

Software Requirements Specification (SRS)

EcoWanderNest: AI-Powered GIS Web Application

1. Introduction

1.1 Purpose

The purpose of this document is to specify the functional, non-functional, and system requirements for the development of *EcoWanderNest*, a GIS and AI-powered web application promoting sustainable tourism in Ahmedabad, India. It provides a framework for developers, project stakeholders, and testers to align on project deliverables.

1.2 Scope

EcoWanderNest is an interactive web-based platform that enables tourists to explore heritage locations while being mindful of their carbon footprint. It integrates GIS, AI/ML, and third-party data (e.g., OpenStreetMap, Google Reviews) to suggest optimized routes, summarize reviews, and calculate carbon emissions. Key features include:

- Route planning (shortest path, carbon-friendly path, and multi-stop optimization)
- AI-summarized reviews of places
- Personalized eco-dashboard with emissions tracker
- Voice assistant and chatbot integration

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition
GIS	Geographic Information Systems
AI/ML	Artificial Intelligence / Machine Learning
OSM	OpenStreetMap
CO ₂	Carbon Dioxide
OTP	One-Time Password
TSP	Travelling Salesman Problem
JWT	JSON Web Token
NLP	Natural Language Processing
API	Application Programming Interface
IPCC	Intergovernmental Panel on Climate Change

1.4 References

- IPCC Guidelines for Emission Factors
- Google Reviews (for tourist feedback)
- Booking.com Sustainability Report (2023)
- IEEE Conference Proceedings on EcoTravel & Route Optimization

1.5 Overview

This document contains the overall system description, functional specifications, external interface requirements, performance, security, and usability requirements. It also includes appendices defining formulas and emission benchmarks.

2. Overall Description

2.1 Product Perspective

EcoWanderNest is a standalone web application structured around:

- A frontend using HTML, CSS, JavaScript, Tailwind CSS
- A backend using FastAPI (Python), PostgreSQL with PostGIS
- AI modules (HuggingFace transformers, LangChain)
- GeoServer for route visualization

It communicates via REST APIs and depends on data fetched from third-party services.

2.2 Product Functions

- Tourist site search with place details
- Route planning (shortest, eco-friendly, and multi-stop)
- AI-based review summarization
- Personalized dashboard (eco-score, emissions, trip history)
- Voice assistant and chatbot for hands-free usage

2.3 User Classes and Characteristics

- **Tourists:** Users planning eco-travel within Ahmedabad. Require intuitive UX.
- **Admins:** Manage database content, monitor system events, handle feedback.

2.4 Operating Environment

- **Frontend:** HTML5, Tailwind CSS, JavaScript, OpenLayers
- **Backend:** Python (FastAPI), PostgreSQL/PostGIS
- **External APIs:** Twilio (OTP), Google Reviews (via Selenium), HuggingFace Transformers, LangChain

2.5 Design and Implementation Constraints

- Works only in Ahmedabad (Phase I)
- Uses static emission factors (no real-time transit data)
- Requires stable Internet connection

2.6 User Documentation

- In-app help section
- Onboarding tooltips
- FAQ page (future scope)

2.7 Assumptions and Dependencies

- External APIs (Google, Twilio, IPCC) remain accessible
 - Server hosting environment supports FastAPI and PostgreSQL
-

3. Specific Requirements

3.1 Functional Requirements

3.1.1 User Registration & Authentication

- Email/Phone-based signup with OTP verification
- Secure password storage (bcrypt)
- JWT-based session management
- Role-based access control (Admin/User)

3.1.2 Map Interface & Search

- Real-time map view using OpenLayers
- Search bar with autocomplete and place preview
- Display place details: name, description, rating, hours, coordinates, image

3.1.3 Routing Features

- **Shortest Route:** Dijkstra's algorithm for source → destination
- **Eco-Friendly Route:** Weighted cost function includes emissions
- **Multi-Stop Route:** TSP optimization using pgr_tsp

- Emissions shown by colour code:
 - Green: Short, low emission
 - Yellow: Moderate
 - Red: High emission

3.1.4 Emissions Tracking & Eco Score

- Emissions = Distance × Emission Factor
- Threshold benchmark: 113 g CO₂/km (Indian CAFE standard)
- **Eco Score** = $100 - (\text{Actual Emissions} / \text{Max Emission}) \times 100$
- Dashboard shows:
 - Emissions saved
 - Net carbon impact
 - Vehicle-wise emission breakdown
 - Emissions over time (trend line)

3.1.5 AI-Driven Reviews

- Review summarization via BART model
- NLP filtering of key points
- Rating visualization (bar chart 1–5 stars)

3.1.6 Chatbot & Voice Assistant

- Chatbot built on LangChain + Ollama LLM
- Voice recognition via Web Speech API
- Text-to-speech support
- Location-based travel recommendations

3.1.7 Personalized Dashboard

- Greeting with username
- Trip planner: add/view trips, mode, destination
- Analytics: total distance, carbon savings, map visualization
- Top visited locations
- Delete past data (reset account)

3.2 Non-Functional Requirements

3.2.1 Performance

- API responses within 2 seconds
- Map loads under 3 seconds
- Concurrent session support: 100+

3.2.2 Security

- Encrypted password storage
- JWT tokens with expiration
- Role-based access to endpoints
- CORS-enabled secure API access

3.2.3 Usability

- Responsive design (mobile and desktop)
- Accessibility features (text-to-speech, alt text)
- Tooltip onboarding guide

3.2.4 Reliability

- System uptime > 99%
- Error logging and crash recovery

3.2.5 Maintainability

- Modular code structure (FastAPI services)
- Logging with alerts for critical actions
- Database normalization with foreign keys

3.2.6 Portability

- Deployable using Docker
- Compatible with Windows/Linux hosting

4. External Interface Requirements

4.1 User Interfaces

- Home: login/signup + explore
- Dashboard: trip planner, emissions stats
- Map: live interaction, route colouring.
- Popups: detail windows for tourist places
- Chat UI: text/voice interaction

4.2 Hardware Interfaces

- Client-side device (mobile, desktop) with browser
- Server-side (cloud/VPS with Python + PostGIS)

4.3 Software Interfaces

- PostgreSQL 14 + PostGIS
 - FastAPI REST endpoints
 - Twilio OTP API
 - HuggingFace Transformers
 - Selenium (Google Reviews Scraper)
 - LangChain + Ollama LLM
-

5. Appendices

A. Emission Factors (g CO₂/km)

Transport Mode	Emission
Bike	26.6
Car	223.6
Bus	515.2
EV/Walk	≈ 0

B. EcoScore Classification

- Green (70–100): High sustainability
- Yellow (40–70): Moderate
- Red (< 40): Low sustainability

C. Key Formulas

1. **Carbon Emission (g CO₂)** = Distance (km) × Emission Factor
2. **Emission Saved** = Threshold Emission – Actual Emission
3. **EcoScore** = 100 – (Actual / Max Emission) × 100
4. **Net Carbon Impact** = Σ Emission Saved – Σ Actual Emissions

Business Requirements Document (BRD) – EcoWanderNest

1. Executive Summary

EcoWanderNest is a GIS and AI-powered web application designed to promote **sustainable tourism in Ahmedabad**. It helps users discover heritage sites and eco-friendly travel options while tracking and reducing their carbon footprint.

2. Business Objectives

- Encourage eco-conscious travel behavior
- Provide tourists with optimized, carbon-efficient travel plans
- Showcase city heritage while promoting sustainability
- Use AI to enhance user experience (e.g., review summarization, chatbot)



3. Stakeholders

- Tourists (end users)
- City Tourism Board
- Application Administrators
- Developers and Data Scientists
- Environmental NGOs

4. Business Requirements

ID	Requirement
BR-1	The system must allow users to plan eco-friendly routes in Ahmedabad
BR-2	The platform should provide AI-based summaries of tourist site reviews
BR-3	The user must be able to track their travel-related carbon footprint
BR-4	Voice and chatbot interaction must be available for ease of use
BR-5	Tourists must see a personalized eco-dashboard with emissions data

5. Scope

-  In-Scope: Route planning, review summarization, emissions tracking, dashboard, chatbot
-  Out-of-Scope: Real-time public transport integration, international destinations

6. Assumptions

- Third-party APIs (Google, Twilio) are accessible
- Internet access is stable during usage
- User base is limited to Ahmedabad in Phase I

7. Constraints

- Static emission factors (no live transport data)
- Must run in browsers (desktop and mobile)

8. Approval

This document is to be reviewed and approved by stakeholders from the city's tourism board and the product development team.

Functional Requirements Specification (FRS) – EcoWanderNest

1. Introduction

FRS defines the core **functions and behaviors** of EcoWanderNest that fulfill the business requirements.

2. Functional Modules

ID	Functional Requirement	Description
FR-1	User Registration & Login	Signup with phone/email, OTP verification, JWT-based sessions
FR-2	Route Planning	Shortest, eco-friendly, and multi-stop route options using GIS algorithms
FR-3	Tourist Place Search	Interactive map, autocomplete search, display place info
FR-4	Emissions Calculator	Calculates carbon emissions per route based on distance and transport type
FR-5	Eco Dashboard	Shows emission history, eco score, savings, and trip stats
FR-6	AI Review Summarizer	NLP-based summary of Google Reviews using BART model
FR-7	Chatbot and Voice Assistant	Text and voice-based interaction, location-aware responses
FR-8	Admin Module	Manage tourist site database, monitor activity logs, handle feedback

3. User Roles

- **Tourist:** Access all eco-features, dashboard, and route planner
- **Admin:** Backend management of places, emissions benchmarks, and user analytics

4. Interface Expectations

- Interactive map with OpenLayers
- Dashboard with data visualizations (charts, metrics)
- Chat window and voice icon for assistant

5. Non-functional Expectations (Briefly Covered)

- 🕒 Fast loading (APIs < 2s)
- 🔒 Secure sessions and data encryption

-  Accessibility & responsive design