Software Requirements Specification



A Carbon Footprint Tracking System for Logistics

Yash Vikas Shinde (70)

# Table Of Contents

1. Introduction
   1. Purpose
   2. Scope
   3. Definitions, acronyms and abbreviation
   4. References
2. Overall description
   1. Product perspective
   2. Product functions
   3. User characteristics
   4. Constraints
   5. Assumptions and dependencies
   6. Apportioning of requirements
3. Specific requirements
   1. External interface requirements
      1. User interfaces
      2. Hardware interfaces
      3. Software interfaces
      4. Communications interfaces
   2. Functional requirements
      1. User Class 1
      2. User Class 2
      3. User Class 3
   3. Performance requirements
   4. Design constraints

# Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

# Purpose

The purpose of this document is to outline the software requirements for the development of a “Carbon Footprint Tracking System for Logistics”. This system aims to help logistics companies, transportation providers, and supply chain managers to monitor, calculate, and reduce carbon emissions generated through goods transportation. The document defines the functional and non-functional requirements, system behaviour, interfaces, and constraints to guide development and ensure sustainability-focused outcomes.

# Scope

The Carbon Footprint Tracking System is a web-based application designed to compute and report carbon emissions from various logistics activities. The system allows users to input transport-related data such as distance, vehicle type, fuel type, and load weight to calculate emissions using verified carbon emission factors. It provides dashboards for real-time emission tracking, route optimization suggestions, and sustainability reports. The goal is to assist companies in achieving environmental compliance and making data-driven decisions to minimize their carbon footprint.

# Definitions, acronyms and abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition** |
| CO₂e | Carbon Dioxide Equivalent |
| GHG | Greenhouse Gas |
| GPS | Global Positioning System |
| API | Application Programming Interface |
| UI | User Interface |
| DEFRA | Department for Environment, Food and Rural Affairs (UK) |
| EPA | Environmental Protection Agency (USA) |
| SRS | Software Requirements Specification |

* 1. References

1. DEFRA. “Greenhouse Gas Reporting: Conversion Factors 2023.” UK Government. [Online].Available:https:/[/www.gov.uk/government/collections/government-](http://www.gov.uk/government/collections/government-)

conversion-factors-for-company-reporting

1. EPA. “Emission Factors for Greenhouse Gas Inventories.”

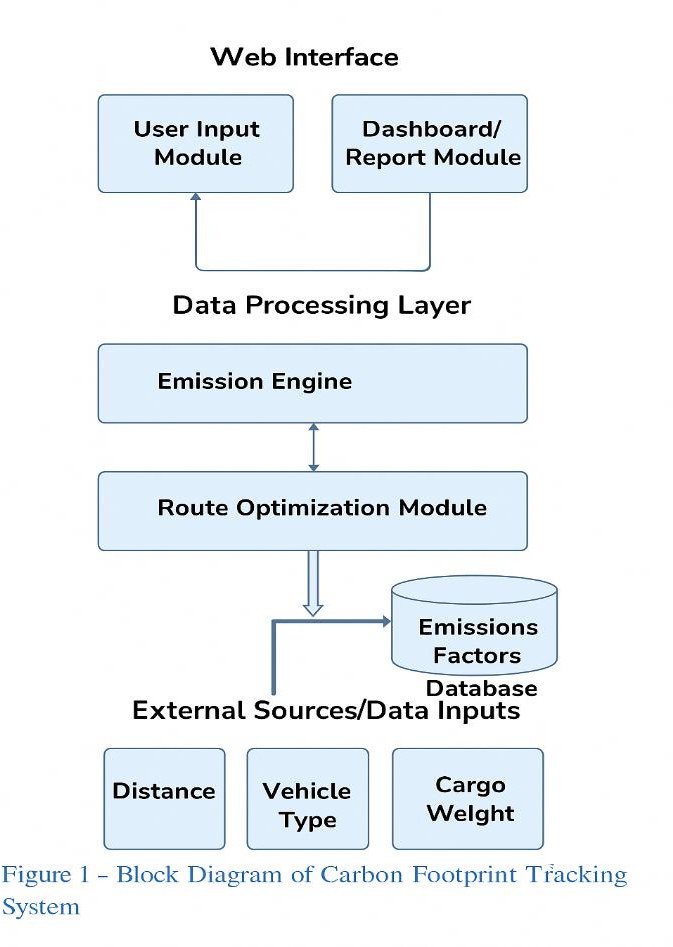
[Online]. Available: <https://www.epa.gov/climateleadership/ghg-emission-factors-hub>

1. IEEE Std 830-1998, “IEEE Recommended Practice for Software Requirements Specifications,” IEEE, 1998.
2. Google Maps Platform Documentation – <https://developers.google.com/maps>

# Overall description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

# Product perspective

The Carbon Footprint Tracking System is designed as a web-based application that assists logistics companies, transport service providers, and sustainability analysts in measuring and minimizing carbon emissions generated through

transportation activities. It is a self-contained product but also supports optional integration with third-party systems such as ERP (Enterprise Resource Planning), TMS (Transport Management Systems), and fleet tracking software.

The application operates using a modular architecture, consisting of:

* **Input Module** – for collecting data such as distance, fuel type, cargo weight, and vehicle type.
* **Emission Engine** – calculates CO₂ equivalent emissions using internationally recognized emission factors (e.g., DEFRA, EPA).
* **Optimization Module** – suggests alternative routes or modes of transport that result in lower emissions.
* **Analytics & Report Module** – generates downloadable, visual reports and dashboards for decision-making and compliance.

The system uses APIs (such as Google Maps API and emission factor datasets) to ensure accuracy in distance measurement and environmental data.

# Product Functions

The Carbon Footprint Tracking System is designed to offer a comprehensive suite of features that assist in monitoring and managing emissions across logistics operations. At its core, the system performs carbon dioxide equivalent (CO₂e) calculations based on user inputs such as travel distance, fuel consumption, vehicle type, and cargo load. These calculations utilize

emission factors sourced from internationally recognized standards, including those provided by DEFRA (UK) and the U.S. EPA.

In addition to basic calculations, the system incorporates real-time route analysis through the integration of mapping APIs. This feature enables users to evaluate various travel paths and identify routes that minimize emissions by avoiding congested or inefficient segments. The platform also includes a transport mode comparison tool, which allows users to assess the environmental impact of choosing between road, rail, air, or sea transport for a given shipment.

To enhance traceability and transparency, the system maintains a detailed emission history log. Every shipment processed through the system is archived along with its associated data, enabling long-term tracking and auditability. This data is made accessible through a dynamic dashboard, which visualizes trends and metrics using pie charts, bar graphs, and line charts, thereby providing intuitive insights into emission patterns.

Furthermore, the system is equipped with a report generation feature that produces downloadable PDF reports. These documents can serve as official carbon footprint certificates or be included in corporate sustainability reports. For system management, an administrative control panel is provided. It allows designated administrators to manage user access, upload updated emission factors, and configure third-party integrations. This structured, multifunctional approach ensures that the system is not only accurate and user-friendly but also scalable and adaptable to evolving environmental compliance needs.

# User Characteristics

The Carbon Footprint Tracking System is designed for four primary user groups: logistics managers, fleet operators, sustainability officers, and system administrators. Logistics managers use the system to oversee emissions across shipments and make informed decisions to optimize routes and reduce carbon output. Fleet operators or drivers are responsible for entering trip data such as distance travelled, vehicle type, and fuel usage, requiring a user- friendly interface and basic computer skills. Sustainability officers analyse emission reports and dashboards to ensure compliance with environmental standards and support ESG reporting. System administrators manage user access, update emission factors, and configure integrations with third-party services. All users are expected to have access to the internet and possess a fundamental understanding of digital tools to operate the system efficiently.

# Constraints

The Carbon Footprint Tracking System faces several key constraints that may impact its accuracy and functionality. First, the reliability of carbon emission calculations is heavily dependent on the accuracy of user-provided input such as travel distance, cargo weight, vehicle type, and fuel used. Additionally, if live API access is unavailable, the system must rely on static emission factor datasets, which require periodic manual updates to remain current. The application also depends on stable internet connectivity for real-time features like route optimization and map-based analysis. Since emission factors can vary regionally, the system is initially configured with UK and US standards (DEFRA and EPA), which may limit accuracy

in other regions unless localized data is added. Moreover, integration with third-party APIs such as Google Maps is subject to quota limits and may incur additional usage costs for organizations with high data traffic.

# Assumptions and Dependencies

The Carbon Footprint Tracking System is developed under several assumptions and external dependencies. It assumes that the emission factor sources used, such as those provided by DEFRA and the EPA, are accurate, standardized, and regularly updated to reflect current environmental data. The system also relies on users to provide complete and accurate input data, including travel distances, vehicle details, and fuel types, as this directly affects the accuracy of emission calculations. Additionally, the availability and responsiveness of external services like mapping APIs are critical for features such as route optimization and distance measurement. It is also assumed that organizations implementing the system have basic digital infrastructure in place, such as GPS tracking or shipment monitoring tools. Finally, the system presumes that standard logistics modes—such as road, rail, air, and sea—are part of the organization’s operational framework.

# Apportioning of requirements

To streamline development, the system requirements are distributed across multiple development phases:

## Phase 1 – Core Functionality

* User registration/login
* Emission calculator with static datasets
* Manual data input form
* PDF report generation

## Phase 2 – Dashboard and Analytics

* Visual representation of emissions
* Emission trends by date/vehicle/route
* Basic filtering and sorting

## Phase 3 – Route Optimization

* Google Maps integration
* Route suggestions based on emission reduction
* Transport mode comparison feature

## Phase 4 – Advanced Features

* API integrations with logistics ERPs
* Emission certificate generation with QR code
* Predictive analytics using ML
* Carbon credit compatibility

# Specific Requirements

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

# External Interface Requirements

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.

* + 1. User Interfaces

The system will provide a clean, structured, and user-centric web interface, accessible through modern web browsers such as Google Chrome, Mozilla Firefox, or Microsoft Edge. The primary user interface will consist of a homepage or dashboard that allows users to enter transport-related information including transport mode (e.g., road, rail, air, or sea), total distance travelled, fuel type (e.g., diesel, petrol, electric), and cargo weight. Each input field will use dropdowns, radio buttons, or numeric input boxes for intuitive data entry. Once the user clicks the "Calculate" button, the carbon footprint will be displayed below in the results area, showing the CO₂ emissions in kilograms or metric tons.

Administrative users will have access to a dedicated admin panel with capabilities such as uploading updated emission factor datasets, managing user roles and permissions, reviewing system logs, and configuring APIs or integration settings. The UI will also include a report generation module where users can download past emission records in PDF format. To enhance user experience, the interface will follow responsive design principles to ensure usability across desktops, tablets, and smartphones. Visual elements such as charts and graphs will be used in the analytics section to aid interpretation of emission trends. Accessibility features like keyboard navigation, proper label associations, and screen reader compatibility will be included to meet standard web accessibility guidelines (WCAG 2.1).

* + 1. Hardware Interfaces

The application does not require any specialized hardware to operate. For end-users, access to a basic internet-enabled device such as a desktop computer, laptop, or smartphone is sufficient. These client devices must be capable of running a modern web browser. On the server side, the application will be hosted on either a dedicated or cloud-based server environment, such as AWS, Azure, or a Linux server. The server must support the deployment of backend technologies such as Node.js, Python (Django or Flask), and database systems like PostgreSQL or MongoDB.

In advanced implementations, the system can interface with GPS hardware or fleet tracking devices to automatically capture real-time transport data. These optional integrations can

enhance the accuracy of emission calculations by eliminating manual input errors. The GPS data may be passed via external tracking systems through REST APIs or direct data streams into the backend system. However, these hardware interfaces are optional and not essential for core system operation.

* + 1. Software Interfaces

The system will interact with multiple software components and services to perform its tasks effectively. It will integrate with external APIs such as the Google Maps API to retrieve distance data, travel time, and alternate route information based on geographic coordinates or address inputs. For emission calculation, the application will use emission factor datasets provided by environmental bodies such as the Department for Environment, Food and Rural Affairs (DEFRA) in the UK or the Environmental Protection Agency (EPA) in the United States. These datasets may be accessed via public APIs or uploaded manually in CSV or JSON formats.

Internally, the application will be built using a combination of frontend frameworks like React.js or Angular for user interaction and backend frameworks such as Node.js or Django for processing logic. The backend system will include business logic for emission calculation, route optimization, and report generation. The system will also connect to a database management system (e.g., MongoDB or PostgreSQL), which stores historical emissions data, user credentials, and transport logs. Optional software interfaces may include integration with Enterprise Resource Planning (ERP) systems or Transportation Management Systems (TMS), allowing for automatic import of logistics data. All external and internal communications between components will follow RESTful API conventions using JSON-formatted requests and responses.

* + 1. Communication Interfaces

All communication between the user’s browser and the backend server will be carried out over the HTTPS protocol to ensure encrypted and secure data transmission. The system will use RESTful APIs for all client-server interactions, enabling consistent and stateless communication. Each module, such as the emission calculator or route optimization service, will expose specific API endpoints that can be called from the frontend. For example, the frontend will send a POST request containing transport data to the emission calculation API and receive a JSON response with computed CO₂ values.

The system may also communicate with external services such as Google Maps, vehicle tracking platforms, or emission factor databases via their respective APIs. These connections will be authenticated using secure API keys or OAuth tokens to prevent unauthorized access. In addition, the system can support outbound communication through email or push notifications. These may be used to send alerts regarding high emission levels, report summaries, or confirmation messages for data submissions. All communication interfaces will follow best practices for data encryption, token-based authentication, and error handling to ensure data privacy, integrity, and compliance with cybersecurity standards.

# Functional Requirements

This section includes the requirements that specify all the fundamental actions of the software system.

* + 1. User Class 1 – Logistic Manager
       1. Functional requirement 1.1

**ID**: **FR1.1**

Title: User Login

Desc: The logistics manager should be able to log in using a username and password. Rat: To access the system functionalities securely.

Dep: None

* + - 1. Functional requirement 1.2

## ID: FR1.2

Title: View Dashboard

Desc: The system shall display a dashboard showing total emissions, route emissions, and graphical analytics.

Rat: To help the logistics manager monitor and analyze emissions. Dep: FR1.1

* + - 1. Functional requirement 1.3

## ID: FR1.3

Title: Generate Emission Report

Desc: The logistics manager should be able to generate and download PDF/Excel reports of emission data.

Rat: For documentation and compliance with environmental reporting standards. Dep: FR1.2

* + 1. User Class 2 – Fleet Operator/Driver
       1. Functional requirement 2.1

## ID: FR2.1

Title: Fleet Operator Login

Desc: The fleet operator should be able to log in securely to the system. Rat: To ensure secure access to the emission calculator.

Dep: None

* + - 1. Functional requirement 2.2

## ID: FR2.2

Title: Input Transport Data

Desc: The user should be able to enter vehicle type, distance, fuel used, and cargo weight into the system.

Rat: Required for accurate carbon emission calculation. Dep: FR2.1

* + - 1. Functional requirement 2.3

## ID: FR2.3

Title: Calculate Emissions

Desc: The system shall calculate emissions based on the entered data and display the CO₂e value.

Rat: To provide immediate feedback on environmental impact. Dep: FR2.2

* + 1. User Class 3 – Sustainability Officer / ESG Analyst
       1. Functional requirement 3.1

## ID: FR3.1

Title: ESG Analyst Login

Desc: The sustainability officer should be able to log in and access analytics features. Rat: To ensure that only authorized users can access sensitive data.

Dep: None

* + - 1. Functional requirement 3.2

## ID: FR3.2

Title: View Emission Trends

Desc: The user should be able to view visual charts showing emission trends over time. Rat: To identify patterns and recommend improvements.

Dep: FR3.1

* + - 1. Functional requirement 3.3

## ID: FR3.3

Title: Filter Emission Data

Desc: The system should allow filtering by date, department, or transport type. Rat: For easier analysis of large datasets.

Dep: FR3.2

# Performance Requirements

The requirements in this section provide a detailed specification of the user interaction with the software and measurements placed on the system performance.

* + 1. System Response Time

## ID: PR1

Title: System Response Time

Desc: The system should respond to user actions such as form submissions or navigation within 2 seconds under normal network conditions.

Rat: To provide a smooth and responsive user experience. Dep: None

* + 1. Concurrent Users Support

## ID: PR2

Title: Concurrent Users Support

Desc: The system shall be able to support at least 500 concurrent users without significant performance degradation.

Rat: Ensures usability in real-time logistics environments with multiple users. Dep: None

* + 1. Emission Calculation Speed

## ID: PR3

Title: Emission Calculation Speed

Desc: The system shall compute and return the emission result within 1 second after the user submits transport data.

Rat: Quick feedback is essential for operational efficiency. Dep: FR2.2, FR2.3

# Design Constraints

This section includes the design constraints on the software caused by the hardware.

* + 1. Web-Based Architecture

## ID: DC1

Title: Web-Based Architecture

Desc: The system must be accessible through standard web browsers and should follow responsive web design principles.

Rat: To ensure cross-platform compatibility and ease of access. Dep: None

* + 1. Third-Party API Dependence

## ID: DC2

Title: Third-Party API Dependence

Desc: The system will rely on external APIs such as Google Maps for distance and route information, and on DEFRA/EPA datasets for emission factors.

Rat: External services must be integrated reliably with fallback handling in case of failure. Dep: FR2.2, FR2.3

* + 1. Technology Stack

## ID: DC3

Title: Technology Stack

Desc: The frontend must be developed using React.js or a similar modern JavaScript framework, and the backend using Node.js or Django.

Rat: Ensures maintainability and scalability of the codebase. Dep: None