Carbon Footprint Tracker

Project Report Submitted

to

MANIPAL ACADEMY OF HIGHER EDUCATION

For Partial Fulfillment of the Requirement for the

Award of the Degree

Of

Bachelor of Technology

in

Computer and Communication Engineering

by

Adri Katyayan

Reg. No.: 210953218

Shubhika Upreti

Reg. No.: 210953252

Under the guidance of

Mrs. Vibha
Assistant Professor-Senior Scale
Department of I&CT
Manipal Institute of Technology
Manipal, Karnataka, India

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Unit of MAHE, Manipal

INDEX

ABSTRACT

This project aimed to develop a carbon footprint tracker that would assist people in effectively managing their carbon footprint. The system was created using Flask-a framework for Python and MySQL for the server side and HTML, CSS, JavaScript, jQuery, and Bootstrap for the user interface. The project has several features, such as users can log in to their accounts and enter data about their daily activities like transportation, waste consumption and diet, and the application calculates the carbon footprint based on this information. The project has followed a thorough methodology that included requirements gathering, design, implementation, and testing. The project was successfully implemented, and it was able to fulfil the initial requirements.

INTRODUCTION

Our online software is a comprehensive solution to make people conscious of their carbon emissions and to improve the environment for future generations. As the consequences of the changing climate become increasingly evident, recognizing and decreasing the emissions we produce are crucial initial actions towards a sustainable future.

By inputting their daily activities, tracking their carbon footprint, and learning about the environmental effects of their everyday decisions, users of our program may easily compute their carbon emissions. This report delves into the development, functionality, and potential implications of our Carbon Footprint Tracker by highlighting its contribution in the making of a greener society and encourage individuals to make better environment-friendly decisions.

The Carbon Footprint Tracker is a web application designed to promote sustainable lifestyle choices. To get started, users create an account and enter data about their daily activities, such as transportation, energy use, and diet. Using a sophisticated algorithm, the application calculates the carbon footprint based on this information. Through interactive charts and graphs, users can visualize their environmental impact and gain a better understanding of their contribution to climate change, aligning with the 12^{th} goal of Sustainable Development, responsible consumption, and production. Additionally, the platform provides personalized recommendations and tips for reducing emissions, motivating users to set and monitor their carbon reduction goals and personal expenses.

BACKGROUND INFORMATION

After noticing the growing demand for a comprehensive system that streamlines and arranges the process of decreasing the emissions and making people aware of their carbon emissions, the Carbon Footprint Tracker project was born. Due to various industrial processes, there is a rise in atmospheric carbon dioxide levels, which in turn causes global temperatures to rise. To tackle these environmental challenges, multiple governments, businesses, and individuals are trying to find ways to reduce their carbon footprints.

The project's goal is to provide simple-to-use software that will improve the overall quality of the environment by calculating the carbon emissions from the day-to-day activities of the users.

PYTHON CONCEPTS USED

- Flask Python Framework
- sqlAlchemy
- pymysql
- numpy
- matplotlib
- Render.com
- Gunicorn

Explanations:

1. Flask Python Framework:
    a. Flask is a lightweight Python web framework this is nicely-ideal for building small to medium-sized net programs. It is easy to analyze and use, and it's far noticeably extensible, making it a great preference for a extensive variety of projects.
    b. Flask applications are normally composed of a series of routes, which can be capabilities which are mapped to unique URLs. When a consumer visits a URL in a Flask utility, the corresponding path characteristic is achieved. Flask also presents some of features for constructing internet programs, inclusive of session control, template rendering, and shape validation.

2. SqlAlchemy:
    a. sqlAlchemy is an item-relational mapper (ORM) for Python that offers a layer of abstraction between Python items and database tables. This makes it easy to work with databases in Python, without having to fear about the underlying SQL info.
    b. SqlAlchemy can be used with various database backends, together with MySQL, PostgreSQL, and SQLite. Capabilities involve object-relational mapping, question generation, and transaction management.

3. Pymysql:
    a. pymysql is a Python wrapper for the MySQL database server. It provides a simple and smooth-to-use interface for connecting to and interacting with MySQL databases.
    b. Pymysql may be used for a variety of obligations, consisting of strolling queries, putting and updating facts, and retrieving facts from MySQL databases. It is a famous preference for developing Python applications that need to interact with MySQL databases.

4. Numpy:
    a. NumPy is a Python library for clinical computing. It offers several functions for operating with arrays, matrices, and other numerical facts systems.
    b. Numpy is a popular choice for developing Python packages that want to carry out medical computing responsibilities, consisting of facts analysis, gadget getting to know, and photo processing.

5. Matplotlib:
    a. matplotlib is a Python library for developing records visualizations. It affords several capabilities for growing charts, graphs, and other types of information visualizations.
    b. Matplotlib is a famous preference for growing Python packages that need to create data visualizations, which include dashboards, reviews, and displays.

6. Render.com:
    a. Render.com is a cloud platform for Python programs. It gives a simple and clean-to-use way to install and manage Python packages in the cloud.
    b. Render.com can be used to set up Python packages of any size, from small non-public tasks to massive corporation packages. It provides several features for managing Python packages, such as automatic deployment, scaling, and tracking.
7. Gunicorn:
    a. gunicorn is a Python internet server this is designed to deal with excessive-site visitors web packages. It is a WSGI server, which means that it could be used to run any Python net framework, which includes Flask.
    b. Gunicorn is a famous choice for deploying Flask packages in production. It affords a number of functions for dealing with high-site visitors internet programs, along with worker control, method management, and logging.

METHODOLOGY

- **Planning and Analysis:**
  We conducted an in-depth analysis of the project requirements and determined the features and functionalities of the web application. We also identified the target audience and their needs. Based on these findings, we created a project roadmap and set project goals and timelines.

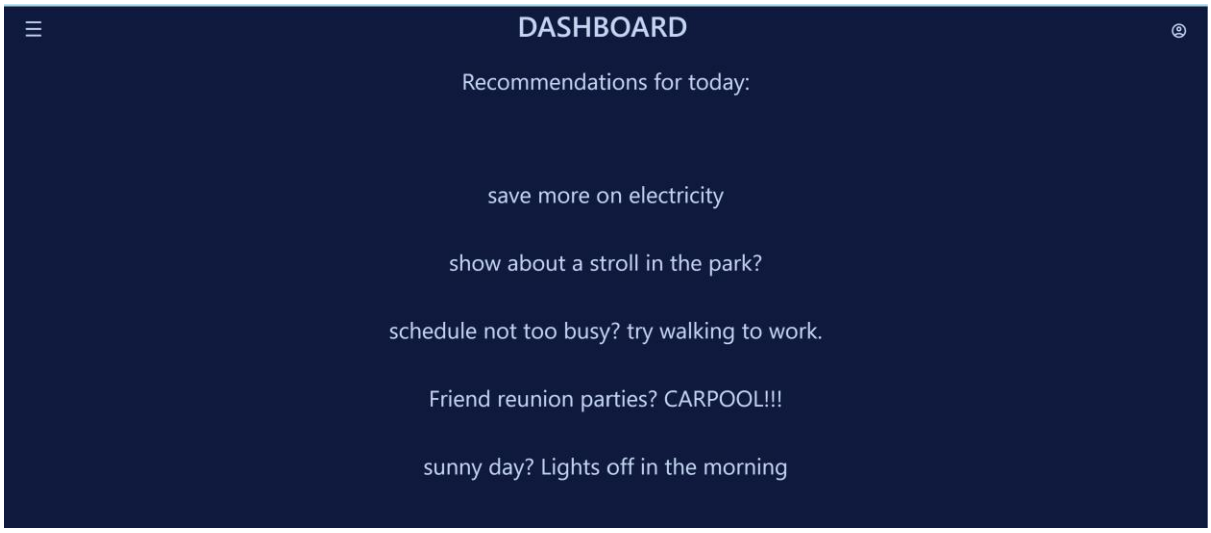
- **Design:**
  We started by designing the UI/UX for the web application using HTML5 and Tailwind CSS. We also defined the database schema using MySQL and created tables to store data for the application. We used ER diagrams to visualize the schema and ensure that it meets the requirements of the application.

- **Development:**
  We developed the web application's backend and frontend components. We used Python and Flask libraries to create the backend server and API, which handles user requests. We used MySQL to create the database schema, tables, and queries. We used HTML, CSS and JavaScript to create the front-end components and connect them to the backend through Flask framework API.

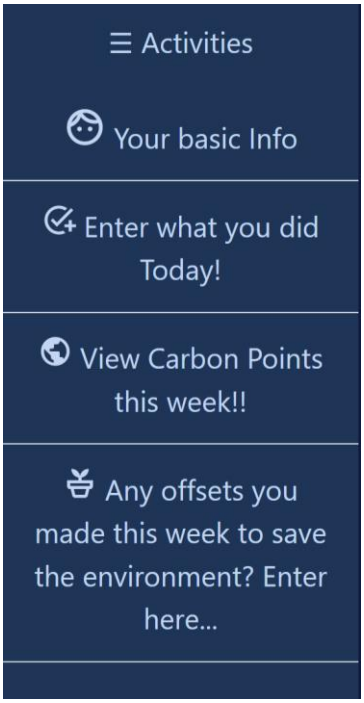
- **Functionality:**
  We implemented several features in the web application, including the ability to add carbon points and view total points, add, and view one's daily activities done in a day, and add and view offsets related to carbon emissions.

IMPLEMENTATION AND RESULTS

1. The project opens to a Dashboard.



2. The user may visit other webpages through the sidebar.



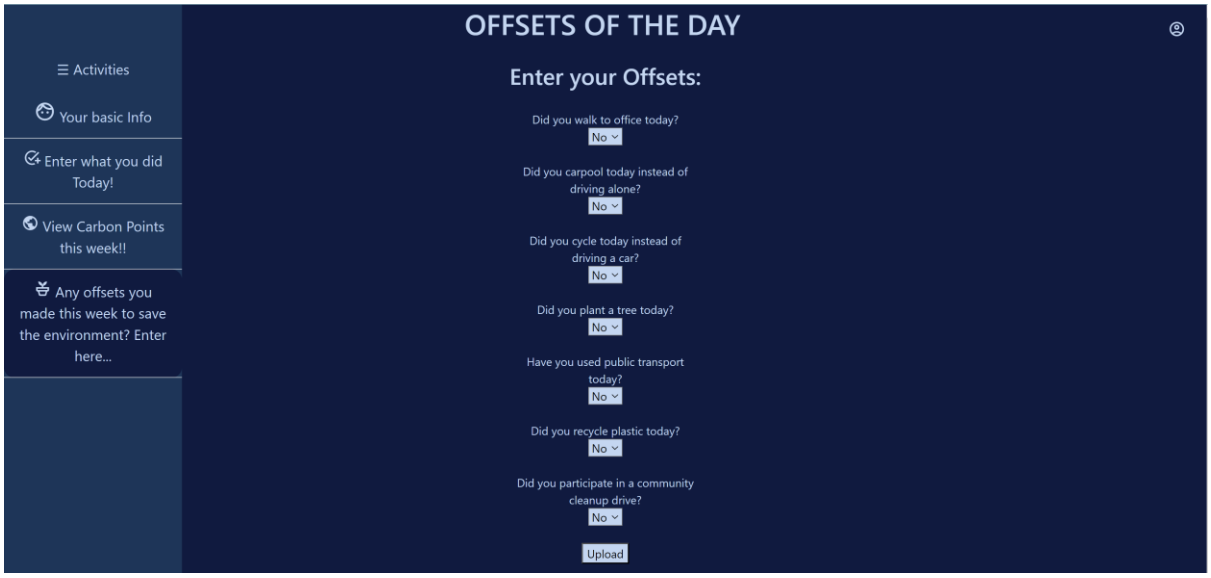
3. The user will be able to view his basic information entered by him when he clicks on “Your Basic Info”. This page displays the user’s personal information that he enters, along with some fixed attributes that will be considered while calculating carbon emissions.
   These fixed attributes include the number of members in the household, size of the lodgings (calculated in terms of number of bedrooms) and total amount of waste produced(calculated in terms of the number of garbage bags discarded on a daily basis).

4. On clicking “Enter what you did today!”, the user gets a form in which he/she may fill what he/she did. Activities such as “Did you throw any waste?”, “Did you drive a car today?”, “What type of meal did you consume?”, “Did you do laundry today?”, “Did you wash utensils?” cause carbon emissions in almost every household and these activities also consume resources such as water and electricity.

   The form asks for further details such as “how many bins?” and “how many kms?” when user answers “yes”.

5. The user has the option of entering any offsets he/she did in a day. Offsets are defined as activities that reduce carbon emissions in the long run and have a positive impact on the environment. The offsets considered in this project are walking to work, carpooling, cycling, planting trees, using public transport, recycling, and community cleanup drives.

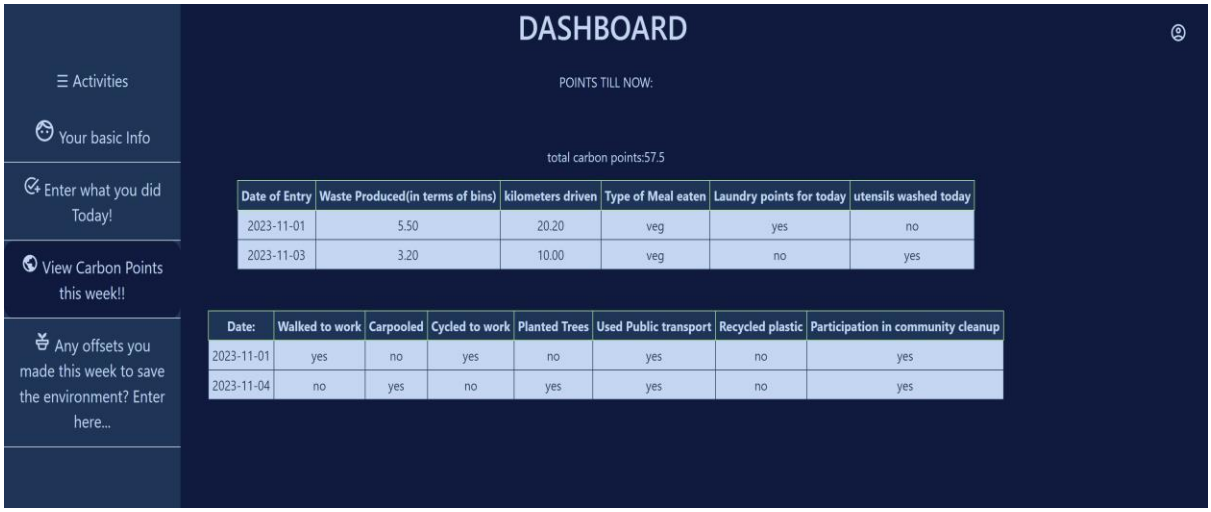
   Each of these activities shall affect the carbon points of the user.



The values entered in offsets.html get uploaded as below:

```
@app.route('/upload_offsets',methods=['GET','POST'])
def upload_offset_to_db():
  if request.method == 'POST':
    user_id=1
    walk = request.form.get('walk')
    carpool = request.form.get('carpool')
    cycle = request.form.get('cycle')
    tree = request.form.get('tree')
    pubtrans = request.form.get('pubtrans')
    recycle = request.form.get('recycle')
    cleanup = request.form.get('cleanup')
    date = datetime.now().strftime('%Y-%m-%d')
    task_for_db=upload_offsets(user_id, walk, carpool, cycle, tree, pubtrans, recycle, cleanup, date)
  return render_template('today.html')
```

6. When user clicks to view carbon points, all his activities that he previously uploaded get displayed in tabular form. His/her carbon points are displayed above the two tables as show in figure below:



7. Carbon points are calculated based on the following:
    a. Fixed attributes:
        i. +(Number of members in family*15)
        ii. +(Number of rooms in the house*30)
        iii. +(Number of wastebins produced*5)
    b. Offsets:
        i. -10 if walked to work.
        ii. -5 if carpooled
        iii. -6 if cycled.
        iv. -20 if planted a tree.
        v. -7 if used public transport.
        vi. -12 if recycled plastic
        vii. -8 if participated in community cleaning activities.
    c. Daily Activities:
        i. +(number of kilometers travelled*5)
        ii. +(total members*3*20) if food eaten is non-vegetarian.
        iii. +(total members*3*5) if food eaten is vegetarian.
        iv. +(if laundry is done, then 70, else 0)
        v. +(if utensils washed then 20, else 0)

CONCLUSION

The "Carbon Footprint Tracker" project, in conclusion, provides users with an entire platform to simplify and improve the standard of their living and the environment. A centralised system is implemented by the application's software for handling user details, family details, day-to-day activities, as well as other important information. Users of this platform can reduce their carbon footprint and enhance their quality of life while also helping the environment simply by putting this software into action.

REFERENCES

[1] SqlAlchemy,[Online]Available: https://docs.sqlalchemy.org/en/20/tutorial/

[2] "Apply graphs on Flask apps", [Online]. Available: https://www.youtube.com/watch?v=E2hytuQvLlE

[3] MySQL, "MySQL", MySQL, 2023. [Online]. Available: https://www.mysql.com/

[4] Tailwind CSS, "Tailwind CSS - Rapidly build modern websites without ever leaving your HTML.", Tailwind CSS [Online]. Available: https://tailwindcss.com/

[5] "pyMySQL Tutorial", pyMySQL Tutorial [Online]. Available: pyMySQL Tutorial - Learn MySQL Fast, Easy and Fun.

[6] "Web Development with Python Tutorial – Flask & Dynamic Database-Driven Web App . Available: https://www.youtube.com/watch?v=yBDHkveJUf4