# PROJECT

(SEMINAR)

# ECO FRIENDLY TRAVEL PLANNER

Instructor – Mr. Naveen Sharma

H.O.D. – Dr. Rita Chhikara

Submitted by:

Vallika Dhawan(22CSU185)

Himanshi Yadav (22CSU085)

Mitali Dogra(22CSU296)

A Report

On

**ECO FRIENDLY TRAVEL PLANNER**

Submitted to

Mr. Naveen Sharma

30th April 2025

TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| S.NO | CONTENTS | PAGE NO. |
| 1. | Abstract | 4 |
| 2. | Introduction | 5 |
| 3. | Tech Stack | 6 |
| 4. | Implementation | 8 |
| 5. | Results | 10 |
| 6. | Conclusion | 14 |
| 7. | References | 15 |

ABSTRACT

The eco-friendly travel planner is designed to optimize travel routes while minimizing environmental impact, providing a sustainable alternative to traditional travel planning. By leveraging Dijkstra's algorithm and the planner computes routes based on user preferences, focusing on the shortest distance and time efficiency. The system promotes eco-friendly travel options by considering factors such as the choice of low-emission vehicles and the encouragement of sustainable transportation methods. With a focus on reducing travel time and environmental impact, the planner adapts to individual travel needs, enhancing convenience while promoting sustainable choices. This project aims to support global sustainability goals by reducing carbon footprints and encouraging more eco-conscious travel habits.

INTRODUCTION

In recent years, the importance of sustainable transportation has grown significantly, as environmental concerns and the need to reduce carbon emissions have become key global priorities. Traditional travel planning systems typically focus on optimizing routes for speed and convenience without factoring in the environmental impact of the travel choices. This gap highlights the need for intelligent systems that balance efficiency with sustainability. The eco-friendly travel planner developed in this report aims to address this issue by providing an optimized travel routing solution that minimizes environmental impact while meeting user preferences for travel time and distance. The system utilizes Dijkstra’s algorithm to calculate the shortest and most time-efficient routes to ensure dynamic and responsive planning. By prioritizing distance and time, the planner offers an efficient alternative to conventional travel methods.

In addition to optimizing for travel efficiency, the planner incorporates eco-conscious factors such as promoting sustainable routes. It also allows users to avoid routes that contribute significantly to higher carbon emissions, offering a route selection that is both practical and environmentally responsible.

This project aims to contribute to sustainable travel practices by developing a system that not only offers users optimal travel routes but also encourages the reduction of carbon footprints through informed decision-making. The ultimate goal is to support global sustainability efforts by providing a tool that facilitates greener travel choices in everyday commuting.

TECH STACK

The development of the Eco-Friendly Travel Planner utilized a comprehensive and efficient set of technologies, tools, and frameworks to implement a scalable, maintainable, and performance-driven system. The key components of the tech stack are as follows:

**1. Programming Languages**

**Java**: Used for backend development, including the implementation of business logic and route optimization algorithms.

**2. Development Tools**

* **Eclipse IDE**: Primary tool for backend development, particularly with Java and Spring Boot integration.
* **Visual Studio Code (VS Code)**: Used for frontend development and API testing, offering extensive plugin support and faster UI development workflows.

**3. Frameworks and Libraries**

* **Spring Boot**: A lightweight Java-based framework for creating RESTful backend services. It provides dependency management, embedded servers, and simplifies overall application configuration.

**4. Architecture**

* **Frontend–Backend Architecture**: Ensures a clear separation of concerns, with the frontend interacting with the backend via RESTful APIs.
* **RESTful APIs**: Facilitate communication between client and server, enabling modularity and extensibility.

**5. Algorithm and Data Structures**

* **Dijkstra’s Algorithm**: Used for finding the shortest or most time-efficient path between locations, considering user preferences.
* **Adjacency List**: A memory-efficient data structure used to represent the graph of connected locations for pathfinding.

**6. User Interface**

* **HTML/CSS/JavaScript**: Standard technologies used for building the web interface, providing users with interactive route planning features.

IMPLEMENTATION

**Backend Architecture**

The backend of the Eco-Friendly Travel Planner is implemented using **Java** and the **Spring Boot** framework. The application follows a layered architecture, dividing responsibilities among various packages: controller, service, repository, model, configuration, and web socket.

**Package Breakdown**

**1. Controller Package**

This package handles HTTP requests and maps them to appropriate services.

* **LocationController.java**: Manages location-related operations.
* **RouteController.java**: Handles route requests, including route generation using Dijkstra's algorithm.
* **StopController.java**: Manages stop-related queries such as rest areas, restaurants, and petrol pumps.
* **UserController.java**: Manages user-related functionalities such as login, registration, and preferences.

**2. Model Package**

Contains the domain models used throughout the system.

* **Graph.java & Edge.java**: Represent the graph and its connections for pathfinding.
* **Location.java, Route.java, Stop.java**: Core entities for the travel planning logic.
* **User.java**: Represents user information and preferences.
* **Other models** like PetrolPump.java, Restaurant.java, Restroom.java, and StopFacilities.java are used to enhance travel planning with nearby stop suggestions.

**3. Repository Package**

This layer interacts with the database and abstracts the data persistence logic.

* Repositories like LocationRepository.java, RouteRepository.java, UserRepository.java, etc., extend Spring Data JPA interfaces and enable CRUD operations.

**4. Service Package**

Contains the business logic and acts as a bridge between controllers and repositories.

* **RouteService.java**: Implements Dijkstra’s algorithm using the Graph and Edge models to calculate the optimal path based on distance or time.
* **StopService.java**: Handles logic for finding nearby rest areas, petrol pumps, and restaurants.
* **UserService.java**: Contains logic for user management and validation.

**5. Configuration Package**

Handles configuration-related files.

WebSocketConfig.java and WebSocketHandler.java:

Configure Web Socket support for real-time communication (e.g., real-time route updates or tracking).

**6. Web socket Package**

RouteWebSocketHandler.java: Manages the Web Socket connections for broadcasting route updates to the frontend in real-time.

**7. Main Application**

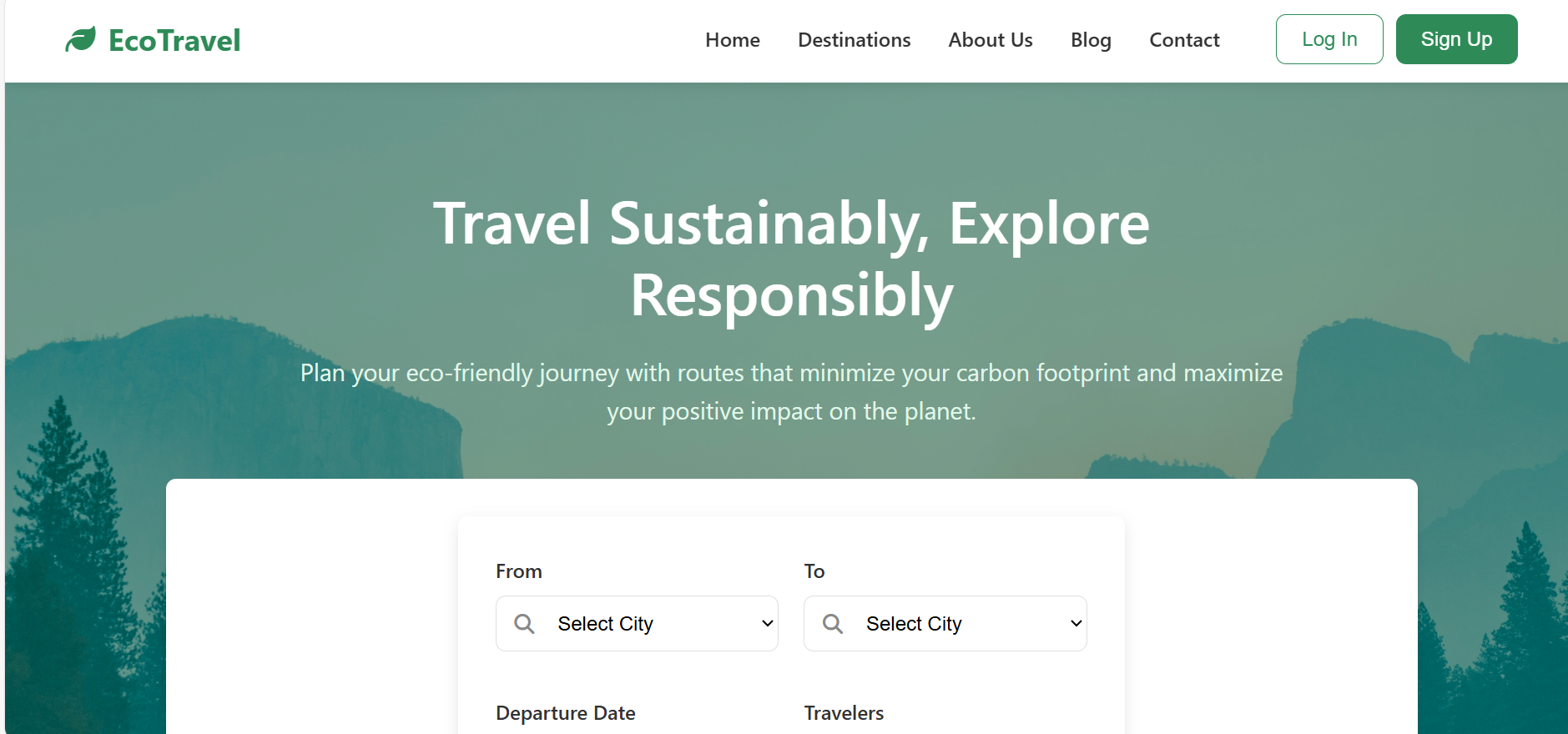
TravelPlannerBackendApplication.java: The main entry point of the Spring Boot application. It initializes the entire application context

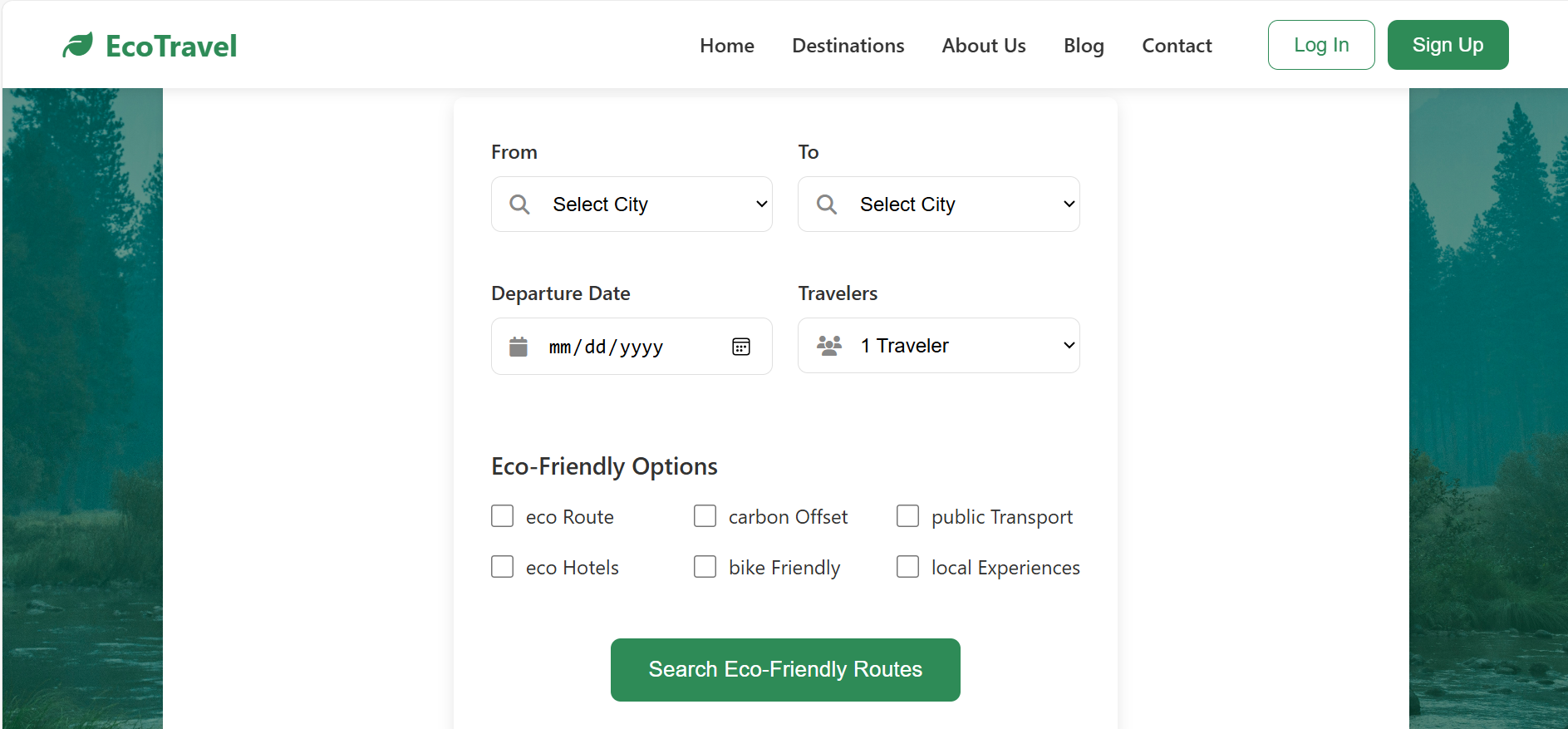
**Core Functionality – Dijkstra’s Algorithm**

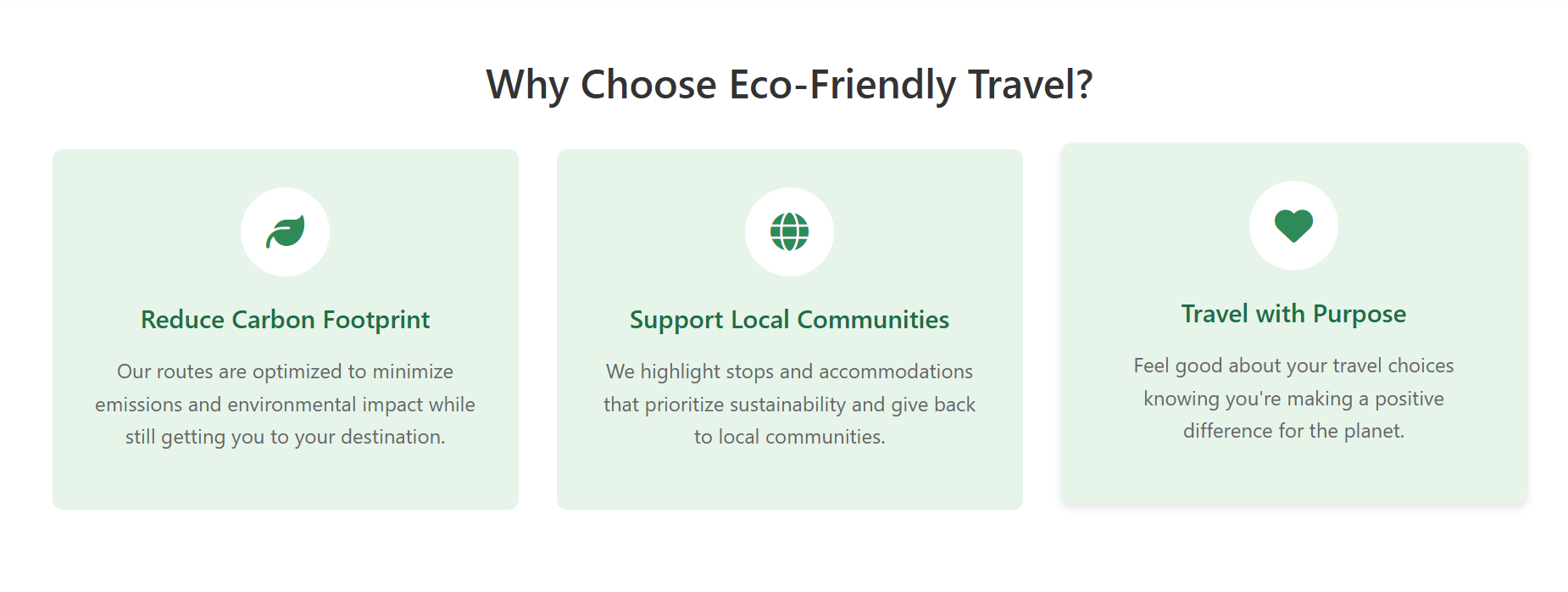
The route optimization is performed in RouteService.java using **Dijkstra’s Algorithm**, which is implemented on a **graph model with an adjacency list**:

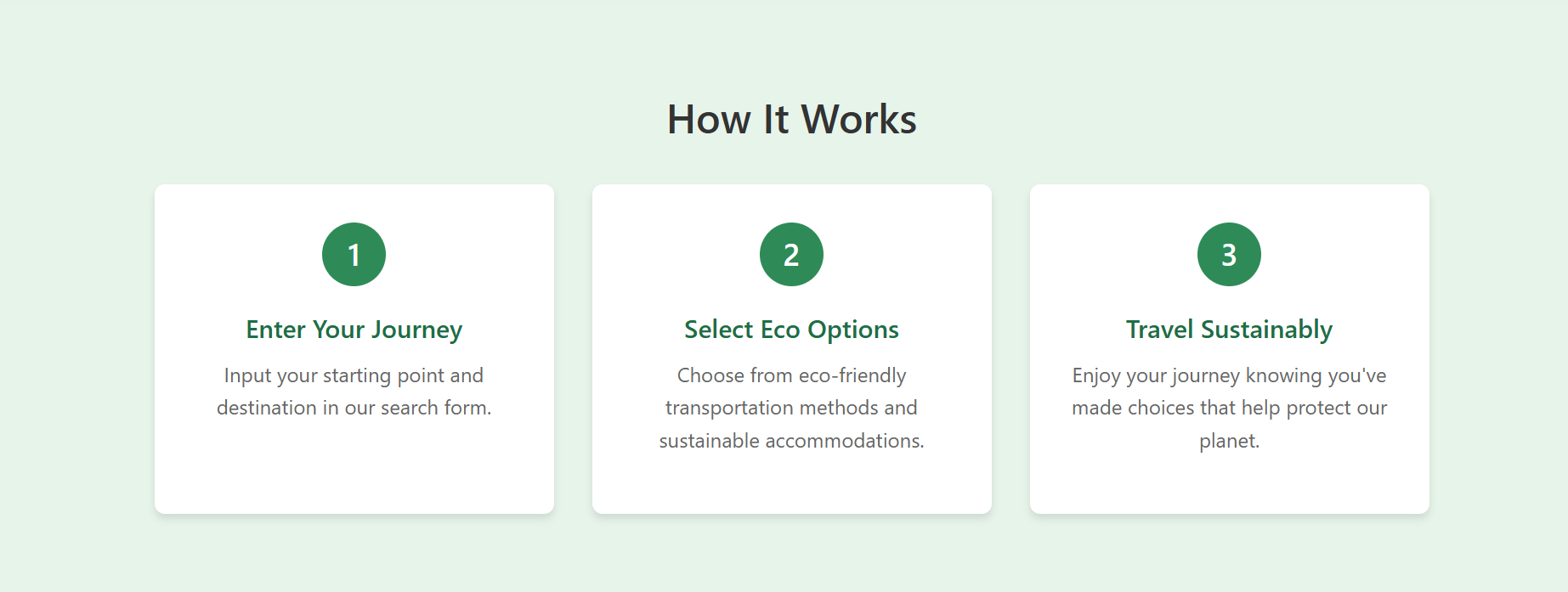
* Each node represents a Location.
* Each edge contains a weight corresponding to either distance or estimated travel time.
* Based on user preference (shortest distance or shortest time), the algorithm dynamically adjusts edge weights and computes the optimal path.

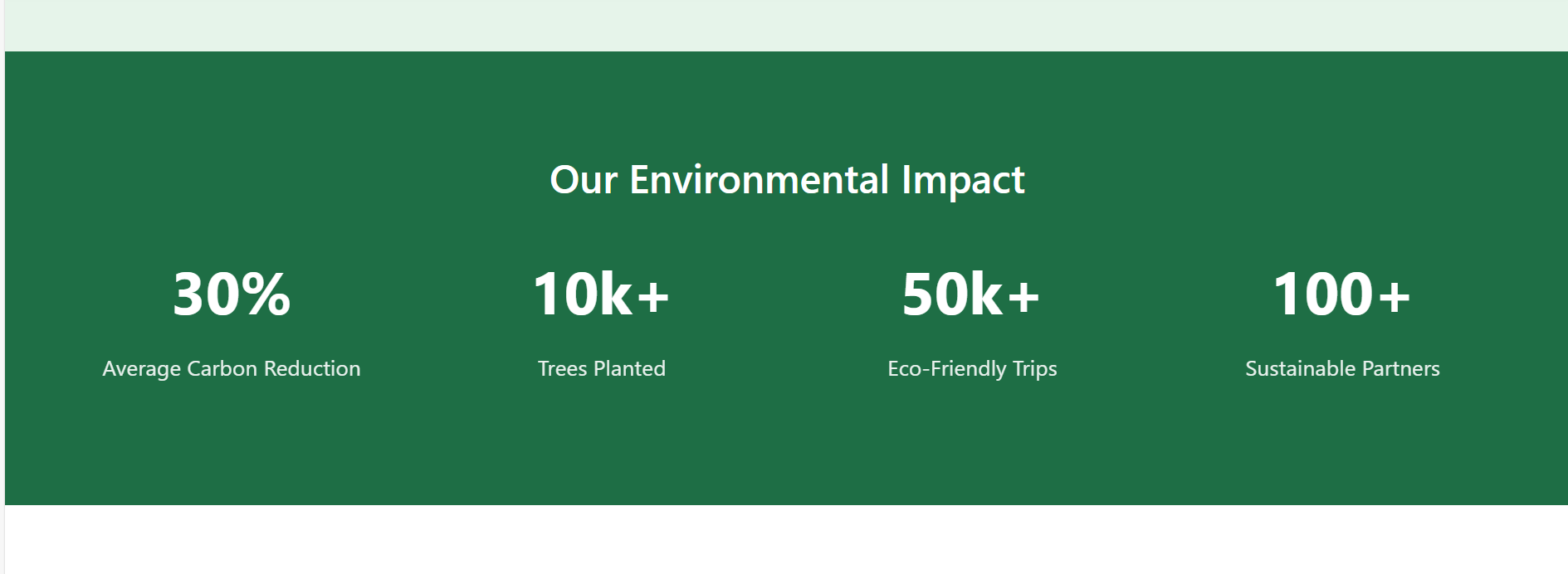
RESULTS

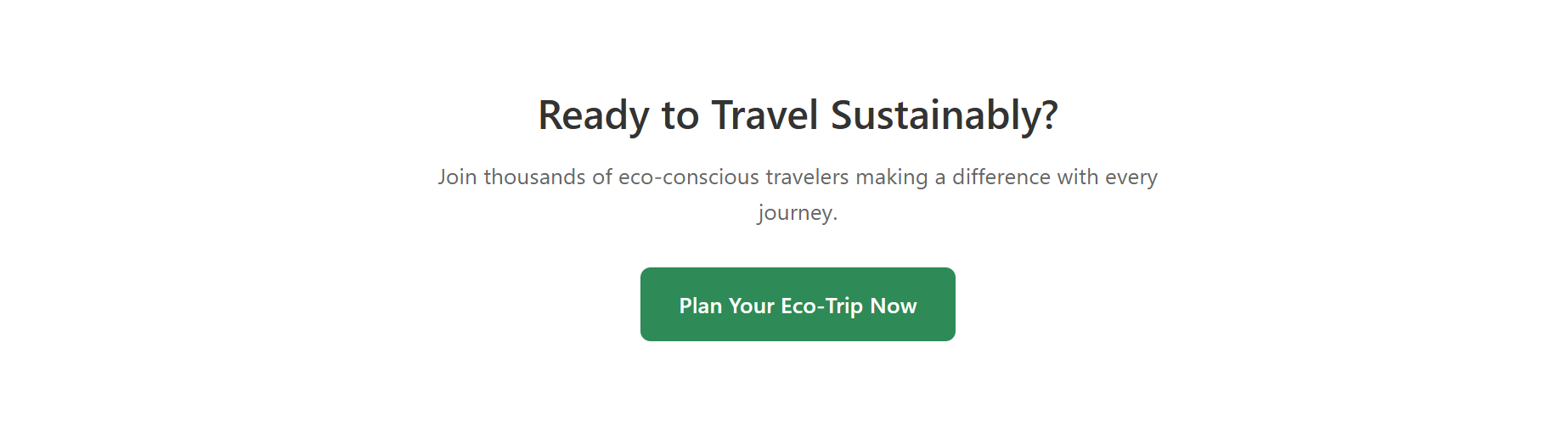


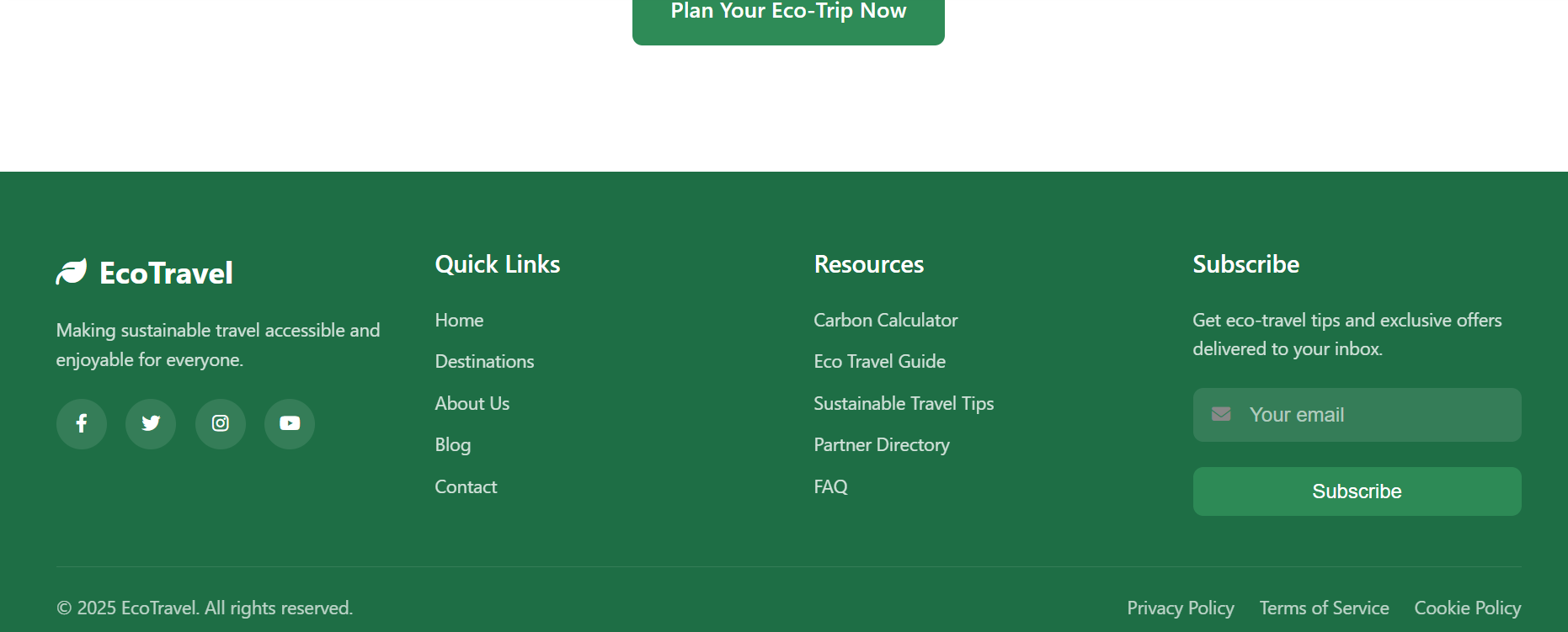


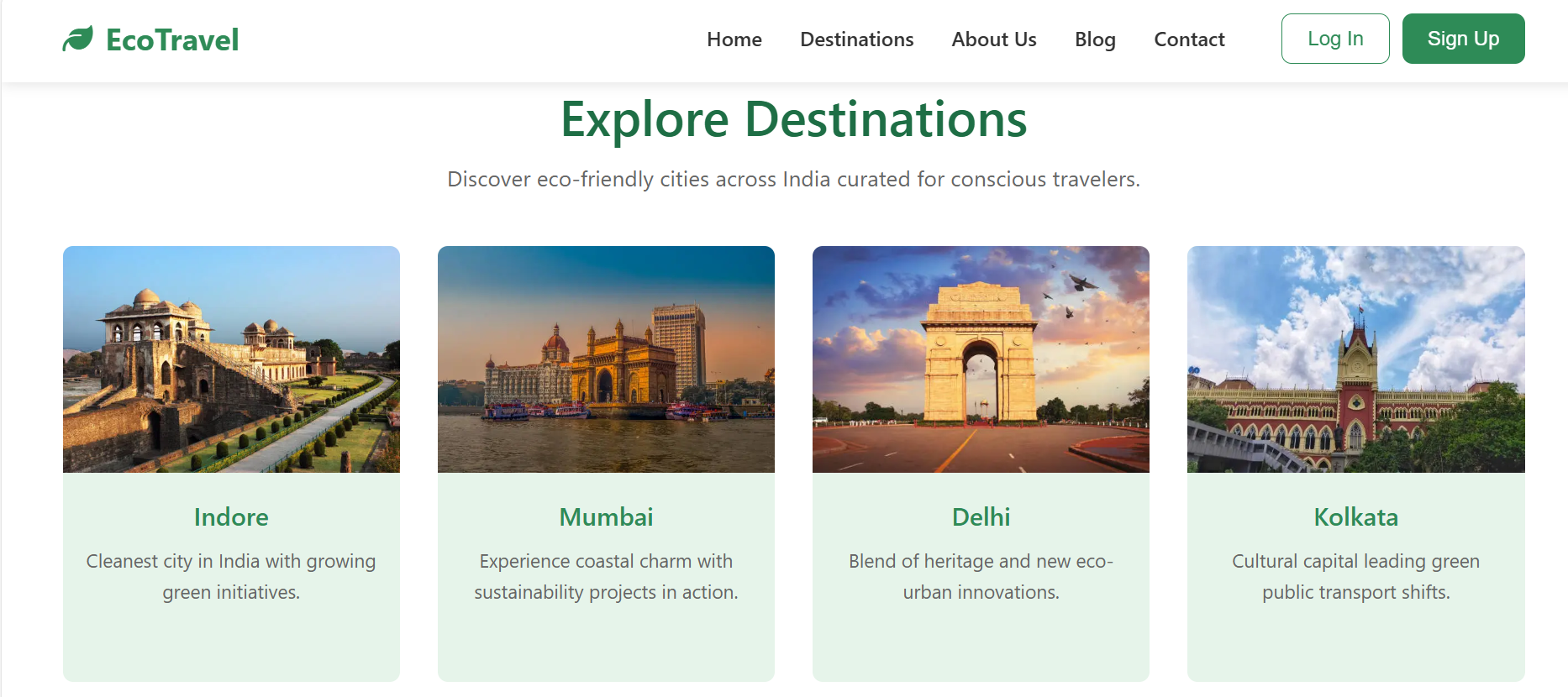


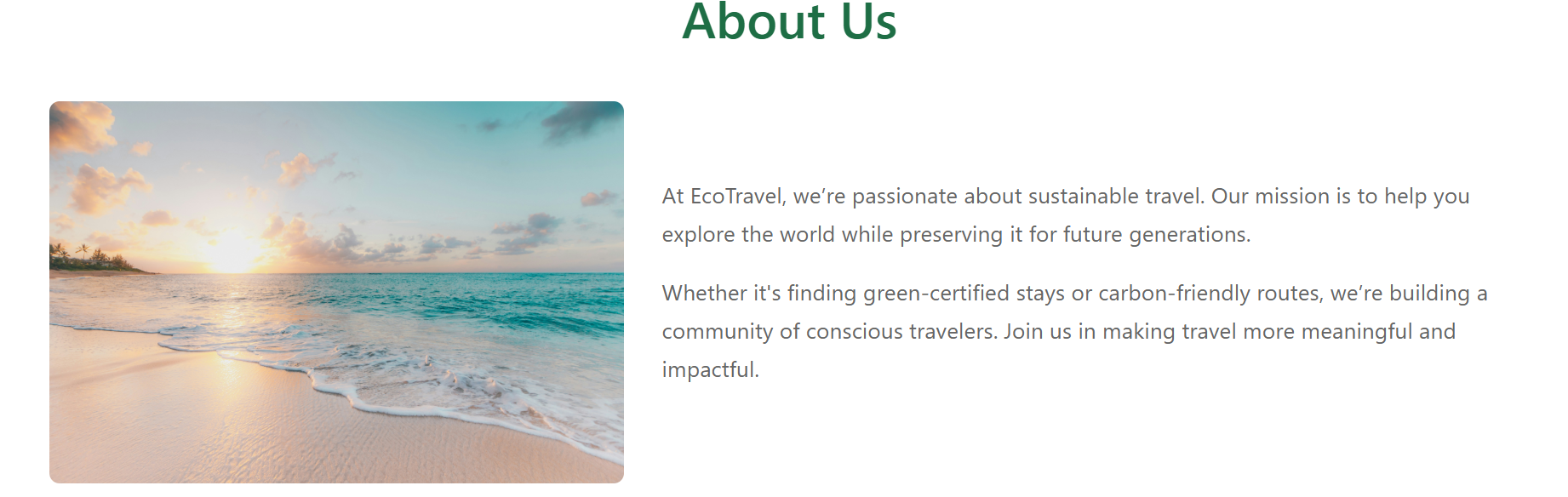


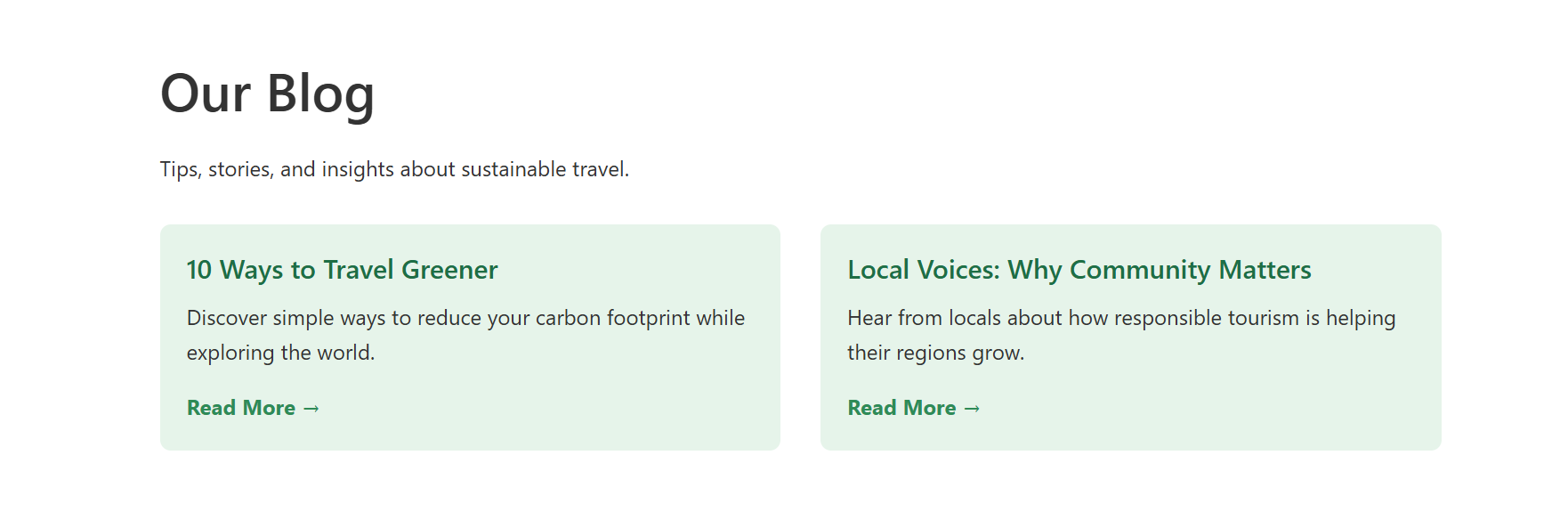


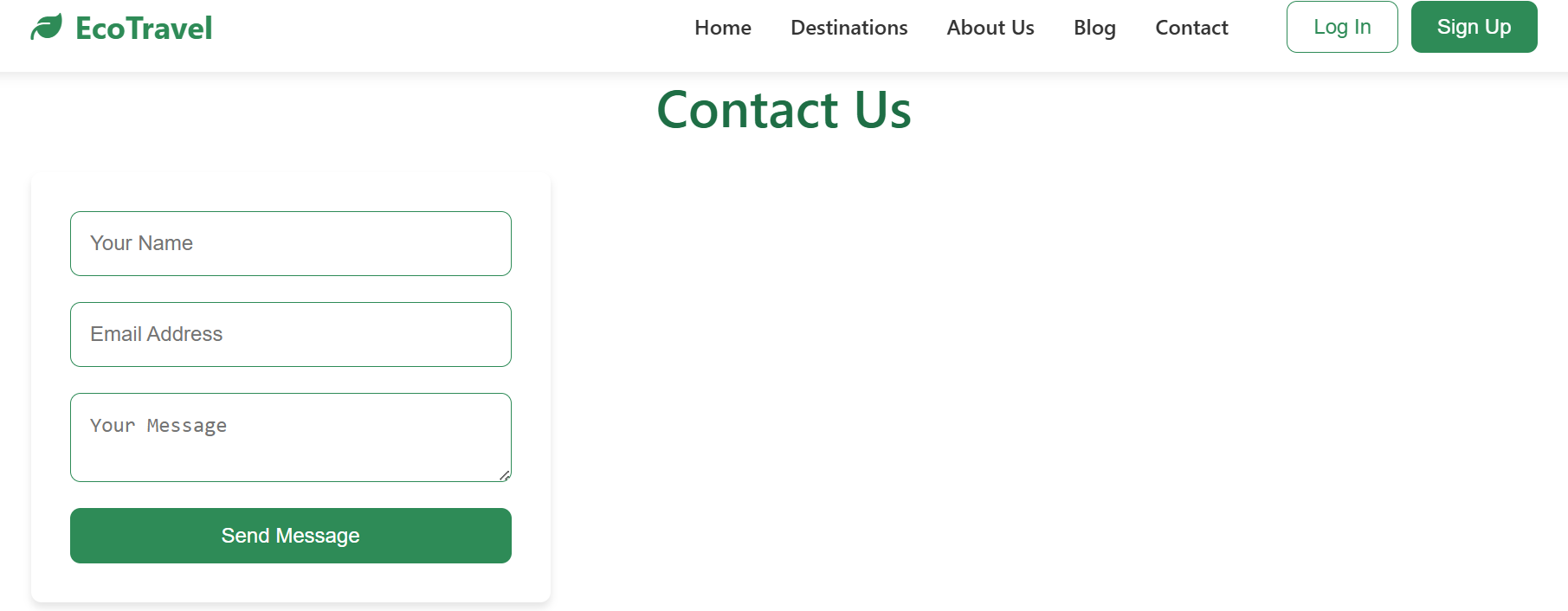


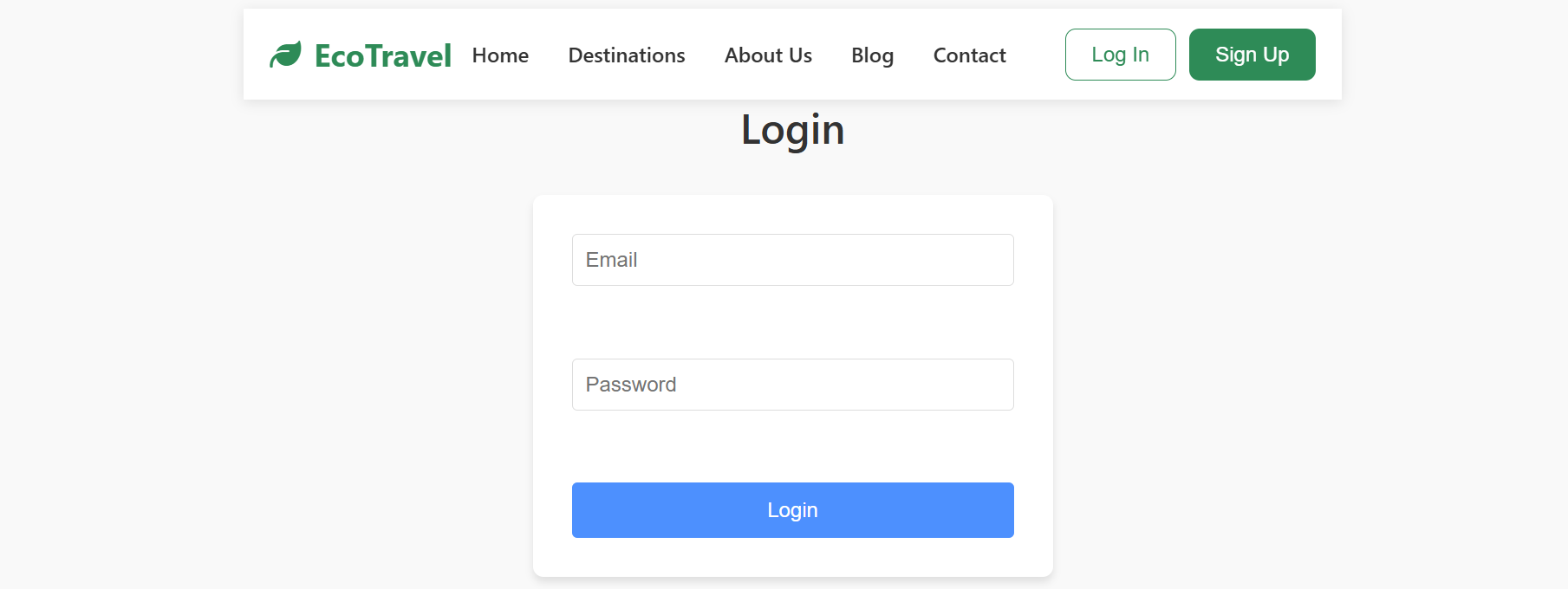


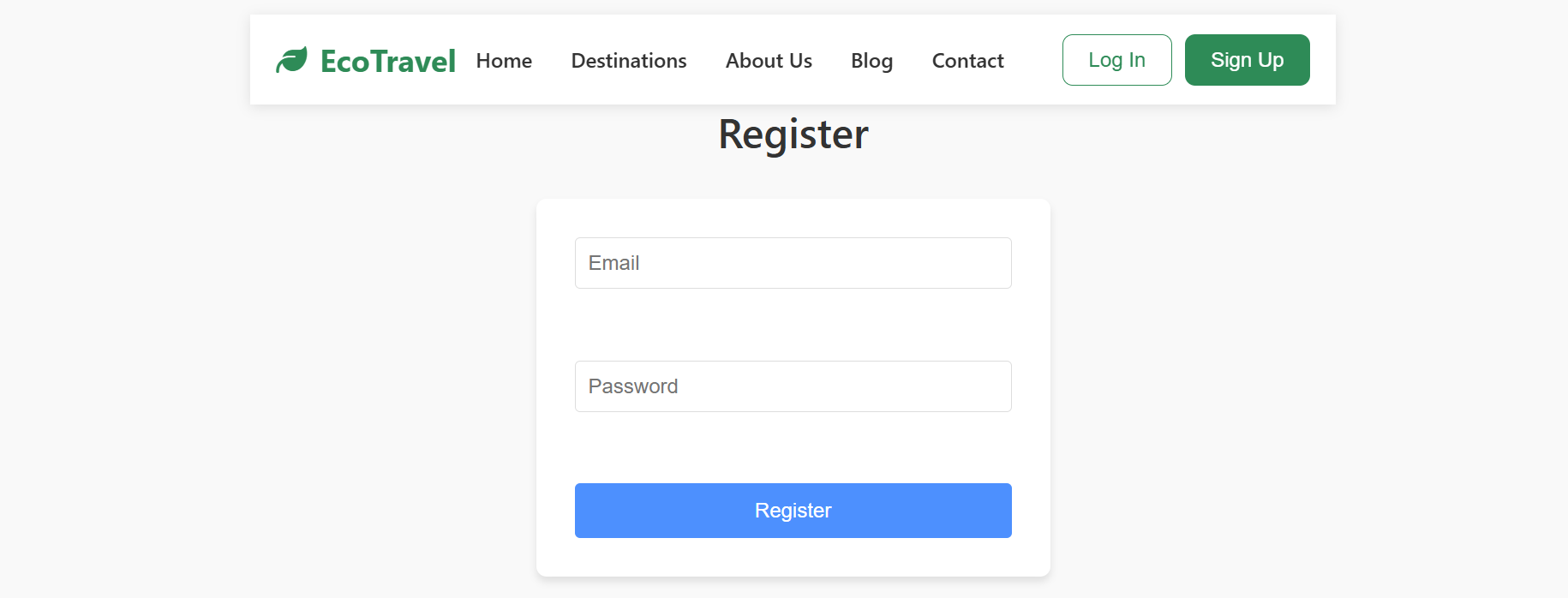












CONCLUSION

The Eco-Friendly Travel Planner successfully demonstrates an efficient and modular approach to modern travel planning by leveraging well-established technologies and algorithms. The project integrates a robust **Java-Spring Boot backend** with a dynamic **JavaScript frontend**, ensuring seamless communication through RESTful APIs. By implementing the **Dijkstra algorithm** over an adjacency list-based graph structure, the system accurately computes optimal travel routes based on user preferences such as minimum distance and minimum travel time.

Throughout the development, a structured software architecture was maintained, comprising distinct layers for **controllers, services, models, and repositories**, which enhances code maintainability and scalability. Tools like **Eclipse** and **VS Code** were utilized to facilitate efficient backend and frontend development, respectively.

Although the current system focuses primarily on optimizing distance and time, it establishes a strong foundation for future enhancements such as the incorporation of scenic routes, toll avoidance, or real-time traffic data integration. Overall, this project offers a technically sound, user-centric solution aligned with the principles of smart, sustainable travel, while also providing a platform for continuous innovation in route optimization and eco-friendly planning.

REFERENCES

1.Dijkstra, E. W. (1959). *A Note on Two Problems in Connexion with Graphs.*

2.Spring Boot Documentation – *Building Applications with Spring Boot.*

3.MDN Web Docs – *JavaScript Guide.*

4.Vite Documentation – *Next Generation Frontend Tooling.*

5.REST API Tutorial – *Introduction to REST APIs.*

6.GeeksforGeeks – *Graph Data Structure And Algorithms.*

7.W3Schools – *Frontend Development (HTML, CSS, JS).*