len(month4)-1][1],month4[0][2],month4[0][0],month4[0][1]])

    crop_month_wise.append([month5[0][3],month5[len(month5)-1][2],month5[len(month5)-1][0],month5[len(month5)-1][1],month5[0][2],month5[0][0],month5[0][1]])

    crop_month_wise.append([month6[0][3],month6[len(month6)-1][2],month6[len(month6)-1][0],month6[len(month6)-1][1],month6[0][2],month6[0][0],month6[0][1]])

    # print(crop_month_wise)
    return crop_month_wise
```



```
def SixMonthsForecastHelper(name):
    current_month = datetime.now().month
    current_year = datetime.now().year
    current_rainfall = annual_rainfall[current_month - 1]
    name = name.split("/")[1]
    name = name.lower()
    commodity = commodity_list[0]
    for i in commodity_list:
        if name == str(i):
            commodity = i
            break
    month_with_year = []
    for i in range(1, 7):
        if current_month + i <= 12:
            month_with_year.append((current_month + i, current_year, annual_rainfall[current_month + i - 1]))
```

```
        month_with_year.append((current_month + i - 12, current_year + 1,
annual_rainfall[current_month + i - 13]))

    wpis = []

    current_wpi = commodity.getPredictedValue([float(current_month), current_year,
current_rainfall])

    change = []


    for m, y, r in month_with_year:

        current_predict = commodity.getPredictedValue([float(m), y, r])

        wpis.append(current_predict)

        change.append(((current_predict - current_wpi) * 100) / current_wpi)

    crop_price = []
    for i in range(0, len(wpis)):

        m, y, r = month_with_year[i]

        x = datetime(y, m, 1)

        x = x.strftime("%b %y")

        crop_price.append([x, round((wpis[i]* base[name.capitalize()]) / 100, 2) , round(change[i], 2)])


    # print("Crop_Price: ", crop_price)

    return crop_price


def CurrentMonth(name):

    current_month = datetime.now().month

    current_year = datetime.now().year

    current_rainfall = annual_rainfall[current_month - 1]

    name = name.lower()

    commodity = commodity_list[0]

    for i in commodity_list:

        if name == str(i):

            modity = i

    ık
```

```
    current_wpi = commodity.getPredictedValue([float(current_month), current_year,
current_rainfall])
    current_price = (base[name.capitalize()]*current_wpi)/100
    return current_price


def TwelveMonthsForecast(name):
    current_month = datetime.now().month
    current_year = datetime.now().year
    current_rainfall = annual_rainfall[current_month - 1]
    name = name.lower()
    commodity = commodity_list[0]
    for i in commodity_list:
        if name == str(i):
            commodity = i
            break
    month_with_year = []
    for i in range(1, 13):
        if current_month + i <= 12:
            month_with_year.append((current_month + i, current_year, annual_rainfall[current_month + i
- 1]))
        else:
            month_with_year.append((current_month + i - 12, current_year + 1,
annual_rainfall[current_month + i - 13]))
    max_index = 0
    min_index = 0
    max_value = 0
    min_value = 9999
    wpis = []
    current_wpi = commodity.getPredictedValue([float(current_month), current_year,
current_rainfall])
    change = []
```

```
for m, y, r in month_with_year:
    current_predict = commodity.getPredictedValue([float(m), y, r])
    if current_predict > max_value:
        max_value = current_predict
        max_index = month_with_year.index((m, y, r))
    if current_predict < min_value:
        min_value = current_predict
        min_index = month_with_year.index((m, y, r))
    wpis.append(current_predict)
    change.append(((current_predict - current_wpi) * 100) / current_wpi)
```



```
    max_month, max_year, r1 = month_with_year[max_index]
    min_month, min_year, r2 = month_with_year[min_index]
    min_value = min_value * base[name.capitalize()] / 100
    max_value = max_value * base[name.capitalize()] / 100
    crop_price = []
    for i in range(0, len(wpis)):
        m, y, r = month_with_year[i]
        x = datetime(y, m, 1)
        x = x.strftime("%b %y")
        crop_price.append([x, round((wpis[i]* base[name.capitalize()]) / 100, 2) , round(change[i], 2)])
    # print("forecasr", wpis)
    x = datetime(max_year,max_month,1)
    x = x.strftime("%b %y")
    max_crop = [x, round(max_value,2)]
    x = datetime(min_year, min_month, 1)
    x = x.strftime("%b %y")
    min_crop = [x, round(min_value,2)]
```

```
ax_crop, min_crop, crop_price
```

```
def TwelveMonthPrevious(name):
    name = name.lower()
    current_month = datetime.now().month
    current_year = datetime.now().year
    current_rainfall = annual_rainfall[current_month - 1]
    commodity = commodity_list[0]
    wpis = []
    crop_price = []
    for i in commodity_list:
        if name == str(i):
            commodity = i
            break
    month_with_year = []
    for i in range(1, 13):
        if current_month - i >= 1:
            month_with_year.append((current_month - i, current_year, annual_rainfall[current_month - i
- 1]))
        else:
            month_with_year.append((current_month - i + 12, current_year - 1,
annual_rainfall[current_month - i + 11]))


    for m, y, r in month_with_year:
        current_predict = commodity.getPredictedValue([float(m), 2013, r])
        wpis.append(current_predict)


    for i in range(0, len(wpis)):
        m, y, r = month_with_year[i]
        x = datetime(y,m,1)
        x = x.strftime("%b %y")
        price.append([x, round((wpis[i]* base[name.capitalize()]) / 100, 2)])
    previous " wpis)
```

```
    new_crop_price =[]
    for i in range(len(crop_price)-1,-1,-1):
        new_crop_price.append(crop_price[i])
    return new_crop_price


if __name__ == "__main__":
    arhar = Commodity(commodity_dict["arhar"])
    commodity_list.append(arhar)
    bajra = Commodity(commodity_dict["bajra"])
    commodity_list.append(bajra)
    barley = Commodity(commodity_dict["barley"])
    commodity_list.append(barley)
    copra = Commodity(commodity_dict["copra"])
    commodity_list.append(copra)
    cotton = Commodity(commodity_dict["cotton"])
    commodity_list.append(cotton)
    sesamum = Commodity(commodity_dict["sesamum"])
    commodity_list.append(sesamum)
    gram = Commodity(commodity_dict["gram"])
    commodity_list.append(gram)
    groundnut = Commodity(commodity_dict["groundnut"])
    commodity_list.append(groundnut)
    jowar = Commodity(commodity_dict["jowar"])
    commodity_list.append(jowar)
    maize = Commodity(commodity_dict["maize"])
    commodity_list.append(maize)
    masoor = Commodity(commodity_dict["masoor"])
    commodity_list.append(masoor)
        Commodity(commodity_dict["moong"])
lity_list.append(moong)
```

```
niger = Commodity(commodity_dict["niger"])
commodity_list.append(niger)
paddy = Commodity(commodity_dict["paddy"])
commodity_list.append(paddy)
ragi = Commodity(commodity_dict["ragi"])
commodity_list.append(ragi)
rape = Commodity(commodity_dict["rape"])
commodity_list.append(rape)
jute = Commodity(commodity_dict["jute"])
commodity_list.append(jute)
safflower = Commodity(commodity_dict["safflower"])
commodity_list.append(safflower)
soyabean = Commodity(commodity_dict["soyabean"])
commodity_list.append(soyabean)
sugarcane = Commodity(commodity_dict["sugarcane"])
commodity_list.append(sugarcane)
sunflower = Commodity(commodity_dict["sunflower"])
commodity_list.append(sunflower)
urad = Commodity(commodity_dict["urad"])
commodity_list.append(urad)
wheat = Commodity(commodity_dict["wheat"])
commodity_list.append(wheat)
Potato = Commodity(commodity_dict["Potato"])
commodity_list.append(Potato)
Onion = Commodity(commodity_dict["Onion"])
commodity_list.append(Onion)

app.run(debug=False,host='0.0.0.0')
```

crops.py

```
def crop(crop_name):

    crop_data = {

    "wheat":["/static/images/wheat.jpg", "U.P., Punjab, Haryana, Rajasthan, M.P., bihar", "rabi","Sri Lanka, United Arab Emirates, Taiwan"],

    "paddy":["/static/images/paddy.jpg", "W.B., U.P., Andhra Pradesh, Punjab, T.N.", "kharif","Bangladesh, Saudi Arabia, Iran"],

    "barley":["/static/images/barley.jpg", "Rajasthan, Uttar Pradesh, Madhya Pradesh, Haryana, Punjab", "rabi","Oman, UK, Qatar, USA"],

    "maize":["/static/images/maize.jpg", "Karnataka, Andhra Pradesh, Tamil Nadu, Rajasthan, Maharashtra", "kharif", "Hong Kong, United Arab Emirates, France"],

    "bajra":["/static/images/bajra.jpg", "Rajasthan, Maharashtra, Haryana, Uttar Pradesh and Gujarat", "kharif", "Oman, Saudi Arabia, Israel, Japan"],

    "copra":["/static/images/copra.jpg", "Kerala, Tamil Nadu, Karnataka, Andhra Pradesh, Orissa, West Bengal","rabi", "Veitnam, Bangladesh, Iran, Malaysia"],

    "cotton":["/static/images/cotton.jpg", "Punjab, Haryana, Maharashtra, Tamil Nadu, Madhya Pradesh, Gujarat","kharif", " China, Bangladesh, Egypt"],

    "masoor":["/static/images/masoor.jpg", "Uttar Pradesh, Madhya Pradesh, Bihar, West Bengal, Rajasthan", "rabi", "Pakistan, Cyprus,United Arab Emirates"],

    "gram":["/static/images/gram.jpg", "Madhya Pradesh, Maharashtra, Rajasthan, Uttar Pradesh, Andhra Pradesh & Karnataka", "rabi", "Veitnam, Spain, Myanmar"],

    "groundnut":["/static/images/groundnut.jpg", "Andhra Pradesh, Gujarat, Tamil Nadu, Karnataka, and Maharashtra", "kharif", "Indonesia, Jordan, Iraq"],

    "arhar":["/static/images/arhar.jpg", "Maharashtra, Karnataka, Madhya Pradesh and Andhra Pradesh", "kharif", "United Arab Emirates, USA, Chicago"],

    "sesamum":["/static/images/sesamum.jpg", "Maharashtra, Rajasthan, West Bengal, Andhra Pradesh, Gujarat", "rabi", "Iraq, South Africa, USA, Netherlands"],

    "jowar":["/static/images/jowar.jpg", "Maharashtra, Karnataka, Andhra Pradesh, Madhya Pradesh, Gujarat", "kharif", "Torronto, Sydney, New York"],

    "moong":["/static/images/moong.jpeg", "Rajasthan, Maharashtra, Andhra Pradesh", "rabi", "Qatar, United States, Canada"],

    "niger":["/static/images/niger.jpg", "Andha Pradesh, Assam, Chattisgarh, Gujarat, Jharkhand",
            Jnited States of American,Argenyina, Belgium"],
```

"rape":["/static/images/rape.jpg", "Rajasthan, Uttar Pradesh, Haryana, Madhya Pradesh, and Gujarat", "rabi", "Veitnam, Malaysia, Taiwan"],

"jute":["/static/images/jute.jpg", " West Bengal , Assam , Orissa , Bihar , Uttar Pradesh", "kharif", "JOrdan, United Arab Emirates, Taiwan"],

"safflower":["/static/images/safflower.jpg", "Maharashtra, Karnataka, Andhra Pradesh, Madhya Pradesh, Orissa", "kharif", " Philippines, Taiwan, Portugal"],

"soyabean":["/static/images/soyabean.jpg", "Madhya Pradesh, Maharashtra, Rajasthan, Madhya Pradesh and Maharashtra", "kharif", "Spain, Thailand, Singapore"],

"urad":["/static/images/urad.jpg", "Andhra Pradesh, Maharashtra, Madhya Pradesh, Tamil Nadu", "rabi", "United States, Canada, United Arab Emirates"],

"ragi":["/static/images/ragi.jpg", "Maharashtra, Tamil Nadu and Uttarakhand", "kharif", "United Arab Emirates, New Zealand, Bahrain"],

"sunflower":["sunflower.jpg", "Karnataka, Andhra Pradesh, Maharashtra, Bihar, Orissa", "rabi", "Phillippines, United States, Bangladesh"],

"sugarcane":["sugarcane.jpg","Uttar Pradesh, Maharashtra, Tamil Nadu, Karnataka, Andhra Pradesh" , "kharif", "Kenya, United Arab Emirates, United Kingdom"],

"Potato":["/static/images/Potato.jpeg","Uttar Pradesh, West Bengal, Bihar,Gujrat","rabi","Netherland,France,Germany"],

"Onion":["/static/images/Onion.jpg","Maharashtra, Madhya Pradesh, Karnataka, Gujarat","kharif","Bangladesh, UAE, Bhutan, Bahrain, Mauritius, Sri Lanka"]

}

return crop_data[crop_name]

requirements.txt

flask

scikit-learn

numpy

pandas

flask-cors

Index.html

```
<!DOCTYPE html>
<html>
<head>
 <title> Krishi Mitra</title>
 <style>
  /* Ensure the container has no scrollbars and fits the content */
  .botpress-webchat-container {
     overflow: hidden !important;
     padding: 0 !important;
     max-width: 100% !important;
     max-height: 100% !important;
     box-sizing: border-box !important;
  }

  /* If the bot's content is placed within another container, target it */
  .botpress-webchat-container .bp-chat-container {
     width: auto !important;
     height: auto !important;
     max-width: 100% !important;
     max-height: 100% !important;
     overflow: hidden !important;
     padding: 0 !important;
     display: block !important;
     box-sizing: border-box !important;
  }

  /* Target the logo */
  .botpress-webchat-container .bp-chat-logo {
     100% !important;
    : auto !important;
```

```
    display: block !important;
  }
 </style>


 <!-- Compiled and minified CSS -->
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.css">
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">


    <!-- Compiled and minified JavaScript -->
    <style>
      @import url('https://fonts.googleapis.com/css?family=Open+Sans&display=swap');

     body {
      font-family: 'Open Sans', sans-serif;
      margin: 0;
      padding: 0;
      background-color:white;
     }

     span, img {
      vertical-align: middle;
     }

     .brand-logo {
      display: inline-flex;
      align-items: center;
     `
```


```
    .brand-logo img {
      margin-right: 10px;
    }

    .brand-logo span {
      font-size: 24px;
      color: #f87d2a;
    }

    .fixed-image {
      background-image: url('/static/Untitled_Export_V2.jpeg'); /* Use your image path */
      background-size: cover;
      background-position: center;
      /* background-attachment: fixed;  */
      height: 100vh; /* Makes the image cover the full viewport height */
      width: 100%;
  }
  </style>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

</head>
<!-- <body class="main"> -->
 <body>

  <!-- my changes -->

  <div class="fixed-image"></div>

  <div class="row">
        s="card-panel col s9">
        ss=" #e0f7fa cyan lighten-5">Top Gainers(Current trends)</h4>
```

```
<table class="striped">
 <thead>
  <tr>
     <th>Item Name</th>
     <th>Price (per Qtl.)</th>
     <th>Change</th>
  </tr>
 </thead>

 <tbody>
 {% for item in context['top5'] %}
     <tr>
       <td>{{item[0]}}</td>
       <td>₹{{item[1]}}</td>
       <td class="valign-wrapper">{{item[2]}}% <img src="../static/gain-icon.png" height="25"
width="25"></td>
     </tr>
 {% endfor %}
 </tbody>
</table>
 <div class="card-panel">
<h4 class="#e0f7fa cyan lighten-5">Top Losers(Current trends)</h4>
<table class="striped">
 <thead>
  <tr>
     <th>Item Name</th>
     <th>Price (per Qtl.)</th>
     <th>Change</th>
  </tr>
 ...  id>
```

```
<tbody>
{% for item in context['bottom5'] %}
 <tr>
  <td>{{item[0]}}</td>
  <td>₹{{item[1]}}</td>
  <td class="valign-wrapper">{{item[2]}}% <img src="../static/loss-icon.png" height="25" width="25"></td>
 </tr>
{% endfor %}
</tbody>
</table>
</div>
</div>
<div class="col s3">
<div class="card grey lighten-3">
 <div class="card-content black-text">
  <h6>Star Commodity Prediction</h6>
  <span class="card-title" id="time">{{context.sixmonths[0][0]}}</span>
  <table class="valign-wrapper">
   <tr>
    <td><h5 id="crop1">{{context.sixmonths[0][1]}}</h5></td>

<td>            </td>
    <td class="right">
     <h4 id="price1">₹{{context.sixmonths[0][2]}}</h4>
    <p id="pos-change" class="valign-wrapper right">{{context.sixmonths[0][3]}}%</p><img id="change1" src="../static/gain-icon.png" height="20" width="20" style="{margin-left: 5;}"></td>

   </tr>
   <tr>
    ><h5 id="crop2">{{context.sixmonths[0][4]}}</h5></td>
```

```
<td>             </td>
          <td class="right"><h4 id="price2">₹{{context.sixmonths[0][5]}}</h4>
          <p id="neg-change" class="valign-wrapper right">{{context.sixmonths[0][6]}}%</p><img id="change2" src="../static/loss-icon.png" height="20" width="20" style="{margin-left: 5;}">
          </td>
        </tr>
       </table>
       <script>
        function sleep(ms) {
          return new Promise(resolve => setTimeout(resolve, ms));
        }
        function updateTime(i){
        $('#time').load('https://krishimitra-gyl4.onrender.com/ticker/'+0+ '/' + i);
        }

        function updateCrop1(i){
          $('#crop1').load('https://krishimitra-gyl4.onrender.com/ticker/'+ 1 + '/' + i);
        }

        function updatePrice1(i){
          $('#price1').load('https://krishimitra-gyl4.onrender.com/ticker/'+ 2 + '/' + i);
        }

        function updateChange1(i){
          $("#change1").attr("src","../static/gain-icon.png");
          $('#pos-change').load('https://krishimitra-gyl4.onrender.com/ticker/'+ 3 + '/'  + i);

        }

        tion updateCrop2(i){
        #crop2').load('https://krishimitra-gyl4.onrender.com/ticker/'+ 4 + '/'  + i);
```

```
}

function updatePrice2(i){
  $('#price2').load('https://krishimitra-gyl4.onrender.com/ticker/'+ 5 + '/'  + i);
}

function updateChange2(i){
  $("#change2").attr("src","../static/loss-icon.png");
  $('#neg-change').load('https://krishimitra-gyl4.onrender.com/ticker/'+ 6 + '/'  + i);

}
```



```
</script>
<script>
    var i = 0
    setInterval(async function(){
    i = (i+1)%5;
    updateTime(i);

    updateCrop1(i);
    await sleep(200);

    updatePrice1(i);
    await sleep(200);

    updateChange1(i);
    await sleep(200);

    updateCrop2(i);
    await sleep(200);
```

```
                updatePrice2(i);
                await sleep(200);

                updateChange2(i);
                await sleep(200);

                /*
                setTimeout(updateCrop1(i), 100);
                setTimeout(updatePrice1(i), 300);
                setTimeout(updateChange1(i), 500);
                setTimeout(updateCrop2(i), 700);
                setTimeout(updatePrice2(i), 800);
                setTimeout(updateChange2(i), 900);
                */

                }, 4000);

            </script>

          </div>
        </div>

      </div>
    </div>
    <h5>Explore by commodity</h5>
    <div class="row">
      <div class="col s3">
        <a href="http://localhost:5000/commodity/paddy" style="color: #000000">
        <div class="card grey lighten-4">
              ass="card-content row valign-wrapper">
              :lass="col s3">
```

```
    <img src="https://img.icons8.com/color/48/000000/rice-bowl.png">
   </div>
   <div class="col s9">
    <span class="card-title">Paddy</span>
   </div>
   </div>
  </div>
 </a>
</div>
 <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/wheat" style="color: #000000">
   <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
     <div class="col s3">
     <img src="https://img.icons8.com/color/48/000000/wheat.png">
    </div>
    <div class="col s9">
     <span class="card-title">Wheat</span>
    </div>
    </div>
  </div>
 </a>
</div> <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/barley" style="color: #000000">
  <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
     <div class="col s3">
     <img src="https://img.icons8.com/color/48/000000/barley.png">
    </div>
          ass="col s9">
          n class="card-title">Barley</span>
```

```
    </div>
    </div>
   </div>
  </a>
  </div> <div class="col s3">
   <a href="https://krishimitra-gyl4.onrender.com/commodity/soyabean" style="color: #000000">
   <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
     <div class="col s3">
     <img src="https://img.icons8.com/color/48/000000/soy.png">
     </div>
     <div class="col s9">
      <span class="card-title">Soya Bean</span>
     </div>
    </div>
   </div>
  </a>
  </div>
 </div>
  <div class="row">
  <div class="col s3">
   <a href="https://krishimitra-gyl4.onrender.com/commodity/cotton" style="color: #000000">
   <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
     <div class="col s3">
     <img src="https://img.icons8.com/color/48/000000/cotton.png">
     </div>
     <div class="col s9">
      <span class="card-title">Cotton</span>
```


```
 </div>
</a>
</div>
 <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/copra" style="color: #000000">
  <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
      <div class="col s3">
      <img src="https://img.icons8.com/color/48/000000/coconut.png">
    </div>
    <div class="col s9">
      <span class="card-title">Coconut</span>
    </div>
    </div>
  </div>
</a>
</div> <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/groundnut" style="color: #000000">
  <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
      <div class="col s3">
      <img src="https://img.icons8.com/color/48/000000/peanuts.png">
    </div>
    <div class="col s9">
      <span class="card-title">Ground Nut Seeds</span>
    </div>
    </div>
  </div>
</a>
```



```
div class="col s3">
="https://krishimitra-gyl4.onrender.com/commodity/rape" style="color: #000000">
```

```
 <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
   <div class="col s3">
   <img src="https://img.icons8.com/color/48/000000/peanuts.png">
  </div>
  <div class="col s9">
   <span class="card-title">Mustard Seed</span>
  </div>
  </div>
 </div>
</a>
</div>
</div>
<div class="row">
<div class="col s3">
 <a href="https://krishimitra-gyl4.onrender.com/commodity/sesamum" style="color: #000000">
 <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
   <div class="col s3">
   <img src="https://img.icons8.com/color/48/000000/sesame.png">
  </div>
  <div class="col s9">
   <span class="card-title">Gingelly Seed(Sesamum)</span>
  </div>
  </div>
 </div>
</a>
</div>
 <div class="col s3">
      ="https://krishimitra-gyl4.onrender.com/commodity/gram" style="color: #000000">
      ss="card grey lighten-4">
```

```
<div class="card-content row valign-wrapper">
<div class="col s3">
<img src="https://img.icons8.com/color/48/000000/paper-bag-with-seeds.png">
</div>
<div class="col s9">
<span class="card-title">Gram</span>
</div>
</div>
</div>
</a>
</div> <div class="col s3">
<a href="https://krishimitra-gyl4.onrender.com/commodity/sugarcane" style="color: #000000">
<div class="card grey lighten-4">
<div class="card-content row valign-wrapper">
<div class="col s3">
<img src="https://img.icons8.com/color/48/000000/bamboo.png">
</div>
<div class="col s9">
<span class="card-title">Sugarcane</span>
</div>
</div>
</div>
</a>
</div> <div class="col s3">
<a href="https://krishimitra-gyl4.onrender.com/commodity/arhar" style="color: #000000">
<div class="card grey lighten-4">
<div class="card-content row valign-wrapper">
<div class="col s3">
<img src="https://img.icons8.com/color/48/000000/paper-bag-with-seeds.png">
```


ass="col s9">

```
    <span class="card-title">Arhar</span>
   </div>
   </div>
  </div>
 </a>
 </div>
</div>
 <div class="row">
 <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/ragi" style="color: #000000">
  <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
     <div class="col s3">
      <img src="https://img.icons8.com/office/48/000000/wheat.png">
     </div>
     <div class="col s9">
      <span class="card-title">Ragi</span>
     </div>
     </div>
  </div>
 </a>
 </div>
 <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/maize" style="color: #000000">
  <div class="card grey lighten-4">
    <div class="card-content row valign-wrapper">
     <div class="col s3">
     <img src="https://img.icons8.com/color/48/000000/corn.png">
     </div>
          ass="col s9">
     ı class="card-title">Maize</span>
```

```
  </div>
  </div>
 </div>
</a>
</div> <div class="col s3">
 <a href="https://krishimitra-gyl4.onrender.com/commodity/moong" style="color: #000000">
 <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
   <div class="col s3">
   <img src="https://img.icons8.com/color/48/000000/paper-bag-with-seeds.png">
  </div>
   <div class="col s9">
    <span class="card-title">Moong</span>
  </div>
  </div>
 </div>
</a>
</div> <div class="col s3">
 <a href="https://krishimitra-gyl4.onrender.com/commodity/masoor" style="color: #000000">
 <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
   <div class="col s3">
   <img src="https://img.icons8.com/color/48/000000/paper-bag-with-seeds.png">
  </div>
  <div class="col s9">
   <span class="card-title">Masoor</span>
  </div>
  </div>
 </div>
```

```
</div>
 <div class="row">
 <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/urad" style="color: #000000">
  <div class="card grey lighten-4">
   <div class="card-content row valign-wrapper">
    <div class="col s3">
    <img src="https://img.icons8.com/color/48/000000/paper-bag-with-seeds.png">
   </div>
   <div class="col s9">
    <span class="card-title">Urad</span>
   </div>
   </div>
  </div>
 </a>
 </div>
 <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/jute" style="color: #000000">
  <div class="card grey lighten-4">
   <div class="card-content row valign-wrapper">
    <div class="col s3">
    <img src="https://img.icons8.com/color/48/000000/potato.png">
   </div>
   <div class="col s9">
    <span class="card-title">Raw Jute</span>
   </div>
   </div>
  </div>
 </a>
```



```
div class="col s3">
="https://krishimitra-gyl4.onrender.com/commodity/niger" style="color: #000000">
```

```
 <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
   <div class="col s3">
   <img src="https://img.icons8.com/color/48/000000/paper-bag-with-seeds.png">
  </div>
  <div class="col s9">
   <span class="card-title">Niger Seed</span>
  </div>
  </div>
 </div>
</a>
</div> <div class="col s3">
 <a href="https://krishimitra-gyl4.onrender.com/commodity/safflower" style="color: #000000">
 <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
   <div class="col s3">
   <img src="https://img.icons8.com/color/48/000000/paper-bag-with-seeds.png">
  </div>
  <div class="col s9">
   <span class="card-title">Kardi Seed</span>
  </div>
  </div>
 </div>
</a>
</div>
</div>
 <div class="row">
 <div class="col s3">
  <a href="https://krishimitra-gyl4.onrender.com/commodity/sunflower" style="color: #000000">
```


ss="card grey lighten-4">

ass="card-content row valign-wrapper">

```
  <div class="col s3">
  <img src="https://img.icons8.com/color/48/000000/potato.png">
 </div>
 <div class="col s9">
  <span class="card-title">Sunflower</span>
 </div>
 </div>
</div>
</a>
</div>
<div class="col s3">
 <a href="https://krishimitra-gyl4.onrender.com/commodity/jowar" style="color: #000000">
  <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
  <div class="col s3">
  <img src="https://img.icons8.com/color/48/000000/potato.png">
 </div>
 <div class="col s9">
  <span class="card-title">Jowar</span>
 </div>
 </div>
</div>
</a>
</div> <div class="col s3">
 <a href="https://krishimitra-gyl4.onrender.com/commodity/bajra" style="color: #000000">
 <div class="card grey lighten-4">
  <div class="card-content row valign-wrapper">
  <div class="col s3">
  <img src="https://img.icons8.com/color/48/000000/potato.png">
```


ass="col s9">

```
    <span class="card-title">Bajra</span>
   </div>
   </div>
  </div>
 </a>
</div> <div class="col s3">
  <a href="http://localhost:5000/commodity/bajra" style="color: #000000">
  <div class="card grey lighten-4">
   <div class="card-content row valign-wrapper">
    <div class="col s3">
    <img src="/static/images/alloo.png" style="width: 48px; height: 48px;">>
    </div>
    <div class="col s9">
    <span class="card-title">Potato</span>
   </div>
   </div>
  </div>
 </a>
</div><div class="col s3">
  <a href="http://localhost:5000/commodity/Onion" style="color: #000000">
  <div class="card grey lighten-4">
   <div class="card-content row valign-wrapper">
    <div class="col s3">
    <img src="/static/images/onion.png" style="width: 48px; height: 48px;">>
   </div>
   <div class="col s9">
    <span class="card-title">Onion</span>
   </div>
   </div>
```

```
 </div>


<script type="text/javascript" charset="utf-8">

 $(document).ready(function(){

   $('.slider').slider({full_width :true});

 });

</script>

<div class="botpress-webchat-container">  <script
src="https://cdn.botpress.cloud/webchat/v2.1/inject.js"></script>

 <script src="https://mediafiles.botpress.cloud/3bc67aa9-7bbf-4745-aae3-
efe4d68f5807/webchat/v2.1/config.js"></script>

</div>
<body>
<script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js"></script>
</body>


</html>
```

commodity.html

```
<!DOCTYPE html>

<html>

<head>

 <title>Krishi Mitra</title>

 <!-- Compiled and minified CSS -->

  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.css">

  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">


  <!-- Compiled and minified JavaScript -->
```



```
      <style>

          div.main {

              padding: 5px 50px 75px 50px;

          }

          body{

           background-color:#A9BA9D;


          }

          .header{

           color: black;

          }

          .custom-card-panel {

  background-color:white; /* Replace with your desired color */

  color: black; /* Text color */

}


  </style>
```

```
<div class="main">
  <div class="nav-wrapper">


    <h3 class="card-panel custom-card-panel center valign-wrapper"><a class="brand-logo" href="#" style="color: green;"></a><img alt="" src="/static/krishilogobg.png" height="100px" width="100px" class="responsive-img" />
        <span>KRISHI MITRA</span></a>
        </a></h3>
          </div>
```



```
class="header">{{context.name}}</h2>
    <div class="row">
      <div class="col s8 m7">
     <div class="card horizontal medium">
      <div class="card-image">
       <img src={{context.image_url}}>
      </div>
      <div class="card-stacked">
       <div class="card-content">
        <table>
          <tr>
           <td>Current Price</td>
           <td><b>₹ {{context.current_price}} / ql</b></td>
          </tr>
          <tr>
           <td>Prime Location</td>
           <td><b>{{context.prime_loc}}</b></td>
```

r>

>

```
<td>Crop Type</td>
<td><b>{{context.type_c}}</b></td>
</tr>
<tr>
<td>Export</td>
<td><b>{{context.export}}</b></td>
</tr>
</table>
</div>
</div>
</div>
</div>
<div class="col s4">
<div class="card white">
<div class="card-content black-text">
<span class="card-title">Brief Forecast</span>
<table>
<tr>
<td><p>Min. crop price time</p>
<td><h5>{{context.min_crop[0]}}</h5></td>
<td>
<h4>₹{{context.min_crop[1]}}</h4>
</td>
</tr>
<tr>
<td><p>Max. crop price time</p>
<td><h5>{{context.max_crop[0]}}</h5></td>
<td>
ı4>₹{{context.max_crop[1]}}</h4>
d>
```

```
          </tr>
         </table>


        </div>
       </div>
      </div>
     </div>
    </div>

     <div class="row" >
      <div class="col s4" style="background-color:#D0D9CD;box-shadow: 5px 5px 8px #78866B; margin:
auto;">

       <h5 style="color: #687169;">Forecast Trends</h5>
        <table class="striped"  style="color: black;">
         <thead>
          <tr>
            <th>Month</th>
            <th>Price (per Qtl.)</th>
            <th>Change</th>
          </tr>
         </thead>

         <tbody>
         {% for item in context.forecast_values %}
          <tr>
           <td>{{item[0]}}</td>
           <td>₹{{item[1]}}</td>
           <td class="valign-wrapper">{{item[2]}}% {% if item[2]>=0 %}<img src= "../static/gain-icon.png"
height="25" width="25">{% else %}<img src= "../static/loss-icon.png" height="25" width="25">{%
endif %}</td>

          !for %}
```

```
      </tbody>
    </table>
</div>

<div class="col s2"></div>
 <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.4.0/Chart.min.js"></script>

  <!-- <div class="chartjs-wrapper col s7 offset-s1">
  <canvas id="chartjs-0" class="chartjs" width="undefined" height="undefined" style=" width:800px
!important;
 height:400px !important;">
    </canvas>
    <script>
    new Chart(
  document.getElementById("chartjs-
0"),{"type":"line","data":{"labels":{{context.forecast_x|safe}},"datasets":[{"label":"Next year
Price","data":{{context.forecast_y}},"fill":false,"borderColor":"rgb(75, 192,
192)","lineTension":0.1}]},"options":{ responsive: true,
  maintainAspectRatio: false,
  scales: {
    yAxes: [{
      display: true,
      ticks: {
        suggestedMin: 1000,
        stepSize : 200
      }
    }]
  }}});</script>
 </div> -->

<div class="chartjs-wrapper col s7 offset-s1">
<canvas id="chartjs-0" class="chartjs" style="width: 800px !important; height: 400px
;background-color: white;margin-bottom:55px"></canvas>
```

```
<script>
    new Chart(
        document.getElementById("chartjs-0"), {
            "type": "line",
            "data": {
                "labels": {{ context.forecast_x|safe }},
                "datasets": [{
                    "label": "Next year Price",
                    "data": {{ context.forecast_y|safe }},
                    "fill": false,
                    "borderColor": "rgb(75, 192, 192)",
                    "lineTension": 0.1
                }]
            },
            "options": {
                responsive: true,
                maintainAspectRatio: false,
                scales: {
                    yAxes: [{
                        display: true,
                        ticks: {
                            suggestedMin: 1000,
                            stepSize: 200
                        }
                    }]
                }
            }
        }
    );
```

```
<!-- <div class="chartjs-wrapper col s7 offset-s1">
  <canvas id="chartjs-1" class="chartjs" width="undefined" height="undefined" style=" width:800px !important;
 height:400px !important;">
  </canvas>
 <script>
  new Chart(document.getElementById("chartjs-1"),{"type":"line","data":{"labels":{{context.previous_x|safe}},"datasets":[{"label":"Previous year price","data":{{context.previous_y}},"fill":false,"borderColor":"rgb(75, 192, 192)","lineTension":0.1}]},"options":{ responsive: true,
    maintainAspectRatio: false,
      scales: {
    yAxes: [{
      display : true,
      ticks: {
        suggestedMin : 1000,
        stepSize : 200
      }
    }]
  }}});</script>
 </div> -->


 <div class="chartjs-wrapper col s7 offset-s1">
  <canvas id="chartjs-1" class="chartjs" style="width: 800px !important; height: 400px !important; background-color: white"></canvas>
  <script>
    new Chart(document.getElementById("chartjs-1"), {
      type: "line",
      : {
      bels: {{ context.previous_x|safe }},
```

```
            datasets: [{
                label: "Previous year price",
                data: {{ context.previous_y|safe }},
                fill: false,
                borderColor: "rgb(75, 192, 192)",
                lineTension: 0.1
            }]
        },
        options: {
            responsive: true,
            maintainAspectRatio: false,
            scales: {
                yAxes: [{
                    display: true,
                    ticks: {
                        suggestedMin: 1000,
                        stepSize: 200
                    }
                }],
                xAxes: [{
                    gridLines: {
                        color: "rgba(135, 206, 250, 0.1)",  /* Light blue gridlines */
                    }
                }]
            }
        }
    });
</script>
</div>
```

```
 </div>


</script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js"></script>
</body>
</html>
```