Computer programming

Hadi ali chebli gh1033528

Carbon Footprint Calculator Report

Problem Statement

In today's world, the reduction of carbon emissions becomes critical in tackling climate change. There is an increased pressure on businesses, governments, and individuals to measure, monitor, and reduce their environmental impact. However, most of these organizations do not have access to an easy-to-use tool that calculates carbon emissions and offers actionable insights. This therefore calls for a Carbon Footprint Calculator that will simplify understanding environmental impact, offer visualizations, and suggest tailored solutions.

The stakeholders benefiting from this solution include:

- Businesses: To identify high-emission areas and implement strategies for reduction.

- Environmental Agencies: To promote carbon-reduction programs and track progress.

- Individuals: To inspire personal sustainable behaviors.

The ultimate objective of the tool is to give users the ability to make informed choices and contribute less to carbon emissions through straightforward analysis and recommendations.

Design and Implementation

Overview

The Carbon Footprint Calculator is a GUI Python application that uses `tkinter`. It will calculate carbon emissions based on the user's electricity usage, natural gas usage, fuel expenses, waste management, and business travel. The tool would give some visual feedback by bar graphs and pie charts, store results in JSON format, and provide some suggestions for reduction.

Key Features

1. User-Friendly Input: Users can input data such as monthly electricity bills and travel distances.

2. Graphical Visualizations: Results can be displayed as bar graphs or pie charts.

3. Data Storage: Emission data is saved in JSON format for future reference.

4. Suggestions: Provides a random suggestion if the total emissions exceed 40,000 kg CO2 annually.

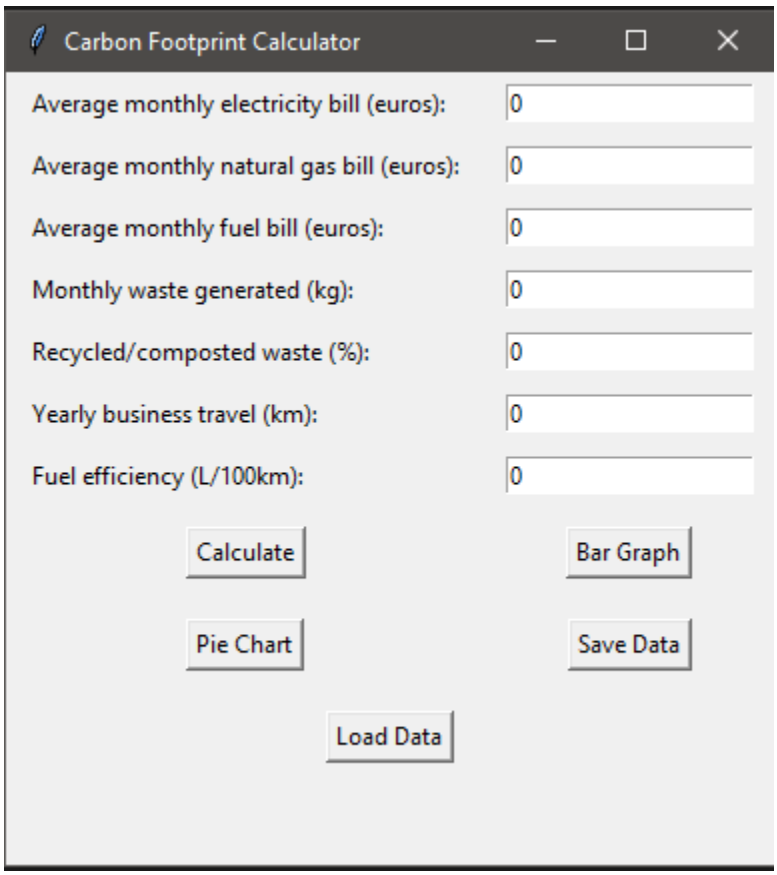
5. Error Handling: Ensures invalid inputs are managed gracefully.

Code Explanation

User Input Handling

The program captures user inputs via a simple and intuitive interface using `tkinter`. Each input is validated to ensure accuracy. For example, numeric inputs are checked to prevent errors such as entering text where numbers are expected. This ensures the program doesn't crash and provides meaningful feedback to users when invalid inputs are detected.

For instance, when the user enters data such as monthly electricity or natural gas bills, the program retrieves and stores these values for further processing. Real-time error handling ensures any invalid or missing data is flagged immediately with user-friendly messages.



Carbon Emission Calculations

The calculator uses formulas derived from the provided reference table to compute emissions. Each component (electricity, gas, fuel, waste, and travel) is processed separately, and the results are aggregated to calculate total emissions.

- Electricity Emissions: Multiplied by a conversion factor to estimate the CO2 equivalent.

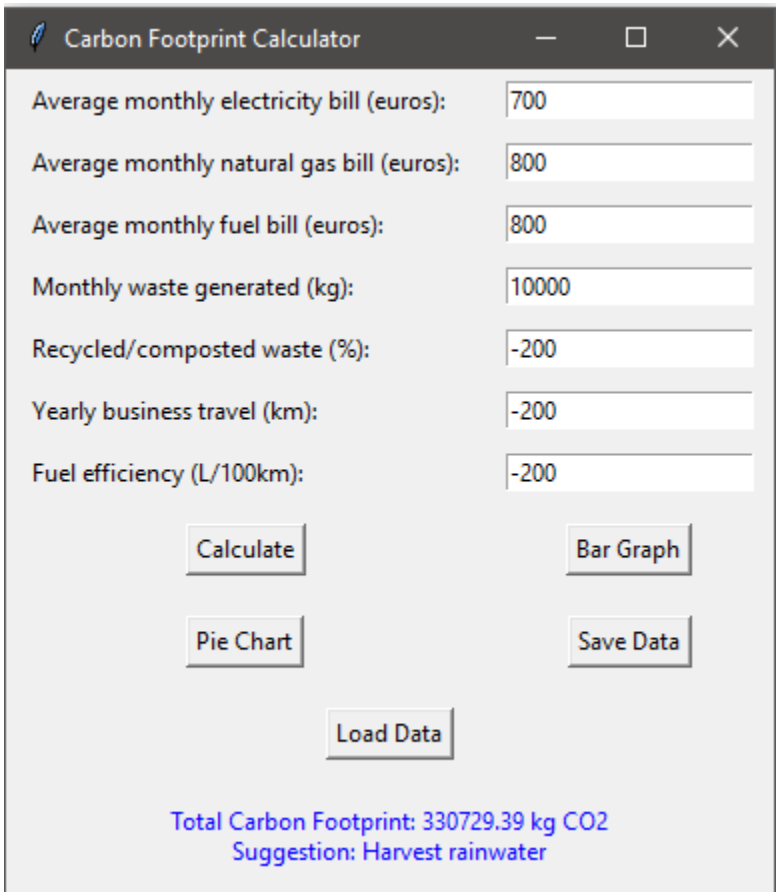
- Natural Gas Emissions: Calculated using the monthly expenditure and a specific emission coefficient.

- Fuel Emissions: Combines monthly costs with a fuel-specific multiplier.

- Waste Emissions: Considers the total waste generated, recycling percentage, and their respective emission factors.

- Travel Emissions: Uses distance traveled and fuel efficiency to compute emissions for business travel.

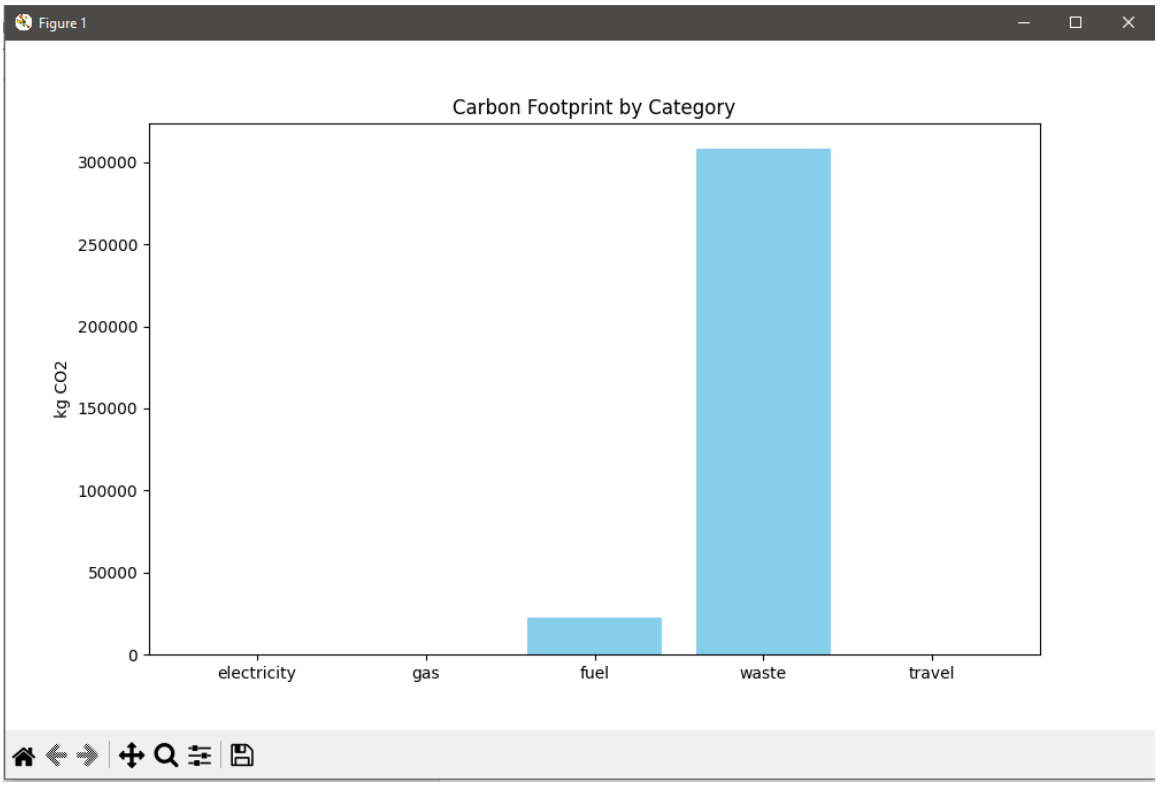
Each calculation is modularized, making the program easy to extend or modify.



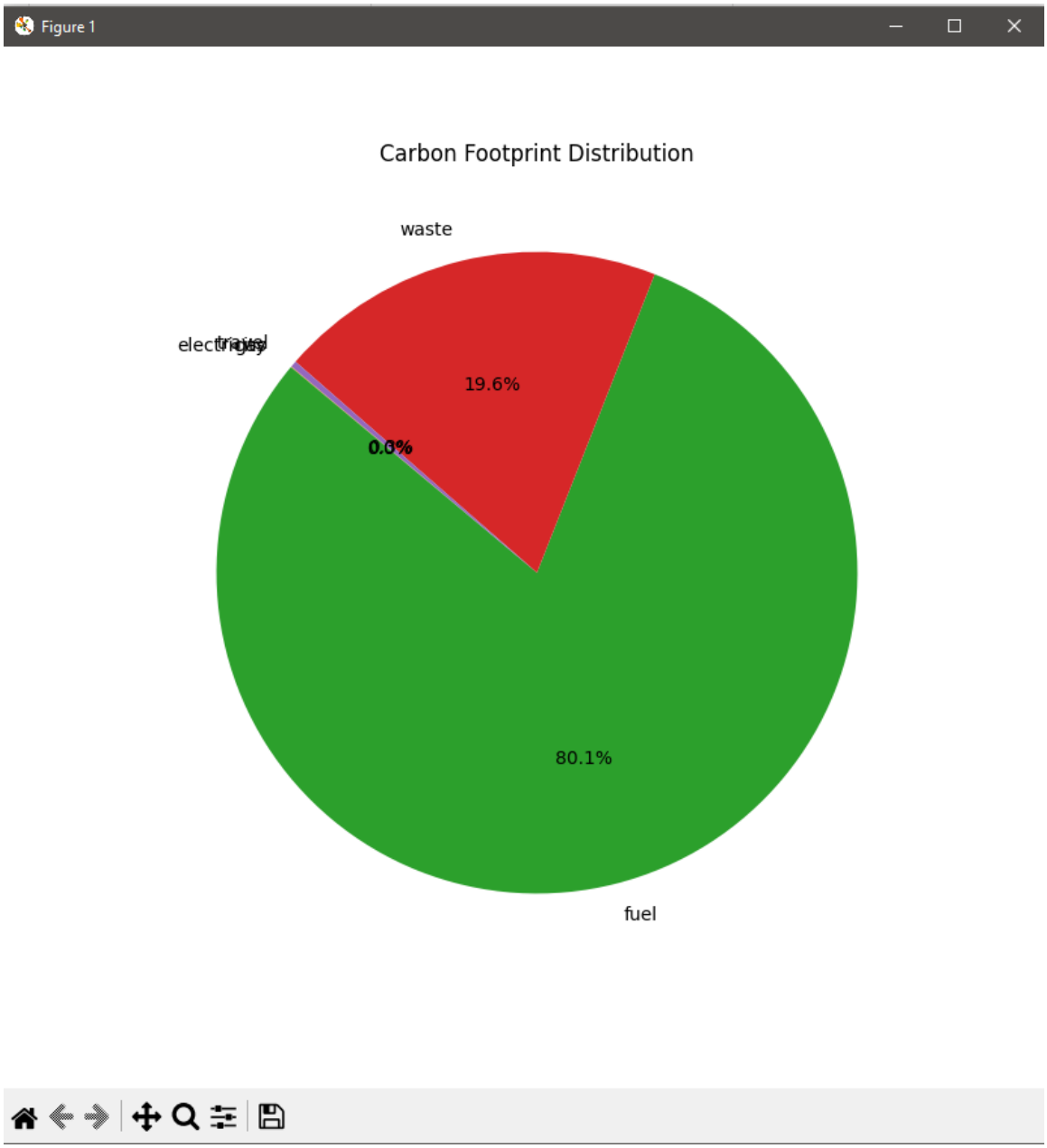
Visualization

The program uses the `matplotlib` library to generate visual representations of carbon emissions. Users can choose between bar charts and pie charts to view their data.

- Bar Charts: Show a category-wise breakdown of emissions for easier comparison.



- Pie Charts: Display the proportionate contribution of each category to the total emissions. If any values are negative (e.g., due to recycling deductions), they are converted to positive for proper visualization without affecting the original data.



This feature makes it easier for users to identify major contributors to their carbon footprint.

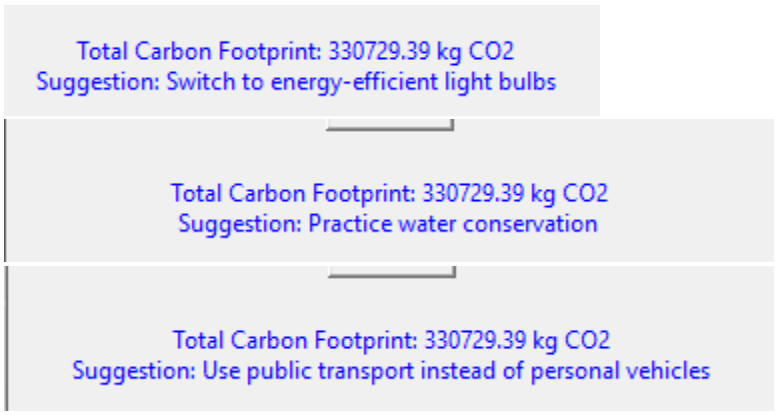
Suggestions for Emission Reduction

If the total emissions exceed 40,000 kg CO2 annually, the program provides a randomly selected suggestion from a list of 30 predefined tips. Examples include:

- Switching to renewable energy sources.

- Improving vehicle fuel efficiency.

- Reducing waste generation and increasing recycling efforts.



The suggestion is stored alongside the results in a JSON file for future reference. This feature personalizes the user experience and encourages actionable changes.

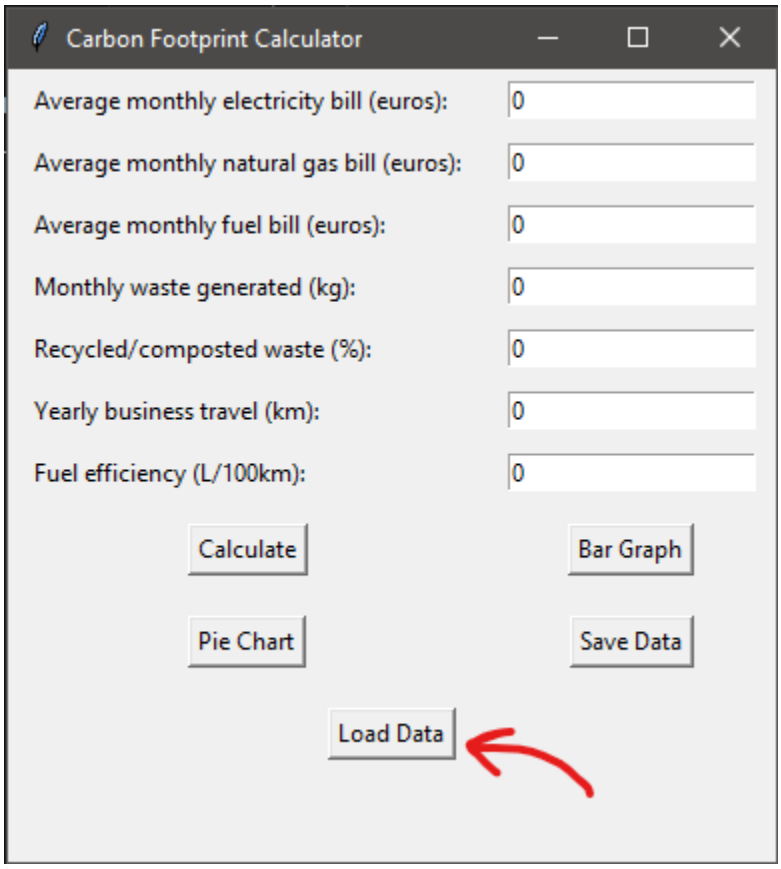
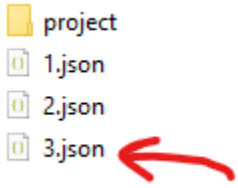
Data Storage

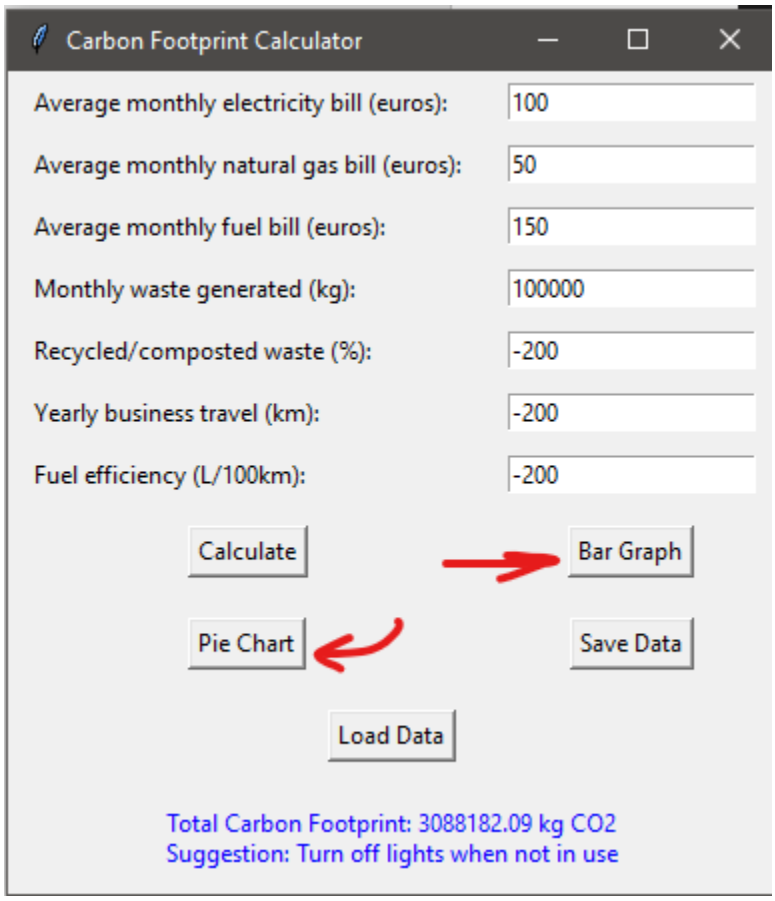
Emission results and user suggestions are saved in a JSON file. This allows users to revisit previous calculations and track their progress over time. Each entry includes:

- The user's input data.
- Calculated emissions for each category.
- Total emissions.
- Any provided suggestion.

The program writes this data into a JSON file after every calculation, ensuring that all results and suggestions are preserved for future use. For example, each JSON entry might look like this:

To retrieve this data, the program reads the JSON file, parses the stored information, and displays it to the user. This ensures users can review past emissions or compare their progress over time.

1. 
Carbon Footprint Calculator
Average monthly electricity bill (euros):
0
Average monthly natural gas bill (euros):
0
Average monthly fuel bill (euros):
0
Monthly waste generated (kg):
0
Recycled/composted waste (%):
0
Yearly business travel (km):
0
Fuel efficiency (L/100km):
0
Calculate
Bar Graph
Pie Chart
Save Data
Load Data
2. 
project
1.json
2.json
3.json

3. 

This persistent storage ensures that the tool is both practical and useful for long-term carbon footprint management. Users can also reload and review past results by accessing the JSON file. The JSON structure ensures flexibility and compatibility with various data analysis tools.

The program handles JSON storage by writing the results after every calculation. This ensures that no data is lost if the application is closed. Additionally, the user can choose to reload this data for comparison, enabling trend analysis over time.

Error Handling

Robust error handling is integrated throughout the application. Examples include:

- Validating user inputs to ensure numeric fields do not contain text.

- Handling division by zero errors, such as when fuel efficiency is set to zero.

- Graceful recovery from unexpected issues, with descriptive error messages to guide users.

Here are a few examples of error handling in the code:

```python
def load_data(filepath):
    try:
        with open(filepath, "r") as f:
            return json.load(f)
    except Exception as e:
        messagebox.showerror("Error", f"Could not load data: {e}")
        return None
```

```python
def plot_bar_graph(data):
    try:
        labels = list(data.keys())[:-1]  # Exclude 'total'
        values = list(data.values())[:-1]

        plt.figure(figsize=(10, 6))
        plt.bar(labels, values, color="skyblue")
        plt.title("Carbon Footprint by Category")
        plt.ylabel("kg CO2")
        plt.show()
    except Exception as e:
        messagebox.showerror("Error", f"Could not plot bar graph: {e}")
```

This ensures the program remains stable and user-friendly, even in edge cases.

Results and Analysis

Sample Input and Output

Input Example

- Electricity: €100/month
- Natural Gas: €50/month
- Fuel: €150/month
- Waste: 50 kg/month
- Recycling: 30%
- Travel: 10,000 km/year
- Fuel Efficiency: 8 L/100 km

Output Example

- Electricity: 600 kg CO2

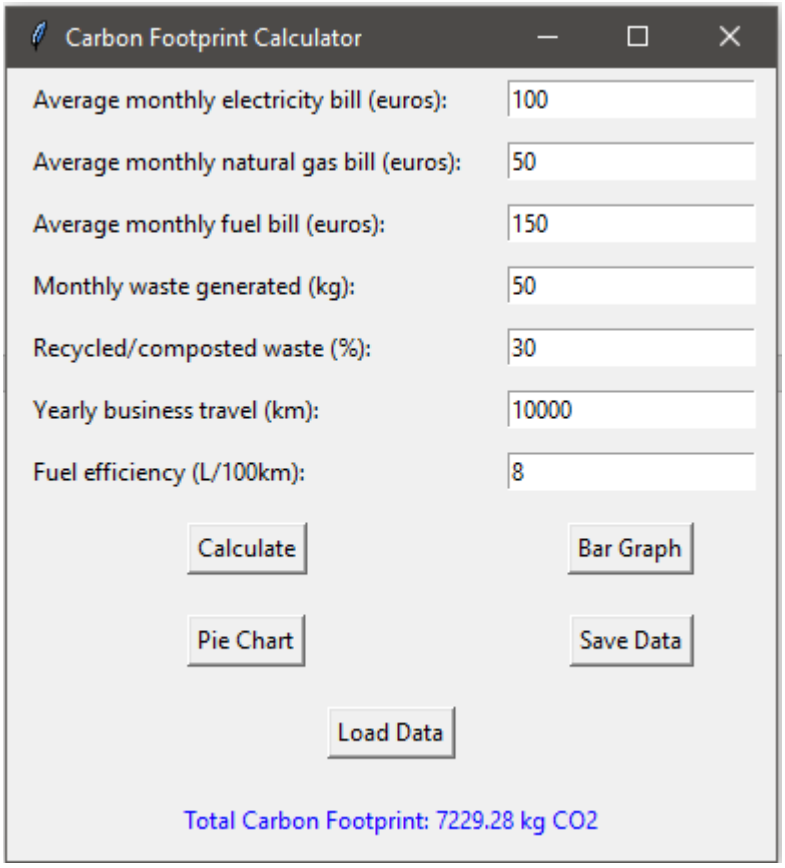
- Natural Gas: 3,180 kg CO2

- Fuel: 4,176 kg CO2

- Waste: 342 kg CO2

- Travel: 2,887.5 kg CO2

- Total Emissions: 7229.28 kg CO2



Visualization

Bar and pie charts allow users to see category-wise contributions to total emissions.

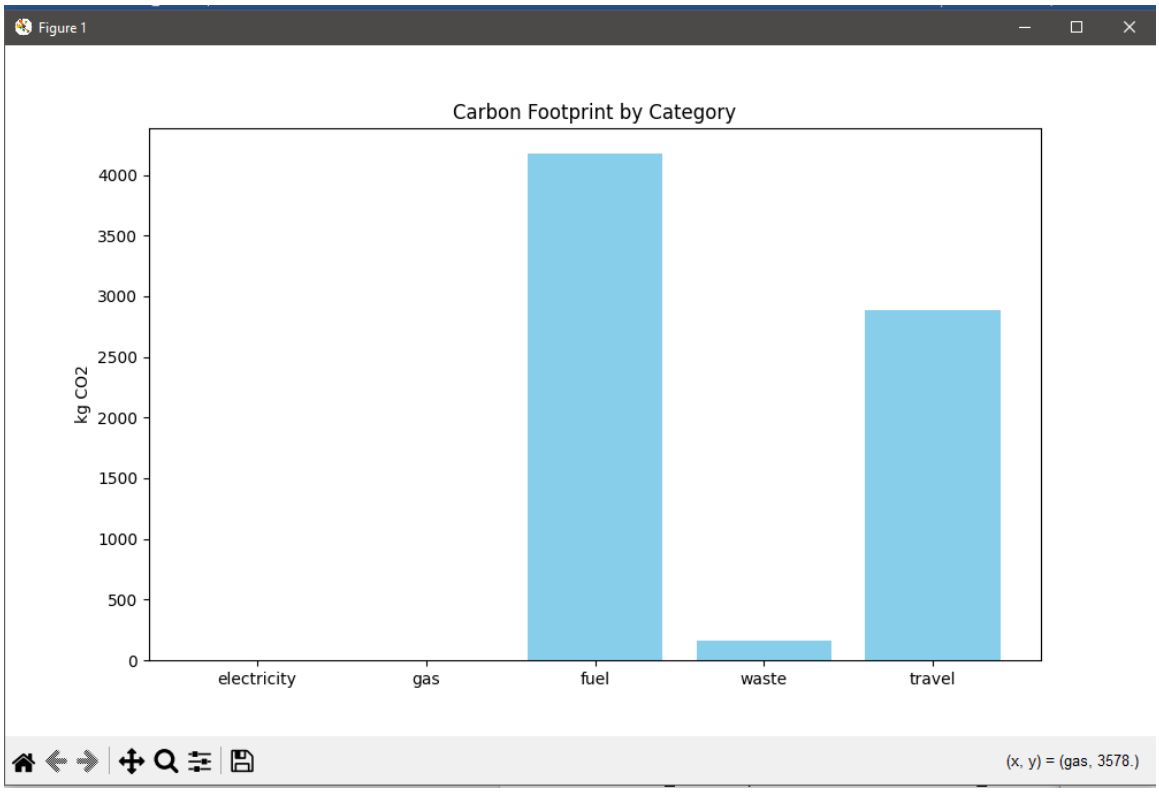
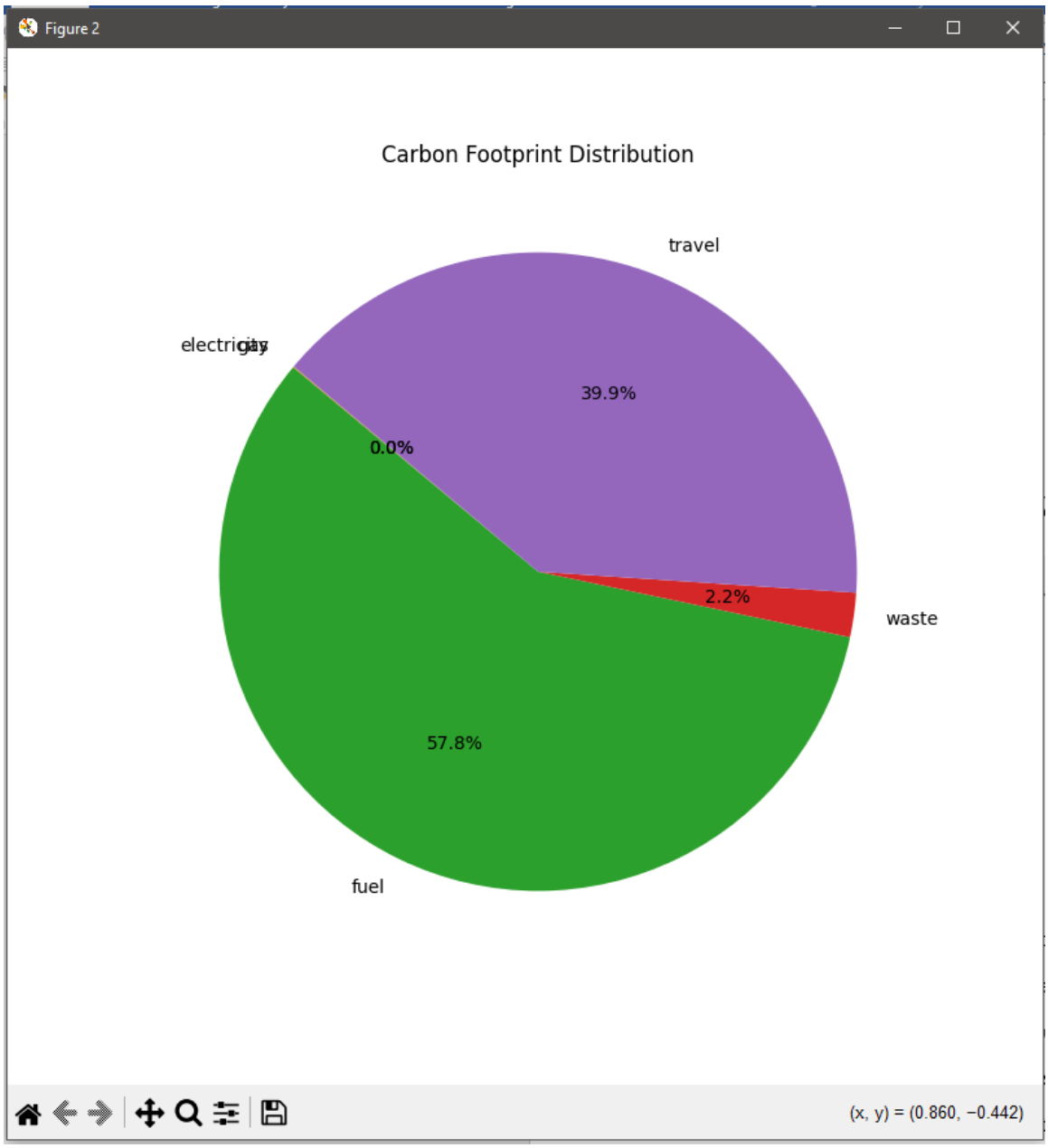


Figure 1
Carbon Footprint by Category
kg CO2
4000
3500
3000
2500
2000
1500
1000
500
0
electricity
gas
fuel
waste
travel
(x, y) = (gas, 3578.)



Conclusion and Future Work

Conclusion

The Carbon Footprint Calculator successfully meets its objectives by providing:

1. Accurate emission calculations based on user inputs.

2. Intuitive visualizations for better understanding.

3. Tailored suggestions to promote sustainability.

4. Robust error handling and data storage features.

It empowers users with knowledge and actionable insights to help contribute to reducing carbon emissions globally.

Future Enhancements

To improve the tool further, the following enhancements are suggested:

1. Integration with APIs: Use APIs to fetch real-time electricity and gas rates for more accurate calculations.

2. Mobile Version: Develop a mobile-friendly version of the tool.

3. Advanced Reporting: Include features for trend analysis and comparison across different timeframes or user groups.

4. Gamification: Add rewards or incentives for users who consistently reduce their carbon footprint.

These enhancements will make the Carbon Footprint Calculator even more powerful and versatile in the fight against climate change.

Github link for project : https://github.com/Hadialishibli/Programming-project

Assessment Submission Form

Student Number (If this is group work, please include the student numbers of all group participants)	Hadi ali chebli 1033528
Assessment Title	M602A Computer Programming (WS0924)AssignmentsIndividual Final Project
Module Code	M602A Computer Programming (WS0924)AssignmentsIndividual Final Project
Module Title	M602A Computer Programming (WS0924)AssignmentsIndividual Final Project
Module Tutor	Willian morrison
Date Submitted	19/12/2024

Declaration of Authorship

I declare that all material in this assessment is my own work except where there is clear acknowledgement and appropriate reference to the work of others.

I fully understand that the unacknowledged inclusion of another person's writings or ideas or works in this work may be considered plagiarism and that, should a formal investigation process confirms the allegation, I would be subject to the penalties associated with plagiarism, as per GISMA Business School, University of Applied Sciences' regulations for academic misconduct.

Signed.....hadi ali chebli..... Date .....19/12/2024.....

(Ctrl)