**1. Project Overview**

The project was developed as a capstone requirement to create a basic working prototype of a carbon calculator. Its primary goal is to showcase the proper understanding and application of Life Cycle Assessment (LCA) principles in real-world scenarios.

* **Target Audience:** LCA consultants and evaluators.
* **Purpose:** To facilitate carbon footprint calculations across various stages (e.g., raw materials, manufacturing) using predefined emission factors stored in a MongoDB database.

**2. Technology Stack**

* **Backend:** Python (FastAPI)
* **Database:** MongoDB
* **Frontend:** HTML, CSS, JavaScript
* **Libraries/Tools:**
  + motor (for MongoDB integration)
  + Jinja2 (for template rendering)
  + openai (for chatbot functionality)

**3. System Architecture**

The system is designed with a separation of frontend and backend components. The frontend interacts with the backend via REST APIs, and the backend processes user inputs asynchronously using FastAPI.

* **Frontend-Backend Interaction:**
  + User inputs data via HTML forms.
  + Backend retrieves emission factors from MongoDB and calculates the carbon footprint.
  + Results are returned to the frontend for visualization.

**4. Backend Structure**

The backend is organized into different modules to handle specific functionalities:

* **Main Application (main.py)**
  + Initializes the FastAPI app, handles CORS, and sets up routing for the main endpoints.
  + Endpoints include: emissions, visualizations, template, scenarios, and chatbot.
  + Example code for main.py provided in the section above.
* **Key Endpoints:**
  + emissions.py: Handles emission factor retrieval and carbon footprint calculation based on user inputs.
  + chatbot.py: Provides chatbot functionality using OpenAI for LCA-related queries.
  + scenarios.py: Manages user scenarios, including CRUD operations for scenario data.
  + template.py: Retrieves predefined templates for product systems and their associated stages.

**5. Configuration (config.py)**

Configuration is handled using Pydantic Settings. The .env file is automatically loaded to configure the MongoDB URI, database name, and OpenAI API key.

* **Example:**

class Settings(BaseSettings):

MONGODB\_URI: str

MONGODB\_NAME: str

OPENAI\_API\_KEY: str

class Config:

env\_file = ".env"

**6. Frontend Structure**

The frontend consists of HTML, CSS, and JavaScript files that provide the user interface and interaction. It communicates with the backend via HTTP requests, submitting user data for processing.

* **HTML Files:** Display forms for user input (e.g., index.html, new\_scenario.html).
* **CSS (styles.css):** Provides styling to ensure a consistent and responsive user interface.
* **JavaScript (script.js):** Handles client-side logic, including AJAX requests to the backend.

**7. MongoDB Usage**

MongoDB is used to store emission factors and user scenarios. Key collections include:

* **Emission Factors:** Stores carbon emission factors for different activities (e.g., raw materials, transportation).
* **Scenarios:** Stores user-generated scenarios that combine emission factors and user input to calculate the total carbon footprint.

**8. Data Flow & Business Logic**

1. **User Input:** User submits data for various stages (e.g., raw materials, manufacturing) through the frontend.
2. **Emission Factor Retrieval:** The backend retrieves the corresponding emission factors from MongoDB.
3. **Carbon Calculation:** The carbon footprint for each stage is calculated by multiplying user inputs by the emission factors.
4. **Output:** The total carbon footprint for the scenario is calculated and sent back to the frontend.

**9. Version Control & Documentation**

* **Version Control:** The project uses Git for version control.
* **README.md:** The documentation provides an overview of the project, setup instructions, and key endpoints.

**10. Deployment**

* **Current Status:** The application has not been deployed and is run locally.
* **Future Plans:** Potential deployment options include using cloud services such as AWS or GCP to host the application.