Smart Carbon Emission Tracker: A Data-Driven Approach for Greener Multi-modal Transportation

Habibur Rahman[1] and Vahid Javidroozi[1]

[1] Birmingham City University, STEAMhouse, Belmont Row, Birmingham, B4 7RQ, United Kingdom

Abstract. The study covers the research, development and evaluation of a smart and sustainable transportation tracker using data from transportation sectors to successfully calculate carbon emissions across various modes of transport such as walking, bicycling, car, and public transit while attempting to reduce emissions that they produce through route planning. The project aims to provide users with transportation routes that minimise carbon emissions, thereby aiding efforts to combat climate change. Key features of the application include integration with the Google Directions API for retrieving real-time traffic data and an interface that presents users with detailed information about route distance, duration, carbon emissions, and alternative transportation modes. The prototype undergoes rigorous testing, including both black box and white box methods, to ensure functionality and user-friendliness. The project's significance lies in its potential to influence commuter behaviour towards more sustainable practices, a critical step in reducing urban carbon footprints. The project supports the UK's objective of achieving net zero emissions by 2050, enhancing both academic and practical approaches to incorporating environmental data into daily technologies and enhancing individual awareness of their carbon footprint. The project lays a foundation for ongoing research and development in the field of smart, sustainable urban transportation.

Keywords: Carbon Emissions, Sustainable Transportation, Python, Carbon Footprint, Smart Transportation, Environmental Impact, Smart City

1 Introduction

The increasing threat of global warming due to greenhouse gases, particularly carbon emissions, has become a significant global concern. Studies indicated that carbon emissions, which increase by 2.5% annually, are a major driver of climate change. In the UK, the transport sector is a key contributor to this issue, accounting for 40% of the nation's total CO2 emissions [1]. With the UK government committing to achieve net zero emissions by 2050 [2], the decarbonisation of transport is crucial. While various solutions such as electric and hydrogen-powered vehicles [3] and the controversial HS2 high-speed rail project [4] have been proposed, there remain concerns about the effectiveness and long-term sustainability of these initiatives. Furthermore, as private transport continues to dominate the UK's transportation methods [5], efforts to

encourage more sustainable practices, including public transportation and low-emission alternatives, are increasingly vital. However, behavioural factors and the attractiveness of private transport pose challenges to achieving widespread adoption of sustainable transportation [6]. Consequently, the integration of smart city technologies into urban transport systems has been explored as a potential solution to improve efficiency and reduce emissions [7]. Nonetheless, while advancements in smart transportation technologies offer various benefits, more focus on environmental sustainability is needed to align these innovations with the broader goal of reducing carbon emissions.

This project explores the transportation sector to find a solution for tracking carbon emissions from various modes of transport by developing a smart sustainable transportation tracker for carbon emissions. The purpose of this project is to close the gap in research involving carbon emissions in the transport sector and to discover a solution that provides alternative routes that reduce carbon footprint in the transport sector.

1.1 Problem Definition

Carbon emissions and sustainability have emerged as one of the greatest challenges of our time. The undeniable reality of climate change is resulting in the rise of temperatures by 1.5 - 5.8°C across the globe, extreme weather such as heat waves, floods, and droughts [8]. The UK specifically has created the Climate Change Act of 2008 which aims at a reduction of 80% of emissions by 2050. This was later amended by the government to reach net zero emissions by 2050 with Prime Minister Boris Johnson saying, “The UK will be home to pioneering businesses, new technologies and green innovation as we make progress to net zero emissions” [9].

Regarding the transport sector specifically, it has been shown that carbon emissions from transportation have been identified as one of the highest contributors to climate change in the UK alone accounting for “34% of all territorial carbon emissions” [10]. This demonstrates the impact that the transport sector alone has on carbon emissions in the UK. Addressing these issues is paramount, not only to mitigate the adverse effects of climate change but to create a brighter future that is both sustainable and efficient.

Despite advances in technology and increasing environmental awareness, there remains a significant gap in the integration of carbon emissions data into everyday decision-making tools such as navigation apps. While some applications provide route optimisation based on time or traffic, few consider the environmental impact of these routes directly. This gap is particularly pronounced in the context of urban transport where decision-making tools can significantly influence commuter behaviour towards more sustainable practices.

This project directly contributes to the reduction of urban carbon emissions by encouraging low-carbon transport choices. It empowers users by providing them with the information needed to make environmentally responsible choices, aligning with growing consumer preferences for sustainability. The application offers valuable data that can assist travel planners in understanding and promoting more sustainable transportation habits and the research adds to the academic and practical understanding of integrating environmental data into everyday technology, providing a foundation for further research and development.

1.2 Project Aim and Objectives

The aim of this research is to develop a smart and sustainable prototype solution for reducing carbon emissions using data from the transportation sector to provide routes using various modes of transport. The following objectives will be addressed in this research:

1. Identify methods for tracking carbon emissions within the transportation sector. This analysis will uncover the principles of carbon emissions and how they are managed in the transport sector. This will ensure an effective prototype is built that is relevant to transportation.
2. Identify various technologies used for providing routes of transportation that have been demonstrated to be effective at planning journeys, while holding focus on their practicality and reduction in carbon emissions.
3. Identify the best method of collecting data on carbon emissions from transportation. This will aid in developing the technology that creates routes that are low in carbon footprint through various modes of transport.
4. Develop a functional prototype application that provides a route via various modes of transport based on carbon emissions data that reduces an individual's carbon footprint. This should include walking, bicycling, car, and public transport.
5. Measure the effectiveness of the application by carrying out a mixed-method survey. This will evaluate the level of success in the prototype outlining its success, areas of development, and future development suggestions.

Addressing these objectives provides several innovative contributions to the field of smart transportation and sustainability. Firstly, the development of the Transportation Tracker prototype introduces a novel approach to integrating real-time carbon emissions data into route planning, filling a crucial gap in current navigation technologies. Unlike existing platforms that prioritise speed or traffic, this system places environmental impact at the forefront, empowering users to make eco-friendly transportation choices. The study also offers significant research value by exploring the intersection of urban mobility and environmental sustainability, providing a data-driven solution that promotes behavioural change towards low-emission transport. Furthermore, the project contributes to the broader academic discourse on smart city technologies by emphasising the need for tools that align technological advancements with climate goals. Through the development and rigorous testing of this prototype, the research not only advances practical applications in transport decarbonisation but also lays a foundation for future innovations that could further enhance sustainability in urban transport systems.

2 Literature Review

The following five key themes are studied to enhance understanding of the topic and develop a working prototype for this project:

2.1 The impact of carbon emissions in the UK

The greenhouse gases are known to trap heat in the Earth's atmosphere causing climate problems and global warming [11]. Carbon emissions are the biggest factor in global warming which shows an increase of 2.5% per year indicating the concern of carbon emissions [12].

Many different contributors of carbon emissions impact the UK with the biggest being energy and transport. [13] explains that both the "energy and transport-related industries are associated with substantial direct emissions providing the most carbon-intensive inputs." According to [1], the UK produces around "1.7 million tonnes of CO2 per year" and 40% of it is associated with the transport industry evidencing the dominance that the transport industry holds in terms of carbon footprint. Carbon emissions have only been increasing over the years and due to this, the UK has created a plan to reduce emissions. [2] explained that 75% of local authorities in the UK have declared a "climate emergency" which ultimately led the UK government to create a goal of reaching net zero carbon emissions by 2050. Whether or not this plan can realistically be achieved is also a concern that [14] have explored and justified that a strategy with all greenhouse gas emitters and the extraction of atmospheric CO2 should be implemented on a large scale.

Focusing on transportation alone the government has created a decarbonisation plan for transport and many solutions towards this goal have been studied with [3] highlighting the use of both electric and hydrogen-powered transport which would significantly reduce carbon emissions. One controversial solution for reducing carbon emissions is the high-speed railway (HS2) which [4] calculates to reduce "25 million tonnes of CO2 over the next 60 years." However, [15] argued that just the construction of the HS2 alone creates a "carbon debt" that will not be repaid for over 60 years. This essentially shows that while solutions are being made there are many concerns as to how they are being created.

[3] elaborate further on the decarbonisation of transport by including human lifestyle factors such as considering electric vehicles as alternative modes of transport. Behavioural factors were also included such as the use of walking and cycling for short-distanced travels. Individually changes like these would be miniscule but as a whole UK population, the change would create a huge impact on the reduction of carbon emissions. [16] investigated human consumption, behaviours, and lifestyle and concluded that these factors have been "insufficiently addressed in the research on climate change mitigation." While both [3] and [16] express different opinions, they both ultimately believe that these changes on a large scale can effectively reduce carbon emissions not only in the UK but across the world. While the arguments of how transport decarbonisation can be achieved and the various evaluations created, it has consistently shown that the UK has not taken enough action to reduce carbon emissions overall in the transport sector highlighting the need for further action to be taken place.

2.2 UK Transportation Methods

According to the [5], the most used transportation method is private transport which includes the use of cars, vans, and taxis with public transportation being the secondary choice. Fig. 1 shows the passenger kilometres travelled by cars, vans, and taxis

increased steadily to a record high of 738 billion in 2019. However, 2020 marked a significant decline, dropping 27% to 536 billion passenger kilometres. Rail travel reached a of peak 16 billion passenger kilometres in 2018. However, there was an 80% decline in 2020. Buses and coaches experienced a long-term decline, with 2019's distance at 33 billion passenger kilometres, less than half of 1960's 79 billion. In 2020, there was a 58% decrease to 14 billion passenger kilometres. Other modes remained relatively stable from 1960 at 20 billion passenger kilometres. However, in 2020, there was a 29% decrease to 14 billion passenger kilometres.

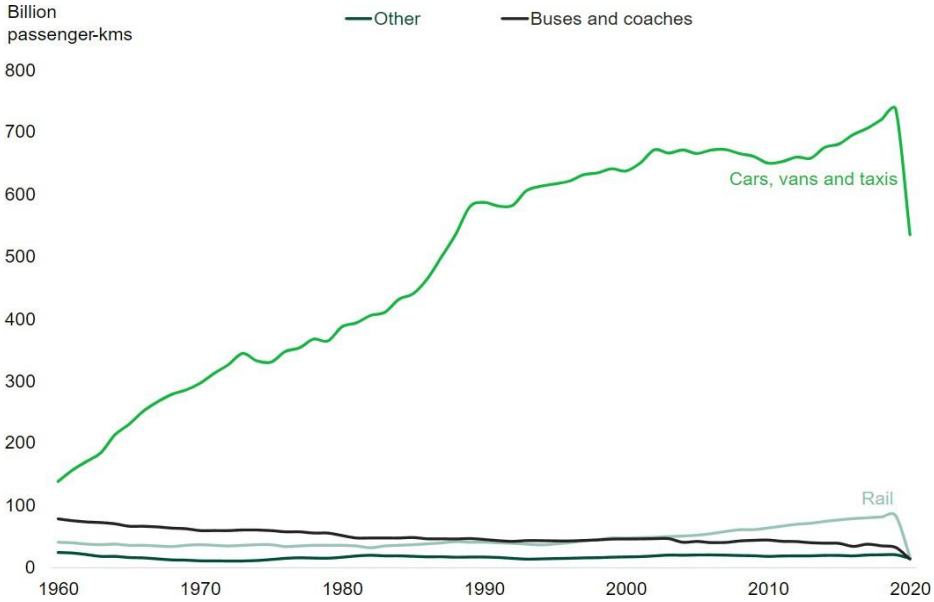


Fig. 1. Chart of passenger kilometres by mode, Great Britain, 1960 to 2020 [5]

The reason for the decline around 2019-2020 is due to the impact of the coronavirus which introduced a nationwide lockdown in the UK which naturally made transport decline. This is further elaborated and supported by [6] that private modes of transport including cars vans and taxis are dominant as they're "perceived to be attractive" due to factors such as comfort, independence, flexibility, speed, and much more. They also agree that public transport such as buses, rails, walking, and cycling decline at a steady rate as factors such as age and income increase. Studies by [17] also show evidence of private transport being dominant however they argue that public transport is just as important as it provides an alternative mode of transport attracting labour and visitors for companies. As shown in Fig. 2, it is evident that there has been a decrease in emissions from modes such as cars, taxis, buses, and other types of emissions.

Mode	1990	2019	% change
Cars and taxis	72.3	67.7	-6%
HGVs	20.5	19.5	-5%
Vans	11.6	19.2	65%
Buses	5.3	3.1	-42%
Domestic shipping	8.5	6.1	-29%
Other emissions	10.0	6.7	-33%
International aviation	15.5	37.0	138%
International shipping	8.1	7.5	-8%

Fig. 2. Greenhouse gas emissions by transport mode, 1990 and 2019, in mtCO2e [5]

However, there are still concerns that while the UK government does report reductions in emissions, recent progress in reducing emissions has been limited and if climate targets are to be met, significant progress is required [18]. [19] support this by also outlining the need to strategically identify sectors that are required to reduce their emissions. As discussed previously, the UK has introduced a net zero emissions plan and within that, they have taken some actions by decarbonising transport. Based on the UK's actions [20] conveyed that the ban on all sales of new petrol/diesel-powered cars in 2035 will leave a gap in the market where alternative methods of transport can be introduced such as battery electric vehicles which are perceived as producing zero emissions when used.

2.3 Smart City Transportation

The term "smart city" has been used since the 1990s to denote cities that are technologically and scientifically advanced. [21] emphasise that smart cities address various aspects of urban life, including governance, economy, living, environment, and transportation. According to [22], smart cities aim to enhance sustainability, quality of life, and citizen services through technology. They encompass social, economic, and environmental sustainability, with environmental sustainability being a primary objective achieved through technological integration [23].

According to [7], smart transportation is defined as the use of sensors embedded into vehicles offering a variety of solutions such as optimised route suggestions that use data from various sources such as mobile devices or sensors placed throughout roads which collect data on roads to estimate traffic congestions to provide the most optimal route to a destination with consideration to both time and distance. [24] elaborated on smart transportation and focused on traffic congestion, but they focused on creating an intelligent parking system using variables such as date, time, weather, distance, and much more. [25] explored the challenges of using real-time data to create smart transportation. While all these attempts at creating smart transportation satisfy the

objectives of a smart city in the sense that they are advanced technology, improve quality of life, and create a smart living, they are not dedicated to directly being environmentally sustainable and reducing carbon emissions.

2.4 Carbon Emissions Data

Developing a solution that aligns with reducing carbon emissions in transport requires reliable data on transport. [26] discusses the use of technology to obtain accurate carbon emissions data by following a two-step process of transport modelling, to provide data on traffic flow, journey lengths, and selected modes of transport combined with the analysis of fuel consumption which includes vehicle type, traffic conditions, environmental conditions, and driver behaviour. Similarly, [27] created a methodology that uses a combination of models which are the International Vehicle Emissions (IVE) model, COPERT model and country-specific emissions factors.

Both [26] and [27] prove similar solutions in collecting carbon emissions using models and estimations but they cannot continuously collect data such as real-time solutions. [28] resolve this by creating a solution for drivers that monitors a vehicle using GPS data. This solution is centred around providing insights on how drivers can become eco-friendly drivers by producing fewer carbon emissions through changes such as behaviours, travel routes, travelling in certain conditions, and much more as shown below in Fig. 3.

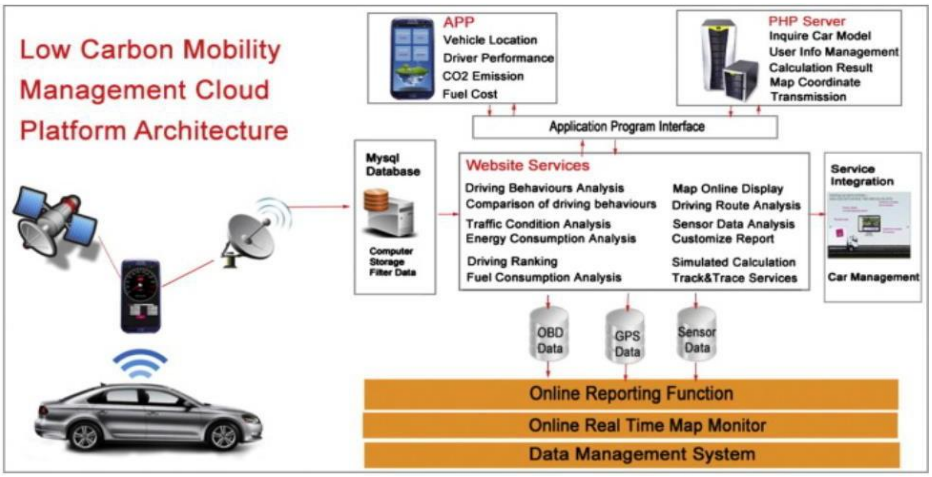


Fig. 3. Low carbon mobility management cloud platform architecture [28]

[29] take real-time carbon emissions data collection a step further by producing a multi-scale emissions calculation of walking, cycling, bus, car, and railways. They produce a solution which provides a visualisation of carbon emissions from various modes of transport. Different variables have also been considered when calculating emissions such as time, travel route, and weather. While these solutions show carbon emissions with various methods of calculations and data processing, they do not act upon eliminating emissions from transport and instead focus on the contribution towards knowledge.

2.5 Navigation Applications

Global Positioning Systems (GPS) are widely used across many sectors to track activity and detect patterns. In the context of transport, GPS is used to monitor transport vehicles and map out transportation networks.

Movement that is collected from GPS is stored in the form of a trajectory which provides an outline of a subject from the origin to the destination [30]. [31] outlines the shift towards mobile applications is continuously growing with navigation applications to be used by 77% of smartphone owners with 87% primarily using navigation applications for driving directions.

As [32] discussed, these applications use route planning algorithms to create a path from the origin to the destination, with the shortest path algorithm being the most common algorithm. The separation between route planning a journey on the road and public transit networks was made due to their distinct variables. Public transport is time-dependent while route planning for private transport can be created at any time. [31] indicate some of the most common applications that use similar concepts of route planning are Google Maps, Waze, and Apple Maps. Google Maps and Apple Maps are described to be very similar with updates being delivered regularly and temporary events releasing within 48-72 hours. Waze is set differently from these applications as it focuses only on cars and no other modes of transportation like Google and Apple Maps, such as navigation for walking, cycling, taxis, buses, railways, and metro.

According to [33], the popularity of these applications is derived from the feature to calculate the shortest route such as walking or cycling and more with Google Maps being outlined as creating "the shortest route possible for each vehicle that wants to travel from one location to another." However, [34] also support the idea that navigation applications provide information on routes based on the shortest path, traffic, speed, time, and other conditions but they also noted that applications do help users calculate their carbon footprint indicating that these applications do make some users carbon aware. [35] expand on route planning with the plan to reduce carbon emissions for logistics including transport however, they focus on reducing carbon emissions for logistics over all sectors.

2.6 Key findings from existing literature

According to the literature review, table 1 summarises the key findings, emphasising the comparative aspects of the studies, by showing how different pieces of research are connected, their areas of focus, and the critical discussions around carbon emissions, transportation, smart city technology, and emissions data:

Table 1: Key findings from existing literature

Key Themes	Authors	Main Points	Comparison and Insights
Impact of Carbon Emissions in the UK	[1][3][11-14] [16]	- GHGs trap heat, leading to climate change. - Carbon emissions in the UK increase by	Some authors (e.g. [3]) focus on decarbonising transport, but others [15] highlight issues

		2.5% per year. - 40% of CO2 emissions come from transport [1]. - Electric/hydrogen transport is suggested as a solution.	like "carbon debt" from infrastructure projects like HS2. [16] bring in the importance of behavioural changes.
UK Transportation Methods	[6][17-20]	- Private transport is dominant due to comfort and independence [6]. - Public transport declines with age/income. - Emissions reductions are not on - track to meet climate targets [18].	Debate exists between the need for better public transport and the slow progress in transport decarbonisation. Some authors see electric vehicles as the solution, while others push for hydrogen-powered transport.
Smart City Transportation	[7][21, 22][24, 25]	- Smart cities aim to enhance the quality of life through technology. - Smart transportation uses sensors for route optimisation. - Intelligent parking systems and - real-time data for smart transport.	The smart city focus is on technological advancements, but concerns arise around their direct impact on environmental sustainability [25]. These systems need to contribute to carbon reduction.
Carbon Emissions Data	[26-29]	- Carbon emissions models estimate transport emissions, but real-time monitoring is better for continuous data. - GPS data can promote eco-driving behaviour. - [29] focus on multi-scale emissions data.	Continuous real-time data collection [28] provides more accurate tracking compared to static models [2], bridging the gap between emissions data and actionable insights.
Navigation Applications	[30-35]	- Navigation apps calculate shortest routes and reduce travel time. - Some applications,	Navigation apps can reduce emissions by promoting efficient routes, but their effectiveness is limited

	like Google Maps, now consider carbon footprints. - Waze focuses exclusively on - Private car navigation.	by user choices. Public transport navigation is seen as important but underdeveloped in these apps.

3 Project Design and Methods

The project design and methods section will discuss the methodology of this research in four main data collection and development phases including literature review, software development, testing, and validation.

3.1 Literature Search Methodology

Literature in various forms from articles, books, journals, reports, and more were reviewed in this research. To ensure all sources are valid, a variety of academic databases, including "IEEE Xplore" and "Science Direct" were used. To cover the main themes of this project, search terms used were "Impact of Carbon Emissions," "Carbon Emissions in the UK," "UK Net Zero Emissions," "UK Transportation Methods," "Low-Carbon Transport," "Sustainable Transportation," "Smart City Transportation," "Smart City Goals," "Smart City Sustainability," "Carbon Emissions Data," "Carbon Emissions Collection Methods" "Processing Carbon Emissions Data," "Carbon Emissions Data in Python," "Navigation Applications," and "Python Navigation Algorithms". Using these themes, several search terms were created, using Boolean operators.

Furthermore, filters on various databases were used such as publication dates (to be between 2016 and 2024), only peer-reviewed articles published in English, and the rigour of their methodology utilised.

3.2 Software development methodology

This research utilised agile software development methodology, due to its flexibility allowing this project to revisit different stages for improvements throughout the project. Furthermore, if any requirements in the development of this project do change then these can be easily implemented.

Design Specification/User Requirements. The solution will ultimately achieve the aims and objectives of this project and to ensure this is properly achieved expected user requirements have been created which are based on the project literature review, limitations, and options. Table 2 outlines the requirements, description, and priority level.

Table 2: The requirements, description, and priority level of the application software

Requirements	Description	Priority
Graphical User	This must be created to allow users to interact with the	High

Interface (GUI)	application. The GUI should be simple and easy to interact with.	
Route planning	The application must be able to plan a route from a starting point to a destination. The route planning system should take into consideration the time, location, traffic/emissions data, and type of transportation. Once these variables are calculated a route is created for the user.	High
Multiple Transport methods	The application creates routes based on multiple transportation methods. To implement this only transport such as walking, cycling, vehicles and railways will be considered. Data on their emissions, efficiency, usage, and many more relevant variables are combined with the route planning system to provide the user with a route that is the lowest in carbon emissions and uses the best possible transport method in the scenario.	High
Map API	The application is created using Python and with the integration of maps API, the application can implement real-time data based on navigation such as traffic, location, time, and transport data. This data can be used to optimise routes to ensure they provide accurate and reliable routes.	High
Traffic Information	Once the route has been calculated and planned it outputs the route to the user. Extra information on their generated route can be provided to ensure the user has the best route provided for them. This information mostly includes any complications that they could face on their route such as delays, congestion and fees that could apply.	Medium

Concept Solution. Upon accessing the application, the user will load directly into a page to input the starting point and location. Once the information has been input the program does all the necessary calculations as shown in the diagram to provide the most optimised route with low carbon emissions and with consideration to different methods of transport to get to the destination. This concept boasts simplicity to the user, but the application features complex conditions to provide routes that ultimately reduce carbon emissions. There will be visualisation elements included to show the route, identify the different modes of transport and estimated time of arrival (ETA) (Fig. 4).

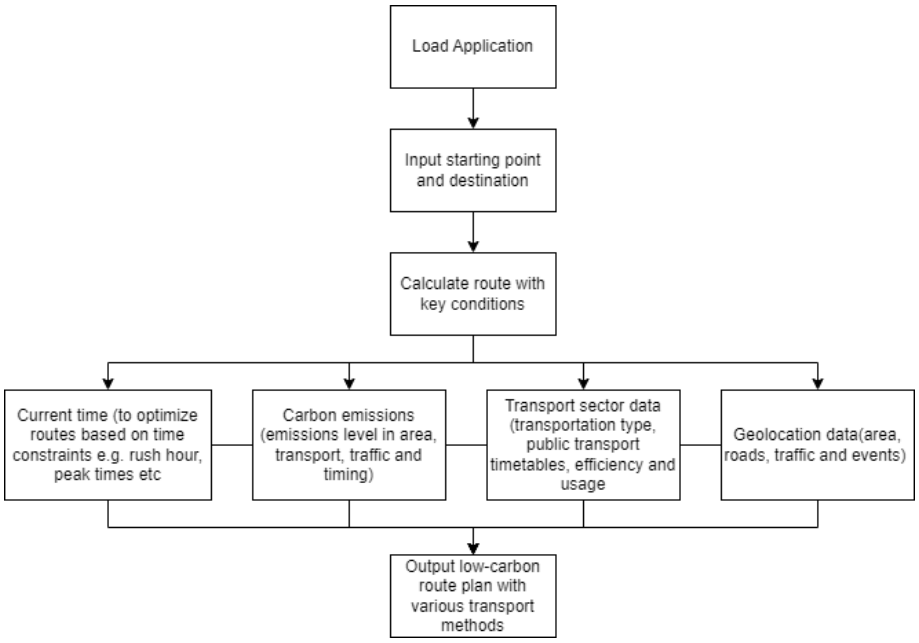


Fig. 4. Conceptual algorithm of processes intended for the solution

Design and Development. Python has been used to develop this application, leveraging existing emissions data and APIs to enhance its functionality. Currently, the primary source of carbon emissions data is the UK Government statistics on emissions from various modes of transport. This data is integrated into the application. The development process involved using Python and its libraries to create components such as the GUI, route planning system, and transport data implementation. Once these components are completed, APIs like Google Maps and GeoPandas are utilised to incorporate geographical data. This focuses on a small area, such as part of a city to generate the most efficient routes in terms of overall efficiency and carbon emissions [36]. Fig. 5 shows the structure of the application.

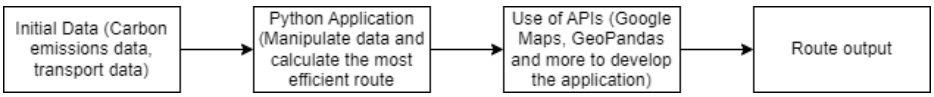


Fig. 5. Conceptual diagram of the development structure for the solution

An interactable GUI is displayed first taking inputs of certain landmarks to the output basic information such as distance and time. Once the journey is confirmed the next part renders a path using various modes of transport to reach the destination. Information such as when to leave, public transport timing, arrival time, distance, carbon emissions produced, and traffic will be shown to the user. Only the most efficient route is outputted. Fig. 6 and 7 show the design concept of the application.

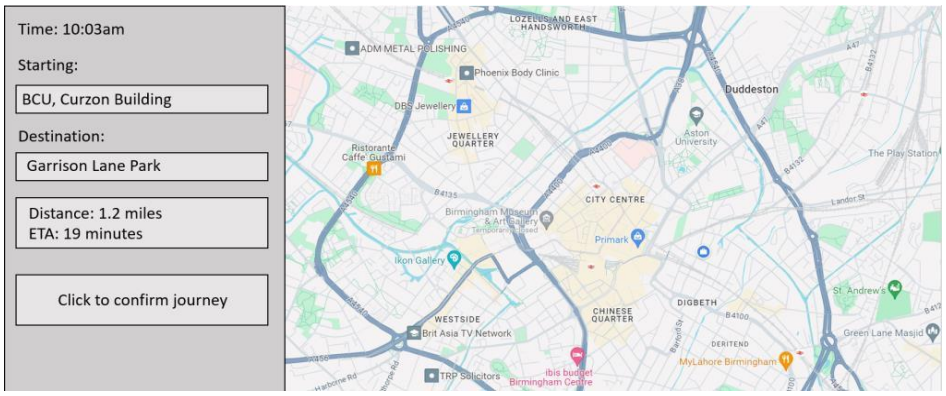


Fig. 6. Conceptual solution GUI to input journey details

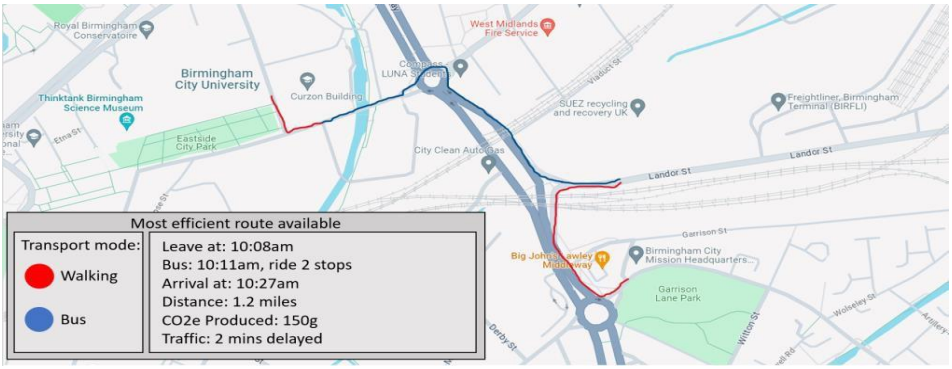


Fig. 7. Conceptual solution route plan outlining the route, transport method and information regarding the route

3.3 Evaluation and testing

An approach of two testing styles has been conducted which are white box and black box testing strategies which tested both the functionality of the solution and logic to ensure the requirements have been achieved.

Black Box Testing. This strategy of testing ensures that the solution is functional to the end user. Black box testing is conducted by a user who has no awareness or knowledge of the source code or internal workings of the solution [37]. In this form of testing the user is purely focused on what the application can do with the primary goal of validating the functionality from the user's perspective which is also displayed as an advantage as the user does not need to know any coding to validate the application [38]. The black box strategy is only used on end-user participants to test that the solution has met all requirements.

White Box Testing. This strategy of testing ensures that the solution functions logically by software testers. This is carried out by software testers as knowledge and experience with code will be required to carry out white box testing. This type of testing involves analysis of the application's code and structure so that the tester has a complete understanding of the application and functions logically as intended [37]. The main advantage of this type of testing is that it can be conducted throughout the entire project at various stages [38]. The white box strategy is only used by the developers of this project as they have all the knowledge of the code and will solely know the intention of the code.

Test Cases. Test cases have also been created for each testing strategy. The black box testing strategy implements usability testing which allows feedback to be collected from participants after using the solution. To collect feedback, both open and close-ended questions with 12 participants were used to gather data on the solution. Table 3 shows 10 questions that were asked by the participants.

These questions have been structured to provide both an indication of how successful the prototype is, the validity of the functionalities implemented and carbon awareness.

Table 3. The questions asked by the participants

Question	Rationale
1. Compared to other navigation apps, how does this app perform in terms of route planning?	This will aid in understanding how the prototype is set apart from other navigation applications. The prototype should raise carbon awareness and for that to happen it needs to be built to the level of leading navigation apps like Google Maps or Waze.
2. What improvements or additional features would you suggest making the app more useful for reducing carbon emissions?	This will aid in gauging user familiarity, prioritising features, and identifying the target audience for the prototype. Understanding different features from user feedback can aid in building a better application.
3. Does the application provide enough information to make you aware that your route is of low carbon emission? If not please explain	This will determine if the route plan data that is displayed on the application provides the user with enough information to make sure that they are aware of their carbon footprint.
4. Which features of navigation apps are most important to you when looking for a route (e.g., real-time traffic updates, ETA accuracy, low carbon route options)?	This will help determine whether to focus more on integrating or improving carbon-efficient routing algorithms or balancing them with other essential navigation features.
5. What aspects do you believe makes this application a smart and sustainable solution for reducing carbon emissions?	This will help to understand user perceptions about what makes the transportation tracker app effective in promoting sustainability. By gauging user opinions on the app's features that contribute to reducing carbon emissions, it can identify which functionalities are most valued and perhaps uncover areas that may need enhancement.

6. Overall, what are your thoughts and opinions on using the app?	This is a simple question to generally understand what the participant's thoughts and opinions are of the prototype where they are free to say anything
7. On a scale from 1 to 10, how satisfied are you with the low carbon emission routes suggested by the app?	This will collect quantitative data on the satisfaction overall with the route planning of the app

White box testing will be carried out by the developer. Table 4 shows the test cases that were carried out by the developer.

Table 4: The test cases

Test ID	Description	Rationale	Test Data	Expected Result	Actual Result	Pass/Fail
A1	Verify whether the application handles origin and destination inputs in the GUI	The user must be able to input their starting point and destination to create a journey	Origin = "Curzon St, Birmingham B4 7XG" Destination = "Wulfruna St, Wolverhampton WV1 1LY"	The application should successfully retrieve route information and display the correct details in the GUI.		
A2	Ensure the application handles invalid inputs correctly	The user must be able to load the application to the main page where they can interact and plan their journey	Origin = "***" Destination = "???"	The application should display an error message indicating that it is unable to retrieve route information.		
A3	Test if the application can correctly read and parse the emission factors from the Excel file.	The emission factors Excel file contains all data relevant to the emissions of each transportation mode to be used for calculating kgCO2	"emission_factors .xlsx" contains transit which produces 0.053kg of CO2. This should be compared to the distance of any journey	The emissions in kgCO2 should be distance * 0.053		

A4	Check the logic for alternative driving routes to ensure they are being captured and displayed correctly.	The application should provide alternative routes of transport with information on each route for driving	A journey that has the best method of transport set to “driving”	The application shows navigation routes with multiple modes of transport
A5	Verify the calculation of traffic delays and it is displayed in the GUI	The application should receive traffic delay data from the Directions API	A journey that has the best method of transport set to “driving” with traffic delays	An estimated traffic delay in minutes should appear on the GUI.
A6	Assess the decision logic for selecting the optimal transportation mode based on emissions and distance.	The application should only return the lowest carbon emission route based on decision logic	A journey that has the least amount of carbon emissions when compared to emissions factor and decision logic	The lowest carbon emission route and transport mode selected
A7	Test the generation of the Google Maps URL and its correctness.	The application should provide alternate routes with different modes of transportation	Any journey should be created first to unlock the “view in google maps” button	The button should lead to a google maps page with the chosen route
A8	Verify walking is a valid transport mode when the condition is met	The application should only select walking if the distance is equal to or less than 2 miles	A journey that is less than 2 miles should be created	The journey should select walking as the transportation mode

A9	Verify bicycling is a valid transport mode when the condition is met	The application should only select walking if the distance is between 3 and 4 miles	A journey that is between 3 and 4 miles should be created	The journey should select bicycling as the transportation mode
A10	Ensure GUI elements appear in their respective positions	The application should have a GUI that is always clear and interactable for the user	Run the application	GUI elements should appear appropriately

4 Implementation Results

Following the agile methodology, the implementation for this project was split into four iterations with each iteration improving from the one before. Only the updates of each iteration will be discussed to provide an understanding of why updates have been made. All testing and evaluation have been conducted on the final iteration of the prototype which is the most functional iteration. Each iteration has a set aim to be achieved before moving on to the next iteration.

4.1 Transportation Tracker for Carbon Emissions V1 – Retrieving data from Google Directions API

The first challenge of the implementation was to use the correct APIs which would retrieve core data relevant to route planning e.g. distance, duration, and transport type. The Google Directions API was implemented as the sole API for this project covering all functionalities. The Google Directions API enhances navigation applications by offering comprehensive routing options for various transport modes, including real-time traffic updates and global coverage making this prototype as dynamic as possible. All transport modes such as walking, bicycling, car and public transport have been retrieved from this API. Leveraging this API could significantly boost user adoption due to Google being both credible and familiar. Additionally, it provided a wide range of developer support and detailed documentation which made the troubleshooting supportive and easy to perform throughout the implementation of this solution.

The first function that was built was “calculate_carbon_emissions” which takes two parameters of both “distance,” from the “get_route_info” function and “mode_of_transport”, which would be taken as user input when the program is executed. The function defines the emission factors for all three modes of transport which are “driving, transit and walking” (transit in the Directions API includes all

public transport e.g. bus, rail, tram etc.). These variable names are specifically chosen as the Directions API uses the same variables for each mode of transport. Once arguments for both parameters in this function are passed the returned value is the result of how much CO2 per kg is produced in the given distance (e.g. if “driving” is the input and the distance is 3km, the function will retrieve the emissions factor for driving which is “0.2” and calculate that by the distance of “3” giving the result of “0.6 kgCO2”). Fig. 8 below shows how the function has been created.

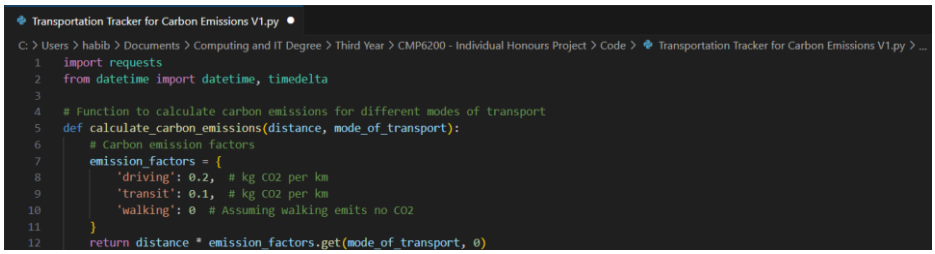


Fig. 8: Function for calculating carbon emissions for the selected mode of transport

The second function built was the “get_route_info” which takes three parameters of “origin”, the starting point “destination”, the end point and “mode_of_transport”. The arguments for these three parameters are retrieved from a URL that uses the Directions API key. The variable “response” uses the requests library to send a request to the constructed URL. The server responds to this request and stores the result in the variable which includes the HTTP status code, headers, and the body containing the data fetched from the API. The “data” variable takes the JSON formatted data from the “response” variable and stores it in a Python dictionary using the “.json()” method provided by the requests library. This allows for easy access and handling of the data, such as extracting routes, distances, durations and more which will be added in later iterations.

The if statement in this function first checks if the status field in the data dictionary equals “OK,” which indicates that the API request was successful, and valid data has been returned. Then the function retrieves the first route from the provided by the Directions API. The API can return multiple routes, but this “route = data['routes'][0]” selects the first one. The variable “distance” stores the journey from the first leg of the route (since a route can have multiple legs, each representing a portion of the journey). The distance is fetched in meters from the API which is converted to kilometres by dividing by “1000”. The “duration” variable stores the duration of the journey as a string format (e.g. “2 hours and 15 minutes”). The “arrival_time” variable calculates the Estimated Time of Arrival (ETA) by converting the current time to seconds and adding it to the duration.

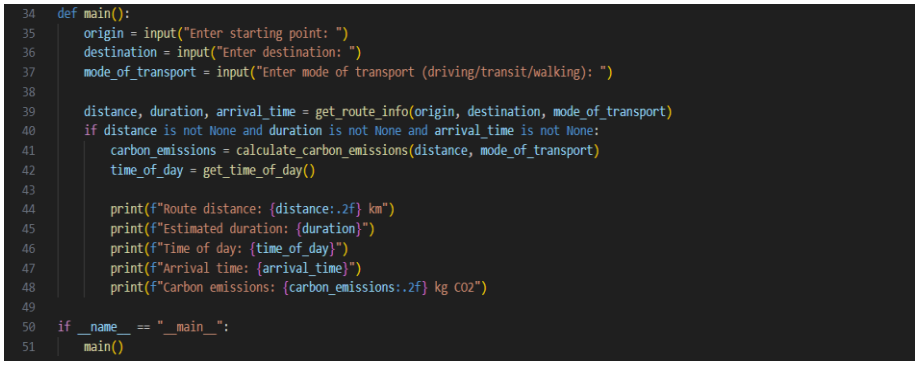
Finally, once the function has all the data in the required variables it returns “distance, duration and arrival_time (in string time format)” ready to be used across the program (e.g. data from distance is passed onto the function in Fig. 9). If the status key in the data dictionary does not equal 'OK', indicating an issue with the API request (like an invalid query or a server error), the code prints an error message displaying the

status, and returns None for all three values, signalling that no valid data was retrieved. Another mini function, “get_time_of_day” is created to simply get the current time and store it in a string format so that it can be used to calculate the ETA. This can all be seen in Fig. 9.

```python
# Function to get route information from Google Maps API
def get_route_info(origin, destination, mode_of_transport):
    api_key = 'AIzaSyAtZeLwt4DtF7I-ZU6V94iAeBrfKGQU5I'
    url = f'https://maps.googleapis.com/maps/api/directions/json?origin={origin}&destination={destination}&mode={mode_of_transport}&key={api_key}'
    response = requests.get(url)
    data = response.json()
    if data['status'] == 'OK':
        route = data['routes'][0]
        distance = route['legs'][0]['distance']['value'] / 1000  # Distance in kilometers
        duration = route['legs'][0]['duration']['text']  # Duration of the journey
        arrival_time = datetime.now() + timedelta(seconds=route['legs'][0]['duration']['value'])
        return distance, duration, arrival_time.strftime("%H:%M:%S")
    else:
        print('Error:', data['status'])
        return None, None, None

# Function to get current time of day
def get_time_of_day():
    now = datetime.now()
    current_time = now.strftime("%H:%M:%S")
    return current_time
```

Fig. 9: Function for retrieving data from Google Directions API and time of day

The final function is “main” which in simple terms, compiles all the results from the functions above to create an output for the user. The start of the output will ask the user to input both the “origin” and “destination” of their journey. It will then ask for the mode of transport that they want to take the journey by which can only be either “walking, driving or transit.” Once these are collected, they are passed as arguments into the “get_route_info” function (Fig. 10) which returns the distance, duration, and arrival time. A conditional check is performed to ensure that each of these variables do not have “None” which is important as it prevents the program from proceeding with null or invalid data, which could lead to errors later in the execution. If the route details are valid, the “calculate_carbon_emissions” function (Fig. 9) is called with the distance and mode_of_transport to calculate the carbon emissions associated with the journey. This can be seen in Fig. 10.



```python
def main():
    origin = input("Enter starting point: ")
    destination = input("Enter destination: ")
    mode_of_transport = input("Enter mode of transport (driving/transit/walking): ")

    distance, duration, arrival_time = get_route_info(origin, destination, mode_of_transport)
    if distance is not None and duration is not None and arrival_time is not None:
        carbon_emissions = calculate_carbon_emissions(distance, mode_of_transport)
        time_of_day = get_time_of_day()

        print(f"Route distance: {distance:.2f} km")
        print(f"Estimated duration: {duration}")
        print(f"Time of day: {time_of_day}")
        print(f"Arrival time: {arrival_time}")
        print(f"Carbon emissions: {carbon_emissions:.2f} kg CO2")

if __name__ == "__main__":
    main()
```

Fig. 10: Function for running the Transportation Tracker for Carbon Emissions V1 program

Finally, the route details, including the distance, estimated travel duration, current time, expected arrival time, and calculated carbon emissions, are printed out. This

provides the user with a complete overview of their journey, using formatted strings to ensure clarity and readability (e.g., distance is formatted to two decimal places). Both inputs for origin and destination can be any location in the UK in any format e.g. street name, town, city, postcode, landmark, etc. This output can also be validated by using the actual Google Maps application which shows the same distance (18.18km = 11.7 miles) and similar duration with a few minutes variance due to API capabilities being different from the actual Google Maps application. This can be seen in Fig. 11 and 12.

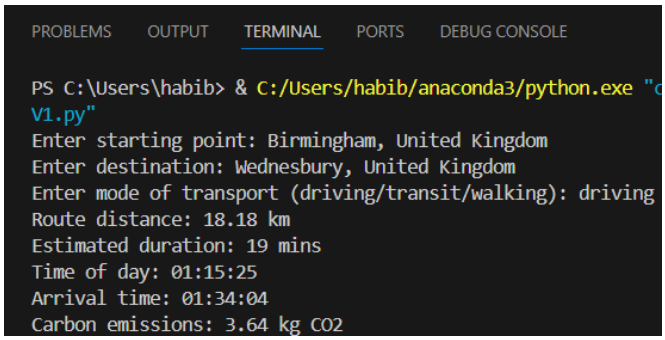


Fig. 11: Output of Transportation Tracker for Carbon Emissions V1

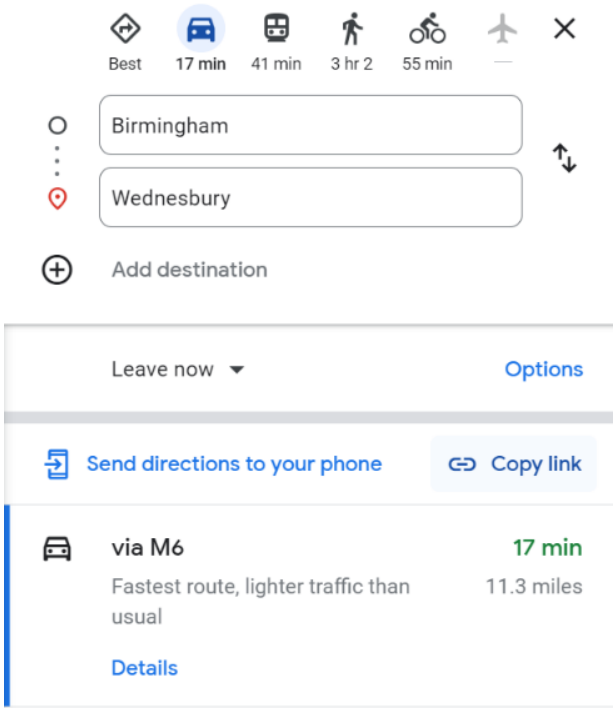


Fig. 12: Google Maps result for validating V1

Overall, the first version of the prototype (Fig. 13) was a huge success in using the Google Directions API but following Table 3 there is much room for improvement which will be carried out over the next iteration. This includes automated transport selection as the aim of the prototype is to reduce the user's carbon emissions and having the program choose the transport will always ensure reduced emissions for route planning.

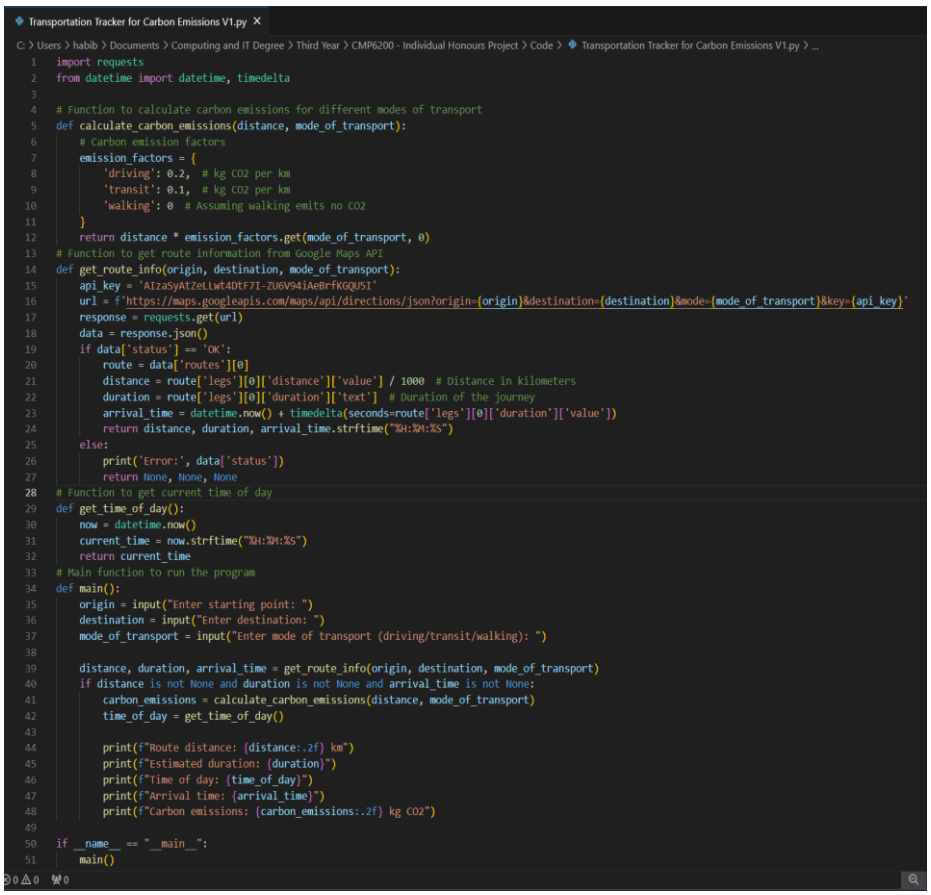


Fig. 13: Transportation Tracker for Carbon Emissions V1 source code

4.2 Transportation Tracker for Carbon Emissions V2 – Automatic transport selection using carbon emissions data

The second iteration uses components from version 1 with the only changes being the addition of automated transportation selection and changing the arrival time logic. The "get_route_info" function has some of its logic changed to implement automated transport selection. The arrival_time has been changed to get the arrival_time directly

from the route selected from the API however there are complications as stated by [39], the documentation indicates a variable known as “arrival_time” but Python cannot find this.

The if statement still works the same by getting the route information from the Directions API however this has been encased into a for loop that loops around for each mode of transport i.e. it will collect information on each route from the origin to the destination for “car, walking and transit.” Once the distance of each transport mode is collected it passes the data as an argument in the “calculate_carbon_emissions” function, which then returns the emissions in kgCO2 and stores it in the “emissions” variable. This is then compared to “min_emissions” which is initially set to infinity meaning the first route will always replace “min_emissions” with the “emissions” value. Once the for loop has been completed the returned mode of transport will always be the transport type with the lowest emission. This can be seen in Fig. 14.

```python
# Function to get route information from Google Maps API
def get_route_info(origin, destination):
    api_key = 'AIzaSyAtZeLLwt4DtF7I-ZU6V94iAeBrfKGQU5I'
    modes_of_transport = ['driving', 'transit', 'walking']
    best_mode = None
    min_emissions = float('inf')
    arrival_time = None
    duration = None

    for mode in modes_of_transport:
        url = f'https://maps.googleapis.com/maps/api/directions/json?origin={origin}&destination={destination}&mode={mode}&key={api_key}'
        response = requests.get(url)
        data = response.json()
        if data['status'] == 'OK':
            route = data['routes'][0]
            distance = route['legs'][0]['distance']['value'] / 1000  # Distance in kilometers
            emissions = calculate_carbon_emissions(distance, mode)
            if emissions < min_emissions:
                min_emissions = emissions
                best_mode = mode
                arrival_time = route['legs'][0]['arrival_time']['text'] if 'arrival_time' in route['legs'][0] else None
                duration = route['legs'][0]['duration']['text']

    return best_mode, min_emissions, arrival_time, duration, distance
```

Fig. 14: Updated function for retrieving data from Google Directions API and automated transport selection based on carbon emissions data

The “main” function that compiles all functions has been updated with the only inputs being the origin and destination. The mode of transport that is selected in the “get_route_info” function will also carry across its relevant route planning data i.e. distance, duration, emissions, arrival_time etc. In the first iteration, there was a form of validation to check if data has been collected from the Directions API however, this has been updated and replaced with an else statement that can also catch other errors if the “get_route_info” function does not work as intended. These changes can be seen in Fig. 15.

```python
# Main function to run the program
def main():
    origin = input("Enter starting point: ")
    destination = input("Enter destination: ")

    best_mode, min_emissions, arrival_time, duration = get_route_info(origin, destination)
    if best_mode:
        time_of_day = get_time_of_day()
        print(f"The most carbon-efficient mode of transport is: {best_mode}")
        print(f"Estimated carbon emissions: {min_emissions:.2f} kg CO2")
        print(f"Estimated duration: {duration}")
        print(f"Arrival time: {arrival_time}")
        print(f"Time of day: {time_of_day}")
    else:
        print("Unable to retrieve route information.")

if __name__ == "__main__":
    main()
```

Fig. 15: Function for running the Transportation Tracker for Carbon Emissions V2 program

Overall, the output is like the first iteration, however, the program in the second iteration gives the most carbon-efficient route to ensure users have the best available carbon-efficient route automatically. However, there are two main concerns with the first being the automated transport selector which only returns walking as the transport mode regardless of distance and the arrival time returning “None” as it does not exist in the Directions API. This will be updated in the third iteration along with the implementation of further functionalities such as implementing a GUI. Both outputs in the console and Google Maps for validation can be seen showing the routing information is still successful in Fig.16 and 17.

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  DEBUG CONSOLE

PS C:\Users\habib> & C:/Users/habib/anaconda3/python.exe
Enter starting point: Birmingham, United Kingdom
Enter destination: Wednesbury, United Kingdom
The most carbon-efficient mode of transport is: walking
Estimated carbon emissions: 0.00 kg CO2
Route distance: 13.03 km
Estimated duration: 3 hours 2 mins
Arrival time: None
Time of day: 22:38:35
```

Fig. 16: Output of Transportation Tracker for Carbon Emissions V2

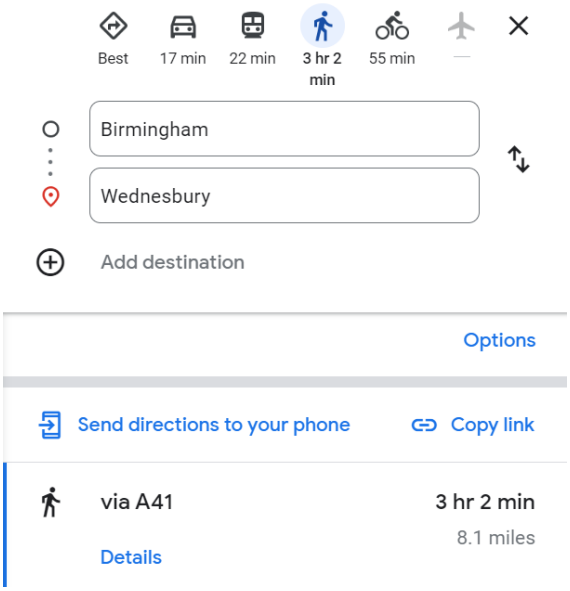
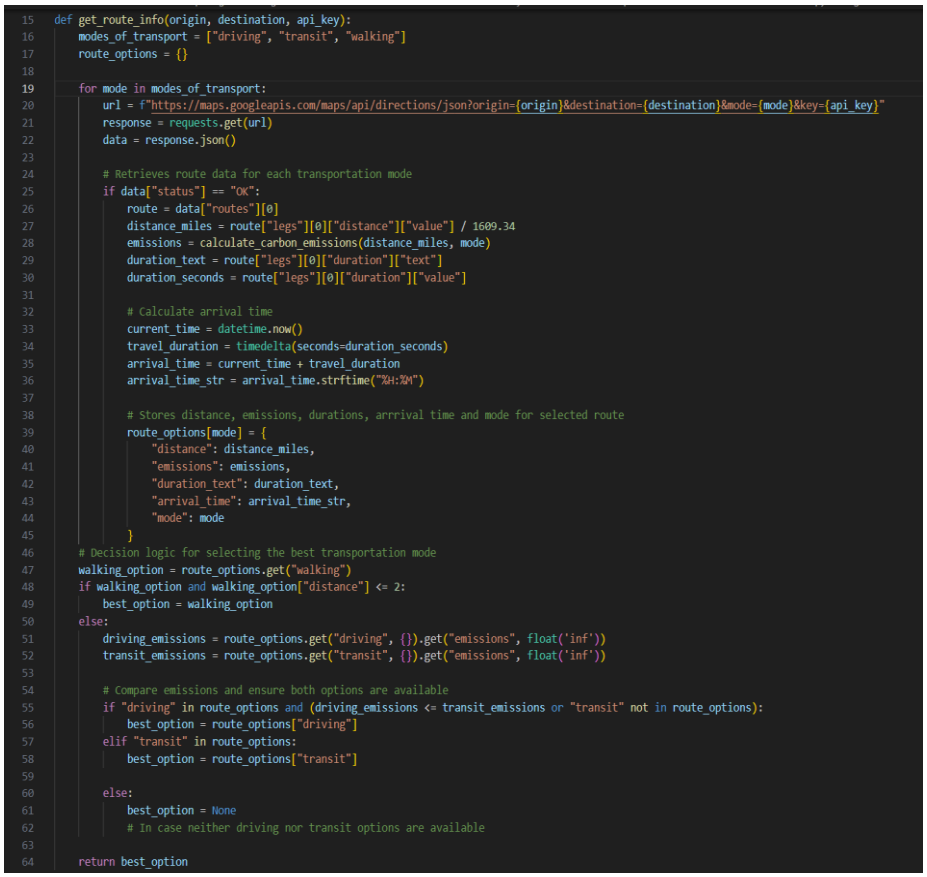


Fig. 17: Google Maps result for validating V2

4.3 Transportation Tracker for Carbon Emissions V3 – Updated emissions calculation logic and GUI

The third iteration uses components from the second iteration to both update and implement new features and improve the prototype to become one step closer to meeting the aim and all requirements of this project. The "get_route_info" function has been updated again featuring a new decision logic for selecting the best transportation mode. The function keeps the same data retrieval element from the Directions API however this time it has been further optimised by storing all the information relating to each route in the "route_options" variable which is in the form of a dictionary.
The new decision logic compares each transport option in an if statement which ensures that walking is only chosen if the distance is less than or equal to 2 miles, otherwise, both driving and transit emissions are compared. Statistically, driving emissions are higher so if driving emissions are less than or equal to transit emissions then driving is the best mode of transport. If transit emissions are lower than driving, then transit is the best mode of transport. If the journey has invalid inputs or no mode of transport, then "None" will be set to indicate an error with the journey. The arrival time error encountered in the second iteration has also been fixed by reverting it to the same logic used in the first iteration and adding the duration to the current time. The units have also been changed to imperial units of miles by dividing the returned distance by 1609.34. This update will give clarity to participants who will test the prototype and will likely be in the UK as Google Maps in the UK defaults to miles instead of kilometres. These updates can be seen in Fig. 18.



```python
def get_route_info(origin, destination, api_key):
    modes_of_transport = ["driving", "transit", "walking"]
    route_options = {}

    for mode in modes_of_transport:
        url = f"https://maps.googleapis.com/maps/api/directions/json?origin={origin}&destination={destination}&mode={mode}&key={api_key}"
        response = requests.get(url)
        data = response.json()

        # Retrieves route data for each transportation mode
        if data["status"] == "OK":
            route = data["routes"][0]
            distance_miles = route["legs"][0]["distance"]["value"] / 1609.34
            emissions = calculate_carbon_emissions(distance_miles, mode)
            duration_text = route["legs"][0]["duration"]["text"]
            duration_seconds = route["legs"][0]["duration"]["value"]

            # Calculate arrival time
            current_time = datetime.now()
            travel_duration = timedelta(seconds=duration_seconds)
            arrival_time = current_time + travel_duration
            arrival_time_str = arrival_time.strftime("%H:%M")

            # Stores distance, emissions, durations, arrrival time and mode for selected route
            route_options[mode] = {
                "distance": distance_miles,
                "emissions": emissions,
                "duration_text": duration_text,
                "arrival_time": arrival_time_str,
                "mode": mode
            }
    # Decision logic for selecting the best transportation mode
    walking_option = route_options.get("walking")
    if walking_option and walking_option["distance"] <= 2:
        best_option = walking_option
    else:
        driving_emissions = route_options.get("driving", {}).get("emissions", float('inf'))
        transit_emissions = route_options.get("transit", {}).get("emissions", float('inf'))

        # Compare emissions and ensure both options are available
        if "driving" in route_options and (driving_emissions <= transit_emissions or "transit" not in route_options):
            best_option = route_options["driving"]
        elif "transit" in route_options:
            best_option = route_options["transit"]

        else:
            best_option = None
            # In case neither driving nor transit options are available

    return best_option
```

Fig. 18: “get_route_info” function updated from Transportation Tracker for Carbon Emissions V2 with new decision logic and arrival time fix

The “main” function has been renamed to “calculate_route” which has been updated to support a Tkinter GUI. Tkinter is a simple library that can be imported to create a functional GUI application with buttons, inputs and more. The values from the origin and destination are not set to be retrieved from a tkinter entry box which will still return a string and function as normal. The message for route planning has been stored in the variable “result” which can change if a route plan does not exist. Extra validation has been added to ensure only valid routes are outputted on the GUI. This can be seen in Fig. 19.

```python
def get_time_of_day():
    now = datetime.now()
    return now.strftime("%H:%M")

def calculate_route():
    origin = entry_origin.get()
    destination = entry_destination.get()
    api_key = "AIzaSyAtZeLLwt4DtF7I-ZU6V94iAeBrfKGQU5I"

    best_option = get_route_info(origin, destination, api_key)
    if best_option is not None and "mode" in best_option:
        time_of_day = get_time_of_day()
        result = (f"Most carbon effcient transport method: {best_option['mode']}\n"
                  f"Total Journey Distance: {best_option['distance']:.2f} Miles\n"
                  f"Estimated carbon emissions: {best_option['emissions']:.2f} kg CO2\n"
                  f"Estimated duration: {best_option['duration_text']}\n"
                  f"Departure time: {time_of_day}\n"
                  f"Arrival time: {best_option['arrival_time']}\n")

    else:
        result = ("Unable to retrieve route information.")
    text_result.config(state=tk.NORMAL)
    text_result.delete(1.0, tk.END)
    text_result.insert(tk.END, result)
    text_result.config(state=tk.DISABLED)
```

Fig. 19: "calculate_route" function with Tkinter support

At the bottom of the Python script, all GUI coding was implemented using Python's tkinter library. Fig. 20 shows the code involved in creating the GUI which mostly consists of creating/setting GUI window properties, creating input fields and confirmation buttons to calculate routes based on user input for origin and destination in the GUI.

```python
# Tkinter GUI Code below

root = tk.Tk()
root.title("Route planner V3")

# Create a style
style = ttk.Style()
style.configure("TFrame", background="darkgray")
style.configure("TButton", background="darkgray", foreground="black")
style.map("TButton", background=[("active", "darkgray")])

# Set the background color
root.configure(bg="darkgray")

# Modify the padding and layout according to the provided image
frame = ttk.Frame(root, padding="30", relief=tk.FLAT)
frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S), padx=20, pady=20)

# Title
ttk.Label(frame, text="Transportation Tracker for Carbon Emissions", font="18", background="darkgray", foreground="black").grid(column=0, row=0, sticky=tk.W)

# Starting point entry
ttk.Label(frame, text="Starting point", background="darkgray", foreground="black").grid(column=0, row=1, sticky=tk.W)
entry_origin = ttk.Entry(frame, width=60)
entry_origin.grid(column=0, row=2, pady=5)

# Destination entry
ttk.Label(frame, text="Destination", background="darkgray", foreground="black").grid(column=0, row=3, sticky=tk.W)
entry_destination = ttk.Entry(frame, width=60)
entry_destination.grid(column=0, row=4, pady=5)

# Calculate button
calculate_button = ttk.Button(frame, text="Confirm Journey", command=calculate_route)
calculate_button.grid(column=0, row=5, pady=10)

# Result display area
text_result = tk.Text(frame, width=50, height=10, wrap="word")
text_result.grid(column=0, row=6, pady=10)
text_result.config(state=tk.DISABLED, bg="white")

# Start the GUI loop
root.mainloop()
```

Fig. 20: Tkinter coding section

The improvements were successful in this iteration and a functional GUI has been made allowing better visualisation than compared to console output. Fig. 21 and 22 show that once a journey is confirmed the details appear below in the white box to display the results. The GUI is also in a constant loop until the user closes the window down so they can query different starting points and destinations for as long as they require. The decision logic can be seen side by side with Fig. 21 showing a distance greater than 2 miles returning transit as the transport mode which is the most reduced in carbon emissions and Fig. 22 shows a journey that is less than 2 miles which encourages users to walk producing no direct carbon emissions. Fig. 23 and 24 show the validation of each route matching created by the Transportation Tracker for Carbon Emissions V3.

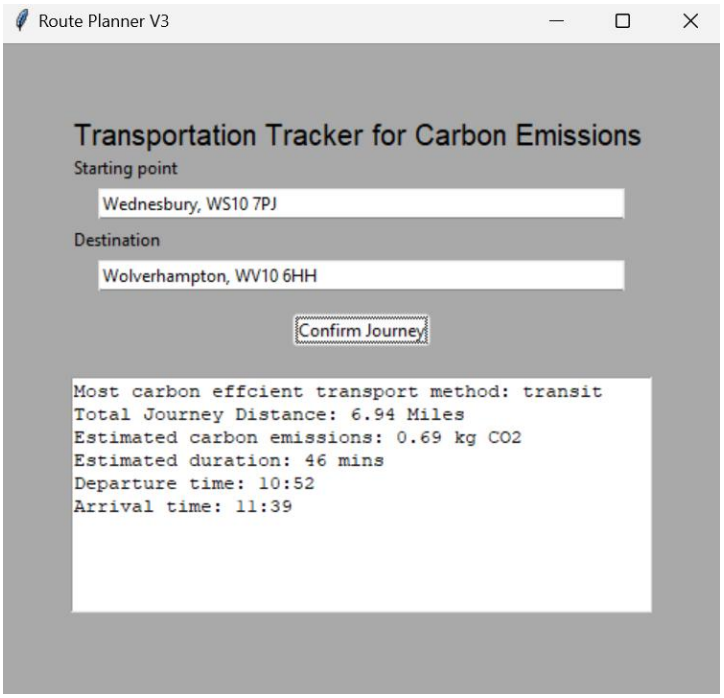


Fig. 21: Route Planner

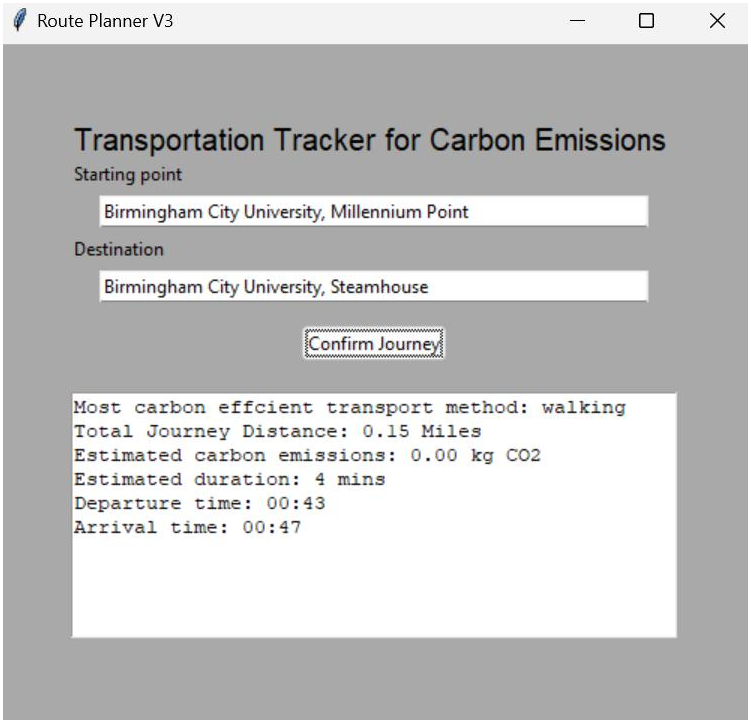


Fig. 22: Output of Transportation Tracker for Carbon Emissions V3 - Walking

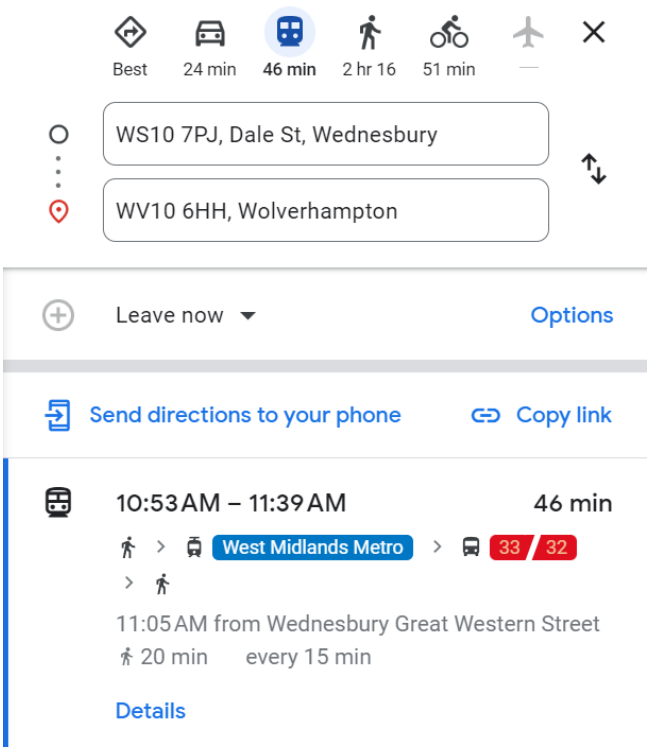


Fig. 23: Google Maps result for validating V3 Transit transportation mode

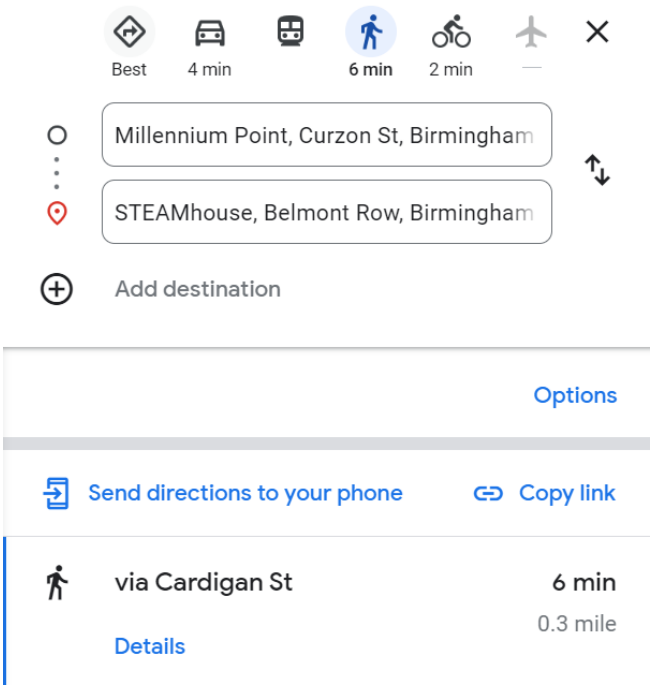


Fig. 24: Google Maps result for validating V3 Walking transportation mode

The next iteration will be the last update which will feature using real-time traffic data from the API to show traffic delays, improved emission factors which will feature file handling allowing future datasets to be implemented for emissions accuracy and more. These features and updates should enhance the capabilities of the prototype making it more dynamic.

4.4 Transportation Tracker for Carbon Emissions V4 – Real-time traffic data, emission factor file handling, alternative transport/routes, and maps visualisation

In this final iteration, many new changes and updates were made to ensure the prototype met the project's aims and objectives, development designs, and user requirements as closely as possible. The first update was to make the emissions factor use file handling instead of static data in the Python script. In Fig. 25 an Excel file is shown where each mode of transport has an emissions factor that has been calculated from data sources. An article by [40] shows the CO2 grams per mile that a car emits on average for each year from 2007 to 2022. This evidence is supported by the Department of Transport and was used in the prototype to find the average emissions from 2007 to 2022 [5]. [41] states that the average carbon emissions per kilometre was 33 grams. Both values for driving and transit are converted into kgCO2 per miles which resulted in 0.053 as the emissions factor for transit and 0.245 for driving. The ideal data would have been to

incorporate real-time carbon emissions data or a dataset directly from a council or public transportation organisation but unfortunately attempts to contact these entities failed with no response from anyone.

B2 =ROUND(AVERAGE(B9:B24)/1000,3)

	A	B	C	D
1	ModeOfTransport	EmissionFactor		
2	driving	0.245		
3	transit	0.053		
4	walking	0		
5	bicycling	0		
6		Average CO2 Emissions in Kilograms per mile		
7				
8	Years	CO2 grams per mile		
9	2007	275.4		
10	2008	272.9		
11	2009	269.7		
12	2010	265.9		
13	2011	262		
14	2012	257.6		
15	2013	252.7		
16	2014	247.5		
17	2015	242.1		
18	2016	236.8		
19	2017	232.1		
20	2018	228.2		
21	2019	223.7		
22	2020	221.4		
23	2021	218.2		
24	2022	215		
25	Driving average CO2 emissions	245.075		

Fig. 25: Emissions factor Excel file

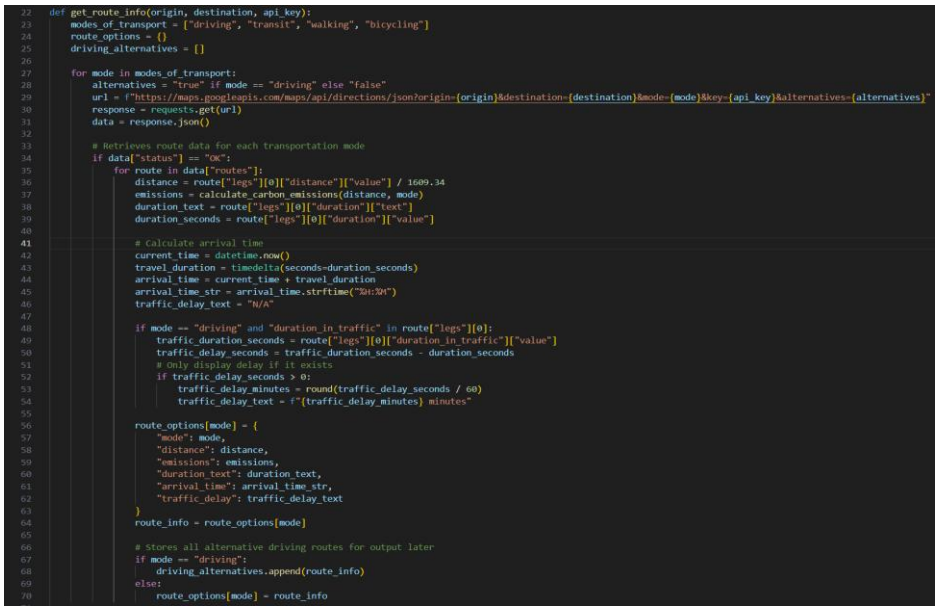
The “calculate_carbon_emissions” function still works the same as in the previous iteration however, the emissions factors are instead retrieved from the Excel file in Fig. 25. “load_emissions_factors” is a new function that takes an Excel file as a parameter. The variable “df” uses the panda’s library to store Excel file data in a Panda DataFrame.

To store the data in the Excel file it needs to be read which is where “openpyxl” is used to read the Excel file. The variable “emissions_factors_dict” retrieves the values for each emission factor and the corresponding index for the mode of transport i.e. 0.245 belongs to driving which both values are retrieved and stored in a dictionary for use in the program later. This can be seen in Fig. 26.

```
def load_emission_factors(excel_file):
    # Reads the Excel file
    df = pd.read_excel(excel_file, engine='openpyxl')
    # Converts the DataFrame to a dictionary where the mode of transport is the key and the emission factor is the value
    emission_factors_dict = pd.Series(df.EmissionFactor.values, index=df.ModeOfTransport).to_dict()

    return emission_factors_dict

def calculate_carbon_emissions(distance, mode_of_transport):
    emission_factors = load_emission_factors('emission_factors.xlsx')

    return distance * emission_factors.get(mode_of_transport, 0)
```

Fig. 26: “load_emissions_factor” and updated “calculate_carbon_emissions” function

The “get_route_info” has 2 new updates, the alternative routes and traffic delays for the transport mode of driving only. The for loop still works the same however the “alternatives” data has been collected from the Directions API which only holds “driving” routes. All “driving” routes are stored in a new list called “driving_alternatives” which will be used later in the program. According to [39], the Directions API holds a variable that stores traffic delay for driving in the “duration_in_traffic” variable. Once this data is obtained it is converted into seconds and calculated against the duration of the journey. If the delay is a negative number or 0 then that means there are no delays in the journey, however, if it exceeds 0 then the delay is converted into minutes and outputted onto the GUI. Fig. 27 shows the code behind these updates.



```python
def get_route_info(origin, destination, api_key):
    modes_of_transport = ["driving", "transit", "walking", "bicycling"]
    route_options = {}
    driving_alternatives = []

    for mode in modes_of_transport:
        alternatives = "true" if mode == "driving" else "false"
        url = f"https://maps.googleapis.com/maps/api/directions/json?origin={origin}&destination={destination}&mode={mode}&key={api_key}&alternatives={alternatives}"
        response = requests.get(url)
        data = response.json()

        # Retrieves route data for each transportation mode
        if data["status"] == "OK":
            for route in data["routes"]:
                distance = route["legs"][0]["distance"]["value"] / 1609.34
                emissions = calculate_carbon_emissions(distance, mode)
                duration_text = route["legs"][0]["duration"]["text"]
                duration_seconds = route["legs"][0]["duration"]["value"]

                # Calculate arrival time
                current_time = datetime.now()
                travel_duration = timedelta(seconds=duration_seconds)
                arrival_time = current_time + travel_duration
                arrival_time_str = arrival_time.strftime("%H:%M")
                traffic_delay_text = "N/A"

                if mode == "driving" and "duration_in_traffic" in route["legs"][0]:
                    traffic_duration_seconds = route["legs"][0]["duration_in_traffic"]["value"]
                    traffic_delay_seconds = traffic_duration_seconds - duration_seconds
                    # Only display delay if it exists
                    if traffic_delay_seconds > 0:
                        traffic_delay_minutes = round(traffic_delay_seconds / 60)
                        traffic_delay_text = f"{traffic_delay_minutes} minutes"

                route_options[mode] = {
                    "mode": mode,
                    "distance": distance,
                    "emissions": emissions,
                    "duration_text": duration_text,
                    "arrival_time": arrival_time_str,
                    "traffic_delay": traffic_delay_text
                }
                route_info = route_options[mode]

                # Stores all alternative driving routes for output later
                if mode == "driving":
                    driving_alternatives.append(route_info)
                else:
                    route_options[mode] = route_info
```

Fig. 27: Updated “get_route_info” function from Transportation Tracker for Carbon Emissions V3 with route alternatives for driving and real-time traffic delays

The decision logic created in the third iteration is still the same with the added if statement for driving alternatives which calculates the carbon emissions for each alternative route and ensures that there are no duplicate driving routes i.e. the best route will not appear in the driving alternatives section of the GUI as its already displayed. As shown in Fig. 25, bicycling was added as a mode of transport and has zero emissions. Bicycling is only selected if the distance is between 3 and 4 miles. Fig. 28 shows these updates.

```
23  def get_route_info(origin, destination, api_key):
74      # This ensures that each alternatives carbon emissions are calculated
75      if driving_alternatives:
76          best_driving_option = min(driving_alternatives, key=lambda x: x["emissions"])
77          route_options["driving"] = best_driving_option
78          # Ensures best_option does not appear in alternative avoiding duplication
79          route_options["driving"]["alternatives"] = [alt for alt in driving_alternatives if alt != best_driving_option]
80
81      # Decision logic for selecting the best transportation mode
82      walking_option = route_options.get("walking")
83      bicycling_option = route_options.get("bicycling")
84
85      if walking_option and walking_option["distance"] <= 2:
86          best_option = walking_option
87
88      elif bicycling_option and 3 <= bicycling_option["distance"] <= 4:
89          best_option = bicycling_option
90
91      else:
92          driving_emissions = route_options.get("driving", {}).get("emissions", float('inf'))
93          transit_emissions = route_options.get("transit", {}).get("emissions", float('inf'))
94
95          # Compare emissions and ensure both options are available
96          if "driving" in route_options and (driving_emissions <= transit_emissions or "transit" not in route_options):
97              best_option = route_options["driving"]
98          elif "transit" in route_options:
99              best_option = route_options["transit"]
100         else:
101             best_option = None
102             # In case neither driving nor transit options are available
103
104     return best_option
```

Fig. 28: Updated “get_route_info” function from Transportation Tracker for Carbon Emissions V3 with new bicycling option and driving alternative logic

A new function has been added which allows the user to open a URL with the click of a button in the GUI. “view_in_google_maps” is a function which takes the origin and destination inputted by the user and adds it to a Google Maps URL which is opened in a web browser. This has been controlled with the use of a button meaning the function will only be executed if the user creates a journey which unlocks the button for the user to click to run the function. “calculate_route” has the added output for traffic delay which will show delays if driving is the selected transport mode. A for loop is created which loops around for each driving alternative there is and outputs this information to the GUI. These updates can be seen in Fig. 29.

```python
def view_in_google_maps():
    origin = entry_origin.get()
    destination = entry_destination.get()
    origin_encoded = urllib.parse.quote(origin)
    destination_encoded = urllib.parse.quote(destination)

    # Open the URL in the web browser
    maps_url = f"https://www.google.com/maps/dir/{origin_encoded}/{destination_encoded}"
    webbrowser.open(maps_url)

def calculate_route():
    origin = entry_origin.get()
    destination = entry_destination.get()
    api_key = "AIzaSyAtZeLLwt4DtF7I-ZU6V94iAeBrfKGQU5I"

    best_option = get_route_info(origin, destination, api_key)
    if best_option is not None and "mode" in best_option:
        time_of_day = get_time_of_day()
        result = (f"Lowest Carbon Footprint Route\n\n"
                  f"Transport method: {best_option['mode'].capitalize()}\n"
                  f"Total Journey Distance: {best_option['distance']:.2f} Miles\n"
                  f"Estimated carbon emissions: {best_option['emissions']:.2f} kg CO2\n"
                  f"Estimated duration: {best_option['duration_text']}\n"
                  f"Departure time: {time_of_day}\n"
                  f"Arrival time: {best_option['arrival_time']}\n"
                  f"Traffic delay: {best_option['traffic_delay']}")
        view_maps_button['state'] = 'normal'
        # If driving is chosen then all other alternatives that the directions API has will be shown
        if best_option["mode"] == "driving" and "alternatives" in best_option:
            counter = 1
            result += "\n\nDriving Alternatives:\n"
            for alt in best_option["alternatives"]:
                counter = counter + 1
                result += (f"\nRoute {counter}\n"
                           f"Total Journey Distance: {alt['distance']:.2f} Miles\n"
                           f"Estimated carbon emissions: {alt['emissions']:.2f} kg CO2\n"
                           f"Estimated duration: {alt['duration_text']}\n"
                           f"Departure time: {time_of_day}\n"
                           f"Arrival time: {alt['arrival_time']}\n"
                           f"Traffic delay: {alt['traffic_delay']}\n")

        elif "alternatives" in best_option and not best_option["alternatives"]:
            result += ("\n No routes available")

    else:
        result = ("Unable to retrieve route information.")
        view_maps_button['state'] = 'disabled'
    text_result.config(state=tk.NORMAL)
    text_result.delete(1.0, tk.END)
    text_result.insert(tk.END, result)
    text_result.config(state=tk.DISABLED)
```

Fig. 29: "view_in_google_maps" and updated "calculate_route" function

There is no notable change for the GUI except that the "view_maps_button" has been added which executes the "view_in_google_maps" function (Fig. 29). Fig. 30 shows this change.

```python
view_maps_button = ttk.Button(frame, text="View Google Maps Route", command=view_in_google_maps)
view_maps_button.grid(column=0, row=9, pady=10)
view_maps_button['state'] = 'disabled'

# Result display area
text_result = tk.Text(frame, width=70, height=20, wrap="word")
text_result.grid(column=0, row=7, pady=10)
text_result.config(state=tk.DISABLED, bg="white")

# Start the GUI loop
root.mainloop()
```

Fig. 30: Updated Tkinter GUI with the added button to view Google Maps

5 Evaluation

As the prototype has been improved on throughout each iteration, the evaluation of the project will involve only the final prototype (Transportation Tracker for Carbon Emissions V4)

Following on from what has been discussed in 3.3, the testing strategies and their results are explained below. The logical testing was conducted by the developer of the prototype following through an updated test case and recording the result. The functional testing was conducted by 12 participants who only knew the prototype's purpose and what input/outputs to focus on. After the test, the participants were given a questionnaire to complete which anonymously collected response data. Three of the participants were selected due to the nature of their background showing some awareness of carbon emissions while the rest were standard selected participants that use navigational applications generally. This was done purposely to see the variance in results between those who are carbon-aware and those who are not.

5.1 Logical Testing Results

This follows the same test case as shown in Table 5 but has been completed below with results. As shown below all 10 tests have been successful. This test case is used to ensure all functions in the prototype work logically as required for users to successfully use the prototype without issues. Refer to the Appendix for corresponding test result evidence.

Table 5. Table captions should be placed above the tables.

Test ID	Description	Rationale	Test Data	Expected Result	Actual Result	Pass /Fail
A1	Verify whether the application handles origin and destination inputs in the GUI	The user must be able to input their starting point and destination to create a journey	Origin = "Curzon St, Birmingham B4 7XG" Destination = "Wulfruna St, Wolverhampton WV1 1LY"	The application should successfully retrieve route information and display correct details in the GUI.	(A.1: Result) Route information is retrieved and correct details are displayed regarding the journey	PASS

A2	Ensure the application handles invalid inputs correctly	The user must be able to load the application to the main page where they can interact and plan their journey	Origin = “***” Destination = “???”	The application should display an error message indicating that it is unable to retrieve route information.	(A.2: Result) The application displays an error message “Unable to retrieve route information”	PASS
A3	Test if the application can correctly read and parse the emission factors from the Excel file.	The emission factors Excel file contains all data relevant to the emissions of each transportation mode to be used for calculating kgCO2	“emission_factors.xlsx” contains transit which produces 0.053kg of CO2. This should be compared to the distance of any journey	The emissions in kgCO2 should be distance * 0.053	(A.3: Result) 89.71 miles * 0.053 = 4.75kgCO2 which correctly uses emissions factors from the Excel file	PASS
A4	Check the logic for alternative driving routes to ensure they are being captured and displayed correctly.	The application should provide alternative routes of transport with information on each route for driving	A journey that has the best method of transport set to “driving”	The application shows the best route and an alternative route by driving	(A.4: Result) The application displays the best route and an additional route with all respective information	PASS
A5	Verify the calculation of traffic delays and it is displayed in the GUI	The application should receive traffic delay data from the Directions API	A journey that has the best method of transport set to “driving” with traffic delays	An estimated traffic delay in minutes should appear on the GUI.	(A.5: Result) The application displays existing delays in the route in minutes	PASS

A6	Assess the decision logic for selecting the optimal transportation mode based on emissions and distance.	The application should only return the lowest carbon emission route based on decision logic	A journey that has the least amount of carbon emissions when compared to emissions factor and decision logic	The lowest carbon emission route and transport mode selected	(A.6: Result) The application always returns the lowest carbon emissions transportation mode using decision logic (transit is selected mostly due to emissions factor being lower)	PASS
A7	Test the generation of the Google Maps URL and its correctness.	The application should provide alternate routes with different modes of transportation	Any journey should be created first to unlock the "view in Google Maps" button	The button should lead to a Google Maps page with the chosen route	(A.7: Result) The application takes origin and destination input and opens a Google Maps URL	PASS
A8	Verify walking is a valid transport mode when the condition is met	The application should only select walking if the distance is equal to or less than 2 miles	A journey that is less than 2 miles should be created	The journey should select walking as the transportation mode	(A.8: Result) A journey under 2 miles uses walking	PASS
A9	Verify bicycling is a valid transport mode when the condition is met	The application should only select walking if the distance is between 3 and 4 miles	A journey that is between 3 and 4 miles should be created	The journey should select bicycling as the transportation mode	(A.9: Result) A journey between 3 and 4 miles uses bicycling	PASS
A10	Ensure GUI elements appear in their respective positions	The application should have a GUI that is always clear and interactable for the user	Run the application	GUI elements should appear appropriately	(A.10: Result) Placement for all elements is correctly displayed	PASS

5.2 Functionality Testing Results

This section presents the results obtained after participants tested the prototype. Each participant completed a questionnaire with questions similar to those shown in Table 3. The overall results indicate that the participants were satisfied and understood the concept of the prototype, as well as its potential to make navigation applications more carbon-aware. Additionally, there were varying responses, which will be discussed in the following section.

Question 1: Compared to other navigation apps, how does this app perform in terms of route planning?

The purpose of this question is to understand how the prototype is compared to other market-leading apps like “Google Maps” or “Waze” from the user’s perspective. Many respondents appreciate the app's “focus on carbon emissions” as a unique approach to highlighting carbon emissions associated with different routes on a navigation app. This feature “sets the app apart” from conventional navigation tools and aligns well with environmentally conscious users. Some users compare the app favourably against mainstream options like Google Maps, noting that while other apps prioritise speed and efficiency, this app provides a valuable alternative by focusing on environmental impact. However, a few users point out that the app sometimes “lags in speed” and “user interface smoothness,” suggesting that while the environmental focus is beneficial, it should not compromise basic functionalities like route efficiency and user convenience.

A few responses highlight the app's effectiveness in presenting information in a “simple” and understandable manner. This suggests that the app is user-friendly, providing information efficiently and concisely. While the focus on low-carbon routes is praised, there is a concern that users might ultimately choose convenience over environmental benefits. As discussed in 4.3, this has indicated a potential gap between the app's objectives and user behaviour in real-world scenarios, where ease and convenience often prevail.

Question 2: What improvements or additional features would you suggest, to make the app more useful for reducing carbon emissions?

The purpose of this question is to understand the weakness of the prototype from the user’s perspective. This information will provide future ideas for features to implement should the prototype have another iteration.

The respondents expressed a desire for more comprehensive information about routes, including “all potential routes” with associated travel times and carbon emissions. This would enable users to make more informed decisions that align with their values on environmental impact. Multiple respondents also suggest improvements to the GUI or even the development of “a full website or standalone app,” indicating that current interface limitations may be impacting user experience and engagement.

There is a strong response to adding further real-time data that affects travel and emissions. This includes “traffic conditions, public transportation schedules, and EV charging station availability.” Such integration would help in planning the most efficient and least polluting routes, potentially encouraging more sustainable travel habits. Suggestions also include the use of “predictive analytics” and “real-time carbon footprint calculations” to enhance route planning and to provide dynamic suggestions that adapt to changing conditions and user behaviours.

The trends and patterns from these responses indicate a need for a navigation tool that not only prioritises environmental impact but also meets high standards of user convenience, flexibility, and real-time responsiveness. This feedback showcases the importance of balancing environmental goals with practical usability and advanced technological integration to widen user adoption and satisfaction.

Question 3: Does the application provide enough information to make you aware that your route is of low carbon emission? If not, please explain

The purpose of the entire project is to develop a prototype that reduces carbon emissions and this question analyses if the users are aware of low carbon footprint routes. Responses to this question indicated that users appreciate that the app offers “some level of carbon emission data,” affirming its utility in providing environmentally conscious routing options. However, there is a recurring request for “more detailed comparisons of carbon emissions” across different route options. Users express a desire for the app to not only display emissions for the top recommended route but also for alternative routes that are not driving (driving is the only transport method that shows alternative routes), enhancing their ability to make informed decisions based on environmental impact.

Several responses also suggest that the app could improve by including features that allow users to input specific details (such as vehicle make and model) to get more personalised emission data. This indicates a need for customisation options that could make the app more relevant to individual circumstances. Users are looking for more than just data they want it to be actionable. This includes having enough information to genuinely assess how different travel choices affect their carbon footprint, thereby enabling them to make more environmentally friendly decisions actively.

While some users confirm the app's utility, there is a sentiment that carbon emission information is not a major factor in decision-making for some. This suggests a potential area for the app to integrate carbon data more effectively with other critical factors like traffic conditions, travel time, or cost. Overall, the feedback indicates positivity on the app's focus on low-carbon routes but also highlights the need for deeper data integration and enhanced functionalities to fully meet user expectations and encourage more environmentally responsible behaviours.

Question 4: Which features of navigation apps are most important to you when looking for a route (e.g., real-time traffic updates, ETA accuracy, low carbon route options)?

The feedback from users regarding the navigation app emphasises a strong preference for real-time “traffic updates and ETA accuracy,” highlighting their importance in practical route planning. Several respondents highlight these features as crucial for making timely and efficient travel decisions. While there is an appreciation for low-carbon route options, especially noted by individuals in roles such as vehicle compliance officers and environmental workers, who express a distinct need for balancing ecological considerations with efficiency, the primary demand still leans towards functionality that supports “quick and accurate” travel planning. They also recognise the importance of having environmentally friendly route options but insist that these should not compromise the core functionalities of “speed and accuracy.”

Overall, the trend suggests that while environmental features are valued, they must be integrated seamlessly with essential navigation functionalities that prioritize real-time data and travel efficiency. Users are looking for a holistic app experience that does not sacrifice traditional navigational features for environmental features but rather blends both seamlessly.

Question 5: What aspects do you believe make this application a smart and sustainable solution for reducing carbon emissions?

The prototype also focuses on being smart and sustainable so that one day the program itself or the concept can be implemented in a fully functioning smart city. The feedback provided by the users shows that the prototype is “sustainable” and provides “low carbon routes,” both of which are crucial features that align well with the goals of smart cities and environmental sustainability. Users recognise the app as a “tool” that not only assists in reducing carbon emissions by suggesting routes with lower environmental impact but also enhances their awareness of ecological considerations during travel. Overall, the feedback suggests that while users value the app's green credentials, they also expect it to function competently as a navigation aid, underlining the need for a balanced approach in its development and improvement.

Question 6: Overall, what are your thoughts and opinions on using the app?

This question purely collects the user’s thoughts and opinions on the application which could reveal more information on the prototype. The responses that generally indicate a positive experience towards the transportation tracker app, with users appreciating its core concept and functionality aimed at reducing carbon footprints. Many respondents recognise the app as “beneficial, innovative, and aligned with environmental values.” The app is seen as a bridge between traditional navigation technologies and enhanced environmental consciousness, highlighting its utility in promoting sustainable travel habits.

While the app is “commended” for its efficiency and straightforward navigation options, there is a call for further enhancements in visual appeal, user interface, and the integration of direct route display capabilities instead of redirecting users to external services like Google Maps. Users suggest that the app could improve in areas such as “UI/UX design” to make it more engaging and user-friendly. Additionally, there is

feedback pointing towards the need for a balance between focusing on carbon emissions and maintaining practical usability and speed, ensuring that the app remains competitive with other navigation tools in terms of functionality.
Overall, the trend suggests that while the app is valued for its environmental focus and basic navigation capabilities, there is potential for growth and improvement that could increase its appeal and effectiveness, making it not only a tool for navigation but also to help engage in environmentally responsible behaviours.

Question 7: On a scale from 1 to 10, how satisfied are you with the low carbon emission routes suggested by the app?
The graph in Fig. 31 presents user satisfaction with the prototype having an average rating of 73.3% out of 100% which has exceeded the expectations set at 60%. The distribution of scores indicates a generally positive response, with most respondents rating their satisfaction at 7 (41.67%). There are peaks at scores of 6 (16.66%), and 9 (16.66%) as well, suggesting some variation in user experience or expectations. There are no ratings below 5 indicating that most users find the prototype either satisfactory or above average. This data suggests that while the app is performing well in its goal to suggest environmentally friendly routes, there is still room for improvement to elevate the user experience to higher satisfaction levels.

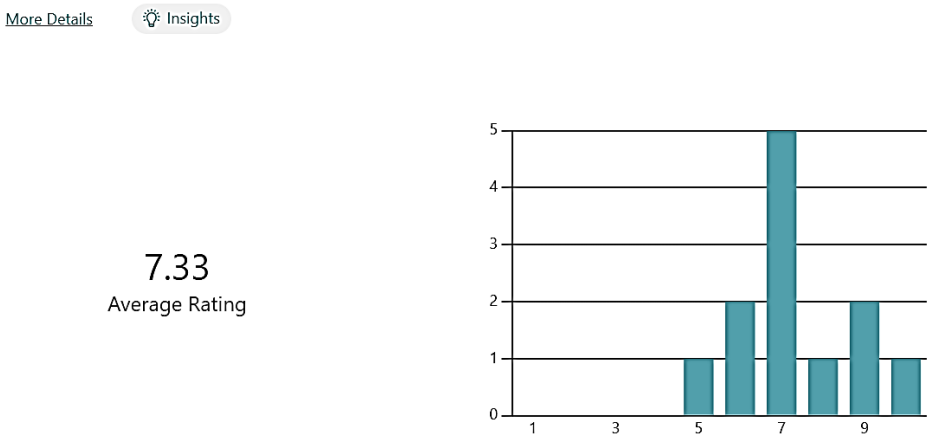


Fig. 31: Participants Survey Result for Question 7

6 Discussion

This research project has successfully developed a smart and sustainable Transportation Tracker prototype that utilises data from various transportation modes to calculate carbon emissions. The prototype has proved to provide users with alternative routes that minimise carbon footprints, aligning closely with the UK's ambitious goals for reducing carbon emissions in the transportation sector through the net zero plan.

Through rigorous literature review, design, implementation, and evaluation phases, the project has addressed all objectives set. It has integrated some real-time traffic data and emissions data to enhance route recommendations and has been fully tested through both black-box and white-box methods, confirming the prototype's effectiveness and user-friendliness.

Moreover, the prototype's capability to provide routes directly to the user concerning emissions data marks a significant advance in navigation technology with an environmental focus which has not been introduced on major applications like Google Maps, Waze, or TomTom. The project has not only met its aim but has also set a foundation for future enhancements, such as expanding the range of transport modes and incorporating more dynamic environmental data, which could further position the Transportation Tracker as a crucial tool in the global effort to mitigate climate change. The positive feedback from user surveys further highlights the prototype's potential impact and relevance in fostering sustainable transportation practices.

The development of the Transportation Tracker for Carbon Emissions represents a critical step towards creating a smart and sustainable transport solution. The prototype is designed to calculate and display low carbon emission routes across various transportation modes, with relevant integrations to real-time traffic data and advanced data processing techniques to optimise route selection based on carbon emissions.

Overall, user feedback highlighted the prototype's effectiveness in providing environmentally conscious route options, though it also revealed areas for improvement, such as the integration of more comprehensive traffic and emissions data to enhance accuracy and usability. The prototype was well-received for its core functionality, allowing users to choose routes based on carbon emissions alongside other traditional navigation factors such as speed and traffic conditions. The iterative development process facilitated through the Agile methodology allowed for continuous refinement, demonstrating a functional application that aligns with the project's aims and objectives.

While feedback in general was mixed, the survey had a consistent indication of a lot of concerns with traffic and speed. During the development stage, this was a concern that had risen whether to prioritise speed or emissions. [42] explained how rerouting apps are used by many people to get out of traffic and find the fastest route but doing so creates even more traffic. Alongside this, they argue that the system must be thought about as a whole so that rerouting algorithms can benefit both citizens and the environment. However, [43] also argues that traditional navigation apps like Google Maps, Waze, and TomTom provide the "best path" to reach a destination, "they do not care about collective effects on the city." Based on their research they shed some light stating that "On the one hand, navigation apps may provide benefits in mitigating carbon dioxide emissions, on the other hand, they may increase the population exposure to pollution in densely populated areas."

Focusing on this project, the aim is to develop a prototype that "reduces" carbon emissions and with the consideration of all the research conducted the decision made was to favour a routing algorithm that focuses on reducing carbon emissions over speed. However, with this focus, most routes that are returned by the prototype favour the use

of public transport as it has some of the most reduced emissions when compared to private vehicles as discussed earlier. Walking and bicycling were refined to ensure that they are not always selected as they directly emit 0 kgCO2 meaning they are always the best mode of transport but it is not ideal nor convenient to users so limits were set on the distance they can travel.

Therefore, we carried out another iteration and attempted to create a system that can provide a series of options for the users so that they choose what is the best option for them considering their current circumstances and of course the amount of carbon emission. For example, if using a car for a route takes 20 minutes from point A to Point B and using public transport takes 30 minutes but with 70% less carbon emissions, they can choose public transport when they are planning for their journey, if they have 10 minutes extra time. This is explained further in 6.1.

6.1 Transportation Tracker for Carbon Emissions V5 – Updated prototype output to display all modes of transportation

The prototype successfully calculates and outputs the best mode of transport that is low in carbon emissions however, due to the emissions factor, the prototype will most likely favour public transportation over driving, bicycling, or walking as shown in the calculation logic to output the best transportation route (Fig. 29). In this branch iteration of V4, the calculation logic for choosing the best route has been removed and instead, all routes via all modes of transportation are displayed using the Directions API. This was done by storing all route information for each mode of transport in a list. The "route_options" list would then be looped until all modes of transportation and their route information were displayed to the user (Fig. 32).

```
def get_route_info(origin, destination, api_key):
                route_info = {
                    "mode": mode,
                    "distance": distance,
                    "emissions": emissions,
                    "duration_text": duration_text,
                    "arrival_time": arrival_time_str,
                    "traffic_delay": traffic_delay_text
                }
                route_options[mode].append(route_info)

    return route_options

def format_route_info(route_options):
    result = ""
    time_of_day = get_time_of_day()
    for mode, routes in route_options.items():
        result += f"Mode of Transport: {mode.capitalize()}\n"
        for idx, route in enumerate(routes):
            result += (f"Route {idx + 1}:\n"
                       f"Total Journey Distance: {route['distance']:.2f} Miles\n"
                       f"Estimated carbon emissions: {route['emissions']:.2f} kg CO2\n"
                       f"Estimated duration: {route['duration_text']}\n"
                       f"Departure time: {time_of_day}\n"
                       f"Arrival time: {route['arrival_time']}\n"
                       f"Traffic delay: {route['traffic_delay']}\n\n")
    return result
```

Fig. 32: Updated "get_route_info" and "format_route_info" function

As a result, the output of the prototype now allows the user to scroll through all driving, public transport, bicycling and walking routes available for a given journey using the Directions API. The information remains the same however, instead of giving only one option to the user, all options are available allowing the user to freely choose any mode of transportation.Providing users with only one option may not effectively persuade them to reduce their carbon footprint. However, by displaying all available results, users can evaluate and select the mode of transport that best suits their needs while being informed about the carbon emissions associated with each route. This comprehensive information encourages users to consider the environmental impact of their choices. By presenting all the details, users are more likely to make informed decisions that align with their preferences and contribute to reducing their carbon footprint.

6.2 Efficiency-Oriented Approaches to Transportation Planning

Efficiency-oriented approaches in transportation planning often focus on optimising specific outcomes, such as minimising travel time, reducing costs, or lowering carbon emissions. While these objectives are critical, they can create trade-offs that impact other priorities, such as user convenience, safety, and accessibility. In this study, the Transportation Tracker prototype adopts an environmental efficiency lens, prioritising carbon emissions reduction. This design aligns with the UK's Net Zero targets and broader sustainability goals, encouraging environmentally conscious travel behaviour.

However, a singular focus on efficiency can have limitations. For instance, prioritising low-carbon routes may not always align with users' preferences for the

fastest or most cost-effective routes. This could lead to lower user adoption rates if the system fails to adequately consider the diverse needs of its users. Additionally, efficiency-oriented systems may inadvertently disadvantage certain groups, such as individuals with mobility challenges, if they do not account for accessibility or social equity.

The pros of efficiency-oriented approaches include their ability to provide quantifiable and actionable insights that align with measurable goals, such as carbon emissions reduction. On the other hand, the cons include a potential lack of flexibility in addressing complex user requirements and broader societal concerns. Balancing efficiency with adaptability and inclusivity is therefore a key challenge in developing transportation systems that cater to multifaceted user needs. Future iterations of the Transportation Tracker prototype could explore incorporating additional criteria, such as travel cost, accessibility, and safety, to create a more holistic decision-making tool.

6.3 Aligning with Multifaceted User Requirements and Social Goals

Modern urban transportation planning must balance individual user needs with collective societal objectives, creating systems that are not only efficient but also equitable, inclusive, and aligned with long-term sustainability goals [44]. The Transportation Tracker prototype takes an important step by addressing the environmental dimension of user decisions. However, user requirements are multifaceted, encompassing priorities such as time, convenience, cost, safety, and even the social value of their choices.

One critical consideration is how the prototype can accommodate diverse user groups with varying needs. For instance, users with mobility constraints may prioritise accessibility over carbon efficiency, while low-income users may prioritise cost savings. The integration of personalised preferences, such as the ability to prioritise cost-effective or fastest routes alongside carbon-efficient options, would allow the system to cater to a broader audience.

Social goals, such as equity in transport access and fostering behavioural shifts toward sustainable practices, also require attention. Urban transport systems play a vital role in promoting social inclusion by ensuring all users have access to affordable and sustainable travel options. By integrating features that promote accessibility, equity, and public health benefits, the prototype could contribute to these broader societal goals.

6.4 Scalability of the Transportation Tracker Prototype

Scalability is a crucial aspect of any digital tool intended for widespread adoption, particularly in urban transportation systems where demand fluctuates significantly. The Transportation Tracker prototype has been developed with scalability as a foundational design principle. By leveraging cloud-based infrastructure and integrating with widely used APIs, the system is capable of handling concurrent user requests efficiently. This approach ensures that the prototype can cater to a growing user base, making it suitable for deployment in densely populated cities or regions with high transportation demands.

The modular architecture of the system further supports scalability. Each component of the system, including data processing, route calculation, and user interface, can be independently scaled to address increased demand. For instance, if the number of users

accessing the tool rises significantly during peak hours, additional computing resources can be dynamically allocated to maintain optimal performance. However, while these scalability features are inherent to the system's design, their effectiveness under extreme demand scenarios, such as city-wide events or emergencies, remains untested. Future work should focus on conducting stress tests and simulations to evaluate system performance under high-load conditions, identifying potential bottlenecks, and refining the system's resource management strategies.

Incorporating these improvements will solidify the Transportation Tracker's capacity to function reliably at scale, thereby enhancing its practical applicability in real-world urban environments. Additionally, engaging with municipalities and transportation agencies could provide opportunities to align the system's scalability with broader urban planning initiatives.

6.5 Adaptability to Diverse Regions and Contexts

The adaptability of the Transportation Tracker prototype is another significant consideration for its application beyond the initial context in which it was developed. Urban environments vary widely in terms of transportation infrastructure, available modes of transit, user preferences, and environmental policies. The prototype addresses this variability by employing a modular and flexible design, allowing it to integrate region-specific data sources with minimal structural changes. For instance, the system relies on open APIs for route and emissions data, which can be easily replaced or supplemented with localised data to reflect the specific characteristics of a new region.

To apply the prototype in other regions, several steps would be necessary. First, collaboration with local authorities and data providers is essential to ensure access to accurate, regionally relevant information. This includes transportation network layouts, real-time traffic data, and local emissions factors. Second, adjustments to the algorithm may be required to account for regional differences in transportation modes. For example, cities like Venice may rely on water-based transportation, while some regions may incorporate emerging technologies such as electric scooters or autonomous vehicles. Third, cultural and behavioural considerations must be considered, as user preferences and priorities can differ significantly across regions.

Piloting the Transportation Tracker in diverse urban settings would provide valuable insights into its adaptability. These trials would allow for the identification of region-specific challenges and opportunities, as well as the refinement of the tool to ensure its relevance and effectiveness in different contexts. By demonstrating adaptability, the prototype can contribute to global efforts to reduce carbon emissions in urban transport, supporting the transition to sustainable and smart cities worldwide.

7 Conclusions

This project addresses the critical issue of carbon emissions in the transportation sector by developing a smart transportation tracker for carbon emissions. It contributes significantly to both academic research and practical applications by providing a functional solution that leverages real-time data to suggest environmentally friendly travel routes. The project's focus on integrating carbon emissions data into everyday

decision-making tools fills a significant gap in current transportation apps, which often prioritise time or traffic considerations over environmental impact. This innovation aligns with broader sustainability goals, such as the UK's net-zero emissions plan, and demonstrates how real-time data integration can substantially influence carbon emissions reduction.

The project underscores the potential of smart transportation systems within smart cities, showcasing how software solutions can be effectively designed to tackle climate change. By offering valuable data, the application empowers users to make informed decisions that favour low-carbon transportation options. This not only supports individual efforts to reduce carbon footprints but also aids urban planners in promoting sustainable transportation habits.

7.1 Contributions of this research

Theoretical Contributions. This research contributes to the emerging body of knowledge on sustainable urban mobility by introducing a novel approach to route optimisation that prioritises environmental impact over speed. While existing navigation systems like Google Maps and Waze focus primarily on reducing travel time, this research advances the theoretical understanding of how digital tools can be designed to promote sustainability by incorporating real-time carbon emission data. It provides a framework for further exploration of integrating environmental factors into smart city solutions and sustainable transportation systems.

The study also adds to the literature on decision-making in the context of environmental sustainability by proposing a model that empowers users to balance personal convenience with ecological responsibility, thus opening new avenues for research in user-centred, eco-friendly technology adoption.

Furthermore, this project contributes to the field of smart city development by demonstrating how transport systems can be enhanced through real-time data analytics, offering insights into the integration of digital solutions that address urban sustainability challenges. The research aligns with and supports global sustainability goals, particularly the UN Sustainable Development Goals (SDGs) and the UK's Net Zero strategy, offering a theoretical model that bridges the gap between digital transformation and environmental objectives.

Practical Contributions. From a practical perspective, the Transportation Tracker prototype offers a functional and scalable tool for reducing carbon emissions in urban transport. Unlike current navigation apps, the prototype considers the environmental cost of different travel modes, providing users with multiple route options that balance carbon savings with travel time. This enables a more informed decision-making process for users, who are increasingly seeking ways to minimise their carbon footprint. The project's practical contribution lies in its ability to influence user behaviour by encouraging sustainable transport choices, which can have a direct impact on lowering urban carbon emissions.

Additionally, this research provides valuable insights for urban planners and policymakers. By integrating real- time traffic and environmental data, the prototype can be a critical tool for cities looking to meet their sustainability targets. The ability to dynamically present all available transport modes, along with their associated emissions, allows for better planning and design of sustainable urban mobility solutions. This project serves as a proof of concept for cities aiming to incorporate smart technologies to tackle climate change and promote greener urban living.

7.2 Limitations

Despite the promising outcomes, the project has several limitations that warrant further exploration:

- Scope of Emissions Covered: While the prototype effectively calculates carbon emissions for different transportation modes, it does not account for indirect emissions associated with activities such as food production for walking and cycling, or the manufacturing processes of bicycles. This oversight can lead to an underestimation of the total environmental impact of these modes of transport.
- Human Behaviour: The effectiveness of the application largely depends on user adoption and behaviour change. While initial feedback indicates increased carbon awareness among some users, broader behavioural shifts are challenging to achieve and require more in-depth analysis and strategic interventions.
- Data Integration and Accuracy: The current application relies on available real-time data, which can sometimes be incomplete or inaccurate. Enhancing the accuracy and comprehensiveness of this data is crucial for the app's reliability and effectiveness.

7.3 Recommendations for Future Research

To build on the achievements of this project and address its limitations, several recommendations for future research and development are proposed:

- Advanced Integration of Real-Time Data: Future iterations of the application should incorporate more dynamic variables such as weather conditions, ongoing road works, and live traffic data. Real-time carbon prediction and advanced routing algorithms could provide more actionable insights, helping users see the immediate impact of their travel choices.
- User Interface and Experience Enhancements: Improving the app's user interface and experience is essential for broader adoption. Future developments should include personalised settings that allow users to prioritise their preferences, such as balancing between the fastest route and the lowest carbon route. Features like saving favourite routes, inputting specific vehicle details for more accurate emissions calculations, and customisable

preferences for route options could significantly enhance user engagement and satisfaction.

- Integration with Public Transportation and Multimodal Travel: Expanding the app to seamlessly integrate with public transportation systems and suggest multimodal travel options can provide a more comprehensive view of available travel choices. This should include recommendations for combining different modes of transport (e.g., driving to a train station and then taking a train) based on time, cost, and carbon footprint, as well as options for ride-sharing and biking.
- Behavioural Interventions: Conducting further research into the behavioural aspects of sustainable transportation can inform the development of strategies to encourage wider adoption of low-carbon travel choices. This could involve integrating educational components that highlight the environmental impact of different transportation modes and showcasing the benefits of sustainable travel through personalized feedback and visual aids.
- Other modes of transportation: While this study focused on commonly used urban transportation modes such as walking, cycling, driving, and public transport, future research could explore less conventional modes of transport like water-based systems and emerging technologies such as passenger drones. For instance, water transport could be relevant in cities like Venice, where canals form the primary infrastructure for mobility. Similarly, as air transport technologies, such as passenger drones, advance, they may play a critical role in urban transportation systems.

By incorporating the recommended improvements, future research can further contribute to the development of effective tools that support global efforts to reduce carbon emissions and combat climate change.

References

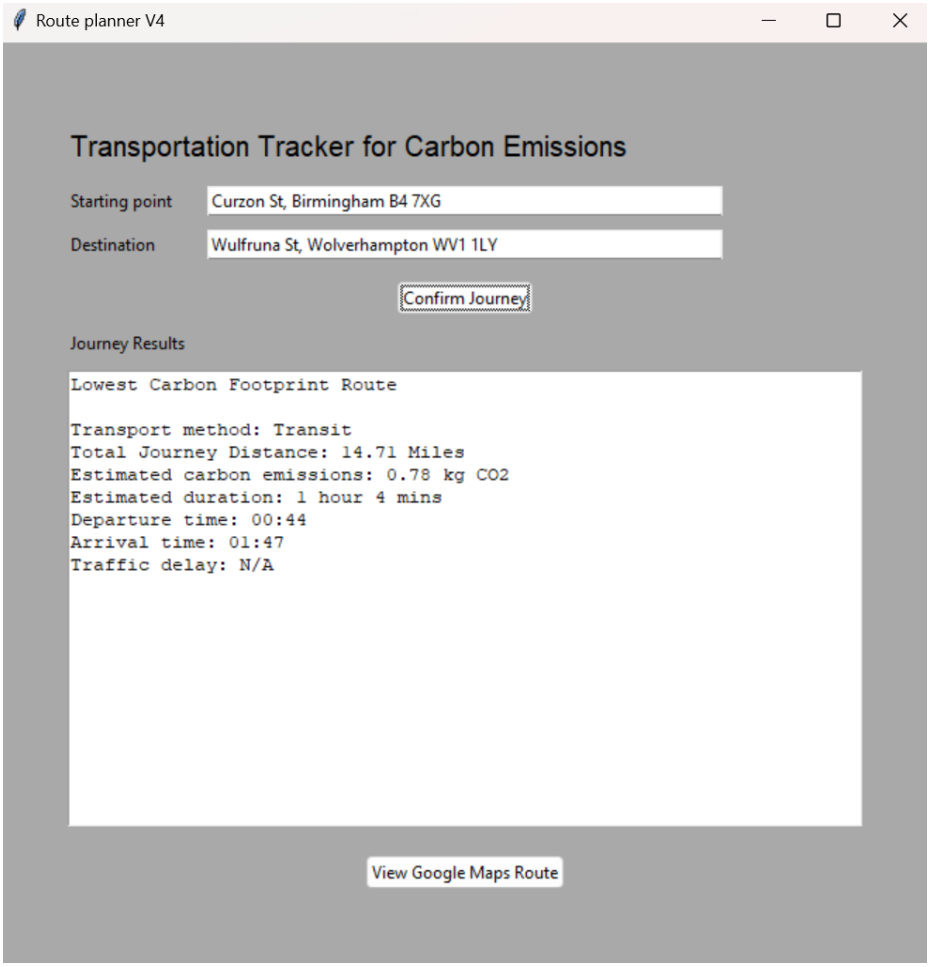
1. Zuo, C., Birkin, M., Clarke, G., et al. (2018) Reducing carbon emissions related to the transportation of aggregates: Is road or rail the solution? Transportation Research Part A: Policy and Practice, 117: 26–38. https://doi.org/10.1016/j.tra.2018.08.006.
2. Gudde, P., Oakes, J., Cochrane, P., et al. (2021) The role of UK local government in delivering on net zero carbon commitments: You've declared a Climate Emergency, so what's the plan? Energy Policy, 154: 112245. https://doi.org/10.1016/j.enpol.2021.112245.
3. Dixon, J., Bell, K. and Brush, S. (2022) Which way to net zero? a comparative analysis of seven UK 2050 decarbonisation pathways. Renewable and Sustainable Energy Transition, 2: 100016. https://doi.org/10.1016/j.rset.2021.100016.
4. Glaister, S. (2021) StackPath. Available at: https://www.instituteforgovernment.org.uk/sites/default/files/hs2-levelling-up-stephen-glaister.pdf [Accessed: 28 December 2023].
5. Department of Transport (2021) Transport Statistics Great Britain: 2021. Available at: https://www.gov.uk/government/statistics/transport-statistics-great-britain-2021/transport-statistics-great-britain-2021#overview-of-transport-statistics/ [Accessed: 3 January 2024].

6. Liu, T., Ceder, A., Bologna, R., et al. (2016) Commuting by Customized Bus: A Comparative Analysis with Private Car and Conventional Public Transport in Two Cities. Journal of Public Transportation, 19(2): 55–74. https://doi.org/10.5038/2375-0901.19.2.4.
7. Zantalis, F., Koulouras, G., Karabetsos, S., et al. (2019) A Review of Machine Learning and IoT in Smart Transportation. Future Internet, 11(4): 94. https://doi.org/10.3390/fi11040094.
8. Olaoluwa, M. O., Taiwo, A. O. & Oteyola, A. O. (2023). Climate Change and Human Infectious Diseases. Https://Services.Igi-Global.Com/Resolvedoi/Resolve.Aspx?Doi=10.4018/978-1-7998-9414-8.Ch002, 31–42. https://doi.org/10.4018/978-1-7998-9414-8.CH002
9. UK Government (2021). UK enshrines new target in law to slash emissions by 78% by 2035. Available at: https://www.gov.uk/government/news/uk-enshrines-new-target-in-law-to-slash-emissions-by-78-by-2035 [Accessed: 4 October 2023]
10. O'Sullivan, C. (2023). 2022 UK Greenhouse Gas Emissions, Provisional Fig.s. Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1147372/2022_Provisional_emissions_statistics_report.pdf [Accessed: 4 October 2023]
11. Jin, T. and Kim, J. (2018) What is better for mitigating carbon emissions – Renewable energy or nuclear energy? A panel data analysis. Renewable and Sustainable Energy Reviews, 91: 464–471. https://doi.org/10.1016/j.rser.2018.04.022.
12. Duan, C., Zhu, W., Wang, S., et al. (2022) Drivers of global carbon emissions 1990–2014. Journal of Cleaner Production, 371: 133371. https://doi.org/10.1016/j.jclepro.2022.133371.
13. Ivanova, D. and Wieland, H. (2023) Tracing carbon footprints to intermediate industries in the United Kingdom. Ecological Economics, 214: 107996–107996. https://doi.org/10.1016/j.ecolecon.2023.107996.
14. Castle, J.L. and Hendry, D.F. (2021) Can the UK achieve net-zero greenhouse gas emissions by 2050? ora.ox.ac.uk, 953(1). Available at: https://ora.ox.ac.uk/objects/uuid:c6e8a6a5-7f94-48d3-9e64-5fdffcf2a399/files/sd217qq34n [Accessed: 28 December 2023].
15. Cornet, Y., Dudley, G. and Banister, D. (2018) High Speed Rail: Implications for carbon emissions and biodiversity. Case Studies on Transport Policy, 6(3): 376–390. https://doi.org/10.1016/j.cstp.2017.08.007.
16. Koide, R., Lettenmeier, M., Akenji, L., et al. (2021) Lifestyle carbon footprints and changes in lifestyles to limit global warming to 1.5 °C, and ways forward for related research. Sustainability Science, 16(6): 2087–2099. https://doi.org/10.1007/s11625-021-01018-6.
17. Berg, J. and Ihlström, J. (2019) The Importance of Public Transport for Mobility and Everyday Activities among Rural Residents. Social Sciences, 8(2): 58. :https://doi.org/10.3390/socsci8020058.
18. Thomas, H. and Serrenho, A.C. (2024) Using different transport modes: An opportunity to reduce UK passenger transport emissions? Transportation Research Part D: Transport and Environment, 126: 103989–103989. https://doi.org/10.1016/j.trd.2023.103989.
19. Pantelidou, H., Casey, G., Chapman, T., et al. (2016) Re-thinking UK transport emissions – getting to the 2050 targets. Proceedings of the Institution of Civil Engineers - Civil Engineering, 169 (4): 177–183. https://doi.org/10.1680/jcien.15.00076.
20. Logan, K.G., Nelson, J.D., Brand, C., et al. (2021) Phasing in electric vehicles: Does policy focusing on operating emission achieve net zero emissions reduction objectives?

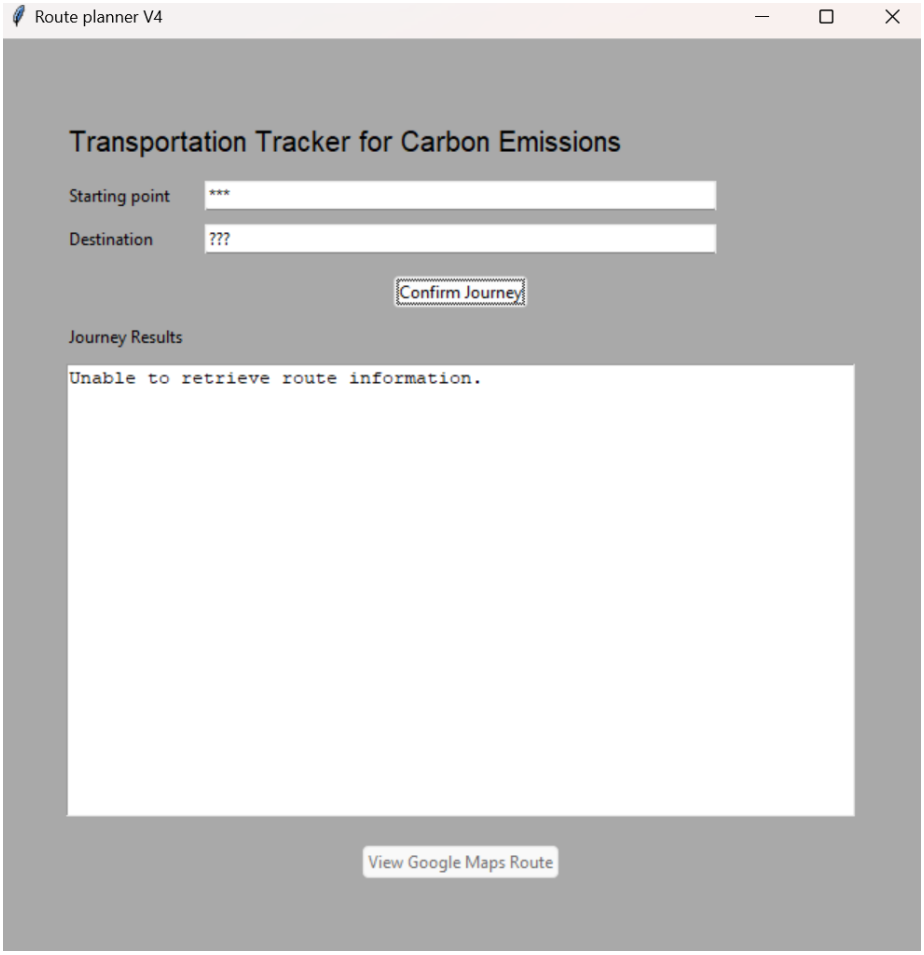
Transportation Research Part A: Policy and Practice, 152: 100–114. https://doi.org/10.1016/j.tra.2021.08.001.
21. Moura, F. and Silva, J.D.A. (2019) Smart Cities: Definitions, Evolution of the Concept and Examples of Initiatives. Available at: https://www.researchgate.net/profile/Filipe-Moura-2/publication/335239465_Smart_Cities_Definitions_Evolution_of_the_Concept_and_Examples_of_Initiatives/links/5e342c0ca6fdccd9657adad1/Smart-Cities-Definitions-Evolution-of-the-Concept-and-Examples-of-Initiatives.pdf [Accessed: 6 January 2024].
22. Ramaprasad, A., Sánchez-Ortiz, A. and Syn, T. (2017) A Unified Definition of a Smart City. Lecture Notes in Computer Science, 10428 (10428): 13–24. https://doi.org/10.1007/978-3-319-64677-0_2.
23. Toli, A.M. and Murtagh, N. (2020) The Concept of Sustainability in Smart City Definitions. Available at: https://www.frontiersin.org/articles/10.3389/fbuil.2020.00077/full/ [Accessed: 6 January 2024].
24. Saarika, P.S., Sandhya, K. and Sudha, T. (2017) "Smart transportation system using IoT," In 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon), Bengaluru, India, 2017, pp. 1104-1107, https://doi.org/10.1109/smarttechcon.2017.8358540.
25. Jan, B., Farman, H., Khan, M., et al. (2019) Designing a Smart Transportation System: An Internet of Things and Big Data Approach. IEEE Wireless Communications, 26(4): 73–79. https://doi.org/10.1109/mwc.2019.1800512.
26. Nocera, S., Ruiz-Alarcón-Quintero, C. and Cavallaro, F. (2018) Assessing carbon emissions from road transport through traffic flow estimators. Transportation Research Part C: Emerging Technologies, 95: 125–148. https://doi.org/10.1016/j.trc.2018.07.020.
27. Kholod, N., Evans, M., Gusev, E., et al. (2016) A methodology for calculating transport emissions in cities with limited traffic data: Case study of diesel particulates and black carbon emissions in Murmansk. Science of The Total Environment, 547: 305–313. https://doi.org/10.1016/j.scitotenv.2015.12.151.
28. Ayyildiz, K., Cavallaro, F., Nocera, S., et al. (2017) Reducing fuel consumption and carbon emissions through eco-drive training. Transportation Research Part F: Traffic Psychology and Behaviour, 46: 96–110. https://doi.org/10.1016/j.trf.2017.01.006.
29. Liu, J., Li, J., Chen, Y., et al. (2023) Multi-scale urban passenger transportation CO2 emission calculation platform for smart mobility management. Applied Energy, 331: 120407. https://doi.org/10.1016/j.apenergy.2022.120407.
30. Ranacher, P., Brunauer, R., Trutschnig, W., et al. (2015) Why GPS makes distances bigger than they are. International Journal of Geographical Information Science, 30(2): 316–333. https://doi.org/10.1080/13658816.2015.1086924.
31. Poska, S., Kaplan, A. and Alford, J. (2021) Impact of Navigation Applications on Traffic Operations. Institute of Transportation Engineers. ITE Journal, 91(2): 37–42. Available at: https://www.proquest.com/docview/2486868602?fromopenview=true&pq-origsite=gscholar&sourcetype=Scholarly%20Journals# [Accessed: 16 January 2024]
32. Hein, G.W. (2020) Status, perspectives and trends of satellite navigation. Satellite Navigation, 1(1). https://doi.org/10.1186/s43020-020-00023-x.
33. Haria, V., Shah, Y., Gangwar, V., et al. (2019) The Working of Google Maps, and the Commercial Usage of Navigation Systems. International Journal of Innovative Research in Technology, 6(5): 2349–6002. Available at: https://ijirt.org/master/publishedpaper/IJIRT148707_PAPER.pdf [Accessed: 6 December 2023].

34. Siuhi, S. and Mwakalonge, J. (2016) Opportunities and challenges of smart mobile applications in transportation. Journal of Traffic and Transportation Engineering (English Edition), 3(6): 582–592. https://doi.org/10.1016/j.jtte.2016.11.001.
35. Guo, J., Wang, X., Fan, S., et al. (2017) Forward and reverse logistics network and route planning under the environment of low-carbon emissions: A case study of Shanghai fresh food E-commerce enterprises. Computers & Industrial Engineering, 106: 351–360. https://doi.org/10.1016/j.cie.2017.02.002.
36. Mishra, S., Singh, N. and Bhattacharya, D. (2021) Application-Based COVID-19 Micro-Mobility Solution for Safe and Smart Navigation in Pandemics. ISPRS International Journal of Geo-Information, 10(8): 571. https://doi.org/10.3390/ijgi10080571.
37. Verma, A., Khatana, A. and Chaudhary, S. (2017) A Comparative Study of Black Box Testing and White Box Testing. International Journal of Computer Sciences and Engineering, 5(12): 301–304. https://doi.org/10.26438/ijcse/v5i12.301304.
38. Jacob, P.M. and Prasanna, M. (2016) "A Comparative analysis on Black box testing strategies." In 2016 International Conference on Information Science (ICIS), Kochi, India, 2016, pp. 1-6, https://doi.org/10.1109/INFOSCI.2016.7845290.
39. Google for Developers (2024) Google Maps Platform Documentation | Directions API. Available at: https://developers.google.com/maps/documentation/directions [Accessed: 4 April 2024].
40. Yurday, E. (2024) Average CO2 Emissions per Car in the UK. Available at: https://www.nimblefins.co.uk/average-co2-emissions-car-uk#nogo [Accessed: 18 April 2024].
41. Office of Rail and Road (2023) Rail emissions April 2022 to March 2023. Available at: https://dataportal.orr.gov.uk/media/1dzb2awz/rail-emissions-2022-23.pdf [Accessed: 18 April 2024].
42. Macfarlane, J. (2019) When apps rule the road: The proliferation of navigation apps is causing traffic chaos. It's time to restore order. IEEE Spectrum, 56 (10): 22–27. https://doi.org/10.1109/mspec.2019.8847586.
43. Cornacchia, G., Matteo Böhm, Mauro, G., et al. (2022) "How routing strategies impact urban emissions." In: Proceedings of the 30th International Conference on Advances in Geographic Information Systems. 1 November 2022. https://doi.org/10.1145/3557915.3560977.
44. United Nations (UN). (2015). Transforming our world: The 2030 agenda for sustainable development. UN. Retrieved from https://www.un.org/sustainabledevelopment/

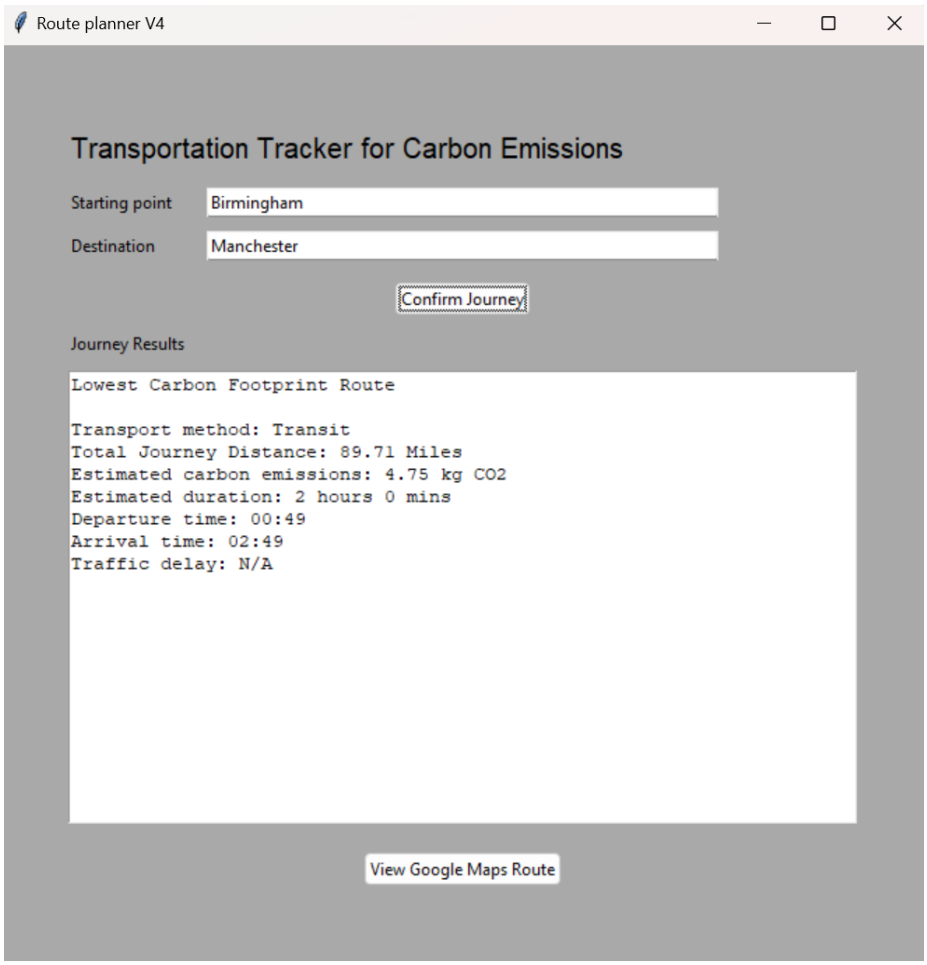
Appendix: Logical Testing Results



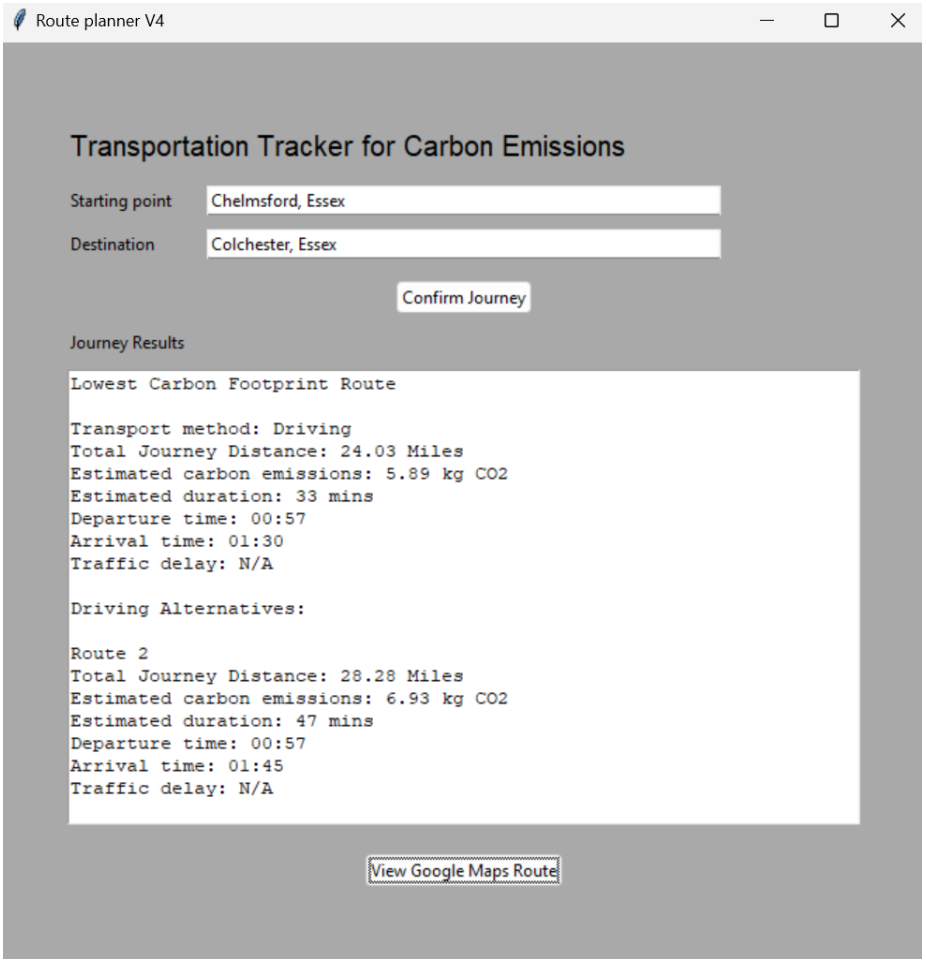
A.1: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
***
Destination
???
Confirm Journey
Journey Results
Unable to retrieve route information.
View Google Maps Route

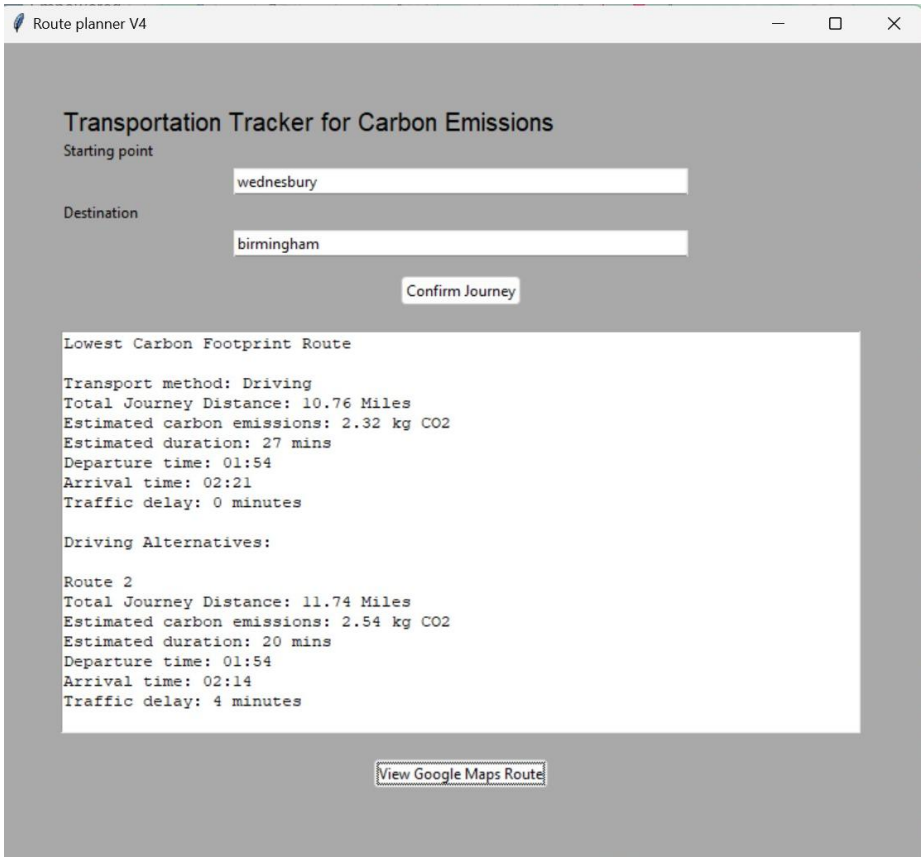
A.2: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
Birmingham
Destination
Manchester
Confirm Journey
Journey Results
Lowest Carbon Footprint Route
Transport method: Transit
Total Journey Distance: 89.71 Miles
Estimated carbon emissions: 4.75 kg CO2
Estimated duration: 2 hours 0 mins
Departure time: 00:49
Arrival time: 02:49
Traffic delay: N/A
View Google Maps Route

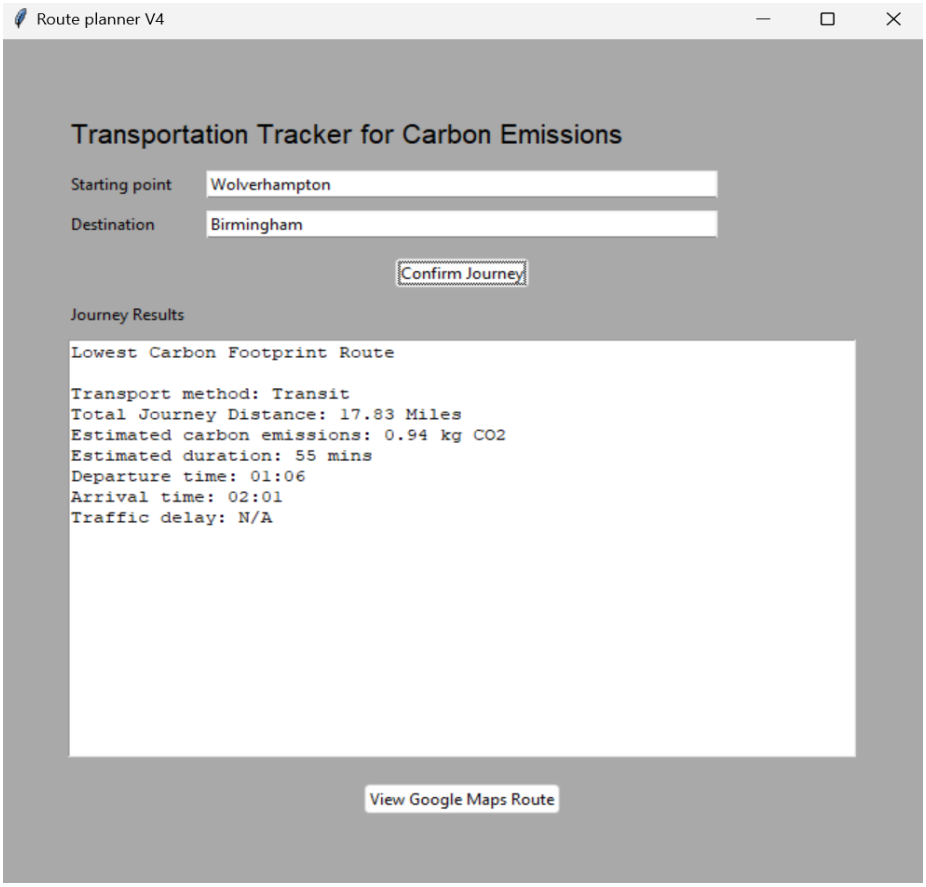
A.3: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
Chelmsford, Essex
Destination
Colchester, Essex
Confirm Journey
Journey Results
Lowest Carbon Footprint Route
Transport method: Driving
Total Journey Distance: 24.03 Miles
Estimated carbon emissions: 5.89 kg CO2
Estimated duration: 33 mins
Departure time: 00:57
Arrival time: 01:30
Traffic delay: N/A
Driving Alternatives:
Route 2
Total Journey Distance: 28.28 Miles
Estimated carbon emissions: 6.93 kg CO2
Estimated duration: 47 mins
Departure time: 00:57
Arrival time: 01:45
Traffic delay: N/A
View Google Maps Route

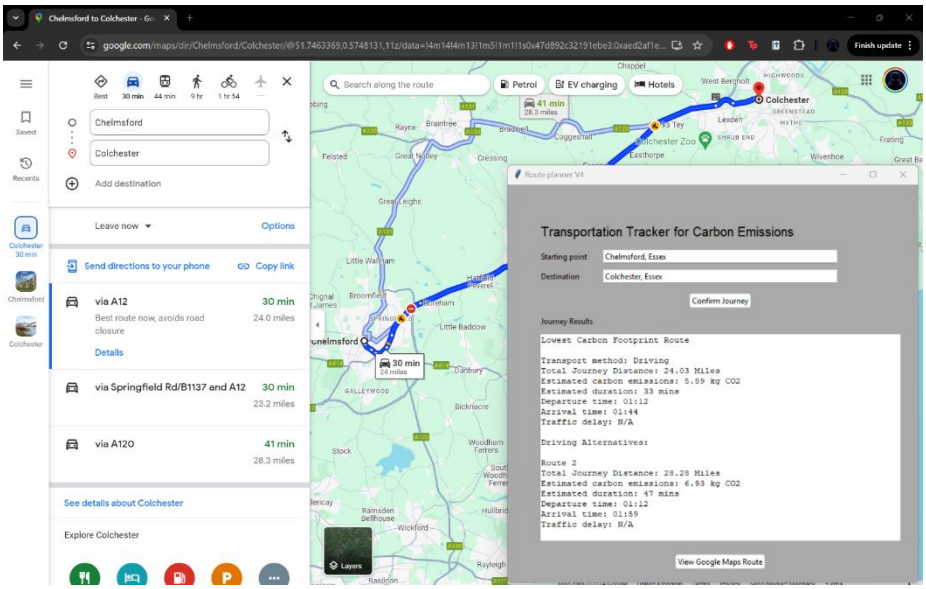
A.4: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
wednesbury
Destination
birmingham
Confirm Journey
Lowest Carbon Footprint Route
Transport method: Driving
Total Journey Distance: 10.76 Miles
Estimated carbon emissions: 2.32 kg CO2
Estimated duration: 27 mins
Departure time: 01:54
Arrival time: 02:21
Traffic delay: 0 minutes
Driving Alternatives:
Route 2
Total Journey Distance: 11.74 Miles
Estimated carbon emissions: 2.54 kg CO2
Estimated duration: 20 mins
Departure time: 01:54
Arrival time: 02:14
Traffic delay: 4 minutes
View Google Maps Route

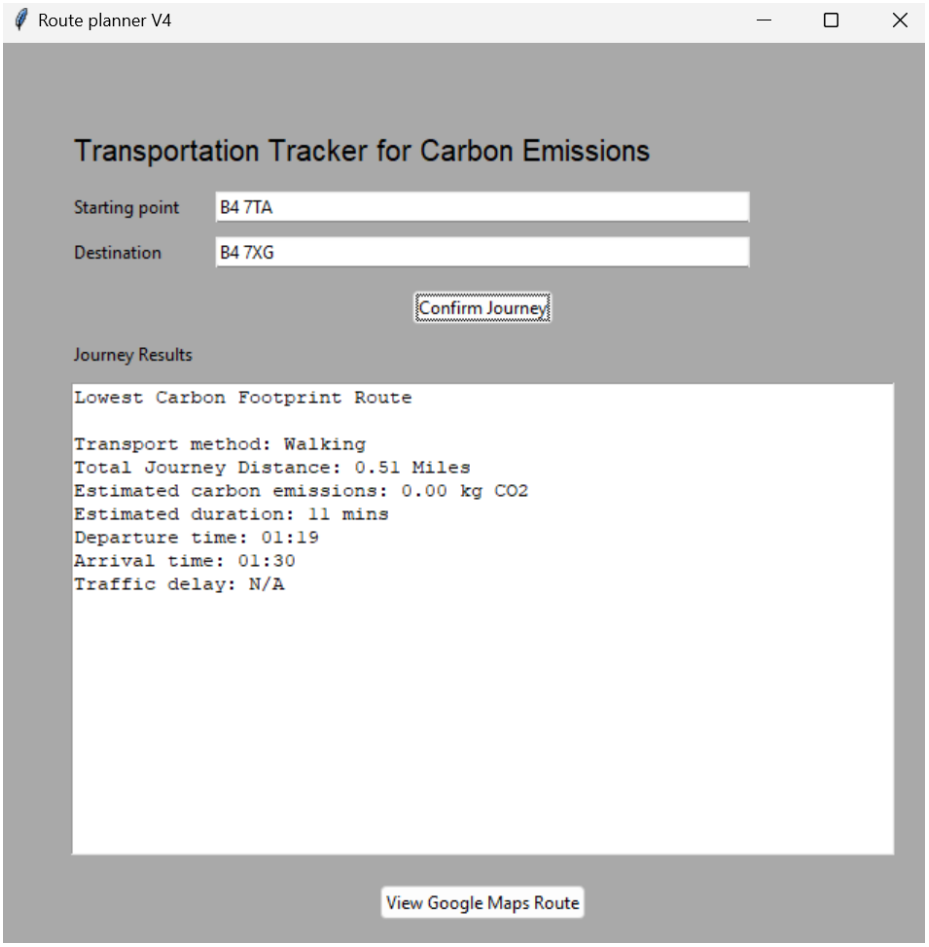
A.5: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
Wolverhampton
Destination
Birmingham
Confirm Journey
Journey Results
Lowest Carbon Footprint Route
Transport method: Transit
Total Journey Distance: 17.83 Miles
Estimated carbon emissions: 0.94 kg CO2
Estimated duration: 55 mins
Departure time: 01:06
Arrival time: 02:01
Traffic delay: N/A
View Google Maps Route

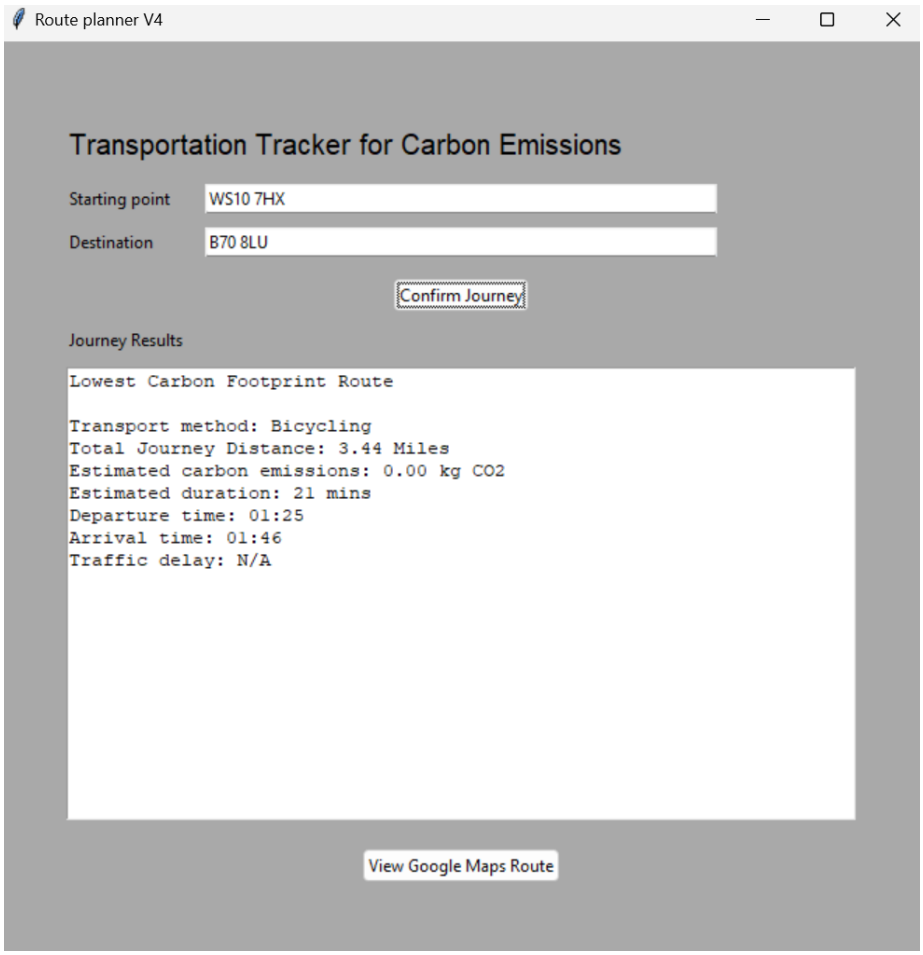
A.6: Result


Chelmsford to Colchester
Chelmsford
Colchester
Add destination
Leave now
Options
Send directions to your phone
Copy link
via A12
30 min
Best route now, avoids road closure
24.0 miles
Details
via Springfield Rd/B1137 and A12
30 min
23.2 miles
via A120
41 min
28.3 miles
See details about Colchester
Explore Colchester
Search along the route
Petrol
EV charging
Hotels
Transportation Tracker for Carbon Emissions
Starting point
Chelmsford, Essex
Destination
Colchester, Essex
Confirm Journey
Journey Results
Lowest Carbon Footprint Route
Transport method: Driving
Total Journey Distance: 24.03 Miles
Estimated carbon emissions: 5.89 kg CO2
Estimated duration: 33 mins
Departure time: 01:12
Arrival time: 01:44
Traffic delay: N/A
Driving Alternatives:
Route 2
Total Journey Distance: 28.28 Miles
Estimated carbon emissions: 6.93 kg CO2
Estimated duration: 47 mins
Departure time: 01:12
Arrival time: 01:59
Traffic delay: N/A
View Google Maps Route

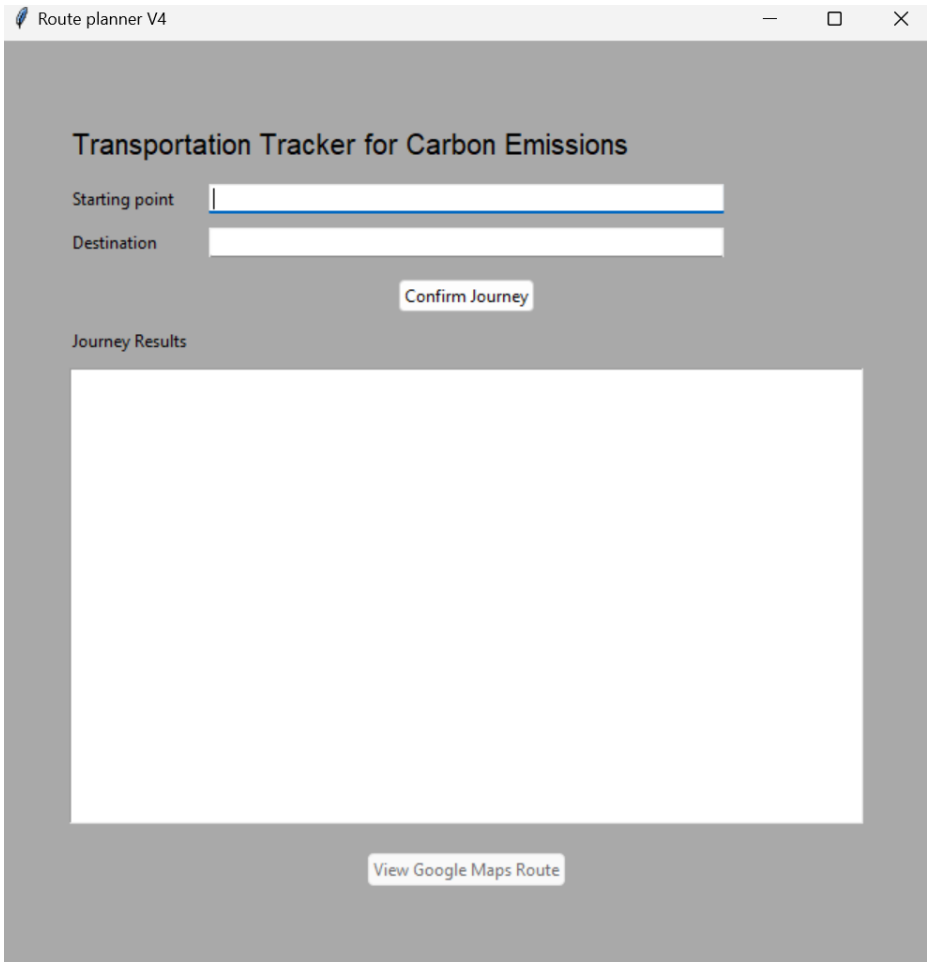
A.7: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
B4 7TA
Destination
B4 7XG
Confirm Journey
Journey Results
Lowest Carbon Footprint Route
Transport method: Walking
Total Journey Distance: 0.51 Miles
Estimated carbon emissions: 0.00 kg CO2
Estimated duration: 11 mins
Departure time: 01:19
Arrival time: 01:30
Traffic delay: N/A
View Google Maps Route

A.8: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
WS10 7HX
Destination
B70 8LU
Confirm Journey
Journey Results
Lowest Carbon Footprint Route
Transport method: Bicycling
Total Journey Distance: 3.44 Miles
Estimated carbon emissions: 0.00 kg CO2
Estimated duration: 21 mins
Departure time: 01:25
Arrival time: 01:46
Traffic delay: N/A
View Google Maps Route

A.9: Result


Route planner V4
Transportation Tracker for Carbon Emissions
Starting point
Destination
Confirm Journey
Journey Results
View Google Maps Route

A.10: Result