Q1. Apply regular expression for form validation. Create your domain-form using Tkinter Module. ➔ Form should contain Text box [For Name, Email Id, Phone number], Dropdown [for Gender], Spinbox [for Year/DoB] and other necessary widgets required for your domain. ➔ Validate Your Name, Email Id, Phone number in the form.

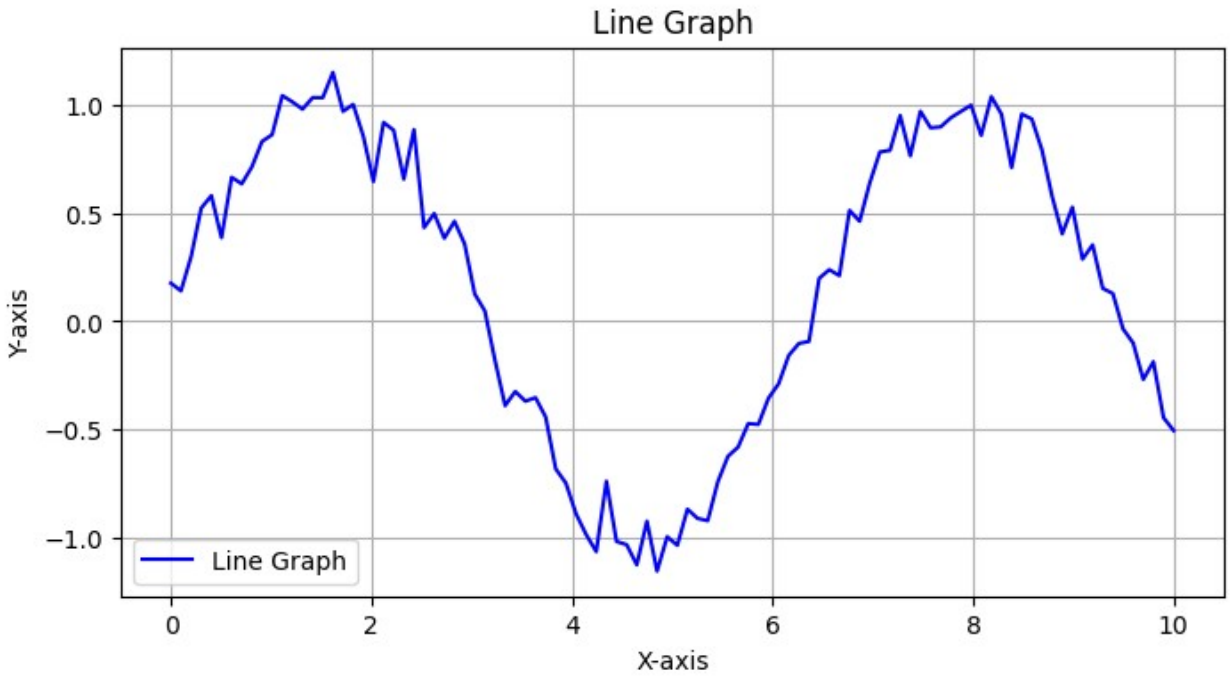
```
import tkinter as tk
from tkinter import ttk
import re

# Function to validate input fields
def validate():
    name = name_entry.get()
    email = email_entry.get()
    phone = phone_entry.get()
    gender = gender_combo.get()
    dob = dob_spinbox.get()

    # Regular expressions for validation
    name_pattern = r"^[A-Za-z\s]+$"
    email_pattern = r"^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+
$"
    phone_pattern = r"^[0-9]{10}$"

    if not re.match(name_pattern, name):
        result_label.config(text="Invalid Name", fg="red")
    elif not re.match(email_pattern, email):
        result_label.config(text="Invalid Email ID", fg="red")
    elif not re.match(phone_pattern, phone):
        result_label.config(text="Invalid Phone Number", fg="red")
    else:
        result_label.config(text=f"Form submitted successfully!\nName:
{name}\nEmail: {email}\nPhone: {phone}\nGender: {gender}\nDOB: {dob}",
fg="green")

# Create the main window
root = tk.Tk()
root.title("Railway Management Form")

# Name
name_label = tk.Label(root, text="Name:")
name_label.pack()
name_entry = tk.Entry(root)
name_entry.pack()

# Email
email_label = tk.Label(root, text="Email ID:")
email_label.pack()
email_entry = tk.Entry(root)
email_entry.pack()
```

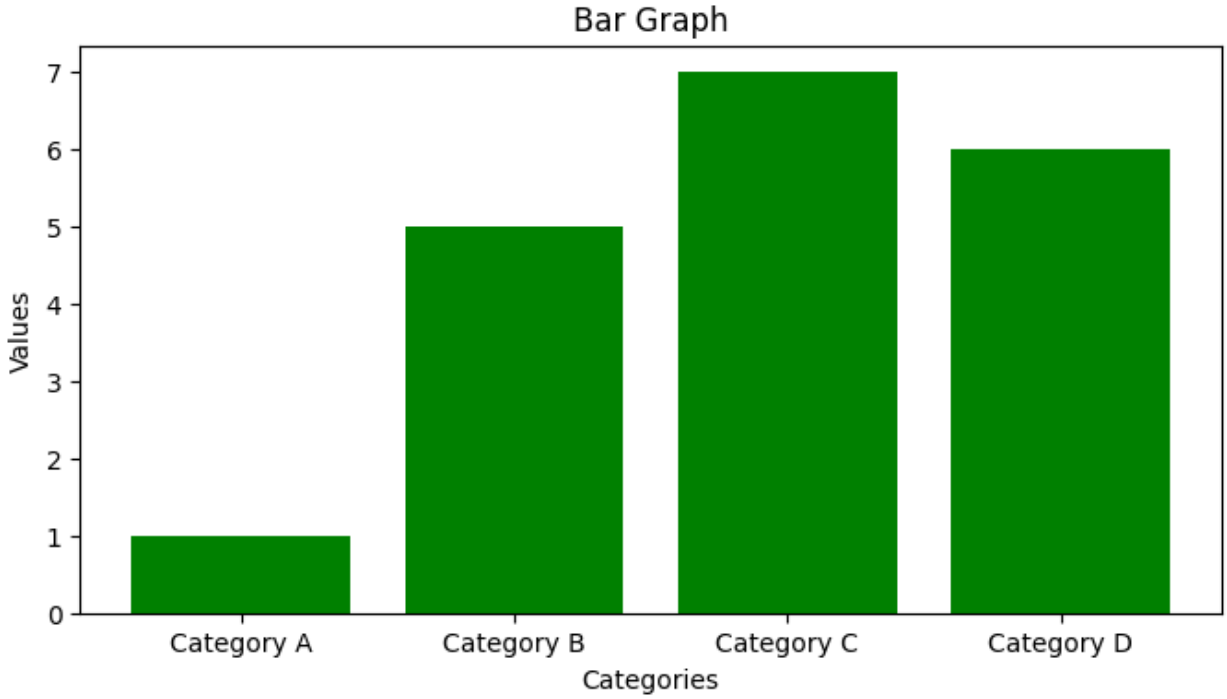
```
# Phone Number
phone_label = tk.Label(root, text="Phone Number:")
phone_label.pack()
phone_entry = tk.Entry(root)
phone_entry.pack()

# Gender
gender_label = tk.Label(root, text="Gender:")
gender_label.pack()
gender_var = tk.StringVar()
gender_combo = ttk.Combobox(root, textvariable=gender_var,
values=["Male", "Female", "Other"])
gender_combo.pack()

# Date of Birth (Spinbox)
dob_label = tk.Label(root, text="Date of Birth:")
dob_label.pack()
dob_spinbox = tk.Spinbox(root, from_=1, to=31, width=2)  # Adjust the
'from_' and 'to' values for your use case
dob_spinbox.pack()

# Submit Button
submit_button = tk.Button(root, text="Submit", command=validate)
submit_button.pack()

result_label = tk.Label(root, text="", fg="green")
result_label.pack()

root.mainloop()
```

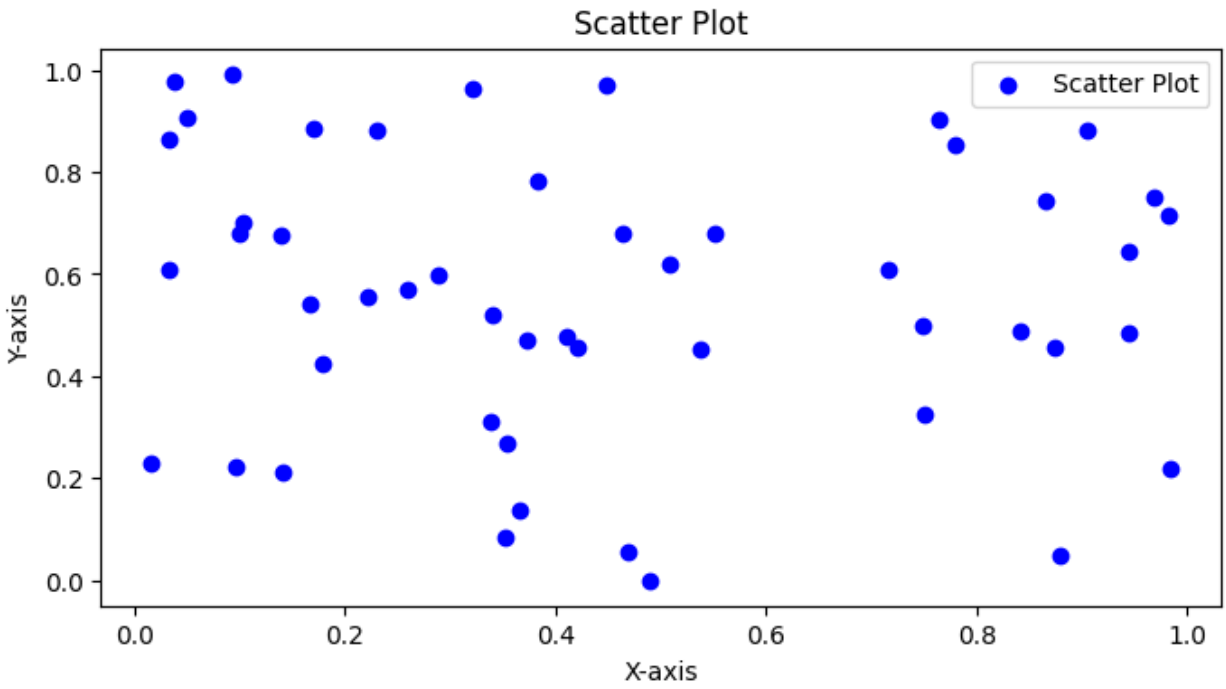
Q2. Perform the Exploratory Data Analysis on your domain-based dataset and demonstrate the retrieved insights using “Matplotlib” modules. Visualize hidden insights using appropriate plots (graphs) [Usage of line plot and scatter plot are mandatory]

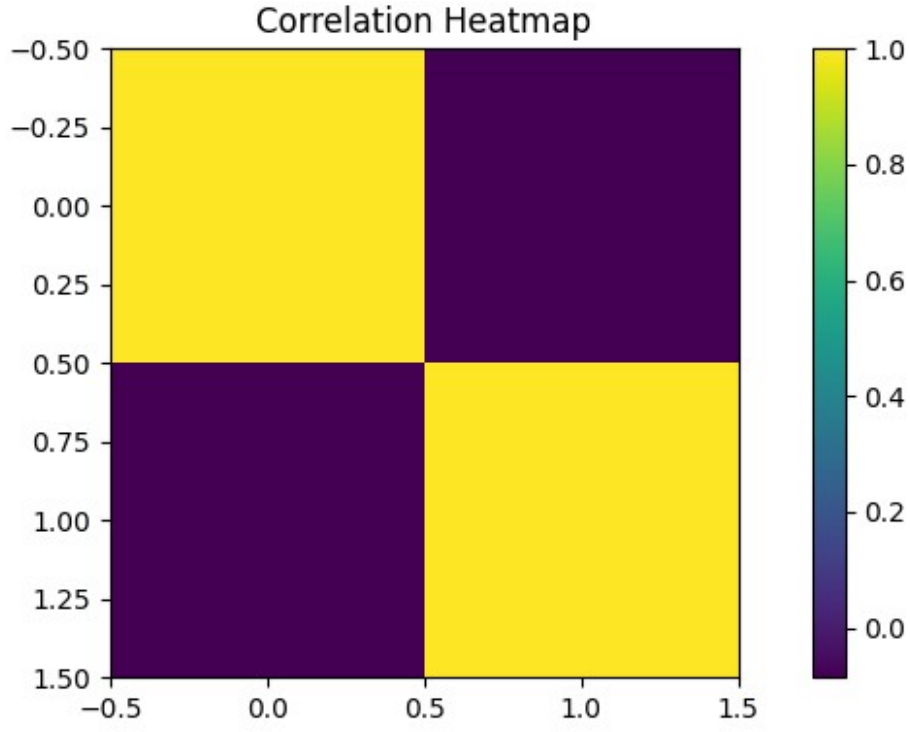
```
import matplotlib.pyplot as plt
import numpy as np

np.random.seed(0)
x = np.linspace(0, 10, 100)
y = np.sin(x) + np.random.normal(0, 0.1, 100)

# Line-Graph
plt.figure(figsize=(8, 4))
plt.plot(x, y, label="Line Graph", color="b")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Line Graph")
plt.legend()
plt.grid(True)
```

```
# Bar-Graph
categories = ["Category A", "Category B", "Category C", "Category D"]
values = np.random.randint(1, 10, len(categories))
plt.figure(figsize=(8, 4))
plt.bar(categories, values, color="g")
plt.xlabel("Categories")
plt.ylabel("Values")
plt.title("Bar Graph")

# Scatter-Plot
x_scatter = np.random.rand(50)
y_scatter = np.random.rand(50)
plt.figure(figsize=(8, 4))
plt.scatter(x_scatter, y_scatter, label="Scatter Plot", color="b",
marker="o")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Scatter Plot")
plt.legend()

# Correlation-Heatmap
correlation_matrix = np.corrcoef(x, y)
plt.figure(figsize=(8, 4))
plt.imshow(correlation_matrix, cmap="viridis", origin="upper")
plt.colorbar()
plt.title("Correlation Heatmap")

plt.tight_layout()
plt.show()
```


Line Graph
1.0
0.5
0.0
−0.5
−1.0
Y-axis
Line Graph
0
2
4
6
8
10
X-axis


Bar Graph
7
6
5
4
3
2
1
0
Values
Category A
Category B
Category C
Category D
Categories


Scatter Plot
Scatter Plot
Y-axis
X-axis
0.0
0.2
0.4
0.6
0.8
1.0


Correlation Heatmap
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50
−0.5
0.0
0.5
1.0
1.5
1.0
0.8
0.6
0.4
0.2
0.0