**import** dash

**from** dash **import** dcc

**from** dash **import** html

**from** dash.dependencies **import** Input, Output

**import** pandas **as** pd

**import** plotly.graph\_objs **as** go

**import** plotly.express **as** px

*# Load the data using pandas*

data = pd.read\_csv('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DV0101EN-SkillsNetwork/Data%20Files/historical\_automobile\_sales.csv')

*# Initialize the Dash app*

app = dash.Dash(\_\_name\_\_)

*# Set the title of the dashboard*

*#app.title = "Automobile Statistics Dashboard"*

*#---------------------------------------------------------------------------------*

*# Create the dropdown menu options*

dropdown\_options = [

{'label': 'Yearly Statistics Report', 'value': 'Yearly Statistics'},

{'label': 'Recession Period Statistics', 'value': 'Recession Period Statistics'}

]

*# List of years*

year\_list = [i **for** i **in** range(1980, 2024, 1)]

*#---------------------------------------------------------------------------------------*

*# Create the layout of the app*

app.layout = html.Div([

*#TASK 2.1 Add title to the dashboard*

html.H1("Automobile Statistics Dashboard"),*#May include style for title*

html.Div([*#TASK 2.2: Add two dropdown menus*

html.Label("Select Statistics:"),

dcc.Dropdown(

id='dropdown-statistics',

options=[

{'label': 'Yearly Statistics', 'value': 'Yearly Statistics'},

{'label': 'Recession Period Statistics', 'value': 'Recession Period Statistics'},

],

value='Select statistics',

placeholder='Select a report'

)

]),

html.Div(dcc.Dropdown(

id='select-year',

options=[{'label': i, 'value': i} **for** i **in** year\_list],

value='Select Year',

style={'width': '80%', 'padding': 3, 'textAlign': 'center', 'font-size': 20}

)),

html.Div([*#TASK 2.3: Add a division for output display*

html.Div(id='output-container', className='chart-grid', style={'display': 'flex'}),])

])

*#TASK 2.4: Creating Callbacks*

*# Define the callback function to update the input container based on the selected statistics*

@app.callback(

Output(component\_id='select-year', component\_property='disabled'),

Input(component\_id='dropdown-statistics',component\_property='value'))

**def** **update\_input\_container**(selected\_statistics):

**if** selected\_statistics =='Yearly Statistics':

**return** False

**else**:

**return** True

*#Callback for plotting*

*# Define the callback function to update the input container based on the selected statistics*

@app.callback(

Output(component\_id='output-container', component\_property='children'),

[Input(component\_id='select-year', component\_property='value'),

Input(component\_id='dropdown-statistics', component\_property='value')])

**def** **update\_output\_container**(selected\_statistics, input\_year):

**if** selected\_statistics == 'Recession Period Statistics':

*# Filter the data for recession periods*

recession\_data = data[data['Recession'] == 1]

*#TASK 2.5: Create and display graphs for Recession Report Statistics*

*#Plot 1 Automobile sales fluctuate over Recession Period (year wise)*

*# use groupby to create relevant data for plotting*

yearly\_rec=recession\_data.groupby('Year')['Automobile\_Sales'].mean().reset\_index()

R\_chart1 = dcc.Graph(

figure=px.line(yearly\_rec,

x='Year',

y='Automobile\_Sales',

title="Average Automobile Sales fluctuation over Recession Period"))

*#Plot 2 Calculate the average number of vehicles sold by vehicle type*

*# use groupby to create relevant data for plotting*

average\_sales = recession\_data.groupby('Vehicle\_Type')['Automobile\_Sales'].mean().reset\_index()

R\_chart2 = dcc.Graph(

figure=px.bar(average\_sales,

x='Vehicle\_Type',

y='Automobile\_Sales',

title="Average Automobile Sales by Vehicle Type during Recession Period"))

*# Plot 3 Pie chart for total expenditure share by vehicle type during recessions*

*# use groupby to create relevant data for plotting*

exp\_rec= recession\_data.groupby('Vehicle\_Type')['Advertising\_Expenditure'].sum().reset\_index()

R\_chart3 = dcc.Graph(

figure=px.pie(exp\_rec,

values='Advertising\_Expenditure',

names='Vehicle\_Type',

title="Total Expenditures by Vehicle Type during Recession Period"))

*# Plot 4 bar chart for the effect of unemployment rate on vehicle type and sales*

unemp\_rate = recession\_data.groupby('Vehicle\_Type')['Automobile\_Sales'].mean().reset\_index()

R\_chart4 = dcc.Graph(

figure=px.bar(unemp\_rate,

x='unemployment\_rate',

y='Automobile\_Sales',

title="Effects of Unemployment Rate on Automobile Sales by Vehicle Type during Recession Period"))

**return** [

html.Div(className='chart-item', children=[html.Div(children=R\_chart1),html.Div(children=R\_chart2)],style={'display': 'flex'}),

html.Div(className='chart-item', children=[html.Div(children=R\_chart3),html.Div(children=R\_chart4)],style={'display': 'flex'})

]

*# TASK 2.6: Create and display graphs for Yearly Report Statistics*

*# Yearly Statistic Report Plots*

**elif** (input\_year **and** selected\_statistics=='Yearly Statistics') :

yearly\_data = data[data['Year'] == input\_year]

*#TASK 2.5: Creating Graphs Yearly data*

*#plot 1 Yearly Automobile sales using line chart for the whole period.*

yas= data.groupby('Year')['Automobile\_Sales'].sum().reset\_index()

Y\_chart1 = dcc.Graph(

figure=px.line(yas,

x='Year',

y='Automobile\_Sales',

title='Automobile Sales for the Year'))

*# Plot 2 Total Monthly Automobile sales using line chart.*

data['Month'] = pd.to\_datetime(data['Month'])

mas = yearly\_data.groupby('Month')['Automobile\_Sales'].sum().reset\_index()

Y\_chart2 = dcc.Graph(

figure=px.line(mas,

x='Month',

y='Automobile\_Sales',

title='Total Automobile Sales by Month'))

*# Total Advertisement Expenditure for each vehicle using pie chart*

exp\_data = yearly\_data.groupby('Vehicle\_Type')['Advertising\_Expenditure'].sum().reset\_index()

Y\_chart4 = dcc.Graph(

figure=px.pie(exp\_data,

values='Advertising\_Expenditure',

names='Vehicle\_Type',

title='Total Advertising Expenditure by Vehicle Type'))

*# Plot bar chart for average number of vehicles sold during the given year*

avr\_vdata = yearly\_data.groupby('Year')['Automobile\_Sales'].mean().reset\_index()

Y\_chart3 = dcc.Graph(

figure=px.bar(avr\_vdata,

x='Year',

y='Automobile\_Sales',

title='Average Vehicles Sold by Vehicle Type in the year {}'.format(input\_year)))

*# Total Advertisement Expenditure for each vehicle using pie chart*

exp\_data = yearly\_data.groupby('Vehicle Type')['Advertising\_Expenditure'].sum().reset\_index()

Y\_chart4 = dcc.Graph(

figure=px.pie(exp\_data,

values='Advertising\_Expenditure',

names='Vehicle\_Type',

title='Total Advertising Expenditure by Vehicle Type'))

*#TASK 2.6: Returning the graphs for displaying Yearly data*

**return** [

html.Div(className='chart-item', children=[html.Div(children=Y\_chart1),html.Div(children=Y\_chart2)],style={'display': 'flex'}),

html.Div(className='chart-item', children=[html.Div(children=Y\_chart3),html.Div(children=Y\_chart4)],style={'display': 'flex'})

]

**else**:

**return** None

*# Run the Dash app*

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

app.run\_server(debug=True)