Problem Descrption on Dataset

To predict electric vehicle sales (electric_vehicle_sold) based on the features such as date, vehicle_category, and maker, we can build a regression model. In this case, since we are predicting the number of vehicles sold (a continuous value), we will use Linear Regression

Summary on the Datasets.

date: The date on which the sales occurred.

vehicle_category: The type of vehicle (e.g., Sedan, SUV).

maker: The manufacturer of the vehicle (e.g., Tesla, Nissan).

electric_vehicle_sold: The number of electric vehicles sold.

Solution on the Problem

Problem Statement: Predict the number of electric vehicles sold based on date, vehicle_category, and maker.

Dataset: Columns: date, vehicle_category, maker, and electric_vehicle_sold.

Approach:

Convert date into year and month.

Encode categorical variables (vehicle_category and maker).

Use Linear Regression for prediction.

In [1]:
```
#-Import Necessary Libraries.

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

Data Collection and Processing

In [2]:
```
#-Load the data set.

vehicle=pd.read_csv("D:\\ML Project\\RPC12_Input_For_Participants\\electric_vehicle_s
```

In [3]:
```
#-Shown the dataset.

print("Show the electric_vehicle datasets:",vehicle)
```

```
Show the electric_vehicle datasets:          date vehicle_category            maker
electric_vehicles_sold
0    01-Apr-21       2-Wheelers     OLA ELECTRIC                      0
1    01-Apr-22       2-Wheelers          OKAYA EV                     0
2    01-May-21       2-Wheelers     OLA ELECTRIC                      0
3    01-Jun-21       2-Wheelers     OLA ELECTRIC                      0
4    01-Jul-21       2-Wheelers     OLA ELECTRIC                      0
..         ...              ...              ...                    ...
811  01-Mar-24       2-Wheelers           BGAUSS                   3070
812  01-Mar-24       2-Wheelers  BATTRE ELECTRIC                    625
813  01-Mar-24       2-Wheelers    KINETIC GREEN                   3915
814  01-Mar-24       2-Wheelers           REVOLT                    585
815  01-Mar-24       2-Wheelers           OTHERS                  10579

[816 rows x 4 columns]
```

Understand the Dataset

In [4]:
```
# Show the first 10-data row and columns.

vehicle.head(10)
```

Out[4]:

	date	vehicle_category	maker	electric_vehicles_sold
0	01-Apr-21	2-Wheelers	OLA ELECTRIC	0
1	01-Apr-22	2-Wheelers	OKAYA EV	0
2	01-May-21	2-Wheelers	OLA ELECTRIC	0
3	01-Jun-21	2-Wheelers	OLA ELECTRIC	0
4	01-Jul-21	2-Wheelers	OLA ELECTRIC	0
5	01-Aug-21	2-Wheelers	OLA ELECTRIC	0
6	01-Sep-21	2-Wheelers	OLA ELECTRIC	0
7	01-Oct-21	2-Wheelers	OLA ELECTRIC	0
8	01-Nov-21	2-Wheelers	OLA ELECTRIC	0
9	01-Apr-21	4-Wheelers	BYD India	0

In [5]:
```
# Show the last 10-data row and columns.

vehicle.tail(10)
```

Out[5]:

	date	vehicle_category	maker	electric_vehicles_sold
806	01-Mar-24	2-Wheelers	BAJAJ	17716
807	01-Mar-24	2-Wheelers	AMPERE	3108
808	01-Mar-24	2-Wheelers	OKINAWA	673
809	01-Mar-24	2-Wheelers	HERO ELECTRIC	316
810	01-Mar-24	2-Wheelers	OKAYA EV	1218
811	01-Mar-24	2-Wheelers	BGAUSS	3070
812	01-Mar-24	2-Wheelers	BATTRE ELECTRIC	625
813	01-Mar-24	2-Wheelers	KINETIC GREEN	3915
814	01-Mar-24	2-Wheelers	REVOLT	585
815	01-Mar-24	2-Wheelers	OTHERS	10579

In [6]:
```
#-Weather checking data shape.

vehicle.shape
```

Out[6]:
```
(816, 4)
```

In [7]:
```
#-Weather checking data information.

vehicle.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 4 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   date                   816 non-null    object
 1   vehicle_category       816 non-null    object
 2   maker                  816 non-null    object
 3   electric_vehicles_sold  816 non-null    int64 
dtypes: int64(1), object(3)
memory usage: 25.6+ KB
```

In [8]:
```
#-Weather checking missing value.

vehicle.isnull()
```

Out[8]:

	date	vehicle_category	maker	electric_vehicles_sold
0	False	False	False	False
1	False	False	False	False
2	False	False	False	False
3	False	False	False	False
4	False	False	False	False
...	...	...	...	...
811	False	False	False	False
812	False	False	False	False
813	False	False	False	False
814	False	False	False	False
815	False	False	False	False

816 rows × 4 columns

In [9]:
```
#-Weather count missing value.

vehicle.isnull().sum()
```

Out[9]:
```
date                    0
vehicle_category        0
maker                   0
electric_vehicles_sold  0
dtype: int64
```

In [10]:
```
#-Weather checking duplicates value.

vehicle.duplicated()
```

Out[10]:
```
0      False
1      False
2      False
3      False
4      False
       ...
811    False
812    False
813    False
814    False
815    False
Length: 816, dtype: bool
```

In [11]:
```
#-Weather count duplicated value.

vehicle.duplicated().sum()
```

Out[11]:
```
0
```

In [12]:
```
#-Weather checking statiscal value.

vehicle.describe()
```

Out[12]:

	electric_vehicles_sold
count	816.000000
mean	2531.998775
std	4771.077333
min	0.000000
25%	42.000000
50%	662.000000
75%	2636.500000
max	44630.000000

In [13]:
```
#-Weather count maker all company.

vehicle["maker"].value_counts()
```

Out[13]:
```
REVOLT                36
OLA ELECTRIC          36
Mahindra & Mahindra   36
OKINAWA               36
Tata Motors           36
HERO ELECTRIC         36
MG Motor              36
KIA Motors            36
BYD India             36
OTHERS                36
Mercedes -Benz AG     36
TVS                   36
Hyundai Motor         36
PCA Automobiles       36
ATHER                 36
AMPERE                36
Volvo Auto India      36
BMW India             36
BAJAJ                 36
OKAYA EV              24
JITENDRA              24
BEING                 24
PURE EV               24
BGAUSS                12
KINETIC GREEN         12
BATTRE ELECTRIC       12
Name: maker, dtype: int64
```

In [14]:
```
#-Weather checking data properties.

vehicle.describe(percentiles=None,include=None,exclude=None).sum()
```

Out[14]:
```
electric_vehicles_sold    56089.576107
dtype: float64
```

In [15]:
```
#-Weather checking dataset correlation.

Corelation=vehicle.corr()
```

In [16]:
```
Corelation
```

Out[16]:

	electric_vehicles_sold
electric_vehicles_sold	1.0

In [17]:
```
# Convert 'date' to datetime format and extract year and month

vehicle['date'] = pd.to_datetime(vehicle['date'])
vehicle['year'] = vehicle['date'].dt.year
vehicle['month'] = vehicle['date'].dt.month
```

In [18]:
```
vehicle
```

Out[18]:

	date	vehicle_category	maker	electric_vehicles_sold	year	month
0	2021-04-01	2-Wheelers	OLA ELECTRIC	0	2021	4
1	2022-04-01	2-Wheelers	OKAYA EV	0	2022	4
2	2021-05-01	2-Wheelers	OLA ELECTRIC	0	2021	5
3	2021-06-01	2-Wheelers	OLA ELECTRIC	0	2021	6
4	2021-07-01	2-Wheelers	OLA ELECTRIC	0	2021	7
...	...	...	...	...	...	...
811	2024-03-01	2-Wheelers	BGAUSS	3070	2024	3
812	2024-03-01	2-Wheelers	BATTRE ELECTRIC	625	2024	3
813	2024-03-01	2-Wheelers	KINETIC GREEN	3915	2024	3
814	2024-03-01	2-Wheelers	REVOLT	585	2024	3
815	2024-03-01	2-Wheelers	OTHERS	10579	2024	3

816 rows × 6 columns

In [19]:
```
# One-hot encode the 'vehicle_category' and 'maker' columns

vehicle = pd.get_dummies(vehicle, columns=['vehicle_category', 'maker'], drop_first=T
```

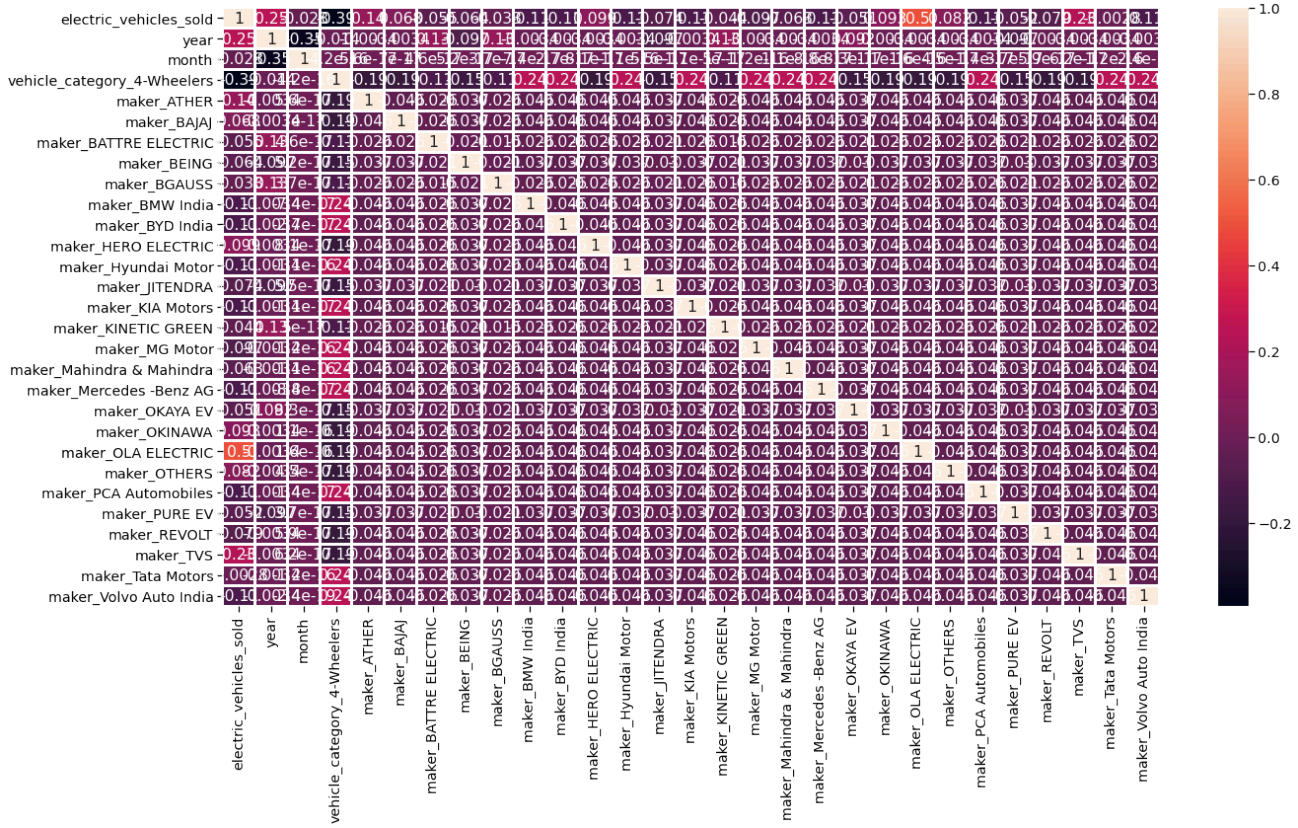
In [20]:
```
vehicle
```

Out[20]:

	date	electric_vehicles_sold	year	month	vehicle_category_4-Wheelers	maker_ATHER	maker_BAJAJ	maker
0	2021-04-01	0	2021	4	0	0	0	
1	2022-04-01	0	2022	4	0	0	0	
2	2021-05-01	0	2021	5	0	0	0	
3	2021-06-01	0	2021	6	0	0	0	
4	2021-07-01	0	2021	7	0	0	0	
...	...	...	...	...	...	...	...	...
811	2024-03-01	3070	2024	3	0	0	0	
812	2024-03-01	625	2024	3	0	0	0	
813	2024-03-01	3915	2024	3	0	0	0	
814	2024-03-01	585	2024	3	0	0	0	
815	2024-03-01	10579	2024	3	0	0	0	

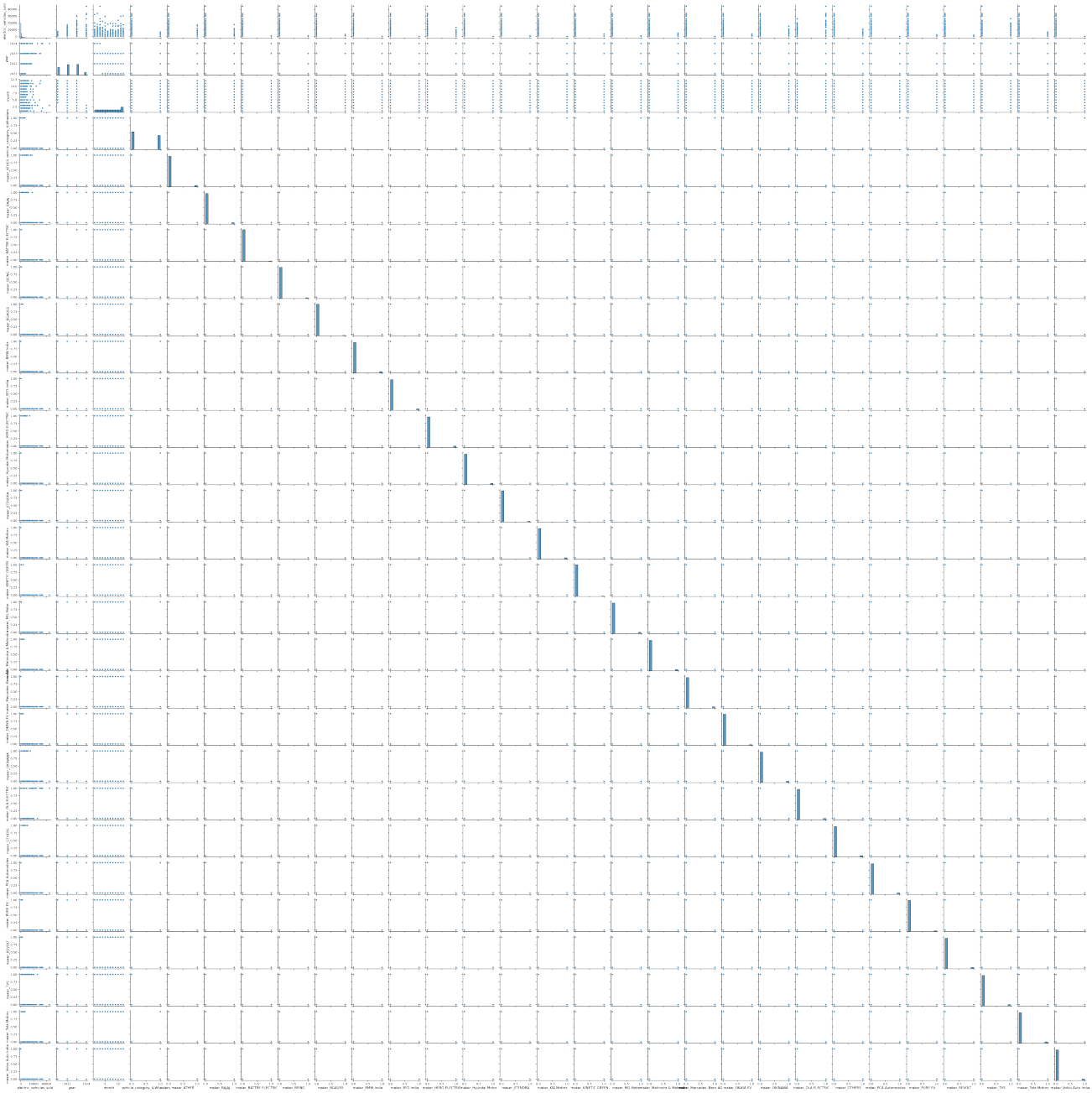
816 rows × 30 columns

In [21]:
```
#--Using seaborn library and plot the heatmap,,using heatmap() method
#sns.heatmap(corelation,xticklabels=corelation.columns,yticklabels=corelation.columns,

plt.figure(figsize=(20,12))
sns.set_context('notebook',font_scale = 1.3)
sns.heatmap(vehicle.corr(),annot=True,linewidth =2)
plt.tight_layout()
```



In [22]:
```
#--Weather checking all data row/columns relationship.

sns.pairplot(vehicle,hue_order=None,height=2.5)
```

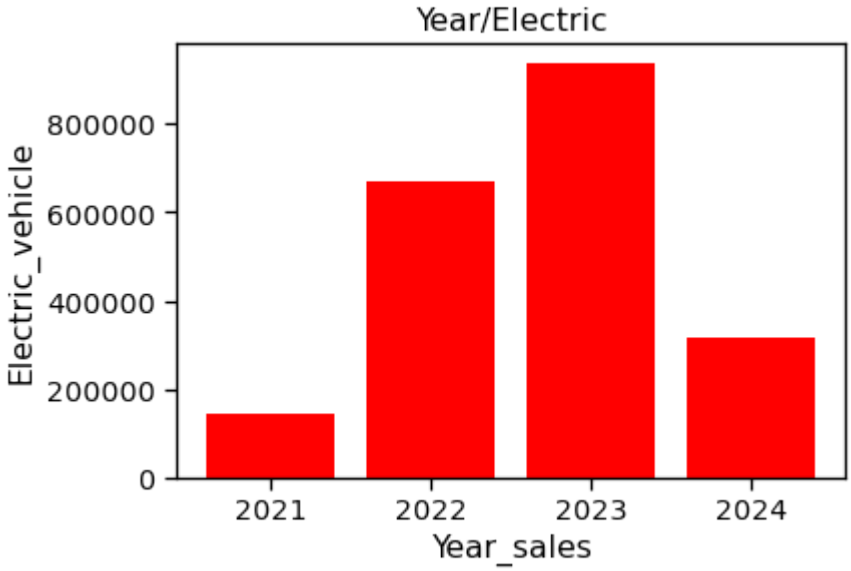
Out[22]:
```
<seaborn.axisgrid.PairGrid at 0x2371e9f7940>
```



In [23]:
```
#-Wather checking year vise electric_vehicle_sold.

sales_per_year = vehicle.groupby('year')['electric_vehicles_sold'].sum()
sales_per_year
```

Out[23]:
```
year
2021    143189
2022    669260
2023    936957
2024    316705
Name: electric_vehicles_sold, dtype: int64
```

In [24]:
```
#--Plot the graph year b/w electric_vehicle_sold

plt.bar(sales_per_year.index, sales_per_year.values, color='red',bottom=None,data=Non

plt.xlabel("Year_sales")
plt.ylabel("Electric_vehicle")
plt.title("Year/Electric")

plt.show()
```

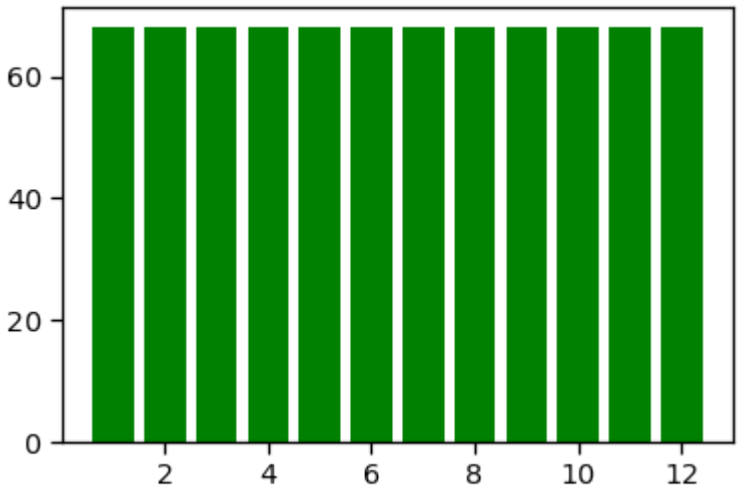


In [25]:
```
#--Plot the graph month b/w electric_vehicle_sold
sales_per_month = vehicle.groupby('month')['electric_vehicles_sold'].count()
sales_per_month
```

Out[25]:
```
month
1     68
2     68
3     68
4     68
5     68
6     68
7     68
8     68
9     68
10    68
11    68
12    68
Name: electric_vehicles_sold, dtype: int64
```

In [26]:
```
#--Plot the graph Month b/w electric_vehicle_sold

plt.bar(sales_per_month.index, sales_per_month.values, color='green', bottom=None, da
```

Out[26]:
```
<BarContainer object of 12 artists>
```



In [27]:
```
# Weather checking columns names

print(vehicle.columns)
```

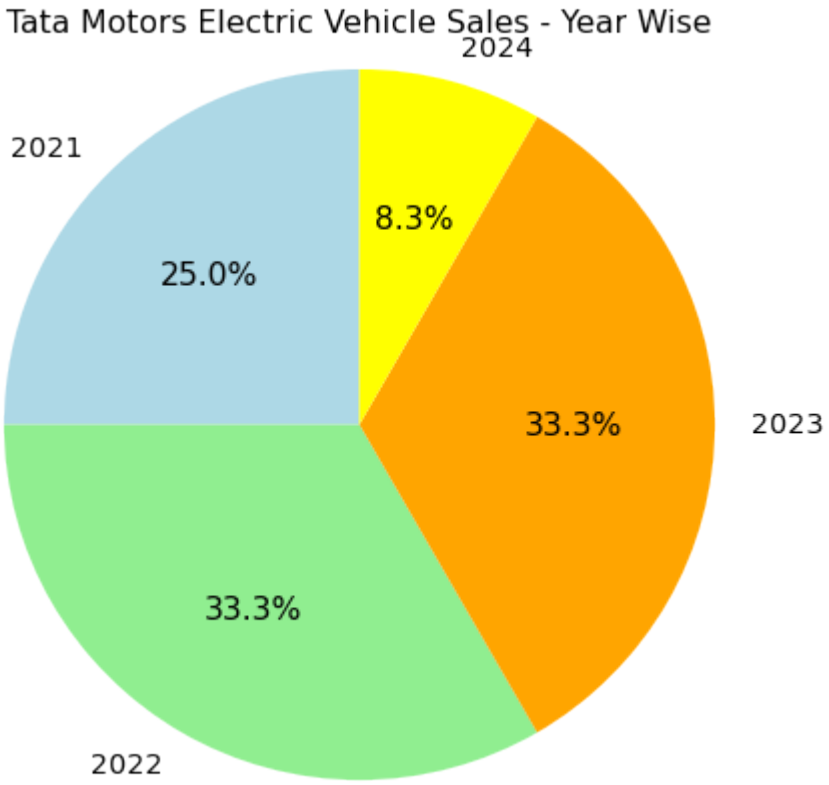
```
Index(['date', 'electric_vehicles_sold', 'year', 'month',
       'vehicle_category_4-Wheelers', 'maker_ATHER', 'maker_BAJAJ',
       'maker_BATTRE ELECTRIC', 'maker_BEING', 'maker_BGAUSS',
       'maker_BMW India', 'maker_BYD India', 'maker_HERO ELECTRIC',
       'maker_Hyundai Motor', 'maker_JITENDRA', 'maker_KIA Motors',
       'maker_KINETIC GREEN', 'maker_MG Motor', 'maker_Mahindra & Mahindra',
       'maker_Mercedes -Benz AG', 'maker_OKAYA EV', 'maker_OKINAWA',
       'maker_OLA ELECTRIC', 'maker_OTHERS', 'maker_PCA Automobiles',
       'maker_PURE EV', 'maker_REVOLT', 'maker_TVS', 'maker_Tata Motors',
       'maker_Volvo Auto India'],
      dtype='object')
```

In [28]:
```
#-Filter the data for tata moters.

tata_sales = vehicle[['year', 'maker_Tata Motors']]

# Group by 'year' and sum the Tata Motors electric vehicle sales.

tata_sales_per_year = tata_sales.groupby('year')['maker_Tata Motors'].sum()
```

In [29]:
```
tata_sales_per_year
```

Out[29]:
```
year
2021     9
2022    12
2023    12
2024     3
Name: maker_Tata Motors, dtype: uint8
```

In [30]:
```
# Plot a pie chart

plt.figure(figsize=(7, 7))
plt.pie(tata_sales_per_year, labels=tata_sales_per_year.index, autopct='%1.1f%%', sta

# Add title.

plt.title('Tata Motors Electric Vehicle Sales - Year Wise')

# Equal aspect ratio ensures the pie chart is circular.

plt.axis('equal')

# Show the plot
plt.show()
```



In [31]:
```
# Group the data by 'year' and sum the Tata Motors electric vehicle sales.

tata_sales_per_year = vehicle.groupby('year')['maker_Tata Motors'].sum().reset_index(
```

In [32]:
```
tata_sales_per_year
```

Out[32]:

	year	maker_Tata Motors
0	2021	9
1	2022	12
2	2023	12
3	2024	3

Define Target Variable and divide into train/test.

In [33]:
```
# Features (year) and target (sales).

X = tata_sales_per_year[['year']]
y = tata_sales_per_year['maker_Tata Motors']
```

In [34]:
```
# Split the data into training and test sets (we can skip this if using all data).

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state
```

In [35]:
```
# Initialize and fit the Linear Regression model.

model = LinearRegression()
model.fit(X, y)
```

Out[35]:
```
LinearRegression()
```

Predicting Scanrio.

In [36]:
```
# Predict future sales (for example, we can predict for 2025, 2026, and beyond).

future_years = np.array([[2025], [2026], [2027], [2028], [2029]])
predicted_sales = model.predict(future_years)
```

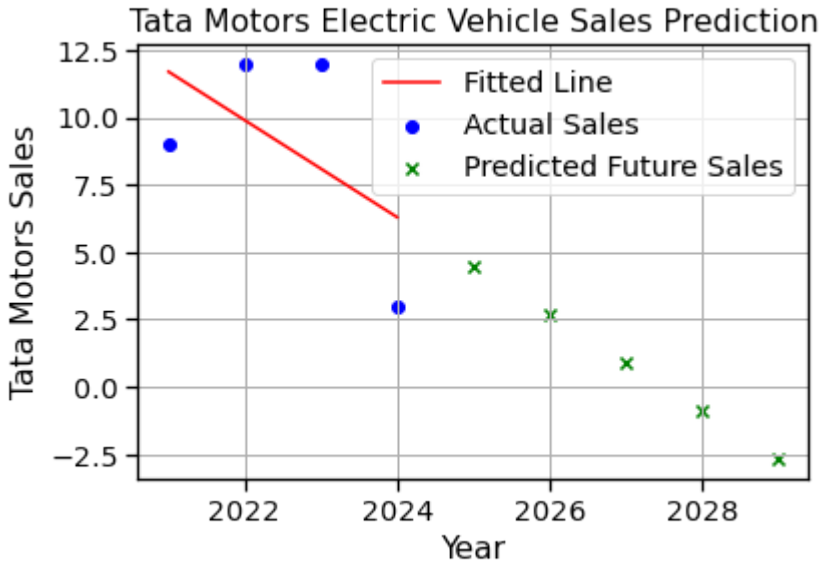
In [37]:
```
predicted_sales
```

Out[37]:
```
array([ 4.5,  2.7,  0.9, -0.9, -2.7])
```

In [38]:
```
# Print predictions for future years.

print("Predicted Tata Motors Sales:")
for year, sale in zip(future_years.flatten(), predicted_sales):
    print(f"Year: {year}, Predicted Sales: {sale:.2f}")
```

```
Predicted Tata Motors Sales:
Year: 2025, Predicted Sales: 4.50
Year: 2026, Predicted Sales: 2.70
Year: 2027, Predicted Sales: 0.90
Year: 2028, Predicted Sales: -0.90
Year: 2029, Predicted Sales: -2.70
```

In [39]:
```
# Visualize the results.

plt.scatter(X, y, color='blue', label='Actual Sales')
plt.plot(X, model.predict(X), color='red', label='Fitted Line')
plt.scatter(future_years, predicted_sales, color='green', marker='x', label='Predicte

# Add labels and title.
plt.xlabel('Year')
plt.ylabel('Tata Motors Sales')
plt.title('Tata Motors Electric Vehicle Sales Prediction')
plt.legend()
plt.grid(True)

# Show the plot
plt.show()



#-The blue dots represent the actual sales data.
#-The red line is the fitted linear regression line.
#-The green "x" marks are the predicted sales for future years.
```



In [ ]:

In [ ]: