**REPORT**

**PyGenExpert Calculator v1.0**

A Comprehensive Multi-Purpose Calculator Application

Class 10 Artificial Intelligence Project Report (CBSE)

**Submitted By:**

Somesh Kumar Satpathy

Class: 10-A

Roll No.: 34

Kendriya Vidyalaya Gajapati

.

.

.

.

.

.

.

.

.

.

***This project report is submitted as part of the Class 10 Artificial Intelligence (417) curriculum for the academic session 2024-25.***

|  |  |  |
| --- | --- | --- |
| **Table of Contents** | | |
| ***Sl. No.*** | **Topic** | ***Page No.*** |
|  | ***Cover Page*** |  |
|  | ***Acknowledgement*** |  |
| ***1*** | **Introduction**   * **1.1 Project Objectives** * **1.2 Development Environment** |  |
| ***2*** | **Project Overview**   * **2.1 Application Architecture** * **2.2 User Interface Design** |  |
| ***3*** | **Features and Functionality**   * **3.1 Basic Calculator** * **3.2 Geometric Calculation** * **3.3 3D Shape Calculator** |  |
| ***4*** | **Technical Implementation**   * **4.1 Code Structure** * **4.2 Key Functions** |  |
| ***5*** | **Code Structure and Organization**   * **5.1 Code Organisation** * **5.2 Best Practices Implemented** |  |
| ***6*** | **Learning Outcomes and Future Enhancement**   * **6.1 Learning Outcomes** * **6.2 Future Enhancements** |  |
| ***7*** | **Conclusion** |  |
| ***8*** | **References** |  |
|  |  |  |

**1. Introduction**

The PyGenExpert Calculator v1.0 is a comprehensive Python-based calculator application developed as part of the Class 10 Artificial Intelligence (417) curriculum. This project demonstrates the practical application of programming concepts and showcases the power of Python in creating user-friendly applications.

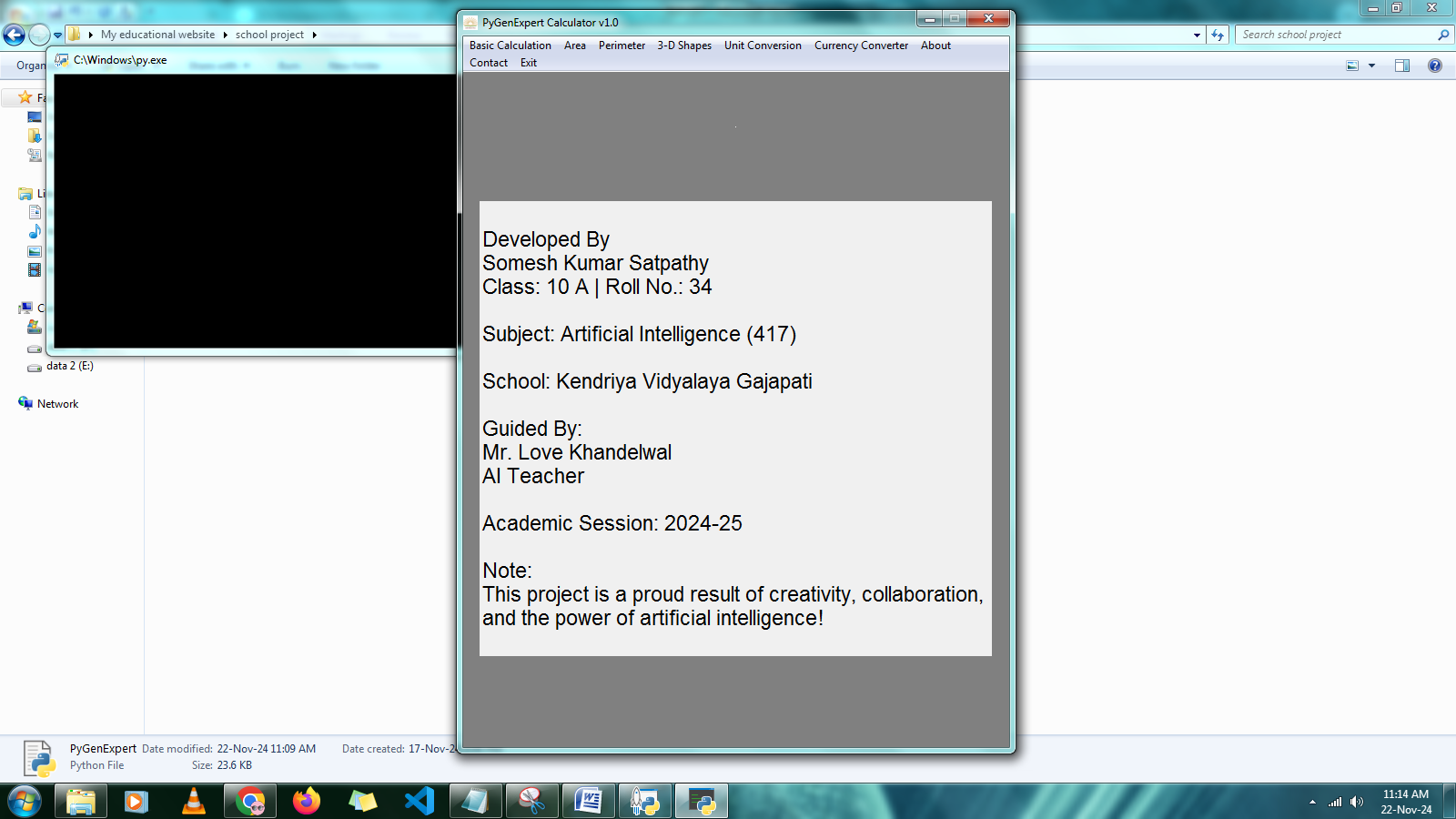


Fig. : GUI of PyGenExpert

**1.1 Project Objectives**

* To create a multi-functional calculator application.
* To implement a graphical user interface using Python's tkinter library.
* To demonstrate understanding of mathematical concepts through programming.
* To provide practical solutions for everyday calculations.

**1.2 Development Environment**

* Programming Language: Python 3.7
* GUI Framework: tkinter
* Additional Libraries: requests (for currency conversion)
* Development Tools: Python IDLE and Visual Studio Code

**Project Overview**

**2.1 Application Architecture**

The PyGenExpert Calculator is built using a modular architecture that separates different functionalities into distinct components:

**1. Basic Calculator Module**

* Arithmetic operations
* Clear function
* Error handling

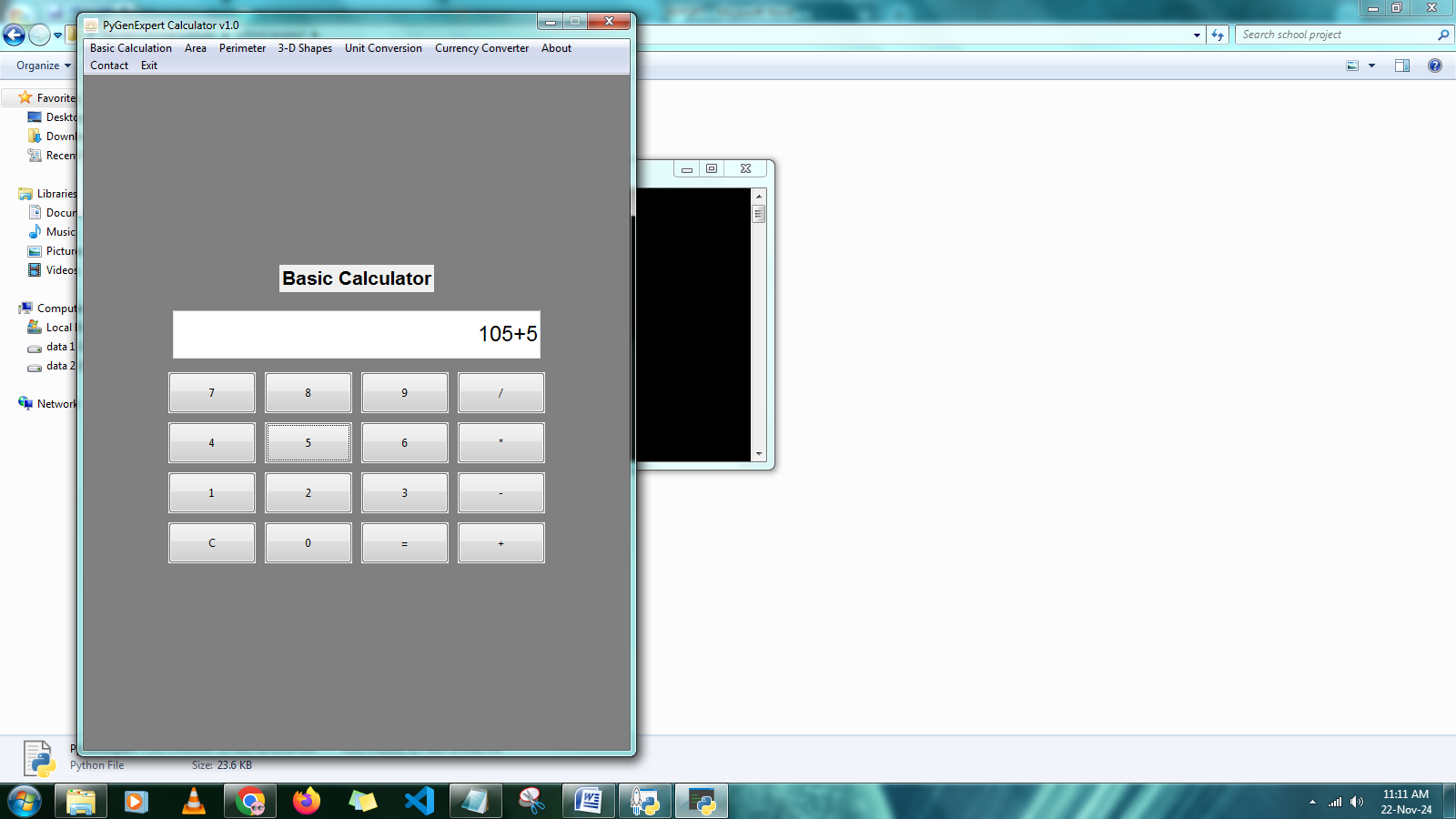
****

Fig. : Calculator

**2. Geometric Calculations**

* Area calculator
* Perimeter calculator
* 3D shape calculations

**3. Unit Conversion System**

* Length conversions
* Weight conversions
* Temperature conversions
* Time conversions

**4. Currency Converter**

* Real-time currency conversion
* API integration
* Multiple currency support

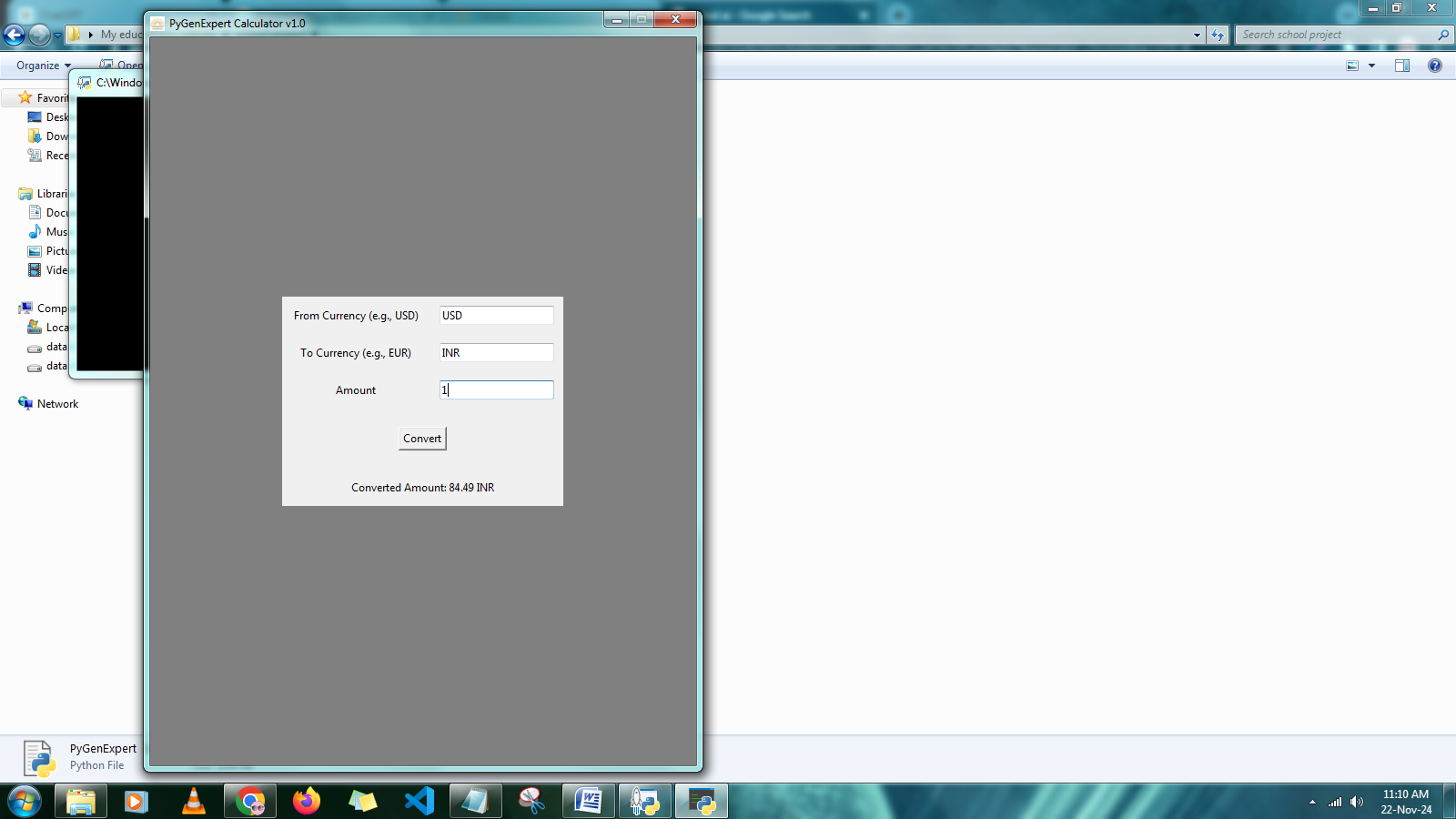
****

Fig. : Currency Converter

**2.2 User Interface Design**

The application features a clean and intuitive graphical user interface with:



Fig. : Menu Bar

* Menu-based navigation
* Tabbed interfaces for different conversions
* Clear input/output fields
* Error messaging system

**Features and Functionality**

**3.1 Basic Calculator**

**Operations Supported:**

* Addition
* Subtraction
* Multiplication
* Division

**Features:**

* Clear function (C)
* Error handling for invalid inputs
* Immediate calculation display

**3.2 Geometric Calculations**

**Area Calculator**

Supports multiple shapes:

* Circle
* Rectangle
* Triangle
* Square
* Trapezium
* Parallelogram
* Semi-Circle

**Perimeter Calculator**

* Calculates perimeter for all supported shapes
* Input validation for dimensions
* Accurate decimal calculations

**3.3 3D Shape Calculator**

**Volume Calculations:**

* Cylinder
* Cube
* Cuboid
* Sphere
* Hemisphere
* Cone

**Surface Area Calculations:**

* Total surface area
* Curved surface area
* Face-wise calculations

**4. Technical Implementation**

**4.1 Code Structure**

**Main application structure**

root = tk.Tk()

root.title("PyGenExpert Calculator v1.0")

root.geometry("600x800")

root.resizable(False, False)

root.wm\_iconbitmap("logo.ico")

root.config(background="grey")

root.mainloop()

**Menu system implementation**

menu\_bar = Menu(root)

# Basic Calculation menu

menu\_bar.add\_command(label="Basic Calculation", command=basic\_calculation)

# Area menu

area\_menu = Menu(menu\_bar, tearoff=0)

shapes = ["Circle", "Rectangle", "Triangle", "Square", "Trapezium", "Parallelogram", "Semi-Circle"]

for shape in shapes:

    area\_menu.add\_command(label=shape, command=lambda s=shape: calculate\_area(s))

menu\_bar.add\_cascade(label="Area", menu=area\_menu)

# Perimeter menu

perimeter\_menu = Menu(menu\_bar, tearoff=0)

for shape in shapes:

    perimeter\_menu.add\_command(label=shape, command=lambda s=shape: calculate\_perimeter(s))

menu\_bar.add\_cascade(label="Perimeter", menu=perimeter\_menu)

# 3-D Shapes menu

shapes\_3d\_menu = Menu(menu\_bar, tearoff=0)

# Submenu for Surface Area

surface\_area\_menu = Menu(shapes\_3d\_menu, tearoff=0)

three\_d\_shapes = ["Cylinder", "Cube", "Cuboid", "Sphere", "Hemisphere", "Cone"]

for shape in three\_d\_shapes:

    surface\_area\_menu.add\_command(label=shape, command=lambda s=shape: calculate\_3d("Surface Area", s))

shapes\_3d\_menu.add\_cascade(label="Surface Area", menu=surface\_area\_menu)

# Submenu for Curved Surface Area

curved\_surface\_area\_menu = Menu(shapes\_3d\_menu, tearoff=0)

for shape in three\_d\_shapes:

    curved\_surface\_area\_menu.add\_command(label=shape, command=lambda s=shape: calculate\_3d("Curved Surface Area", s))

shapes\_3d\_menu.add\_cascade(label="Curved Surface Area", menu=curved\_surface\_area\_menu)

# Submenu for Volume

volume\_menu = Menu(shapes\_3d\_menu, tearoff=0)

for shape in three\_d\_shapes:

    volume\_menu.add\_command(label=shape, command=lambda s=shape: calculate\_3d("Volume", s))

shapes\_3d\_menu.add\_cascade(label="Volume", menu=volume\_menu)

menu\_bar.add\_cascade(label="3-D Shapes", menu=shapes\_3d\_menu)

menu\_bar.add\_command(label="Unit Conversion", command=unit\_conversion)

menu\_bar.add\_command(label="Currency Converter", command=show\_currency\_converter)

# About menu

menu\_bar.add\_command(label="About", command=about)

# Contact menu

menu\_bar.add\_command(label="Contact", command=contact)

# menu\_bar.add\_cascade(label="Unit Conversion", menu=unit\_menu)

menu\_bar.add\_command(label="Exit", command=root.destroy)

# Configure the menu bar in the root window

root.config(menu=menu\_bar)

**4.2 Key Functions**

**Basic Calculator Implementation**

def basic\_calculation():

    for widget in root.winfo\_children():

        widget.pack\_forget()

    # Create the calculator frame

    calc\_frame = tk.Frame(background="grey")

    calc\_frame.pack(expand=True)

     # Remove home page

    # Display a title label for the calculator

    title\_label = tk.Label(calc\_frame, text="Basic Calculator", font=("Arial", 16, "bold"))

    title\_label.grid(row=0, column=0, columnspan=4, pady=10)

    # Entry widget to display calculations

    calc\_entry = ttk.Entry(calc\_frame, justify="right", font=("Arial", 18))

    calc\_entry.grid(row=1, column=0, columnspan=4, ipadx=5, ipady=10, padx=10, pady=10, sticky="ew")

    # Function to update the entry widget with button text

    def click(event):

        current = calc\_entry.get()

        text = event.widget.cget("text")

        if text == "=":

            try:

                result = eval(current)

                calc\_entry.delete(0, tk.END)

                calc\_entry.insert(tk.END, str(result))

            except Exception as e:

                calc\_entry.delete(0, tk.END)

                calc\_entry.insert(tk.END, "Error")

        elif text == "C":

            calc\_entry.delete(0, tk.END)

        else:

            calc\_entry.insert(tk.END, text)

    # Keypad layout in grid form

    buttons = [

        "7", "8", "9", "/",

        "4", "5", "6", "\*",

        "1", "2", "3", "-",

        "C", "0", "=", "+"

    ]

    # Create keypad buttons using grid layout

    row, col = 2, 0  # Starting row index is 2 to leave space for title and entry

    for button in buttons:

        btn = ttk.Button(calc\_frame, text=button, style="TButton")

        btn.grid(row=row, column=col, ipadx=10, ipady=10, padx=5, pady=5, sticky="nsew")

        btn.bind("<Button-1>", click)

        col += 1

        if col > 3:

            col = 0

            row += 1

    # Make the grid layout responsive

    for i in range(4):

        calc\_frame.columnconfigure(i, weight=1)

    for i in range(5):

        calc\_frame.rowconfigure(i, weight=1)

**5. Code Organization and Best Practices**

**5.1 Code Organization**

The project follows these organizational principles:

**1. Modular Design**

* Separate functions for different calculations
* Clear separation of UI and logic
* Reusable code components

**2. Error Handling**

* Input validation
* Try-except blocks for calculations
* User-friendly error messages

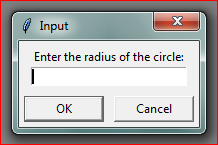
****

Fig. : Calculating the Area of Circle

**3. Documentation**

* Function docstrings
* Inline comments
* Clear variable naming

**5.2 Best Practices Implemented**

* Consistent coding style
* Proper indentation
* Meaningful variable names
* Efficient algorithm implementation
* Comprehensive error handling

**6. Learning Outcomes and Future Enhancements**

**6.1 Learning Outcomes**

Through this project, I have gained:

**1. Technical Skills**

* Python programming proficiency
* GUI development experience
* Understanding of mathematical implementations
* API integration knowledge

**2. Soft Skills**

* Project planning and management
* Problem-solving abilities
* Documentation skills
* Attention to detail

**6.2 Future Enhancements**

1. Potential improvements for future versions:
2. Scientific calculator functionality
3. Graphing capabilities
4. History feature for calculations
5. Custom theme options
6. Additional unit conversions

**Conclusion**

The PyGenExpert Calculator v1.0 successfully demonstrates the practical application of programming concepts learned in Class 10. The project not only serves as a functional calculator but also showcases the integration of various mathematical concepts with modern programming techniques.

The development process has provided valuable insights into software development practices and enhanced my understanding of both programming and mathematics. The modular design ensures that the application can be easily maintained and expanded in the future.

**References**

1. Python Documentation - python.org

2. Tkinter Documentation

3. CBSE Class 10 Artificial Intelligence (417) Curriculum

4. ChatGPT, AI language model created by OpenAI - https://chatgpt.com/

5. YouTube - youtube.com

6. Google Search - www.google.com

****** 

