Data analytics using Python is a powerful approach for extracting insights and knowledge from data. Python, with its rich ecosystem of libraries and tools, is particularly suited for data analysis, visualization, and machine learning. Here is an overview of the key steps and libraries involved in data analytics using Python:

1 Key Steps in Data Analytics

1. **Data Collection**: Gathering data from various sources such as databases, web scraping, APIs, or CSV files.
2. **Data Cleaning**: Handling missing values, removing duplicates, and correcting data types.
3. **Data Exploration**: Summarizing the main characteristics of the data using descriptive statistics and visualization.
4. **Data Analysis**: Applying statistical techniques and machine learning models to analyze data.
5. **Data Visualization**: Creating visual representations of data to communicate findings effectively.
6. **Reporting**: Documenting the analysis process and results.

2 Essential Python Libraries for Data Analytics

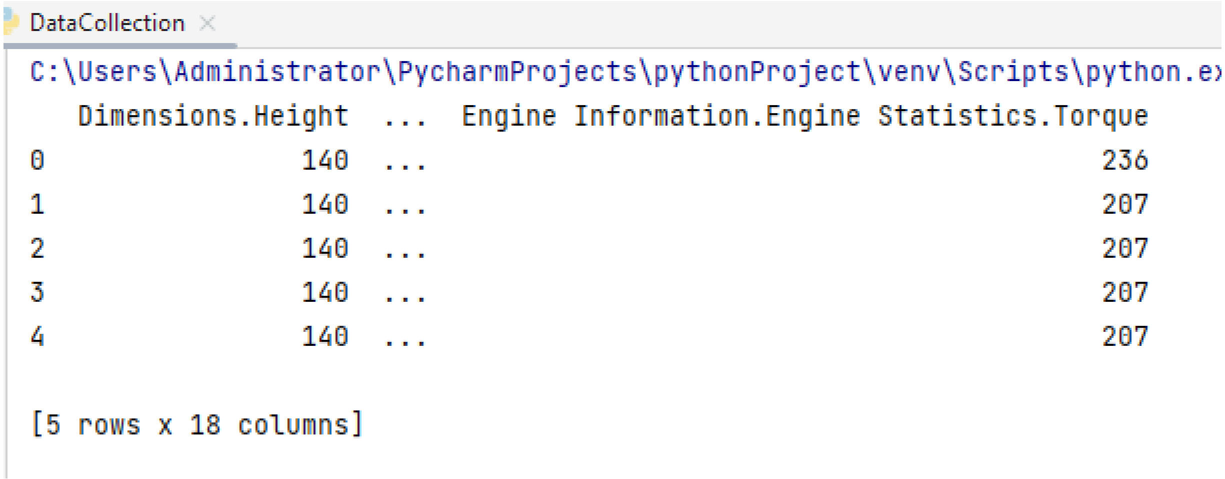
1. **NumPy**: A fundamental package for numerical computations.
2. **Pandas**: A powerful library for data manipulation and analysis.
3. **Matplotlib**: A library for creating static, animated, and interactive visualizations.
4. **Seaborn**: A statistical data visualization library based on Matplotlib.
5. **SciPy**: A library for scientific and technical computing.
6. **Scikit-learn**: A machine learning library for predictive data analysis.
7. **Statsmodels**: A library for statistical modeling.

3 Sample Workflow

4 1. Data Collection

Let's start by loading a dataset using Pandas.

```
import pandas as pd



# Load dataset

data = pd.read_csv('your_dataset.csv')



# Display the first few rows of the dataset

print(data.head())
```

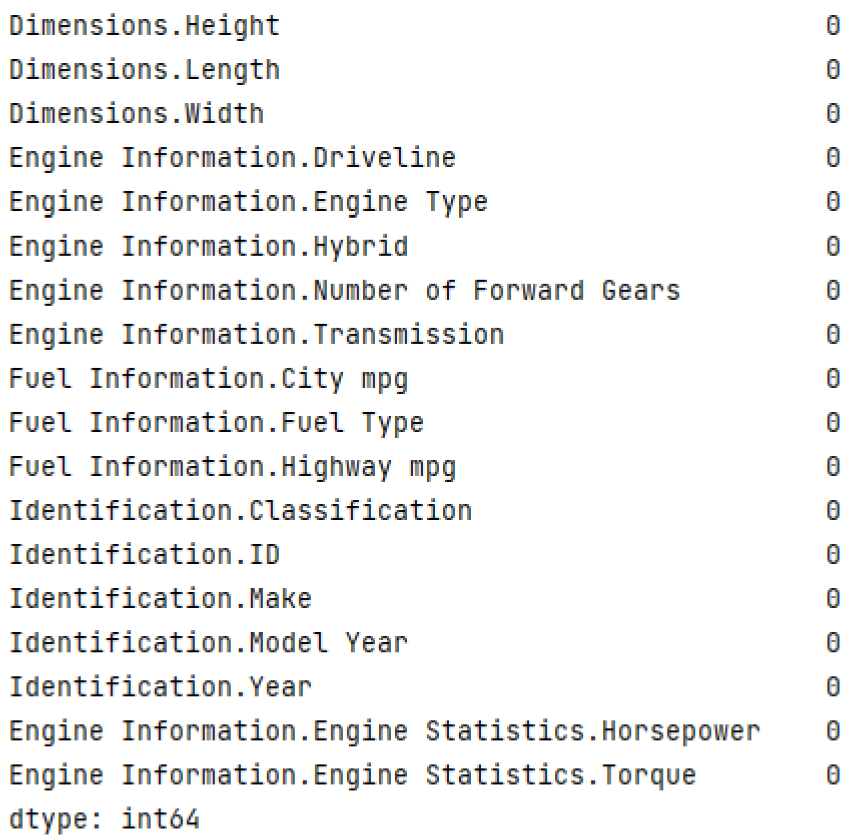


```
C:\Users\Administrator\PycharmProjects\pythonProject\venv\Scripts\python.ex
   Dimensions.Height  ...  Engine Information.Engine Statistics.Torque
0                140  ...                                          236
1                140  ...                                          207
2                140  ...                                          207
3                140  ...                                          207
4                140  ...                                          207

[5 rows x 18 columns]
```

2. Data Cleaning

Handle missing values and data types.

```
# Check for missing values

print(data.isnull().sum())



# Fill missing values or drop rows/columns

data = data.fillna(method='ffill')



# Convert data types if necessary

data['column_name'] = data['column_name'].astype('int')
```



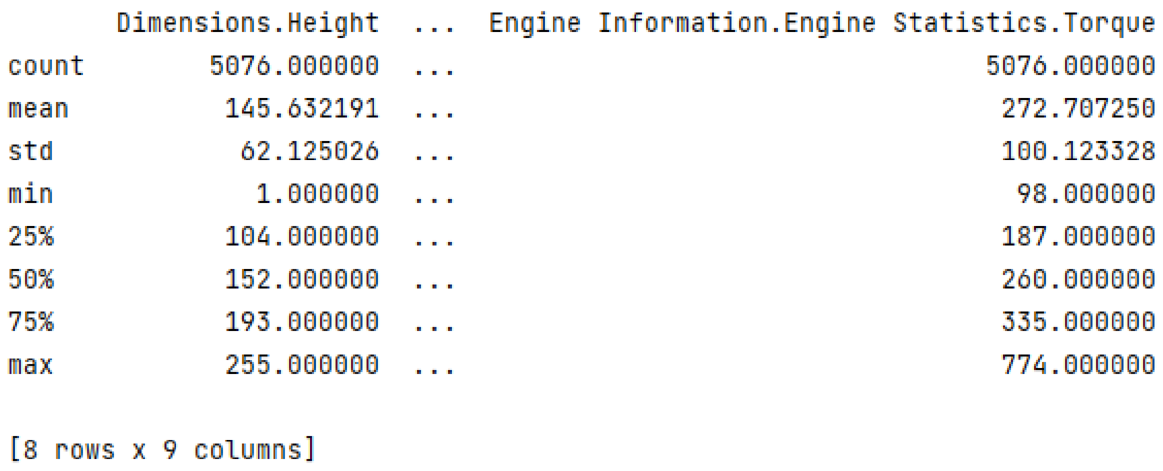
```
Dimensions.Height                                  0
Dimensions.Length                                  0
Dimensions.Width                                   0
Engine Information.Driveline                       0
Engine Information.Engine Type                     0
Engine Information.Hybrid                          0
Engine Information.Number of Forward Gears         0
Engine Information.Transmission                    0
Fuel Information.City mpg                          0
Fuel Information.Fuel Type                         0
Fuel Information.Highway mpg                       0
Identification.Classification                      0
Identification.ID                                  0
Identification.Make                                0
Identification.Model Year                          0
Identification.Year                                0
Engine Information.Engine Statistics.Horsepower    0
Engine Information.Engine Statistics.Torque        0
dtype: int64
```

3. Data Exploration

Generate summary statistics and visualize the data.

# Summary statistics

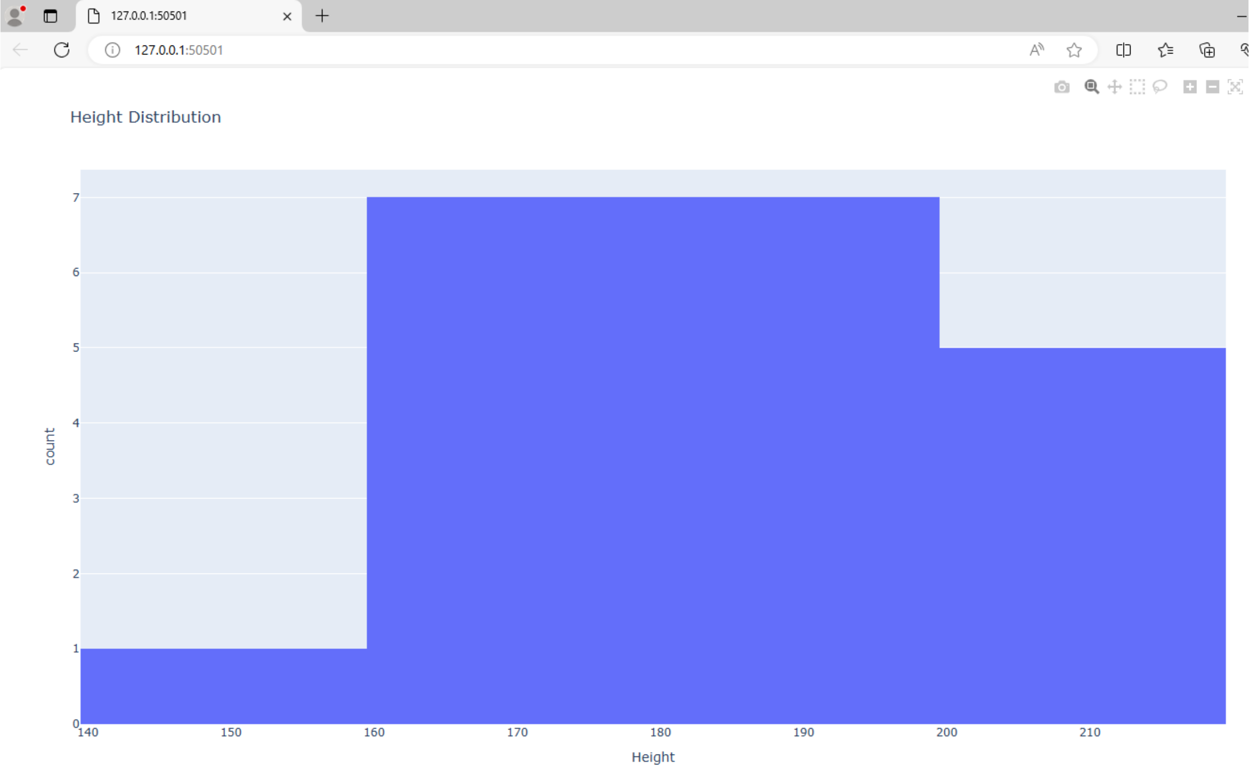
print(data.describe())



```
       Dimensions.Height  ...  Engine Information.Engine Statistics.Torque
count        5076.000000  ...                                  5076.000000
mean          145.632191  ...                                   272.707250
std            62.125026  ...                                   100.123328
min             1.000000  ...                                    98.000000
25%           104.000000  ...                                   187.000000
50%           152.000000  ...                                   260.000000
75%           193.000000  ...                                   335.000000
max           255.000000  ...                                   774.000000

[8 rows x 9 columns]
```

# Visualize data distribution

```
import plotly.express as px
import pandas as pd

# Sample data
data = pd.DataFrame({
    'Height': [150, 160, 170, 180, 190, 200, 210, 160, 170, 180, 190, 200,
175, 185, 195, 205, 215, 165, 175, 185]
})

# Create histogram
fig = px.histogram(data, x='Height', title='Height Distribution')

# Display the plot
fig.show()
```

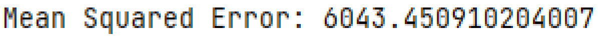


4. Data Analysis

Apply machine learning models using Scikit-learn.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

```
# Load data from CSV file
data = pd.read_csv('cars.csv')  # Replace 'your_file.csv' with the path to
your CSV file


# Assuming the CSV has columns named 'Dimensions.Height',
'Dimensions.Length', and 'Dimensions.Width'
# If the columns have different names, adjust the column names accordingly
X = data[['Dimensions.Height', 'Dimensions.Length']]
y = data['Dimensions.Width']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Train a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
```

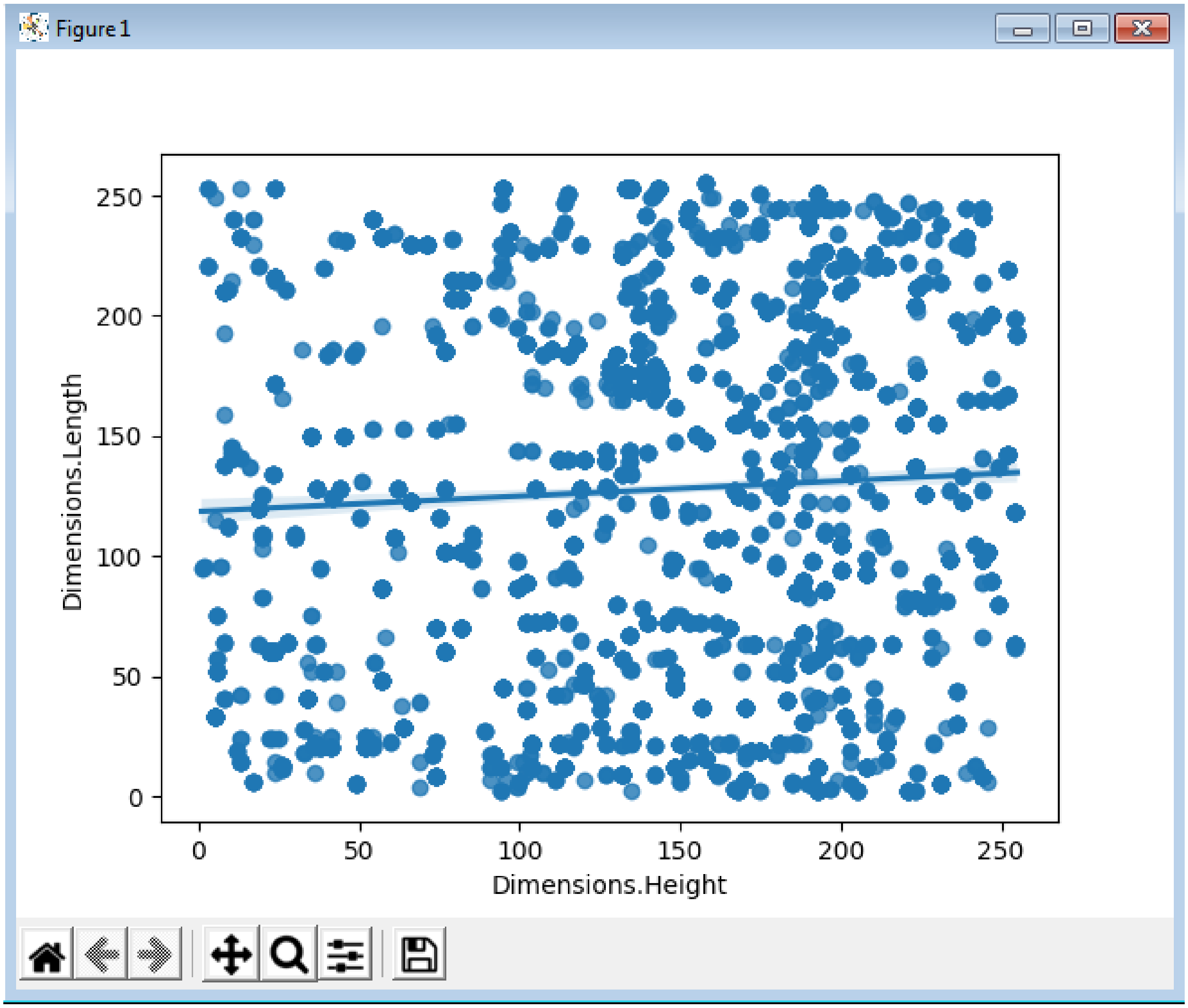


```
Mean Squared Error: 6043.450910204007
```

5. Data Visualization

Create more detailed visualizations.

```
import seaborn as sns
import matplotlib.pyplot as plt
# Scatter plot with regression line
sns.regplot(x='Dimensions.Height', y='Dimensions.Length', data=data)
plt.show()
```



6. Reporting

Summarize the findings in a report.

report = f"""

Data Summary:

{data.describe()}

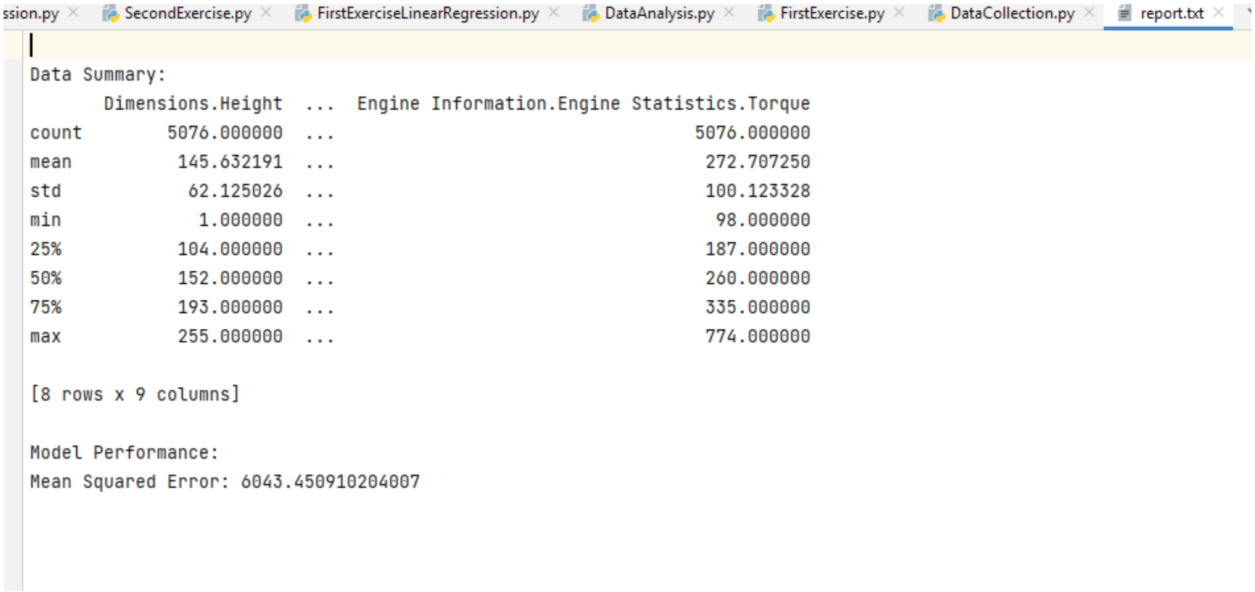
Model Performance:

Mean Squared Error: {mse}

"""

with open('report.txt', 'w') as file:

file.write(report)

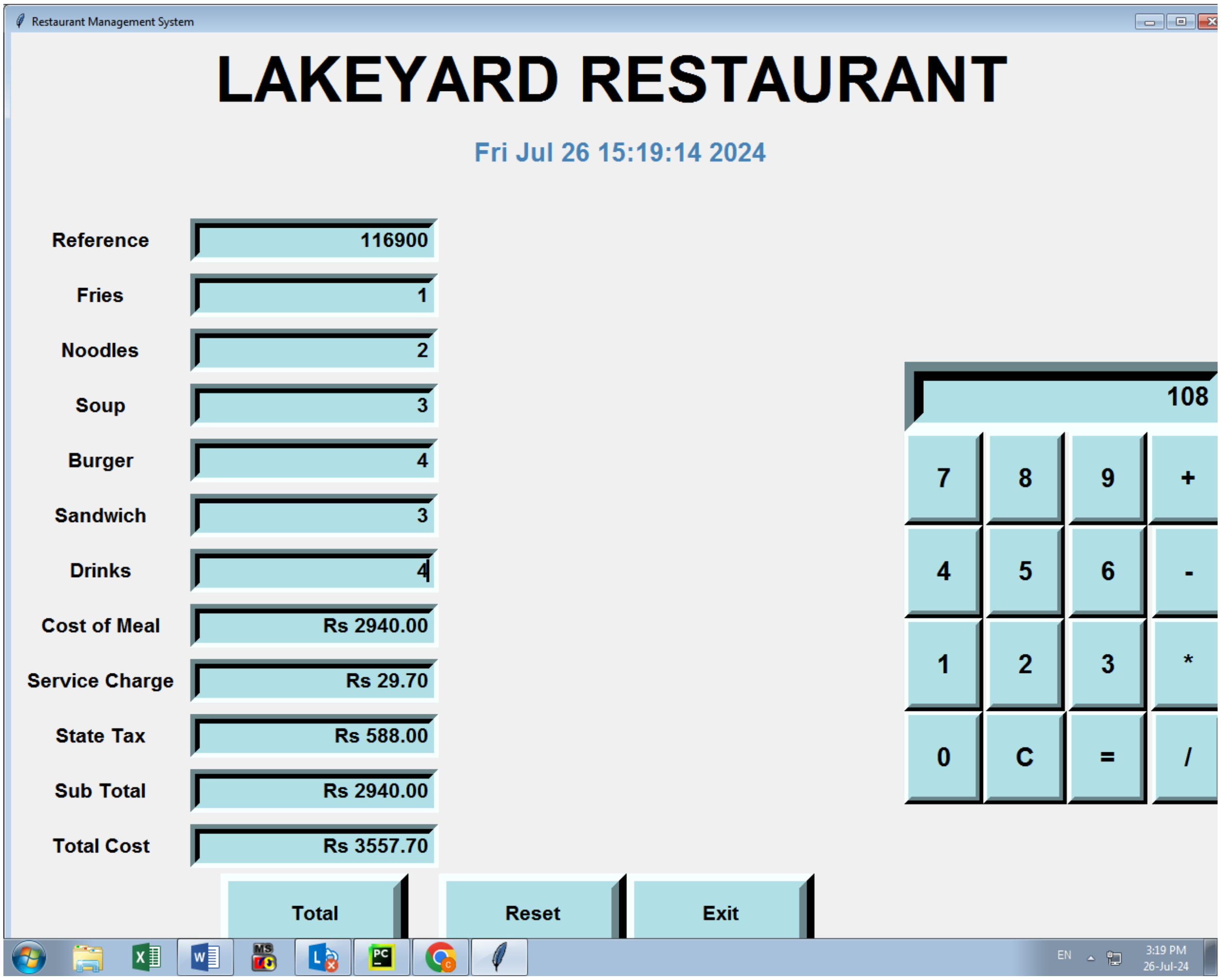


```
Data Summary:
       Dimensions.Height  ...  Engine Information.Engine Statistics.Torque
count        5076.000000  ...                                  5076.000000
mean          145.632191  ...                                   272.707250
std            62.125026  ...                                   100.123328
min             1.000000  ...                                    98.000000
25%           104.000000  ...                                   187.000000
50%           152.000000  ...                                   260.000000
75%           193.000000  ...                                   335.000000
max           255.000000  ...                                   774.000000

[8 rows x 9 columns]

Model Performance:
Mean Squared Error: 6043.450910204007
```

```
from tkinter import *
import random
import time

root = Tk()
root.geometry("1600x8000")
root.title("Restaurant Management System")

# Create frames
Tops = Frame(root, width=1600, relief=SUNKEN)
Tops.pack(side=TOP)

f1 = Frame(root, width=800, height=700, relief=SUNKEN)
f1.pack(side=LEFT)

f2 = Frame(root, width=300, height=700, relief=SUNKEN)
f2.pack(side=RIGHT)

# Display time
localtime = time.asctime(time.localtime(time.time()))

lblInfo = Label(Tops, font=('helvetica', 50, 'bold'), text="LAKEYARD RESTAURANT ", fg="Black", bd=10, anchor='w')
lblInfo.grid(row=0, column=0)

lblInfo = Label(Tops, font=('arial', 20, 'bold'), text=localtime, fg="Steel Blue", bd=10, anchor='w')
lblInfo.grid(row=1, column=0)

# Variables
rand = StringVar()
```

```
Fries = StringVar()
Noodles = StringVar()
Soup = StringVar()
SubTotal = StringVar()
Total = StringVar()
Service_Charge = StringVar()
Drinks = StringVar()
Tax = StringVar()
Cost = StringVar()
Burger = StringVar()
Sandwich = StringVar()

# Calculator variables
text_input = StringVar()
operator = ""

# Functions
def Ref():
    x = random.randint(10908, 500876)
    randomRef = str(x)
    rand.set(randomRef)

    CoFries = float(Fries.get()) if Fries.get() else 0
    CoNoodles = float(Noodles.get()) if Noodles.get() else 0
    CoSoup = float(Soup.get()) if Soup.get() else 0
    CoBurger = float(Burger.get()) if Burger.get() else 0
    CoSandwich = float(Sandwich.get()) if Sandwich.get() else 0
    CoD = float(Drinks.get()) if Drinks.get() else 0

    CostofFries = CoFries * 140
    CostofDrinks = CoD * 65
    CostofNoodles = CoNoodles * 90
    CostofSoup = CoSoup * 140
    CostBurger = CoBurger * 260
    CostSandwich = CoSandwich * 300

    TotalCost = CostofFries + CostofDrinks + CostofNoodles + CostofSoup +
CostBurger + CostSandwich
    PayTax = TotalCost * 0.2
    Ser_Charge = TotalCost / 99

    CostofMeal = "Rs", str('%.2f' % TotalCost)
    Service = "Rs", str('%.2f' % Ser_Charge)
    OverAllCost = "Rs", str('%.2f' % (TotalCost + PayTax + Ser_Charge))
    PaidTax = "Rs", str('%.2f' % PayTax)

    Service_Charge.set(Service)
    Cost.set(CostofMeal)
    Tax.set(PaidTax)
    SubTotal.set(CostofMeal)
    Total.set(OverAllCost)


def qExit():
    root.destroy()
```

```
def Reset():
    rand.set("")
    Fries.set("")
    Noodles.set("")
    Soup.set("")
    SubTotal.set("")
    Total.set("")
    Service_Charge.set("")
    Drinks.set("")
    Tax.set("")
    Cost.set("")
    Burger.set("")
    Sandwich.set("")


# Calculator functions
def btnClick(numbers):
    global operator
    operator = operator + str(numbers)
    text_input.set(operator)


def btnClearDisplay():
    global operator
    operator = ""
    text_input.set("")


def btnEqualsInput():
    global operator
    try:
        sumup = str(eval(operator))
        text_input.set(sumup)
        operator = ""
    except:
        text_input.set("ERROR")
        operator = ""


# Create labels and entries for each menu item and cost details
labels = [
    ("Reference", rand),
    ("Fries", Fries),
    ("Noodles", Noodles),
    ("Soup", Soup),
    ("Burger", Burger),
    ("Sandwich", Sandwich),
    ("Drinks", Drinks),
    ("Cost of Meal", Cost),
    ("Service Charge", Service_Charge),
    ("State Tax", Tax),
    ("Sub Total", SubTotal),
    ("Total Cost", Total)
]

for i, (text, var) in enumerate(labels):
    lbl = Label(f1, font=('arial', 16, 'bold'), text=text, bd=16, anchor="w")
    lbl.grid(row=i, column=0)
    txt = Entry(f1, font=('arial', 16, 'bold'), textvariable=var, bd=10,
insertwidth=4, bg="powder blue", justify='right')
    txt.grid(row=i, column=1)

# Buttons for calculating, resetting, and exiting
btnTotal = Button(f1, padx=16, pady=8, bd=16, fg="black", font=('arial', 16,
'bold'), width=10, text="Total",
                  bg="powder blue", command=Ref)
btnTotal.grid(row=13, column=1)

btnReset = Button(f1, padx=16, pady=8, bd=16, fg="black", font=('arial', 16,
'bold'), width=10, text="Reset",
                  bg="powder blue", command=Reset)
btnReset.grid(row=13, column=2)

btnExit = Button(f1, padx=16, pady=8, bd=16, fg="black", font=('arial', 16,
'bold'), width=10, text="Exit",
                 bg="powder blue", command=qExit)
btnExit.grid(row=13, column=3)

# Calculator display
txtDisplay = Entry(f2, font=('arial', 20, 'bold'), textvariable=text_input,
bd=20, insertwidth=4,
                   bg="powder blue", justify='right')
txtDisplay.grid(columnspan=4)

# Calculator buttons
calculator_buttons = [
    ('7', 1, 0), ('8', 1, 1), ('9', 1, 2), ('+', 1, 3),
    ('4', 2, 0), ('5', 2, 1), ('6', 2, 2), ('-', 2, 3),
    ('1', 3, 0), ('2', 3, 1), ('3', 3, 2), ('*', 3, 3),
    ('0', 4, 0), ('C', 4, 1), ('=', 4, 2), ('/', 4, 3),
]

for (text, row, col) in calculator_buttons:
    Button(f2, padx=16, pady=16, bd=8, fg="black", font=('arial', 20,
'bold'), text=text, bg="powder blue",
           command=lambda t=text: btnClick(t) if t not in ('C', '=') else
btnClearDisplay() if t == 'C' else btnEqualsInput()).grid(row=row,
column=col)

root.mainloop()
```

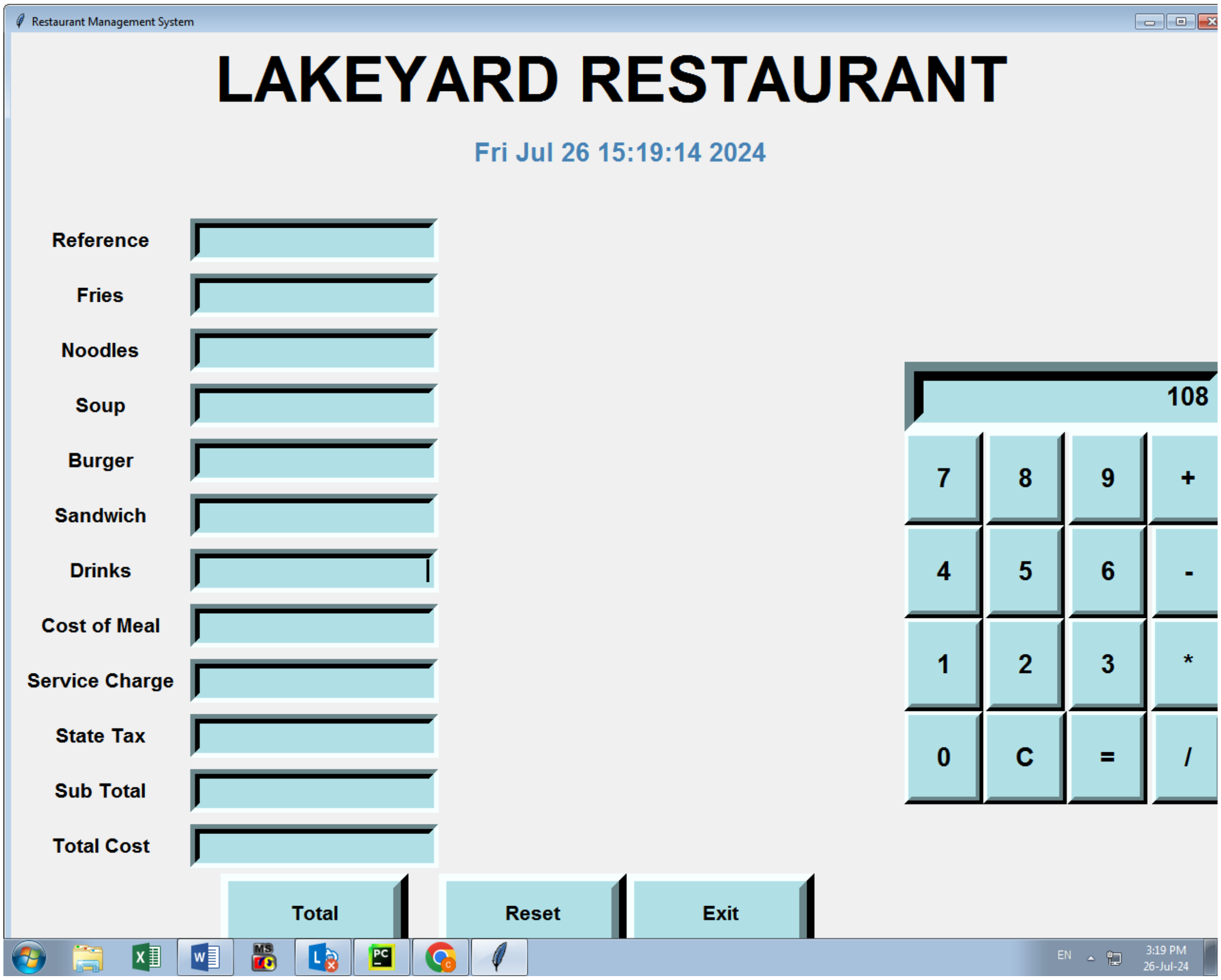


LAKEYARD RESTAURANT

Fri Jul 26 15:19:14 2024

Reference	116900
Fries	1
Noodles	2
Soup	3
Burger	4
Sandwich	3
Drinks	4
Cost of Meal	Rs 2940.00
Service Charge	Rs 29.70
State Tax	Rs 588.00
Sub Total	Rs 2940.00
Total Cost	Rs 3557.70

108



Restaurant Management System

LAKEYARD RESTAURANT

Fri Jul 26 15:19:14 2024