PROJECT SYNOPSIS

Graphical Data Representation System using Python and MySQL

Credits:
Raj Rishi Rana
Satyam Kumar

CERTIFICATE

This is to certify that Mr. <u>Raj Rishi Rana</u> of std. XII Navy Children School, Mumbai has put in sincere and dedicated efforts towards the project. His work has been highly appreciative.

_____   _____
Sign of Head of Department   **Sign of Principal**

_____
School Stamp

ACKNOWLEDGEMENT

Table of Contents

Software Specification

- **Programming Language**:

  - `Front-end` : **Python 3.13**

    Python is a high-level, versatile programming language known for its simplicity and readability, making it a popular choice for beginners and experts alike. Developed in the late 1980s by Guido van Rossum, Python emphasizes clear, concise syntax, allowing developers to focus on problem-solving rather than complex code structures. Its extensive standard library supports tasks ranging from web development and data analysis to artificial intelligence and scientific computing. Python's flexibility and adaptability stem from its compatibility with various libraries and frameworks, such as Django for web development, NumPy and Pandas for data manipulation, and TensorFlow for machine learning. With a large, supportive community and continuous development, Python has become a preferred language for a wide range of applications across industries, known for accelerating development and enhancing productivity.

  - `Back-end` : **MySQL 8.0**

    MySQL is a widely used open-source relational database management system (RDBMS) known for its speed, reliability, and flexibility. Developed in the mid-1990s by Swedish developers, MySQL quickly gained popularity due to its ease of use and powerful capabilities, especially in web applications and data management. MySQL is based on Structured Query Language (SQL), which allows users to create, manipulate, and manage databases effectively. It supports a wide range of platforms, including Windows, Linux, and macOS, and integrates seamlessly with popular programming languages like Python, PHP, and Java. MySQL is highly scalable, capable of handling large data volumes, and includes robust features for data security, transaction processing, and concurrency. Its popularity as the backbone of numerous web applications, including social media sites and e-commerce platforms, is supported by its active community and ongoing development under Oracle Corporation.

- **Modules Used**:

  - `tkinter`: For creating the graphical user interface.
  - `matplotlib.pyplot`: For plotting graphs.
  - `mysql.connector`: For connecting and querying MySQL databases.
  - `numpy`: For array handling.
  - `math`: For mathematical operations.

- **Database**: MySQL
- **IDE Used**: PyCharm (JetBrains IDEs)
- **Operating System**: Linux

Project Description

This project, **"Graphical Data Representation System using Python and MySQL"** aims to simplify **data visualization** for mathematical equations and database tables. The program uses a Tkinter GUI, where users can input either a mathematical equation or the name of a MySQL database and table. The program processes the input and generates a graphical representation of the data.

- **Objective**: To provide an easy-to-use tool for visualizing data and mathematical functions.
- **User Input Options**:
    - **Mathematical Equation**: The user enters a formula, and the program calculates values using `numpy` and plots the result.
    - **Database Table**: The user provides the name of a MySQL database and table, and the program retrieves data and plots it.

This system leverages Python's data processing and graphical capabilities to create a user-friendly experience for data visualization.

Implementation

1. Importing modules

```python
1    import tkinter as tk
2    import mysql.connector as ml
3    import matplotlib.pyplot as mlt
4    import numpy as np
5    from math import *
```

2. Initializing

```python
8    # creating tkinter window
9    root = tk.Tk()
10   root.geometry("300x300")
11   root.title("Visualizer")
12
13   # setting up connection between python and MySQL
14   cn = ml.connect(host='localhost', user='root', passwd='1234')
15   curr = cn.cursor()
```

3. Defining functions

```python
17   def start():    rajrish5541f
18       """Asking user what he wants to plot: Equation or Database"""
19       global what
20       what = tk.StringVar()
21       what.set("What do you want to plot?")
22       drop = tk.OptionMenu(root, what, value:"Mathematical Equation", *values:"MySQL database")
23       drop.pack()
24
25       tk.Button(root, text='Next', command=eqn_or_db).pack()
```

```python
27   def eqn_or_db():    rajrish5541f
28       # e1 is error1 which is encountered when user didn't select any option from the first drop-down menu
29       e1 = tk.Label(root, text="Please select an option from the drop down menu !!!")
30       if what.get()=="Mathematical Equation":
31           try:
32               e1.destroy()
33           except: pass
34
35           inp_eqn()
36       elif what.get()=="MySQL database":
37           try:
38               e1.destroy()
39           except: pass
40
41           fetch_dbs()
42       else:
43           e1.pack()
44
```

```python
45   def inp_eqn():    rajrish5541f
46       """This function is meant to input Mathematical Equation."""
47
48       # creating global variables
49       global eq, eq_st, eq_end
50       tk.Label(root, text='Enter equation:').pack()
51
52       # eq is the textbox where user will enter the equation
53       eq = tk.Entry(root, width=25)
54       eq.pack()
55       tk.Label(root, text='Define domain:').pack()
56
57       # domain is a tkinter frame which contains 2 labels and 2 textboxes
58       # where user will define the domain of the function
59       domain = tk.Frame(root)
60       tk.Label(domain, text='From: ').grid(row=0,column=0)
61       eq_st = tk.Entry(domain, width=5)
62       eq_st.grid(row=0,column=1)
63       tk.Label(domain, text='To: ').grid(row=0, column=2)
64       eq_end = tk.Entry(domain, width = 5)
65       eq_end.grid(row=0, column=3)
66       domain.pack()
67
68       tk.Button(root, text='Plot', command=plot_eqn).pack()
69
70   def plot_eqn():    rajrish5541f
71       """This function is meant to plot the equation retrieved by the inp_eqn() function."""
72
73       # setting up variables which store the equation and domain
74       eqn = eq.get()
75       eqn_st = int(eq_st.get())
76       eqn_end = int(eq_end.get())
77
78       # xval is the numpy array of all the x-coordinates
79       xval = np.arange((eqn_st), eqn_end, 0.1)
80       x = []
81
82       # There is possibility that the function is not defined for some values of x
83       # Hence a for loop is required to separate out those x-coordinates
84       # this loop will split xval array into smaller arrays which hold the x-coordinates where function is defined
85       for i in xval:
86           try:
87               y = eval(eqn.replace(_old:'x', _new:f"({i})"))
88               x.append(i)
89           except:
90               mlt.plot(*args:x,[eval(eqn.replace(_old:'x', _new:f"({i})")) for i in x], 'b-')
91               x=[]
92       mlt.plot(*args:x,[eval(eqn.replace(_old:'x', _new:f"({i})")) for i in x],'b-', label=eqn)
93
94       mlt.legend()
95       # Finally showing up the graph
96       mlt.show()
97
98   def fetch_dbs():    rajrish5541f
99       """This function is meant to only fetch names of databases and return the name of chosen database"""
100
101       # setting up global variable
102       global dbname
103
104       # executing MySQL query
105       curr.execute('show databases;')
106
107       # creating a tkinter StringVar which will hold the name of chosen database
108       dbname = tk.StringVar()
109       dbname.set("Select database name")
110
111       # fetching the records
112       db1 = curr.fetchall()
113       dbs = []     # this will hold names of databases
114
115       # this for loop will create the list which will hold the names of all the databases
116       for i in db1:
117           dbs.append(i[0])
118       drop = tk.OptionMenu(root, dbname, *values:*dbs)
119       drop.pack()
120
121       # once databases are fetched, tables will be fetched
122       tk.Button(root, text='Next', command=fetch_tbs).pack()
123
```

```python
def fetch_tbs():    rajrish5541
    """This function is used to fetch table names"""

    # Declaring global variables
    global tbname

    # creating a tkinter StringVar which will hold the name of chosen table
    tbname = tk.StringVar()
    tbname.set('Select table name')

    # executing MySQL queries
    curr.execute('use {};'.format(dbname.get()))
    curr.execute('show tables;')
    drop = tk.OptionMenu(root, tbname, *values: *[i[0] for i in curr.fetchall()])
    drop.pack()

    # if table names are fetched corretly then column names will be fetched
    tk.Button(root, text='Next', command=fetch_clms).pack()


def fetch_clms():    rajrish5541
    """This function will fetch column names"""

    # declaring global variables
    global x,y,tname
    tname = tbname.get()

    # executing MySQL query
    curr.execute('describe {};'.format(tbname.get()))
    all_clms = curr.fetchall()
    desirable = ['int', 'decimal', 'numeric', 'float', 'double']

    # useable_clms is a list of only those column names which can be plotted, i.e which are numeric
    useable_clms=[]
    for i in desirable:
        for j in all_clms:
            if i in j[1]:
                useable_clms.append(j[0])

    # select_clms is a tkinter frame which allows user to choose
    # which colums to plot on x-axis and which one on y-axis
    select_clms = tk.Frame(root)
    x = tk.StringVar()
    x.set('X-axis')
    y = tk.StringVar()
    y.set('Y-axis')
    drop1 = tk.OptionMenu(select_clms, x, *values: *useable_clms)
    drop1.grid(row=0,column=0)
    drop2 = tk.OptionMenu(select_clms, y, *values: *useable_clms)
    drop2.grid(row=0,column=1)
    select_clms.pack()

    # Once we get column names, we will plot the graph
    tk.Button(root, text='Plot', command=plot_db).pack()
```
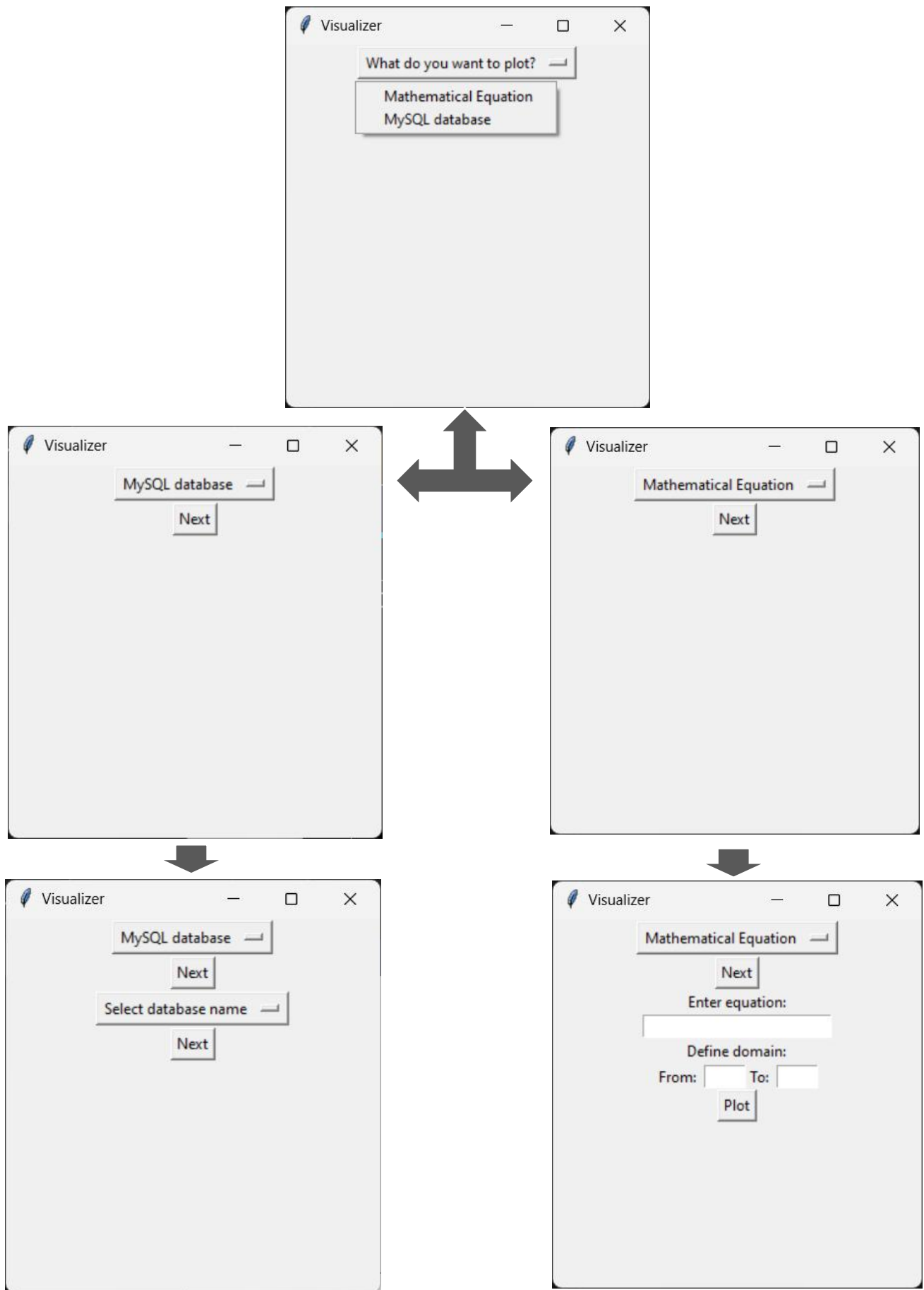
```python
178   v def plot_db():  ⚬ rajrish5541f
179       """This function is used to plot the graph of the db"""
180
181       # declaring variables
182       xaxis = x.get()
183       yaxis = y.get()
184
185       # executing query to get the records
186       curr.execute('select {},{} from {} order by {}'.format(*args: xaxis,yaxis,tname,xaxis))
187       entries = curr.fetchall()
188       xval, yval = [i[0] for i in entries], [i[1] for i in entries]
189
190       # This will plot the absolute coordinates
191       mlt.plot(*args: xval, yval, marker='o', linestyle='')
192
193       curr.execute('select {},{} from {} group by {} order by {}'.format(*args: xaxis,f'avg({yaxis})',tname,xaxis,xaxis))
194       entries = curr.fetchall()
195       xval, yval = [i[0] for i in entries], [i[1] for i in entries]
196
197       # This will plot the line of average values
198       mlt.plot(*args: xval,yval,'r--', label='Average')
199
200       # setting x and y lables for the graph
201       mlt.xlabel(xaxis)
202       mlt.ylabel(yaxis)
203
204       mlt.legend()
205       # showing the graph
206       mlt.show()
207
```

```python
208   start()
209
210   root.mainloop()
211
212   '''
213   @Credits:
214       ● RAJ RISHI RANA aka rajrish5541f
215       ● SATYAM KUMAR
216   '''
217   💡
```

Source code available at github.com -
CS-11-12/Projects/Project Synopsis at main · rajrish5541f/CS-11-12
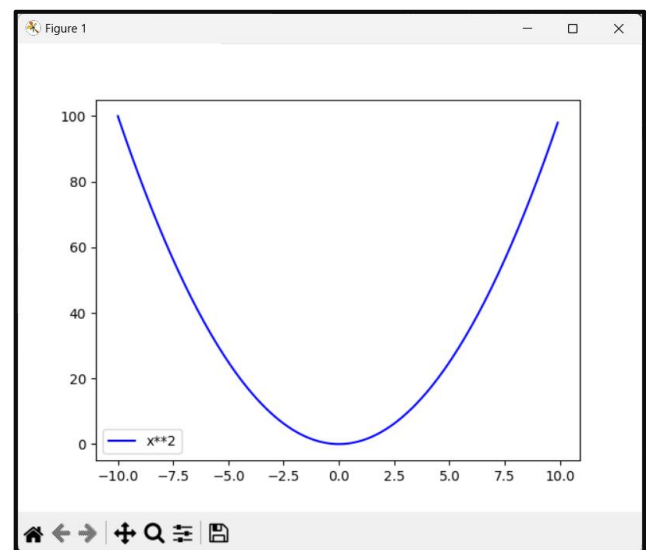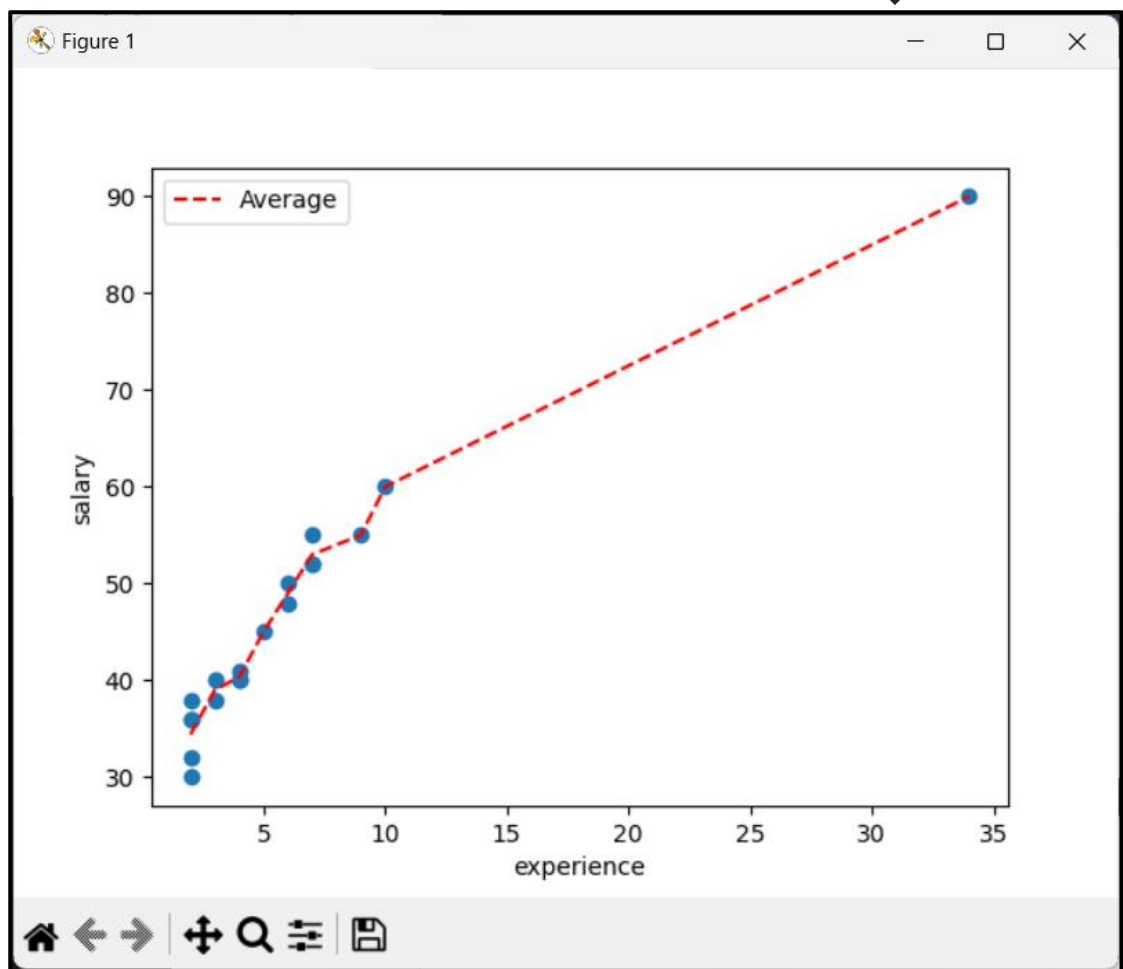
Bibliography

- **Python Documentation**: https://docs.python.org/
- **Matplotlib Documentation**: https://matplotlib.org/
- **MySQL Documentation**: https://dev.mysql.com/
- **Stack Overflow**: https://stackoverflow.com/
- **Youtube Handles**:
  - https://youtube.com/@freecodecamp/
  - https://youtube.com/@SurfsharkAcademy/
  - https://youtube.com/@Codemycom/
  - https://youtube.com/@CodeWithHarry/
  - https://youtube.com/@CodingWithLewis/

Future Scope

- **Enhanced Visualization Options**: Adding support for more types of plots like bar charts, pie charts, and histograms for varied data analysis.

- **Extended Database Support**: Expanding to support additional databases like PostgreSQL, SQLite, or cloud databases.

- **Complex Mathematical Expressions**: Allowing users to input more complex expressions with multiple variables.

- **Data Analysis Tools**: Incorporating basic data analysis functions (e.g., mean, median, mode) for database data to provide deeper insights.