Carbon Footprint Tracking Application System Architecture Documentation

Vinay Kumar Satrasala,Mohammed Shahad

September 3, 2024

1 Introduction

This document provides a detailed description of the system architecture for the Carbon Footprint Management System. The architecture is designed to be modular, scalable, and easy to maintain, utilizing microservices and modern web technologies.

2 Overview of the Architecture

The system architecture is designed with a microservices approach, where each service is independent and interacts with others through well-defined APIs. The architecture is composed of the following components:

- **Web Client (Angular):** The frontend of the application, developed using Angular, provides a user-friendly interface for users to interact with the system.
- **API Gateway:** Acts as a single entry point for all the requests coming from the Web Client. It routes the requests to the appropriate microservices, ensuring security, load balancing, and monitoring.
- **User Management Service:** Handles all operations related to user accounts, including registration, authentication, and profile management.
- **CarbonFootPrint Service:** The core service responsible for managing and processing carbon footprint data. This service interacts with MongoDB for data storage.
- **Reporting Service:** Provides reporting functionalities, generating various reports based on the data processed by the CarbonFootPrint Service. The Feign Client is used for seamless communication between the CarbonFootPrint Service and the Reporting Service.

- **MongoDB:** The NoSQL database used for storing all the data related to users and their carbon footprint records. It ensures efficient data retrieval and scalability.

3 Detailed Component Description

3.1 Web Client (Angular)

The Web Client is the user interface of the system, where users can log in, view their carbon footprint data, and generate reports. It is developed using Angular, a popular framework for building single-page applications. The Web Client communicates with the backend services via the API Gateway.

3.2 API Gateway

The API Gateway acts as the single entry point for all requests from the Web Client. It is responsible for routing requests to the appropriate microservices based on the request path and method. The API Gateway also handles cross-cutting concerns such as authentication, rate limiting, and logging.

3.3 User Management Service

The User Management Service manages all user-related operations, including user registration, authentication, and profile updates. It ensures that only authenticated users can access the system, and it securely stores user credentials.

3.4 CarbonFootPrint Service

This service is the heart of the system, responsible for managing and analyzing carbon footprint data. It receives data from the Web Client via the API Gateway, processes it, and stores it in MongoDB. The service also interacts with the Reporting Service through the Feign Client to generate reports based on the stored data.

3.5 Reporting Service

The Reporting Service is responsible for generating detailed reports based on the carbon footprint data. It communicates with the CarbonFootPrint Service via the Feign Client, ensuring that the reports are accurate and up-to-date.

3.6 MongoDB

MongoDB is a NoSQL database used for storing all user data and carbon footprint records. It is chosen for its scalability, flexibility, and ability to handle large volumes of data efficiently. MongoDB interacts directly with both the User Management Service and the CarbonFootPrint Service.

4 Architecture Diagram

The following diagram illustrates the high-level architecture of the Carbon Footprint Management System:

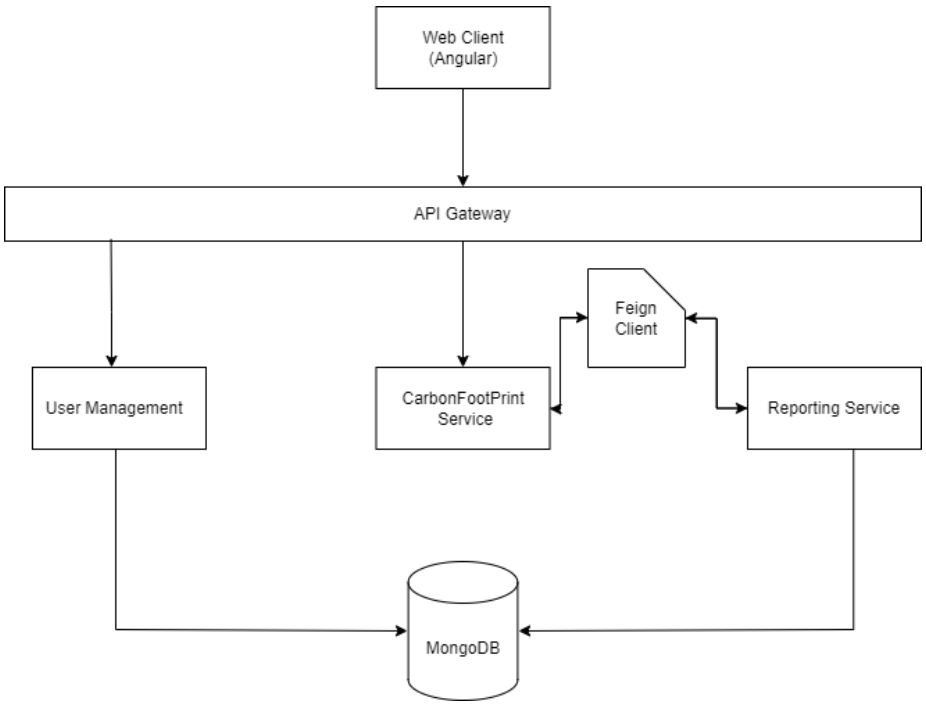

Figure 1: High-Level Architecture of the Carbon Footprint Management System

5 Conclusion

This architecture is designed to be modular and scalable, allowing for easy maintenance and expansion. Each component is independent and communicates with others via well-defined APIs, ensuring that the system is resilient and can handle large volumes of data efficiently.