Carbon Footprint Calculator

A report submitted to Manchester Metropolitan University for the degree of Bachelor of Science

in the Faculty of Science and Engineering

A blue and white logo

Description automatically generated

2024

By Habiba Begum

21312966@stu.mmu.ac.uk

Theme: Computational Systems

Department of Computing and Mathematics

# Contents

Table of Contents

[Contents 3](#_Toc166322669)

[List of figures 5](#_Toc166322670)

[Chapter 1 – Introduction 7](#_Toc166322671)

[1.1 Background Information 7](#_Toc166322672)

[1.2 Aims 7](#_Toc166322673)

[1.3 Objectives 7](#_Toc166322674)

[1.4 Learning Outcomes 7](#_Toc166322675)

[Chapter 2 Literature survey 7](#_Toc166322676)

[2.1 Languages and Frameworks 7](#_Toc166322677)

[2.2 Methodology 7](#_Toc166322678)

[2.2 Layout and Useability 7](#_Toc166322679)

[2.3 MVC (Model View Controller) 7](#_Toc166322680)

[2.4 Carbon Footprint Formula 7](#_Toc166322681)

[2.5 Data Collection 7](#_Toc166322682)

[2.6 Current System 7](#_Toc166322683)

[Current system 1 7](#_Toc166322684)

[Current system 2 9](#_Toc166322685)

[Current system 3 12](#_Toc166322686)

[Current system 4 14](#_Toc166322687)

[Chapter 3 – Design 16](#_Toc166322688)

[3.1 Task Planning 16](#_Toc166322689)

[3.2 Language and Framework Choices 17](#_Toc166322690)

[3.3 Requirements and Specifications 17](#_Toc166322691)

[3.4 Structural Design 18](#_Toc166322692)

[3.4.1 Functionality Detailed Design 19](#_Toc166322693)

[3.4.2 Usability Detailed Design 21](#_Toc166322694)

[3.5 Design Evaluation (500) 21](#_Toc166322695)

[Chapter 4 - Implementation 22](#_Toc166322696)

[Chapter 5 - Evaluation: (2000 words) 39](#_Toc166322697)

[Chapter 6 - Conclusion: (500 words) 43](#_Toc166322698)

[References 43](#_Toc166322699)

# List of figures

[Figure 1 Table of car CO₂ emissions (*Version Information* 2008) 7](#_Toc166327285)

[Figure 2 Input to Current System 1 https://www.carbonindependent.org/index.html 7](#_Toc166327286)

[Figure 3 results from current system 1 https://www.carbonindependent.org/index.html 7](#_Toc166327287)

[Figure 4 results from current system 1 https://www.carbonindependent.org/index.html 7](#_Toc166327288)

[Figure 5 input of current system 2 CoolClimate Calculator (berkeley.edu) 7](#_Toc166327289)

[Figure 6 input of current system 2 CoolClimate Calculator (berkeley.edu) 7](#_Toc166327290)

[Figure 7 input of current system 2 CoolClimate Calculator (berkeley.edu) 7](#_Toc166327291)

[Figure 8 user pledge for current system 2 CoolClimate Calculator (berkeley.edu) 7](#_Toc166327292)

[Figure 9 leaderboard for current system 2 CoolClimate Calculator (berkeley.edu) 7](#_Toc166327293)

[Figure 10 input for current system 3 https://www.carbonfootprint.com/calculator.aspx 7](#_Toc166327294)

[Figure 11 input for current system 3 https://www.carbonfootprint.com/calculator.aspx 7](#_Toc166327295)

[Figure 12 results from current system 3 https://www.carbonfootprint.com/calculator.aspx 7](#_Toc166327296)

[Figure 13 input for current system 4 7](#_Toc166327297)

[Figure 14 result for current system 4 7](#_Toc166327298)

[Figure 15 user pledge for current system 4 7](#_Toc166327299)

[Figure 16 task plan 7](#_Toc166327300)

[Figure 17 input to output diagram Created using https://online.visual-paradigm.com/ 7](#_Toc166327301)

[Figure 18 calculator page design 7](#_Toc166327302)

[Figure 19 reducing carbon page design 7](#_Toc166327303)

[Figure 20 installing R shiny 7](#_Toc166327304)

[Figure 21 r shiny installing 7](#_Toc166327305)

[Figure 22 using r shiny library 7](#_Toc166327306)

[Figure 23 testing basic code 7](#_Toc166327307)

[Figure 24 error found 7](#_Toc166327308)

[Figure 25 resolving error 7](#_Toc166327309)

[Figure 26 testing if page runs 7](#_Toc166327310)

[Figure 27 code for tabset 7](#_Toc166327311)

[Figure 28 checking tabset runs 7](#_Toc166327312)

[Figure 29 error: code not running 7](#_Toc166327313)

[Figure 30 installing shinyWidget package 7](#_Toc166327314)

[Figure 31 code for background colour 7](#_Toc166327315)

[Figure 32 checking colour changes 7](#_Toc166327316)

[Figure 33 sidebar layout code 7](#_Toc166327317)

[Figure 34 checking sidebar works 7](#_Toc166327318)

[Figure 35 error for sidebar 7](#_Toc166327319)

[Figure 36 bslib library 7](#_Toc166327320)

[Figure 37 input tabpanel code 7](#_Toc166327321)

[Figure 38 checking input tabs work 7](#_Toc166327322)

[Figure 39 creating inputs for tabs 7](#_Toc166327323)

[Figure 40 checking radio buttons work 7](#_Toc166327324)

[Figure 41 assigning inputs values 7](#_Toc166327325)

[Figure 42 checking server adds values together 7](#_Toc166327326)

[Figure 43 calculating tab footprints 7](#_Toc166327327)

[Figure 44 checking footprint calculated 7](#_Toc166327328)

[Figure 45 bar chart code 7](#_Toc166327329)

[Figure 46 running bar chart code 7](#_Toc166327330)

[Figure 47 checking graph reacts 7](#_Toc166327331)

[Figure 48 carbon footprint calculation 7](#_Toc166327332)

[Figure 49 comparison graph code 7](#_Toc166327333)

[Figure 50 running graphs 7](#_Toc166327334)

[Figure 51 graph reacts test 7](#_Toc166327335)

[Figure 52 input option for graph choice 7](#_Toc166327336)

[Figure 53 code for graph selection 7](#_Toc166327337)

[Figure 54 changing colour of bar 7](#_Toc166327338)

[Figure 55 bar colour test 7](#_Toc166327339)

[Figure 56 bar (2) colour test 7](#_Toc166327340)

[Figure 57 code for user notes 7](#_Toc166327341)

[Figure 58 user notes show test 7](#_Toc166327342)

[Figure 59 reduce footprint code 7](#_Toc166327343)

[Figure 60 reduce footprint test 7](#_Toc166327344)

[Figure 61 source code 7](#_Toc166327345)

[Figure 62 source code test 7](#_Toc166327346)

[Figure 63 user testing results (1) 7](#_Toc166327347)

[Figure 64 user testing results (2) 8](#_Toc166327348)

[Figure 65 User testing results (3) 8](#_Toc166327349)

[Figure 66 User testing results (4) 9](#_Toc166327350)

# Chapter 1 – Introduction

## 1.1 Background Information

Since the Industrial Revolution, humanity has embarked on an unprecedented journey of growth and innovation, continually pushing the boundaries of progress. This relentless pursuit of advancement has led to the rapid evolution of technology, transportation, and agriculture. One of the major technological achievements was the creation of the internet, which revolutionised global communication and connectivity. Devices have been continuously developed and upgraded to meet the evolving needs of society. However, alongside these remarkable achievements comes a significant environmental challenge: humanity has also generated a substantial carbon footprint as a result of these developments.

A carbon footprint is “the extent of GHG emissions resulting from a person's activities” (Franchetti & Apul, 2013). Carbon dioxide (CO₂) is a fundamental greenhouse gas essential for regulating the Earth's climate. Together with methane (CH₄), nitrous oxide (N₂O), and water vapor (H₂O), CO₂ forms a critical layer in the atmosphere known as the greenhouse gas layer. This layer acts like a thermal blanket (Ledley et al., 1999), allowing sunlight to enter the Earth's atmosphere while trapping some of the outgoing infrared radiation. This natural process, known as the greenhouse effect, helps maintain the planet's temperature at a level suitable for life.

Several factors contribute to CO₂ emissions across different sectors. In transportation, vehicles are powered by fossil fuels such as petrol and diesel, which emit CO₂ during combustion. However, advancements like eco modes in cars mitigate emissions by automatically shutting off the engine when idle. Agriculture plays a role in the increased emissions as animals which graze on plants also release greenhouse gases (GHGs). Moreover, modern agricultural practices have facilitated the global transportation of crops to regions where they are unavailable or out of season, further contributing to CO₂ emissions. Additionally, technology is a significant contributor, with emissions occurring throughout a device's life cycle. From production, where devices already contain embedded carbon, to disposal, which releases carbon, technology has a notable environmental footprint. Furthermore, the lack of software support for older device models leads to premature disposal, reducing their potential lifespan and increasing emissions. Additionally, electricity generated from non-renewable sources like fossil fuels adds to CO₂ emissions. While renewable energy sources offer cleaner alternatives, their intermittent nature sometimes necessitates reliance on non-renewable sources to meet demands, underscoring the importance of minimising their use.

There are numerous effective strategies individuals can employ to reduce their CO₂ emissions. Utilising renewable energy sources like wind and solar power whenever feasible is a key approach. This could be achieved by installing solar panels on houses. Although fossil fuels may be used to meet electricity demands at times, reducing electricity usage during these instances can help mitigate CO₂ emissions. Additionally, minimising vehicle transportation for short distances or opting for alternative modes of transport can significantly decrease emissions from transportation. Moreover, software developers can contribute by creating applications compatible with older device models, extending their lifespan, and reducing the need for premature disposal, thereby lowering overall emissions. By implementing these strategies, individuals can play a vital role in reducing CO₂ emissions and mitigating climate change.

The rising cost of living has led to an increase in CO₂ emissions, as people are commuting longer distances for work and opting for cheaper imported goods. Consequently, higher electricity and heating expenses have heightened concerns about carbon footprints. Transitioning to renewable energy sources like solar power offers a solution, reducing electricity bills and environmental impact. Additionally, heightened awareness of the significant contribution of CO₂ emissions to climate change has prompted individuals to take action. By calculating their carbon footprint, people can identify emission hotspots and focus on reducing them. This proactive approach not only promotes environmental stewardship at the individual level but also contributes to broader efforts to mitigate climate change. Through efforts to reduce their carbon footprint, individuals play a vital role in protecting the planet and enhancing quality of life for both current and future generations.

One of the most omnipresent devices in everyday life, is the smartphone (Montag et al., 2015). It epitomises portability, accompanying users everywhere they go, while also serving as a gateway to the internet from virtually any corner of the globe. To ensure widespread access to the carbon calculator, developing a web application accessible from any internet-enabled electronic device is the optimal approach.

## 1.2 Aims

The aim of this project is to develop a software application that will enable users to measure, visualise, and analyse their carbon usage in order to help them lessen their carbon footprint. To help users lessen their carbon impact, the programme will provide customised methods. Another aim is to write a report that describes the process in detail.

## 1.3 Objectives

1. Conduct a literature survey to evaluate the effectiveness and user interfaces of existing carbon tracking apps. Investigate the specifications necessary for a carbon tracker.
2. Identify the most advantageous features of other carbon calculators.
3. Design the user interface with the target audience in mind.
4. Implement and test the user interface.
5. Develop and test the system's backend using R Shiny and R.
6. Perform alpha and beta testing to gather feedback from end users and ensure the system meets requirements.
7. Adjust the system based on test findings.
8. Analyse performance to evaluate the system and generate a report.

## 1.4 Learning Outcomes

Throughout this project, my goal is to demonstrate a good understanding of programming language principles, compiler operations, and computational thinking. By using the tools and methods essential in the software development lifecycle, I will explain how the software industry functions.

Moreover, I aim to show my critical thinking ability by effectively communicating ideas and solutions to both expert and non-expert audiences through oral and written means. Through practical application, I will illustrate how programming languages can be used to create tailored computational solutions for various challenges.

Furthermore, I will analyse, design, and implement algorithms using a variety of languages and techniques suitable for the task at hand, thereby demonstrating a versatile skillset and a strong problem-solving approach.

# Chapter 2 Literature survey

## 2.1 Languages and Frameworks

Creating a website becomes imperative to ensure accessibility for users with internet connectivity across various devices, making it the ideal solution for the carbon calculator. Amongst the multitude of frameworks available, two notable contenders are R Shiny and Visual Studio Code (VS Code). VS Code distinguishes itself with its versatility and user-friendly interface tailored for web development, boasting robust support for multiple languages and frameworks. Its feature-rich environment is meticulously crafted to streamline the development process. VS Code further aids with indispensable functionalities such as syntax highlighting, bracket matching, and robust debugging capabilities, allowing for efficient error resolution (Visual Studio, 2021). Notably, VS Code furnishes an Integrated Development Environment (IDE) catering to a broad spectrum of languages, including Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript. In the field of web development, these languages hold significant importance. They allow to conceptualise, create, and deploy web applications with precision. By skilfully integrating HTML, CSS, and JavaScript, it ensures that websites not only have aesthetic appeal but also function seamlessly, providing users with a smooth experience. Despite its advantages, VS Code presents some potential drawbacks. Performance issues may arise, particularly on less powerful hardware, and mastering all its features may require time due to its steep learning curve. Additionally, VS Code may offer fewer built-in features compared to full-fledged IDEs and configuring it to integrate with external tools can be complex. Occasionally, stability issues or bugs may disrupt workflow. R Shiny presents numerous advantages for website development, particularly for applications reliant on data-driven functionalities. Its seamless integration with R facilitates the effortless incorporation of statistical analysis and data visualisation into web applications, catering to projects that demand extensive data interaction and exploration. Moreover, R Shiny offers a sophisticated level of interactivity and customisation, allowing to craft dynamic and responsive user interfaces tailored to specific project requirements. This is further complemented by its reactive programming model, which simplifies the management of user inputs and updates, thereby augmenting the overall user experience. Nonetheless, despite its strengths, R Shiny is not without its drawbacks. Its heavy reliance on R as the underlying language may present challenges for developers unfamiliar with its syntax and conventions. Additionally, the performance of R Shiny applications may falter when tasked with handling large datasets or executing complex computations, resulting in sluggish response times and diminished user satisfaction. Furthermore, while R Shiny offers considerable flexibility in terms of design and functionality, it may fall short in providing the same degree of control and customisation as other established web development frameworks. In summary, users tend to evoke emotions more readily when confronted with visual depictions of their carbon emissions compared to the average. These visuals offer a clear comparison, indicating whether they emit more or less than the average, thereby eliciting a stronger emotional response. Consequently, R Shiny emerges as the optimal framework due to its capacity to generate graphs as outputs (Beeley, 2018).

## 2.2 Methodology

The two primary software development methodologies commonly used today are the Waterfall and Agile methodologies. The Waterfall methodology is a linear and sequential approach to software development, similar to steps cascading down like a waterfall (Gurung et al., 2020). The method involves dividing the project into distinct phases such as gathering requirements, design, implementation, testing, deployment, and maintenance. Requirements are meticulously gathered and documented upfront, serving as the foundation for subsequent phases. The design is then crafted based on these established requirements. Once the design is finalised, implementation ensues, followed by rigorous testing to identify and rectify any defects. This methodology does not allow moving to the next phase until the previous phase has been completed (Pargaonkar, 2023). One of the defining characteristics of the Waterfall method is its adherence to strict deadlines for each phase, ensuring progress toward project completion. However, the method's rigidity also means that once a phase is completed, it cannot be revisited, potentially posing challenges if requirements evolve or errors are discovered later in the project, changes are difficult to accommodate once a phase is complete, making it less flexible and potentially leading to delays if requirements change later in the project. (Pargaonkar, 2023). It emphasises thorough documentation and is best suited for projects with stable requirements. It is commonly used in industries with strict regulatory requirements (Ahmad, 2024). The Agile methodology is an iterative and flexible approach to software development, characterised by dividing projects into small increments called ‘sprints’, which typically last 1-4 weeks to complete. Agile methodology takes the view that production teams should start with simple and predictable approximations to the final requirements and then continue to increment the requirements throughout the software development (Kumar & Bhatia, 2012). This allows for the requirements and the solutions to evolve based on continuous feedback from stakeholders. Emphasis is placed on delivering working software frequently, prioritising features based on customer value and feedback. Adaptability to changing requirements and environments is key, making Agile suitable for projects with evolving or uncertain requirements where rapid delivery and flexibility are essential. However, the perpetual cycle of requirement gathering, and refinement can present challenges, potentially leading to overwhelming workloads and jeopardising project timelines if not managed effectively. Consequently, agile teams must strike a delicate balance between responsiveness to change and maintaining a sustainable pace of work. Nonetheless, the benefits of agile, including enhanced collaboration, increased flexibility, outweigh its challenges. While Waterfall presents a structured and systematic approach well-suited to projects with stable requirements, Agile offers flexibility and adaptability, rendering it ideal for projects with evolving or uncertain requirements. Agile would be the most suitable methodology for the carbon calculator, as research will be ongoing throughout the project, potentially leading to changes in requirements.

## 2.2 Layout and Useability

Navigating through applications can pose a significant challenge, particularly for individuals with limited experience in technology, such as novice users and the elderly. This difficulty in navigation frequently results in frustration and may prompt users to abandon the application altogether. To counteract this issue and uphold universal accessibility, the implementation of effective strategies becomes imperative. One such strategic method involves embracing the principles outlined in the Jakob Nielsen Layout methodology. At the core of this methodology lies the integration of a comprehensive help guide directly into the application interface. This feature encourages users by providing detailed explanations and step-by-step instructions on utilising the carbon calculator, thereby fostering confidence in their navigational abilities. Moreover, simplifying the layout, as proposed by (Nielsen, 2005), and refining design elements can significantly enhance the user experience. Intuitive navigation menus, coupled with clear labelling and strategically positioned prompts, serve to guide users seamlessly through the application, reducing confusion and frustration. By prioritising user-friendliness and inclusivity in design, applications can better cater to their diverse user base, thereby fostering long-term engagement and satisfaction.

Shifting the focus to the digital canvas of the carbon calculator website, the importance of enhancing its visual appeal becomes evident. The meticulous selection of colours assumes critical significance, not only for their aesthetic appeal but also for their ability to evoke emotional responses from users. As highlighted by (Mohammad, 2013), effective colour choices can significantly enhance semantic coherence and elicit desired emotional responses. While green, with its intrinsic association with the natural world and environmental consciousness, emerges as an appropriate choice given the thematic relevance of the carbon calculator, it is essential to acknowledge its susceptibility to colour-blindness. In this light, blue presents itself as a potential alternative, being more visible to individuals with colour vision deficiencies (Rigden, 1999). Furthermore, the calming aura exuded by blue, reminiscent of tranquil bodies of water, seamlessly aligns with the overall environmental themes. Thus, by carefully employing colour theory within the framework of the Jakob Nielsen layout methodology, the website can optimally cater to diverse user needs while fostering heightened engagement and accessibility. This comprehensive approach not only enhances user experience but also demonstrates the commitment to inclusivity and accessibility in digital design, ensuring that all users can seamlessly navigate and engage with the carbon calculator application. Expanding further, it is evident that catering to the needs of diverse user demographics is crucial for the success of any digital application. Novice users, often overwhelmed by technology complexities, require intuitive interfaces and clear guidance to navigate effectively. Similarly, age-related cognitive loss may provide difficulties for senior users, requiring design features that cater to their requirements. Embracing a user-centric approach enables to create applications accessible and welcoming to users of all ages and backgrounds. By integrating these considerations into the design process, ensures that applications are not only functional but also inclusive, providing a seamless experience for all users.

## 2.3 MVC (Model View Controller)

In the context of developing a carbon calculator, the Model-View-Controller (MVC) architectural pattern serves as a structured approach to software design, recommended for its ability to facilitate the creation of scalable and maintainable applications. Following MVC principles, the model component contains the core logic and data manipulation functionalities relevant to carbon emissions calculation. In order to guarantee precision and dependability in emissions computation, this section concentrates on data representation. The view component applies to the presentation layer responsible for rendering user interface elements, including visual representations of carbon footprints and interactive features. By adhering to the MVC model, outlines user interface design and functionality, promoting code modularity and ease of maintenance. The Model component manages data related to carbon emissions with precision and efficiency. It encompasses functionalities such as representing emission types (e.g., transportation) and calculating emissions based on user input (e.g., mileage). Additionally, it enables real-time data processing, allowing users to input various parameters and receive immediate feedback on their carbon footprint. This dynamic functionality enhances user engagement and promotes awareness of environmental impact, encouraging users to make informed decisions to reduce their carbon footprint effectively. The View component serves as the interface through which users interact with the carbon calculator, presenting data and information in a clear and visually appealing manner. It includes elements like input fields for user data entry (e.g., mileage), displays showing calculated emissions, and graphical representations like charts or graphs illustrating emission breakdowns by category. Furthermore, it employs responsive design principles, ensuring compatibility across different devices and screen sizes. This adaptability enhances accessibility and usability, enabling users to access the carbon calculator smoothly from various platforms. The Controller module acts as the mediator between the Model and View components, coordinating user interactions and application flow (Deacon, 2000). It receives user input from the View, processes it, communicates with the Model to perform computations, and updates the View with the calculated emissions. Adopting the MVC pattern ensures a well-structured and maintainable application architecture, which allows for efficient management and scalability of the codebase over time. By separating concerns and promoting code modularity, code may be readily extended, or specific components of the application be modified, without impacting other parts of the system. Moreover, the clear separation of responsibilities between the Model, View, and Controller components reduces the possibility of errors. In context, when the user changes one of the input fields on the carbon calculator this triggers an event captured by the view component (Bucanek, 2009), which then sends the updated data to the controller. The controller, acting as the mediator, receives this input and processes it accordingly. Next, the controller communicates with the model to update the relevant data structures and trigger any necessary calculations. The model component processes the new input data, recalculates the carbon emissions based on the changes, and updates its internal state accordingly. Once the calculations are complete, the model notifies the controller of the updated carbon emissions data. Finally, the controller instructs the view component to refresh the user interface, displaying the newly calculated emissions data to the user. Through this coordinated process, the MVC architecture ensures that any changes made by the user are accurately reflected in the carbon calculator's calculations and user interface.

## 2.4 Carbon Footprint Formula

After performing extensive research on carbon footprint calculations, it became apparent that the formulas available for estimating carbon footprint emissions are limited due to copyright restrictions. Nevertheless, the research revealed the presence of two different calculations. These formulas succeed at producing thorough results by analysing user inputs. They provide information about an individual's carbon footprint, taking into account aspects such as energy use, travel patterns, and behaviours. One of the potential calculations was:

**Multiply your monthly electric bill by 105**

**Multiply your monthly gas bill by 105**

**Multiply your monthly oil bill by 113**

**Multiply your total yearly mileage on your car by .79**

**Multiply the number of flights you’ve taken in the past year (4 hours or less) by 1,100**

**Multiply the number of flights you’ve taken in the past year (4 hours or more) by 4,400**

**Add 184 if you do NOT recycle newspaper**

**Add 166 if you do NOT recycle aluminium and tin**

**Add 1-8 together for your total carbon footprint**

(Just Energy, 2013)

The calculation formula estimates an individual's carbon footprint by taking into consideration a variety of variables such as energy use, mode of transportation, frequency of air travel, and recycling behaviours. It involves multiplying specified inputs, such as monthly utility bills and vehicle mileage, with predefined values that indicate the related carbon emissions. Furthermore, adjustments are made based on recycling behaviours. The total results from each stage are combined to calculate the entire carbon footprint estimate. The second calculation was:

**X (CP)**

**X** is the score generated by the question, **C** is the average footprint of your country, and **P** is the weighting percentage.

Home energy (19%)

Shopping (7%)

Diet (20%)

Commute (19%)

Flights (non-weighted)

(Ben, 2022)

This calculation estimates an individual's carbon footprint by taking into account various aspects of their lifestyle. Each aspect, such as home energy usage, shopping habits, diet, and travel, adds to the overall score. The score determined by each aspect is multiplied by the average carbon footprint of the individual's country and the weighting percentage unique to each aspect. The values obtained is combined to determine the total carbon footprint estimate. Flights are also included in the computation, but they are not given a precise percentage weight.

When the two calculations are compared in terms of complexity, accuracy, flexibility, convenience of use, and variable sensitivity, they show considerable differences. Formula 1 performs complex computations, inspecting several elements such as energy bills, transport modes, and recycling habits to provide a thorough breakdown. Formula 2, on the other hand, simplifies the process by producing a single score based on weighted averages from various areas. While Formula 1's thorough technique may improve accuracy through exact data measurement, it demands more effort and time from users to obtain and enter precise data. Formula 2, on the other hand, is based on national average data, which simplifies things but may sacrifice particular details. Formula 1's flexibility allows for customisation depending on individual consumption habits, whereas Formula 2 sticks to standardised national averages, which limits adaptability. Formula 1 accuracy is dependent on accurate data inputs, but Formula 2 relevance is dependent on national average accuracy and weighting percentages that correspond to individual lifestyles.

The carbon calculation process should be simple and not require considerable data collection from users. In this circumstance, Formula 2 emerges as the ideal alternative due to its simplicity and the ability to provide consumers with a full breakdown of their individual percentages across categories. However, a minor concern arises regarding the total percentage weight, which currently stands at 65%. As a result, additional categories may be required to provide a more comprehensive estimate of the individual's overall carbon footprint.

## 2.5 Data Collection

The inclusion of the average carbon emission per capita in the UK in the carbon footprint calculator is critical for allowing a thorough comparison of individual user results to the established average. To maintain the required level of precision in the data collected, different sources were carefully selected to assure accuracy and reliability in the information displayed to the user.

After reviewing numerous sources of data, it is apparent that carbon emissions per capita in the United Kingdom have been relatively consistent in recent years. (Tiseo 2023) and (McKay 2024) estimate 4.7 metric tonnes of carbon emissions per person in 2022. Additionally, (CO2 emissions (metric tonnes per capita) - United Kingdom 2021) documented 4.6 metric tonnes per capita in 2020. This suggests the results are reliable. These may be utilised to build a strong foundation for the carbon footprint calculator, since users will be able to compare their personal carbon footprint to the UK average. The standard average per person will be set at 4.7 metric tonnes. Using this representative value, users have the ability to compare their own carbon footprint values to the known UK average. This approach determines that the calculator's output are reliable. In addition to this, it also increases the usefulness in raising awareness and encouraging users to adopt more sustainable behaviour.

The chosen carbon footprint calculation does not add up to 100%; so, more data must be collected to account for the remaining 35%. Home-based activities are an important element of everyday living that contribute considerably to the carbon footprint (Wiedenhofer et al., 2018). Adjusting the weighted proportion of the elements listed in the calculation and adding a new category: water, ensures that the total reaches 100%. Water distribution has a significant impact on an individual's carbon footprint; energy is also required to sanitise the water and distribute it around through pipes (Boulos & Bros, 2010). Given the frequency with which water is used in daily activities such as washing, flushing, and bathing, the overall impact on carbon emissions becomes apparent.

For the questions in each section to sum up to a number between 0 and 2, the highest corresponding choice values must sum to a total of 2. Different choices chosen will outcome to different values, which results in a different carbon footprint result.

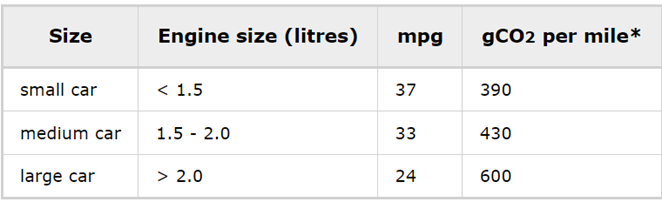


Figure 1 Table of car CO₂ emissions (*Version Information* 2008)

The table above was used to determine the values for the car transportation questions, looking at the table having a large car and driving it daily should result in the highest value of 2.

For flights, the data of 270 grams of CO₂e per kilometre (g CO₂e per km) was collected from (Tiseo 2023), representing the impact of a domestic flight per person. This result was used to determine the value of the flights questions.

The carbon footprint of shopping is produced by the materials required to make products along with the electricity needed to manufacture, the materials used to package, and the transportation required to ship. (Panzone et al., 2021). Using this information, shopping daily should be a high value as the carbon footprints of most products are large.

Being a carnivore produces a lot of carbon emissions as 4.87 kg CO2 equivalent is product per kg of pig carcass - (Philippe & Nicks, 2015) so therefore the corresponding value to having a mainly meat diet should be the highest number. Having a plant diet also produces co2 as plants that are out of season are transported from other countries which produces carbon.

Non-renewable electricity is produced from burning fossil fuels like coal and oil, this in turns produces carbon dioxide. Renewable electricity is produced from using renewable sources like solar power and wind power which produce no co2.

## 2.6 Current System

Current system 1 <https://www.carbonindependent.org/index.html>

A screenshot of a computer

Description automatically generated

Figure 2 Input to Current System 1 <https://www.carbonindependent.org/index.html>

A screenshot of a graph

Description automatically generated

Figure 3 results from current system 1 <https://www.carbonindependent.org/index.html>

A screenshot of a computer

Description automatically generated

Figure 4 results from current system 1 <https://www.carbonindependent.org/index.html>

The carbon calculator includes information for each section of the calculation and specifies how much co2 emission is produced from it.

The benefit of the calculator is that the questions are simple and understandable and there are helpful notes alongside the questions to help guide users to the appropriate choice. The carbon calculator also has a read more within the notes which allows the users to see the data that was used to calculate their results. Even though the questions are simple to understand, the answer choices are not specific, for example one of the choices were “small house/ flat (12,000 kWh)” as shown In figure 2, this information is a bit confusing as some people will not know how much kWh their house produces to know if it is considered a small house. The results show a graph of the user's carbon footprint compared to other countries, however, not all countries are included so the user's country may not be included as shown in figure 3 A bar chart is also produced showing the carbon emissions based on the answers given by the user for the household questions as shown In figure 4.

The layout of the carbon calculator looks overcrowded with the helpful text on the side, which might overwhelm users and result in user disengagement.

The best parts of this calculator are that the questions are split into 2 sections of house and personal so the user knows what type of questions may be asked. There is basic multiple-choice questions so the use won’t feel overwhelmed with having to know information, but it also allows users to input figures to have a more tailored result rather than an estimate.

Current system 2 [CoolClimate Calculator (berkeley.edu)](https://coolclimate.berkeley.edu/calculator)

A screenshot of a computer

Description automatically generated

Figure 5 input of current system 2 [CoolClimate Calculator (berkeley.edu)](https://coolclimate.berkeley.edu/calculator)

The questions are simple to understand but some questions like the gross annual income of the household, not everyone will know. The graph on the side has lots of useful information to help users to better understand their carbon footprints. The block underneath the graph shows the percentage better or worse the user is against the average footprint in the US and changes colour from red and green to help the user understand better. The graph bars are split into the sections they represent and further split into the subsections and the length of them show how much carbon emissions they produce. If the user hovers over the graph, the exact co2 emission produced of each section is also displayed.

A screenshot of a graph

Description automatically generated

Figure 6 input of current system 2 [CoolClimate Calculator (berkeley.edu)](https://coolclimate.berkeley.edu/calculator)

A screenshot of a website

Description automatically generated

Figure 7 input of current system 2 [CoolClimate Calculator (berkeley.edu)](https://coolclimate.berkeley.edu/calculator)

The calculator allows for the user to choose if they would like to have the simple inputs which are vague and are just input slides. The advanced features require inputs for each section which would then lead to a more tailored result.

A screenshot of a screen

Description automatically generated

Figure 8 user pledge for current system 2 [CoolClimate Calculator (berkeley.edu)](https://coolclimate.berkeley.edu/calculator)

 The final page of the calculator, guides users to helpful information to help them to reduce their co2 emissions and displays the cost of the action, money saved by doing it and co2 reduced by completing.

A screenshot of a computer screen

Description automatically generated

Figure 9 leaderboard for current system 2 [CoolClimate Calculator (berkeley.edu)](https://coolclimate.berkeley.edu/calculator)

The carbon calculator engages users by having a leaderboard to have the users compete with improving their carbon footprint.  This helps to boost user engagement and helps users to see their progress of reducing their carbon footprint.

The best parts of this carbon calculator are that it has a simple layout, the questions are split into sections about travel, food, etc. It shows the user a reactive graph that changes based on their inputs. Some disadvantages are that it doesn’t show a comparison against the average so the user can’t visually see the difference.

Current system 3 <https://www.carbonfootprint.com/calculator.aspx>

A screenshot of a calculator

Description automatically generated

Figure 10 input for current system 3 <https://www.carbonfootprint.com/calculator.aspx>

The carbon calculator allows for the users to choose the period of time they want to calculate their carbon footprint for. This is useful for users who struggle to remember information about the full year and tailors the carbon calculator to best fit them. The calculator also takes in the country the user wants to see which helps them compare their result to their countries average.

A screenshot of a computer

Description automatically generated

Figure 11 input for current system 3 <https://www.carbonfootprint.com/calculator.aspx>

The questions asked, require specific answers and do not have estimate answers for users who do not know the information. This would lead to user becoming overwhelmed by not knowing the information.

A screenshot of a cell phone

Description automatically generated

Figure 12 results from current system 3 <https://www.carbonfootprint.com/calculator.aspx>

The result compares the user's footprint to the country average and the world average and show the exact values at the bottom. The result doesn’t have a colour scheme so it might be difficult for users to understand as the footprint size may not be enough for them to understand their footprint is low/high.

### Current system 4

<https://www.nature.org/en-us/get-involved/how-to-help/carbon-footprint-calculator/>

A screenshot of a web page

Description automatically generated

Figure 13 input for current system 4

<https://www.nature.org/en-us/get-involved/how-to-help/carbon-footprint-calculator/>

The calculator requires specific information, which not everyone knows. This will lead to a more precise carbon footprint for users but for others it will cause disengagement as they do not know the answers so they will be deterred from using it.

A screenshot of a graph

Description automatically generated

Figure 14 result for current system 4

<https://www.nature.org/en-us/get-involved/how-to-help/carbon-footprint-calculator/>

Allows users the option to see their results as a bar chart or a pie chart. This result page gives lots of information about the user's carbon footprint. It shows the user graphical representation of their footprint across the sections, shows their total carbon footprint and a percentage against the average to show if the user is above or better than the average. The colour scheme of the graphs does not help user to understand if they are above the average, this is only depicted by the percentage in the corner showing if they are above or below the average.

A screenshot of a computer screen

Description automatically generated

Figure 15 user pledge for current system 4

<https://www.nature.org/en-us/get-involved/how-to-help/carbon-footprint-calculator/>

The calculator has a section for the user to pledge to improve their carbon footprint which shows the statistics of the carbon emission and money saved and how much it costs to do. This allows users to think about the future and make plans to improve their carbon footprint.

# Chapter 3 – Design

## 3.1 Task Planning

Before initiating the project's implementation phase, thorough preparation is required to ensure that all tasks are organised and completed on time. This planned approach not only makes the implementation process run more smoothly, but it also allows for adequate time to be allocated to each activity, increasing overall efficiency and lowering the probability of setbacks.

To do this, a thorough analysis to identify each task required for the development of the carbon calculator project needs to be conducted. The tasks were then carefully organised and prioritised according to their importance and dependencies. Tasks that are critical to the calculator's basic functioning were prioritised since they form the backbone of the product and must be done first to ensure its functionality. This consists of tasks like reading and storing inputs, calculating the footprint, and producing a graph.

Following the prioritisation of fundamental tasks, attention should be shifted to additional tasks including webpage design and visualisation. While these tasks are critical to the product's overall user experience, they are not required for the basic carbon calculator operation. As a result, they were planned to be addressed following the completion of essential tasks, providing for a more focused and systematic approach to project development.

This method of carefully planning and organising work ensures an effective implementation process, ultimately leading to the successful completion of the carbon calculator project.

A screenshot of a screen

Description automatically generated

Figure 16 task plan

## 3.2 Language and Framework Choices

As stated in the extensive literature survey, the language chosen for this project is R, which is complemented by the use of R Shiny as the framework. The chosen decision aligns with the goal stated at the beginning of the report, particularly with the inclusion of graphical visualisations. This option allows to successfully improve the presentation of data, promoting greater understanding for the user of their carbon footprint. R Shiny also allows for the graphs to be reactive so they will change in real time with the users inputs which will further help to engage users as they can experiment with inputs and see the resulting graph.

## 3.3 Requirements and Specifications

The proposed carbon calculator web application has multiple functional features to help users calculate their carbon footprint. The application will take in user's inputs and based off their inputs; it will produce an output to show the user their carbon footprint.

The graphs will react based on the users input and illustrate this by changing the carbon footprint shown. It will also show and display 2 different graphs, one to show the weighted value of each section and another to show the comparison against the average footprint, this will help users to understand and focus on certain parts of their footprint that need improvement. The application should also further help users by tailoring their ways to reduce based on their input choices.

Functional requirements:

* Input: Provide sections so that the questions are split and do not overwhelm the user.
* Calculation server: Implement server to process user inputs and calculate the carbon footprint and section footprints.
* Graphs: Present the calculated carbon footprint to users in a clear and understandable graphical representation.
* Carbon Reduction Tips: Offer suggestions to users on how they can reduce their carbon footprint based on their input data.

Non-functional requirements:

* Usability: Ensure the calculator is user-friendly and intuitive, with clear help guide.
* Accuracy: The calculator must produce accurate results based on input.
* Accessibility: carbon calculator must be accessible to everyone.
* Compatibility: The system should be compatible with various web browsers.

## 3.4 Structural Design

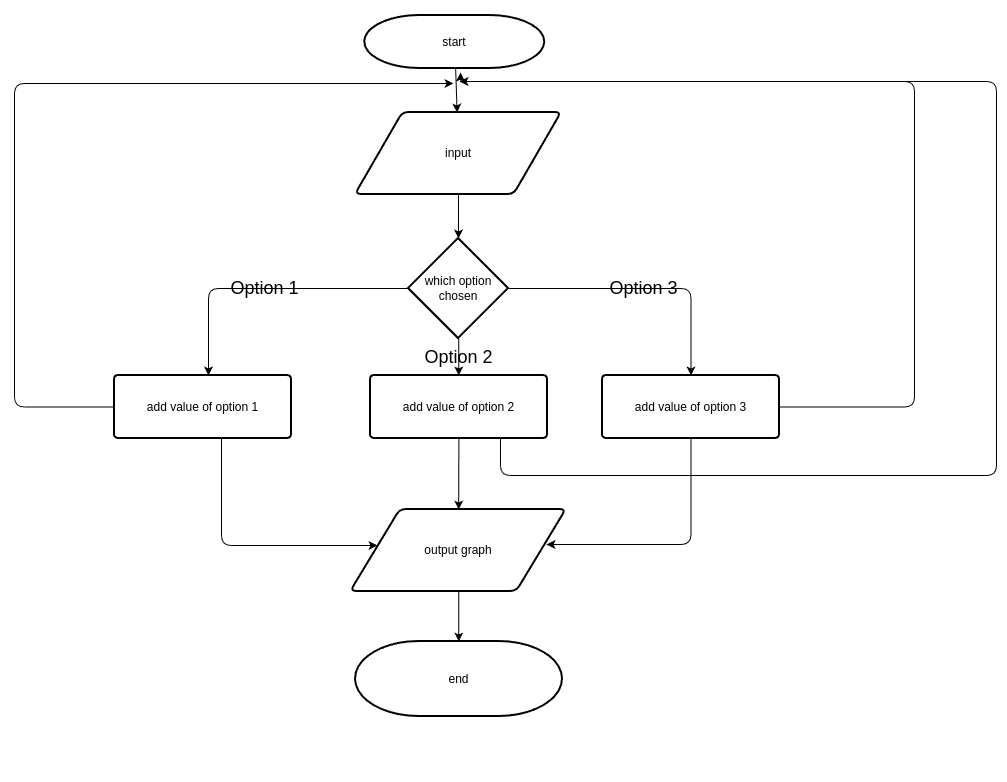


Figure 17 input to output diagram Created using https://online.visual-paradigm.com/

As shown by the diagram of the Input to Output process above,

The process starts with the input, which is the user choosing one of the options for the question. The system will store that answer and check which choice is chosen and give the corresponding value assigned to that choice. All the answers to the questions in a section will be totalled to find the final sum, which should range from a value of 0 to 2. The system then either goes back to the next input to add or the output graph is shown based on the resulting total from the input choices.

Calculation of the carbon calculator for the graphs:

The system totals the values of the selected choices for each section, this number will change if the input choices are changed again. The total value of each section is then multiplied by its assigned weighted percentage to result in the sections’ footprint. All of the footprints from each section will be outputted on a graph. The user will 2 options for graph choice that they can choose from. The other graph choice will be their carbon footprint compared to the average. The carbon footprint is calculated by totalling the section footprints.

## 3.4.1 Functionality Detailed Design

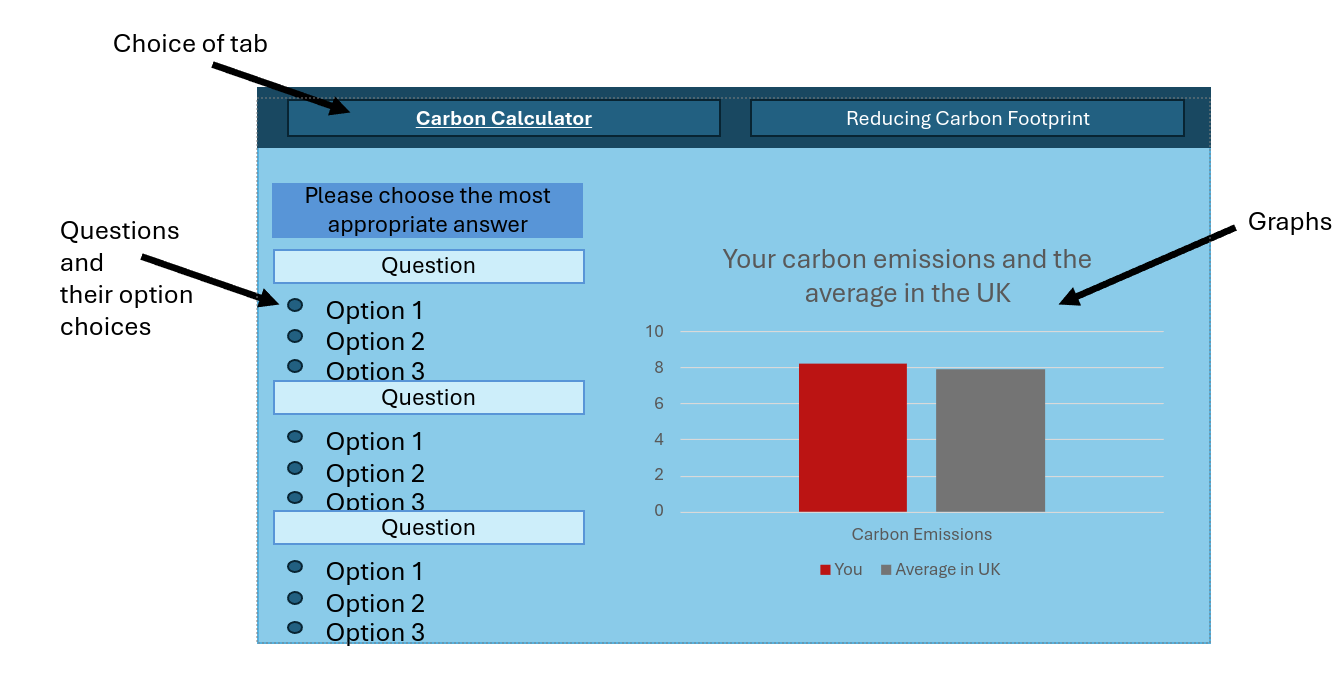


Figure 18 calculator page design

The design above is of the main carbon calculator page. This page will hold the multichoice questions that the user needs to answer. The graph will be produced on the right-hand side of the inputs so the user can compare their inputs to the graph outputted. Having the inputs on the left also helps to make the page more visually appealing to the user as the page will look organised. This design also helps users to navigate as everything for the carbon calculator is on one page so users who are not as confident will not be overwhelmed will having to use multiple tabs.

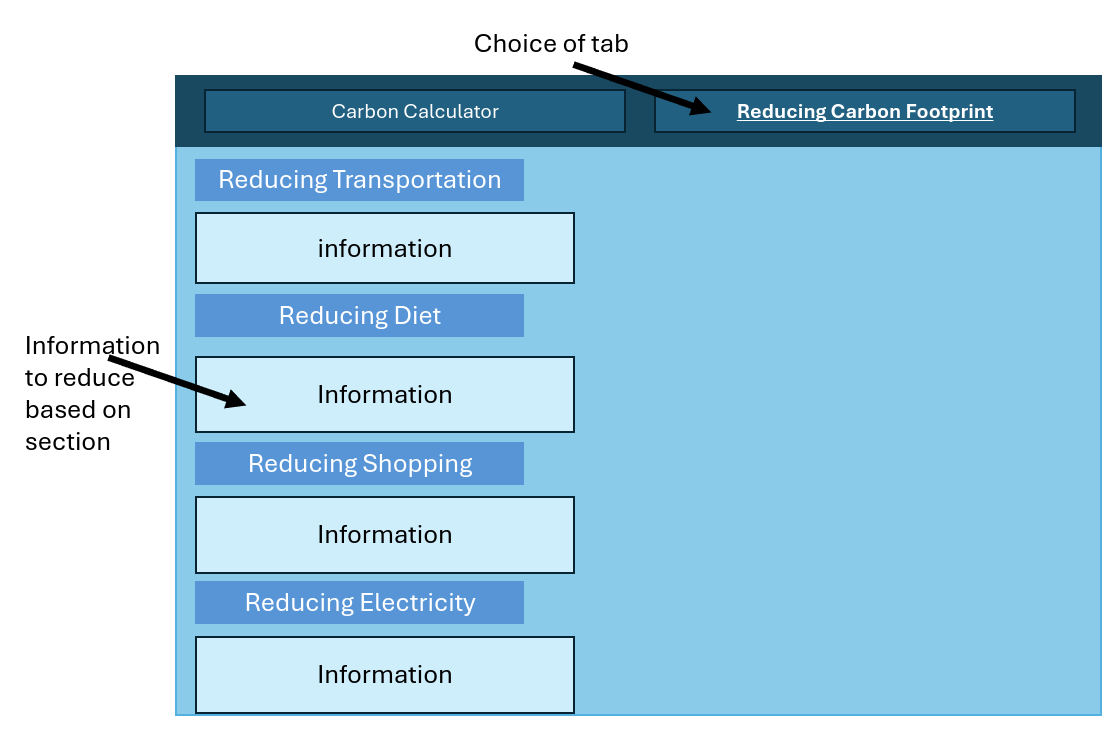


Figure 19 reducing carbon page design

The design above is for the 2nd page which is the reducing carbon footprint page. This page will hold information for the user to help them reduce each section of the carbon footprint. The reducing carbon footprint could have been put with the carbon calculator on a single page, however, this would make the page more cluttered and may make the user feel overwhelmed. By using another page for the reducing carbon footprint, users are able to engage and focus on the information more.

## 3.4.2 Usability Detailed Design

To help the users that are unsure of how to use the page, there will be a help guide which consists of a few sentences placed around the page to help them to navigate around. Due to the Simple design, this may not be necessary but will still be included to ensure that users feel confident in using the calculator.

To further help with usability, the colour scheme has been changed so that it is better suited towards users who are colourblind. They are more able to see blue than green, so the blue background and blue selection colours have been chosen in replacement. This allows the calculator to be inclusive and improves the accessibility of a wider range of users.

## 3.5 Design Evaluation (500)

The design demonstrates simplicity, which will be advantageous for novice users and elderly individuals who may not be confident with the use of the website. The neat layout minimises the probability of users becoming overwhelmed with information. Furthermore, the use of the help guide throughout the website increases navigational ease for users. This will help with user engagement as the novice/elderly will not be deterred from using the calculator.

The border between inputs and the graph, allows users the advantage of simultaneously seeing both at the same time. This allows users to compare their input choices to the output graph. In addition to this, users have the ability to experiment with the carbon calculator and see which choices would get them the lowest carbon footprint and in turn may influence their behaviour to reduce carbon emissions.

The inclusion of different graph types allows the users the option to see the diverse visualisations of their footprint, whether that is the footprint for each section, or their overall carbon footprint compared to the UK average. This will help users to see which areas they should focus on to reduce their carbon footprint.

There is also a reducing carbon footprint page to further guide users to improving their carbon emissions. This will allow users to see the different ways they can become more carbon conscious and overall influence a change in their behaviour.

# Chapter 4 - Implementation

Steps in Implementation - Starting with Basic Operations Required for the Calculator to Work:

**1)** The procedure began with the R Shiny library package being installed in the RStudio Desktop environment. To enable the following development steps, this required sifting through the R environment's package repository and installing the necessary package.

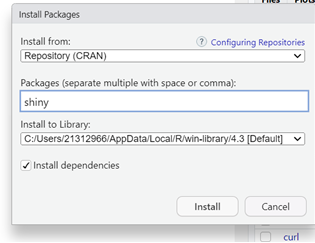


Figure 20 installing R shiny



Figure 21 r shiny installing

**2)** To make sure the Shiny library was available and prepared for use in further development tasks, within the R script, R Shiny was called after the installation was successful.

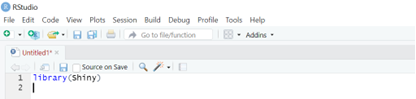


Figure 22 using r shiny library

**3)** Then followed the development of a fluid page and the server that contained the website. This initial assessment was carried out to confirm the page was functioning and the library's proper installation was an essential step in this phase.

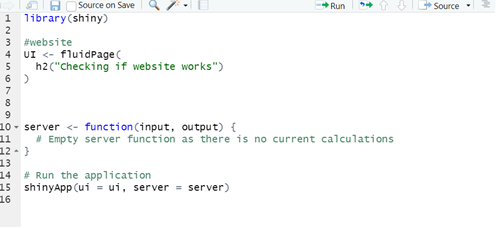


Figure 23 testing basic code

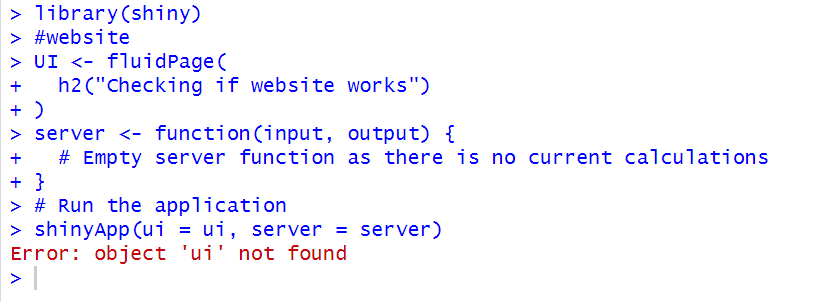


Figure 24 error found

A thorough code study was initiated by an error message that was found and indicated that there was no 'ui'. To fix the code with correct syntax and functionality, the reference to 'UI' was changed to 'ui', therefore resolving the error.

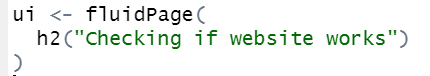


Figure 25 resolving error

The code was executed, and the webpage was displayed.

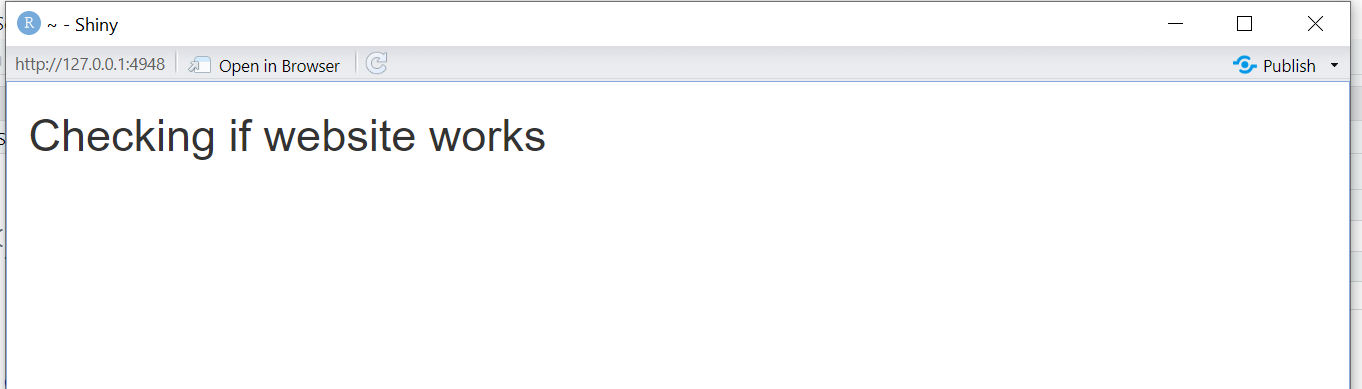


Figure 26 testing if page runs

**4)** Moving on to the tab panel development stage, I carefully structured the interface using the complete Shiny instructions, particularly the Tabsets tutorial supplied by (*Tabsets* 2014).

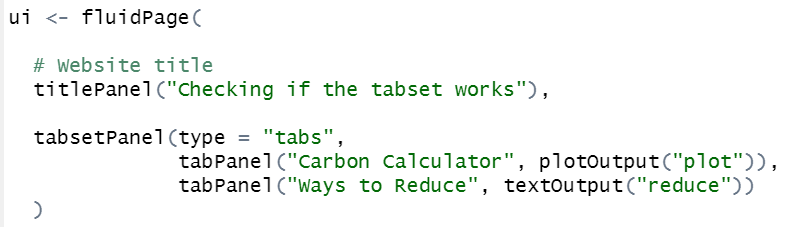


Figure 27 code for tabset

Currently, just two tabs are available. The first tab contains the carbon calculator, which produces output in the form of graphs that can be shown visually. The second tab, labelled "Ways to Reduce," displays textual output containing information on how to reduce one's carbon footprint.

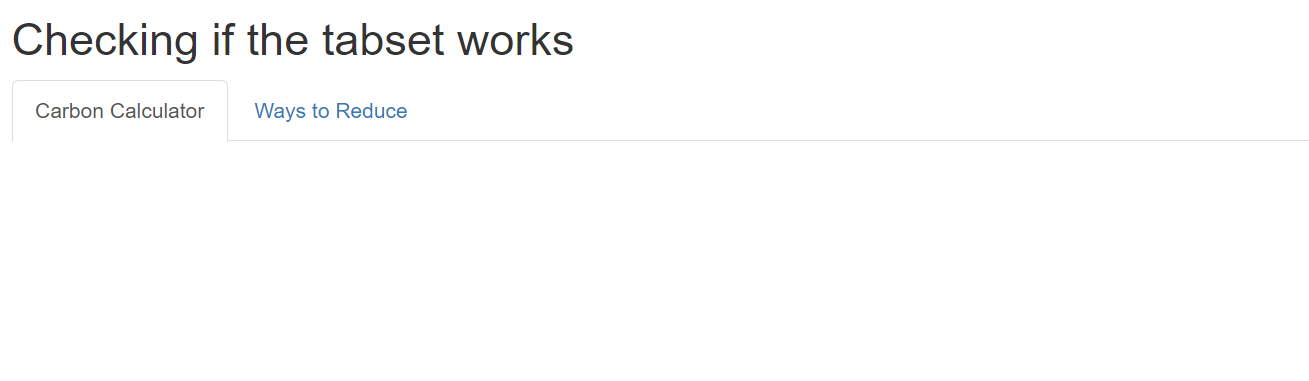


Figure 28 checking tabset runs

**5)** I came across an issue in which R Shiny failed to properly execute the code, instead running only the comments and skipping code blocks. Despite lengthy study to determine the core source of this anomaly, my efforts were unsuccessful, since forums addressing similar concerns went unanswered.

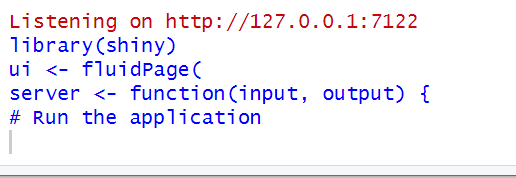


Figure 29 error: code not running

I determined that the problem was caused by the website running concurrently in the background, which I failed to quit before re-executing the code.

**6)** I proceeded along with the task of changing the background colour of the webpage. To accomplish this, I started by installing the ShinyWidget package. This package allows for considerable customisation possibilities, such as specifying and adjusting the website's colour palette.

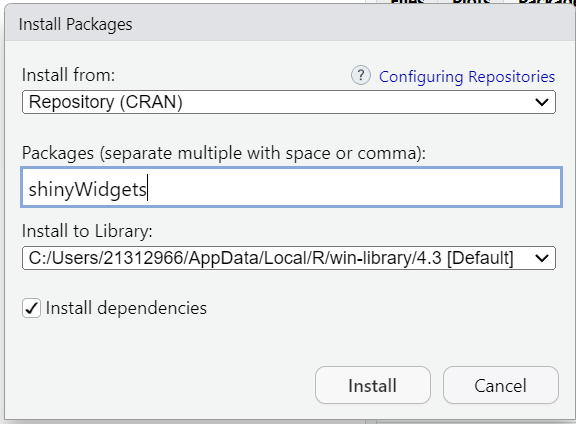


Figure 30 installing shinyWidget package

I installed the previous package and included its library into the code. Following that, I set the backdrop colour to reflect the designated hue of Columbia Blue, as per the design elements.

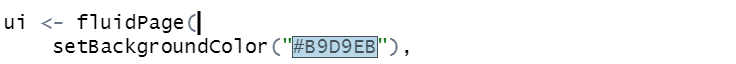


Figure 31 code for background colour

Subsequently, I ran the code to ensure that the website's backdrop adopts the designated Columbia Blue hue as intended.

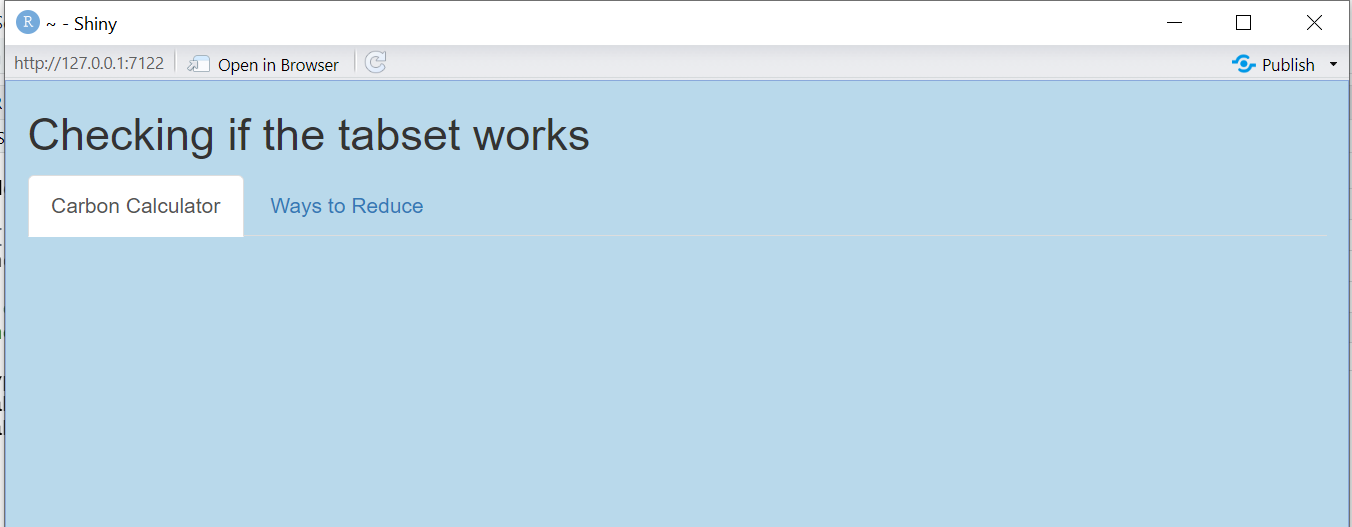


Figure 32 checking colour changes

**7)** I started the design process by working on the Carbon Calculator page. Initially, I divided the page into two separate sections: one for input parameters and another for showing graphical output.

I created the sidebar layout using resources from the R Shiny help website. (*Build a user interface* 2014).



Figure 33 sidebar layout code

I executed the code to check it works correctly.

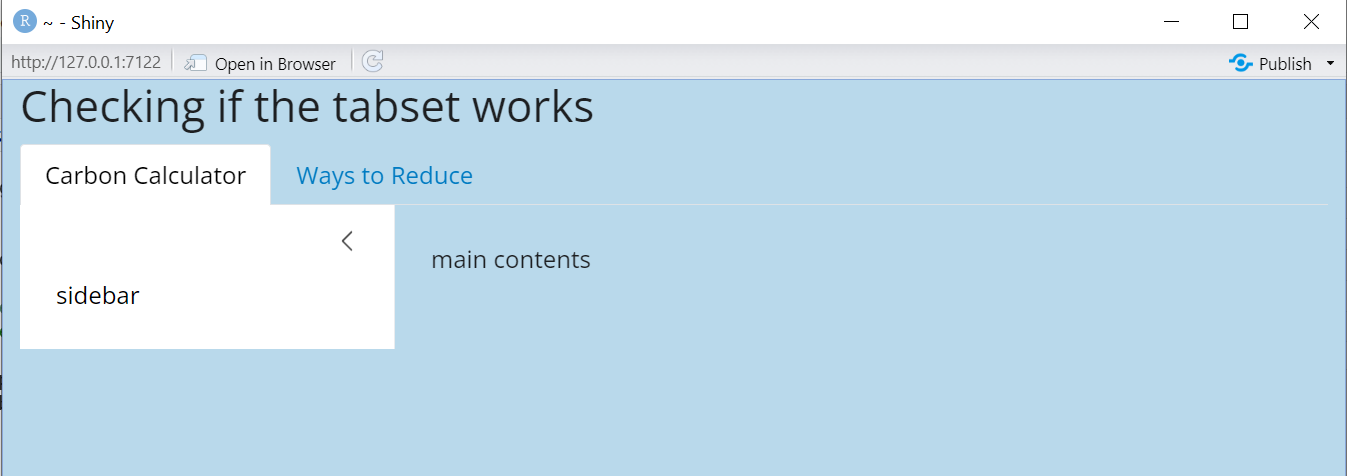


Figure 34 checking sidebar works

**8)** I obtained an error while switching from the University RStudio environment to my personal RStudio environment.

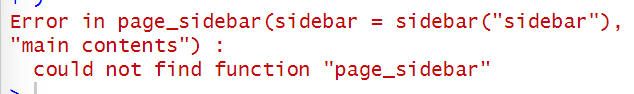


Figure 35 error for sidebar

I was unsure about the origin of this error, so I checked the website from which I got the instructions for developing the side panel layout.



Figure 36 bslib library

I forgot to include the bslib library, rendering RStudio unable to execute the code because it relies on utilities given by this library.

**9)** I began with the task of separating different areas of inputs by adding tab panels to the side panel.

A computer screen shot of text

Description automatically generated

Figure 37 input tabpanel code

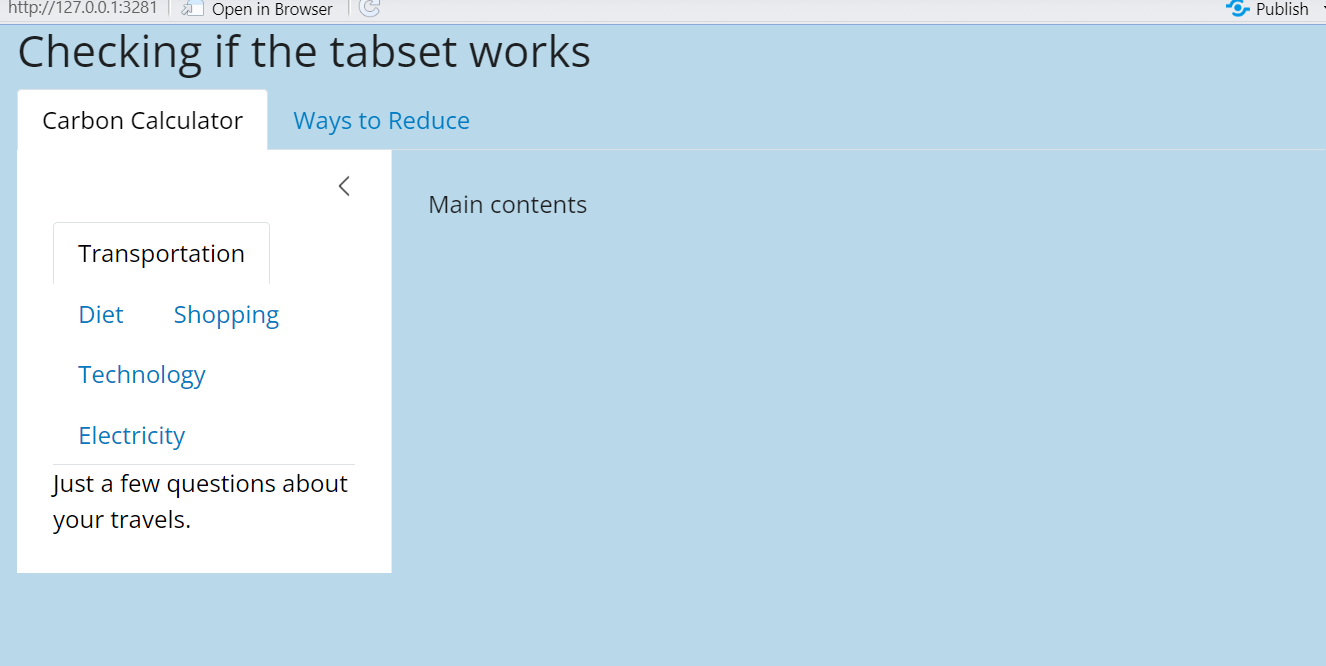


Figure 38 checking input tabs work

**10)** I incorporated the relevant questions along with the related radio button responses to the appropriate tabs. For this, I utilised the vertical radio button code provided by (*Shinywidgets overview* 2019).

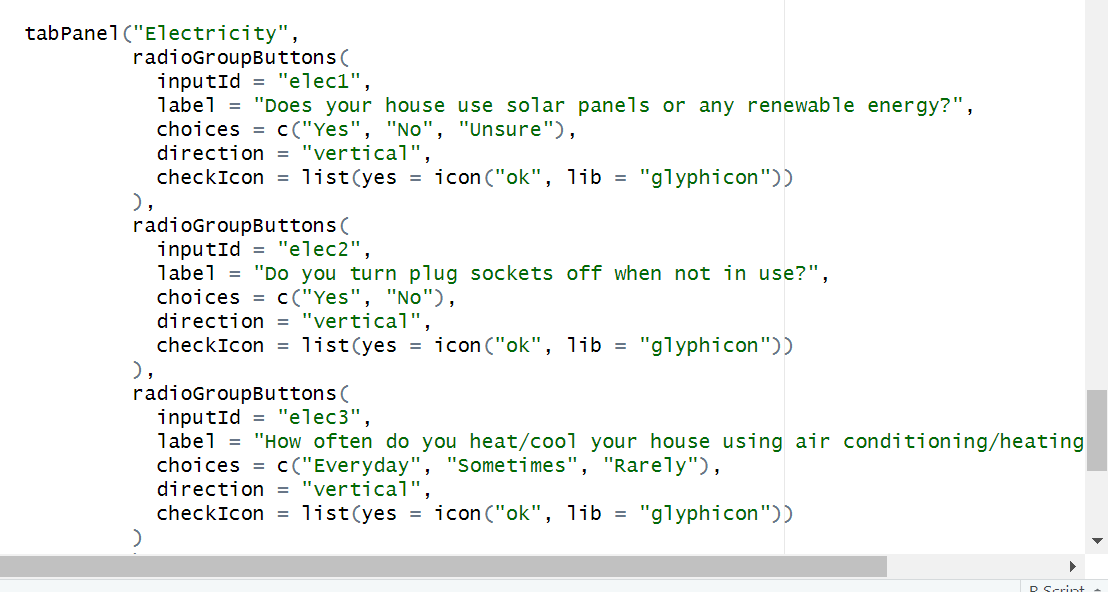


Figure 39 creating inputs for tabs

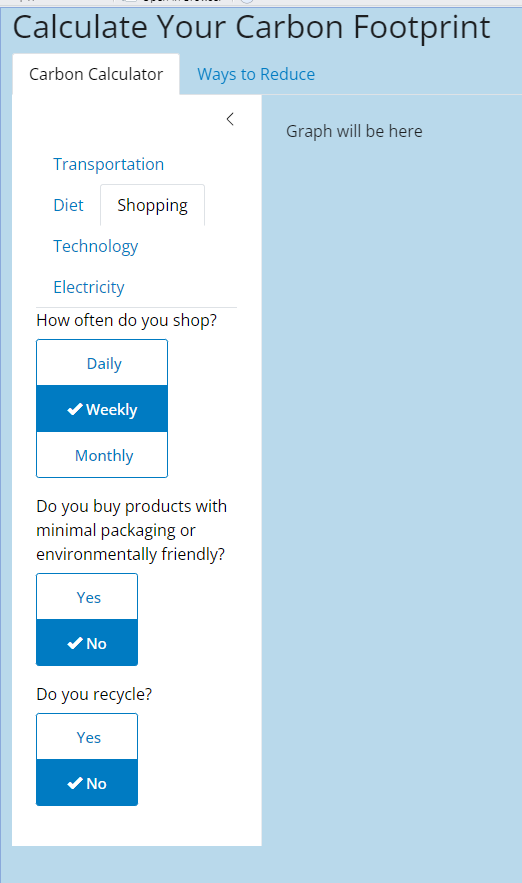


Figure 40 checking radio buttons work

**11)** I proceed to collect the user's inputs and assign numerical values to each based on their decisions. For this task, I used the approach provided at (Ben, 2022), where each input adds to a numerical value between 0 and 2.

A screenshot of a computer program

Description automatically generated

Figure 41 assigning inputs values

A screenshot of a computer

Description automatically generated

Figure 42 checking server adds values together

**12)** Using the user-selected inputs, I calculated the individual's carbon footprint, suggested as X(CP). The combined results from the questionnaire responses establish the value of X, which ranges between 0 and 2. Then, the method includes multiplying X by a constant assigned as C, which represents the UK's per capita carbon emissions (4.7). In addition, the outcome is adjusted by the percentage weights assigned to several categories, which are: transportation (19%), diet (20%), shopping (7%), home energy (19%), flights (15%), and water usage (10%).

A computer code with black text

Description automatically generated with medium confidence

Figure 43 calculating tab footprints

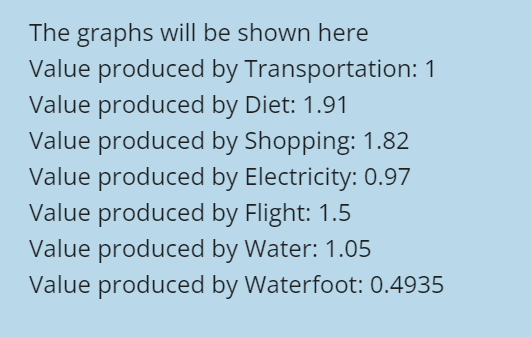


Figure 44 checking footprint calculated

**13)** Create graphical visualisations for each section of the carbon calculator with the plotly library, following the instructions at (*Bar charts in R* 2013).

A close-up of a computer screen

Description automatically generated

Figure 45 bar chart code

To check that the graphs are responsive to user inputs, I will run a test to confirm that the visualisations dynamically adapt based on the new input values. I've included the basic graph below, before any input changes were made.

A graph of a bar chart

Description automatically generated with medium confidence

Figure 46 running bar chart code

Following input changes, the updated values are shown in the graph below.

A screenshot of a graph

Description automatically generated

Figure 47 checking graph reacts

**14)** I created a distinct bar chart to compare the individual's entire carbon footprint to the average carbon footprint in the UK. To begin, I computed the individual's carbon footprint by adding the results from each step of the calculator.

A close-up of a computer screen

Description automatically generated

Figure 48 carbon footprint calculation

As a result, I used the average carbon footprint per capita in the UK, derived from extensive literature research, as an average for comparison. Using this information, I created a graph to illustrate the difference between the individual's carbon footprint and the established average for the UK population.

A computer screen shot of a program

Description automatically generated

Figure 49 comparison graph code

I ran validation tests by gradually modifying the input settings to see how responsive the graph was to these changes.

The initial graph is shown below:

A screenshot of a computer screen

Description automatically generated

Figure 50 running graphs

Following changes to the inputs, the graphs below show the updated parameters:

A screenshot of a computer

Description automatically generated

Figure 51 graph reacts test

**15)** Recognising the potential for user confusion caused by the presence of two graphs on a single page, I designed a solution that included an additional radio button. This allows users to choose their preferred graph for visualisation. I used the input radio button component from (*Shinywidgets overview* 2019) to add the user choice option into the main panel interface.

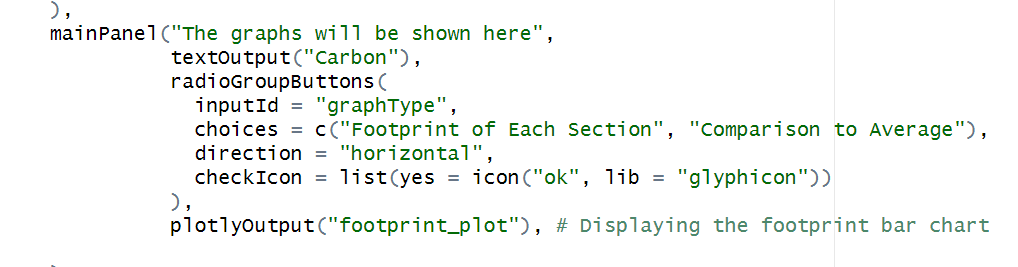


Figure 52 input option for graph choice

Then, within the server component, I added an if statement to determine the user's preferred output. This conditional statement permits the display of the appropriate graph based on the user's selection.

A screenshot of a computer program

Description automatically generated

Figure 53 code for graph selection

After making these changes, I tested thoroughly to confirm that both graphs dynamically adjust to user input, resulting in successful functionality validation.

**16)** Recognising the weak interpretation of the simple graphs, particularly when determining the quality of a user's carbon footprint, I decided to improve their visual appeal. To accomplish this, I created a colour differentiation approach in which the footprints linked with each topic were coloured differently if their respective value surpassed the value, one. Additionally, for the comparative graph, I used a similar colour scheme to emphasise carbon footprints that exceeded the average value.

relying on resources such as the practical guide (*Plotting 2.0: Shiny* 2018), and referencing insights from the book, (Wickham, 2021).

I used appropriate techniques for incorporating colour modifications into the graphs.

A screenshot of a computer program

Description automatically generated

Figure 54 changing colour of bar

Following that, I undertook extensive testing to ensure that the colour changes implemented were effective, and that the graphs appropriately reflected the changes in carbon footprint values.

A screenshot of a graph

Description automatically generated

Figure 55 bar colour test

A screenshot of a computer

Description automatically generated

Figure 56 bar (2) colour test

**17)** Recognising that the graphs lacked context without accompanying information, I included descriptive text underneath each graph. This paragraph explains the graph's purpose and the meaning of the colours used.

By involving this instructive information, readers can better grasp the visual representations in the graphs, allowing for a deeper understanding of their carbon footprint inquiry.

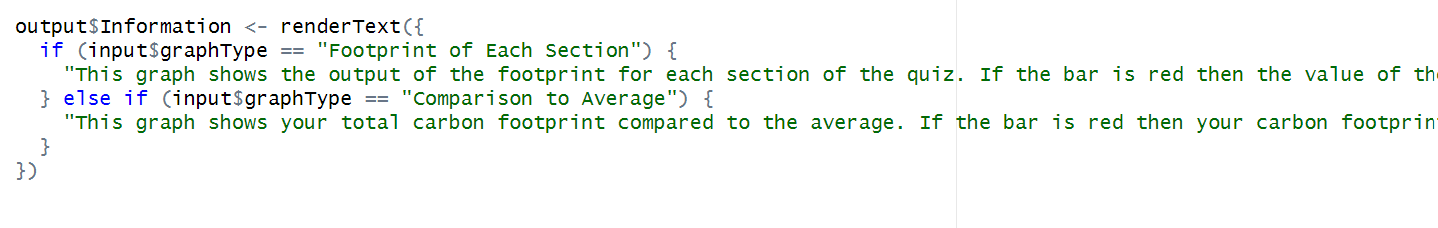


Figure 57 code for user notes

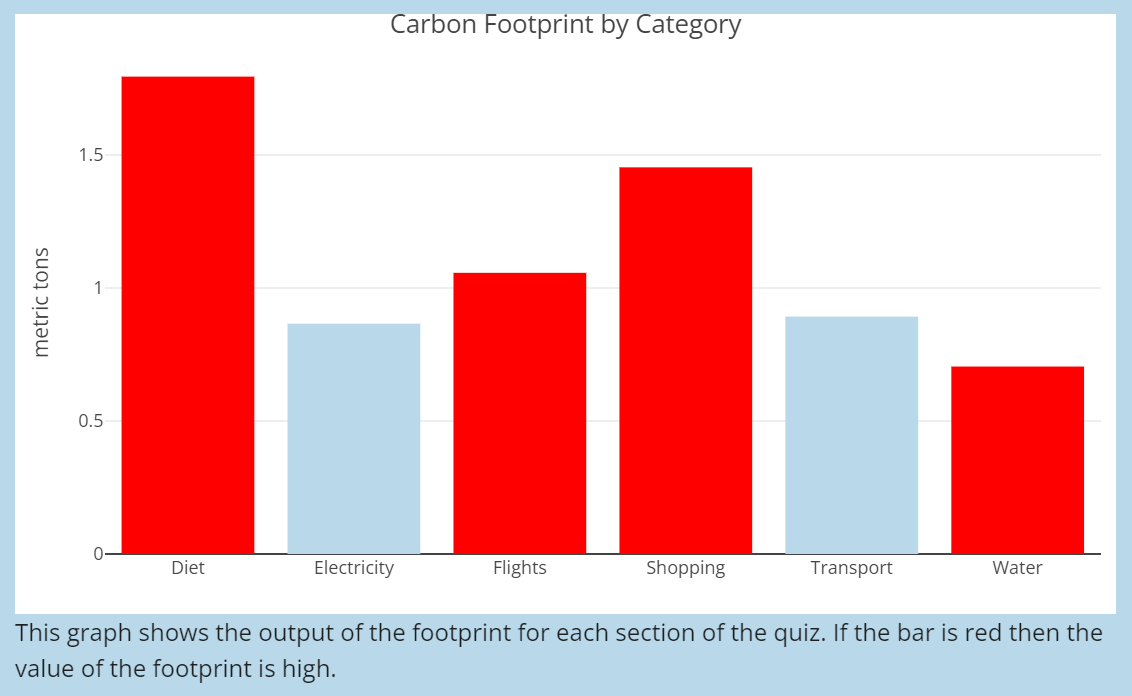


Figure 58 user notes show test

**18)** Incorporating data within the "Reduce Footprint" tab to assist users

A close-up of a computer screen

Description automatically generated

Figure 59 reduce footprint code

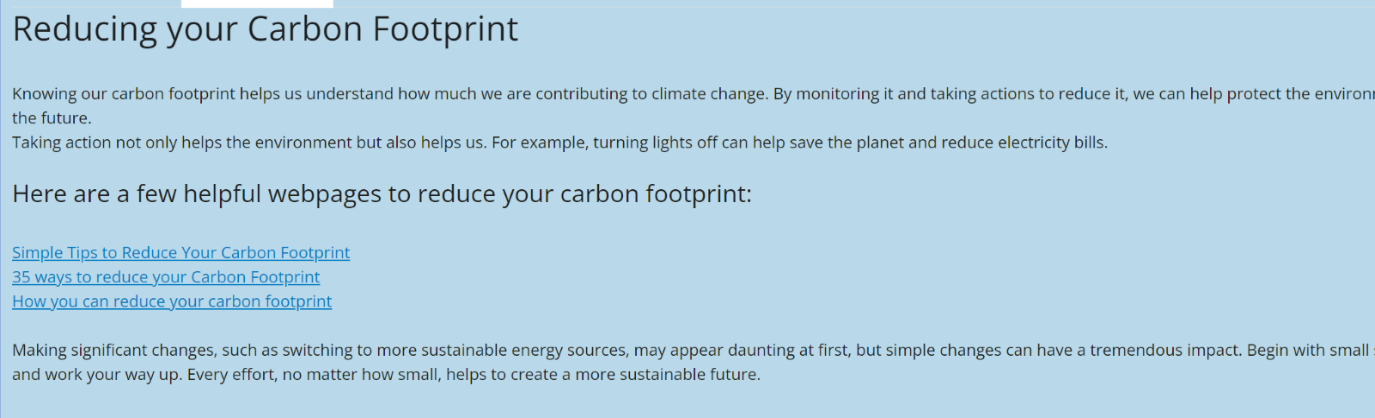


Figure 60 reduce footprint test

**19)** Acknowledging the source information is essential to maintaining integrity and giving due credit to the original creators. As a result, I have added references to the used data beneath the graphs, where it was incorporated. This provides transparency and adheres to ethical standards in information transmission.

A computer code with green and white text

Description automatically generated

Figure 61 source code

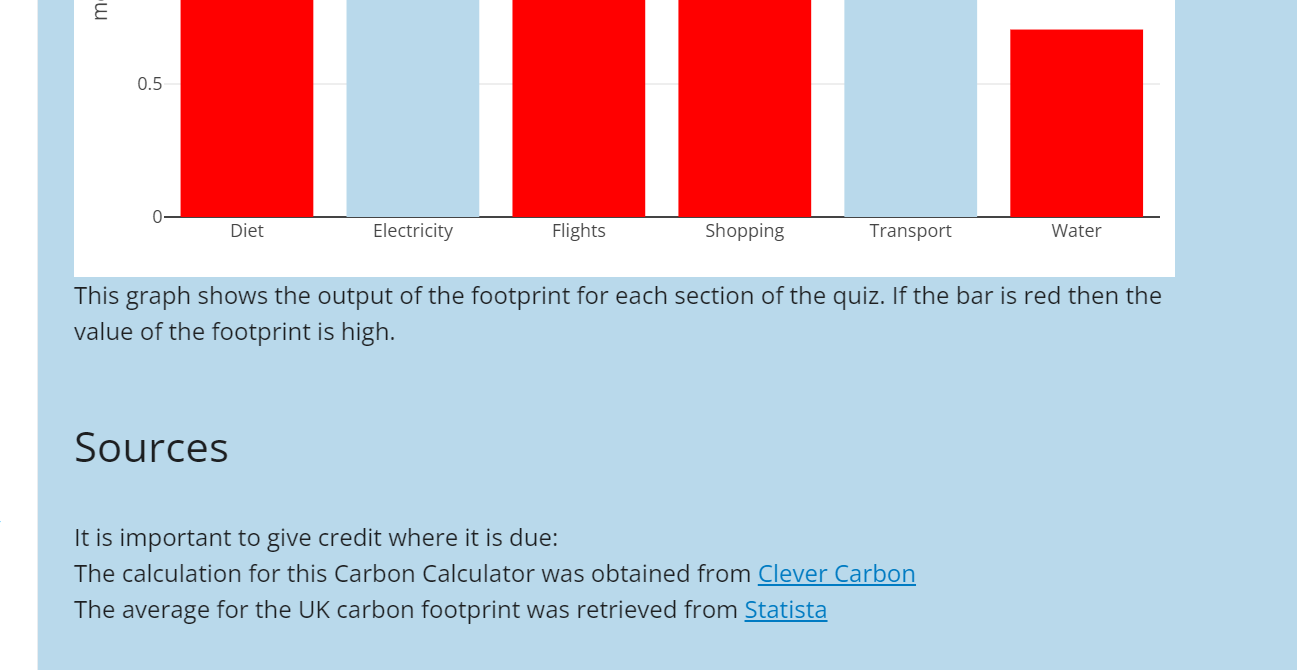


Figure 62 source code test

# Chapter 5 - Evaluation

To evaluate the carbon calculator, I asked 5 end users to test the application and fill out a questionnaire based on their experience. The results of the questionnaire were recorded and are shown below.

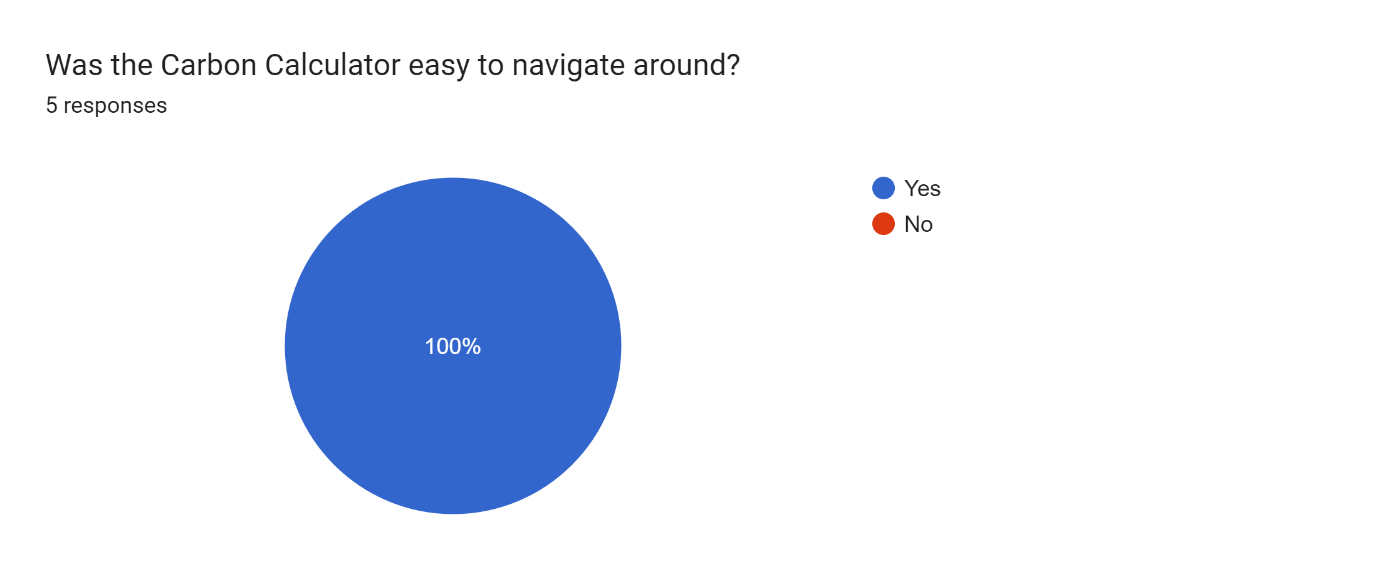


Figure 63 user testing results (1)

The figure above shows that the simple layout designed for the carbon calculator was successful as the users all responded with a yes to being able to navigate around the application. This deliberate simplicity in design was specifically intended to meet the needs of novice and elderly users, making them feel confident and at ease when working with the application. However, one noticeable shortcoming in the evaluation was the lack of feedback from novice or elderly users, which may have offered useful insights into possible areas for further improvement and development.

A screenshot of a cell phone

Description automatically generated

Figure 64 user testing results (2)

User feedback confirms the carbon calculator's accuracy in estimating carbon footprints, as seen by mainly high ratings of 4s and 5s for usefulness. One user expressed similarities to current carbon calculators, which require more precise information to assist users to calculate a more accurate result for their carbon footprint. In contrast, my calculator is targeted towards users who want quick estimates without having to enter a lot of information, which is consistent with its accessibility and user-friendly design aims. By meeting a wide range of user needs, the tool promotes inclusivity and usability, allowing a bigger audience to utilise the carbon calculator.

A screenshot of a phone

Description automatically generated

Figure 65 User testing results (3)

The users did not like the carbon calculator because it was boring and did not entice user engagement. One way to solve this would be to include a game that the user can interact with, for example a recycling game where the user earns points for the number of items that are recycled in the correct bins. Something like this would help to engage users whilst also teaching them how to recycle and reduce carbon footprint. One users said basic colours, these colours were chosen for inclusion of all users but there could be a future improvement of adding a button that lets the user select the colour scheme so that they are more engaged. Another user said the questions were basic and not precise, but the product was aimed at users who only wanted an estimate as they do not know the information that the more precise calculators ask for, however, I could include another input that allows users to be precise with their answers and get more tailored footprint results.

A screenshot of a cell phone

Description automatically generated

Figure 66 User testing results (4)

Based on feedback from users, shown in figure above, while there was some good response towards the calculator's graphs, the general response of the calculator was minimal with concerns expressed about its visual attractiveness. Given that the graphs were the only feature of the calculator that received positive feedback, it emphasises the importance of improving the design and layout to better match user expectations. As a result, in light of previous feedback suggesting that the calculator was seen as plain, there is an obvious need to prioritise visual improvements. By altering the appearance and introducing more engaging design components, such as interactive features or visually appealing layouts, the carbon calculator may improve the entire user experience and satisfaction levels.

Comparing the final product and report to the aims and objectives:

The aim of this project was to create a carbon calculator that helps users to track, visualise and understand their carbon footprint. The product created, creates graphs that shows the user their carbon footprint compared to the average and the footprint of each section. The aim of helping users visualise has been met. The product also has colour on the graphs so if the footprint is too high then the bar will turn red which will indicate to the user that it urgently needs to be improved. The aim of helping users understand their footprint has been met. The product is unable to track the user’s footprint as the product does not permanently store the user’s footprint, which mean that the final product aim has not been met. This can easily be resolved by creating a database and connecting it to the calculator so that users can easily log in and access previous carbon footprint values. Further research could be taken on how to calculate the exact carbon values for each section and ask more questions but not too precise that users do not know the answer.

I have successfully conducted a literature survey to evaluate the effectiveness and user interfaces of existing carbon tracking apps. I was able to identify the most advantageous features of the carbon calculator in the literature survey. I thought about the target audience and designed the user interface with them in mind. I successfully implemented and tested the user interface. I developed and tested the system's backend using R Shiny and R. I performed user testing to check if the system met user requirements. I created a report that holds all my research and implementation. Overall, I have met all of my objectives.

End-user testing showed areas in which the product may be improved. A number of improvements could potentially be done in future versions to solve these issues. To begin, incorporating a database feature would allow users to track their progress over time, offering important insights and promoting ongoing improvement of the user’s carbon emissions. Furthermore, given user comments on visual appeal, implementing a colour scheme customisation option will respond to individual preferences while increasing satisfaction among users.

Furthermore, introducing elements of gamification like leaderboards or interactive games may greatly increase user engagement. These elements, which encourage competition and motivate users to lessen their carbon impact, can provide a more engaging and inspiring experience. Overall, these modifications plan to improve the product's usability, visual appeal, and effectiveness in guiding long-term behaviour change.

# Chapter 6 - Conclusion

The creative piece resulted in a useful and practical product; a successful carbon calculator capable of calculating users' carbon footprint. Through this project, I gained an improved understanding of carbon footprints, understanding the difficulty of handling varying user requirements while trying for universal satisfaction. It became clear that customer preferences differ, and reaching agreement on product features is a difficult process. Additionally, this project was a good learning experience for me, since it improved my knowledge of the R programming language. In addition, this project provided valuable insight into the ethical considerations developers face when creating products to be ethically accepting and fit the majority of user’s needs.

# References

1. (2018) *Plotting 2.0: Shiny*. Available at: <https://therbootcamp.github.io/BaselRBootcamp_2018April/_sessions/D3S3_PlottingII/PlottingII_practical.html> (Accessed: 06 May 2024).
2. Ahmad, N. (2024) *5 most popular types of SDLC models*, *LambdaTest*. Available at: <https://www.lambdatest.com/blog/sdlc-models/> (Accessed: 12 April 2024).
3. *Bar* (2013) *Bar charts in R*. Available at: <https://plotly.com/r/bar-charts/> (Accessed: 06 May 2024).
4. Beeley, C. (2018) *Web application development with R using shiny: Build stunning graphics and interactive data visualizations to deliver cutting-edge analytics*. Birmingham, UK: Packt Publishing.
5. Ben (2022) *How to calculate carbon footprint: Carbon Footprint Calculator*, *clever carbon*. Available at: <https://clevercarbon.io/how-to-calculate-your-carbon-footprint/> (Accessed: 26 April 2024).
6. Boulos, P.F. and Bros, C.M. (2010) ‘Assessing the carbon footprint of water supply and Distribution Systems’, *Journal AWWA*, 102(11), pp. 47–54. doi:10.1002/j.1551-8833.2010.tb11338.x.
7. Bucanek, J. (2009) ‘Model-view-controller pattern’, *Learn Objective-C for Java Developers*, pp. 353–402. doi:10.1007/978-1-4302-2370-2\_20.
8. *Build a user interface* (2014) *Shiny*. Available at: <https://shiny.posit.co/r/getstarted/shiny-basics/lesson2/> (Accessed: 06 May 2024).
9. *CO2 emissions (metric tons per capita) - united kingdom* (2021) *World Bank Open Data*. Available at: https://data.worldbank.org/indicator/EN.ATM.CO2E.PC?locations=GB (Accessed: 07 May 2024).
10. Deacon, J. (2000) *Model-View-Controller (MVC) Architecture*, *JOHN DEACON Computer Systems Development, Consulting & Training*. Available at: <https://d1wqtxts1xzle7.cloudfront.net/50526307/MVC-libre.pdf?1480020702=&response-content-disposition=inline%3B+filename%3DModel_View_Controller_MVC_Architecture.pdf&Expires=1713392997&Signature=a1EgkzehSeTRFr2yTl0d8WgnR4MAECl4I-WFRNJ0ZeQJqRaVNIsnEd5xnZDE0hKfR0cPTX4mcIwsyjlQXc4EnDH87RJdl9tjqlMcWoGbxENIlO0Ieo31XWi4ssppKyCanYHvMQXTAAHuFrp1cKbZfX9vYHF2ccRjhx-IIno~sECI65KsPrSPEMz5C67pBB09v9q4bJm0ab6idwJv2gHj4GDUwR8yR60fwj4fQOrFAtWQQoLUri7owttEBe3CDhj5yW8QF7q8mq7onBe2AfEVV1JH-yzymbKpG4JKJJy2yAnRw9ll9RRLkfBIkB8QRm34xjMFGFbjJLuW0EUdOztjvQ__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA> (Accessed: 17 April 2024).
11. Franchetti, M.J. and Apul, D. (2013) *Carbon Footprint Analysis: Concepts, methods, implementation, and case studies*. Boca Raton, FL, Florida: Taylor & Francis.
12. Gurung, G., Shah, R. and Jaiswal, D.P. (2020) ‘Software development life cycle models-A comparative study’, International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 6(4), pp. 30–37. doi:10.32628/cseit206410.
13. Just Energy, J.E. (2013) *Calculating your carbon footprint*, *How To Calculate Your Carbon Footprint*. Available at: <https://justenergy.com/blog/how-to-calculate-your-carbon-footprint/> (Accessed: 26 April 2024).
14. Kumar, G. and Bhatia, P.K. (2012) (PDF) impact of agile methodology on software development process, Impact of Agile Methodology on Software Development Process. Available at: https://www.researchgate.net/publication/255707851\_Impact\_of\_Agile\_Methodology\_on\_Software\_Development\_Process (Accessed: 08 April 2024).
15. Ledley, T.S. *et al.* (1999) ‘Climate change and greenhouse gases’, *Eos, Transactions American Geophysical Union*, 80(39), pp. 453–458. doi:10.1029/99eo00325.
16. Mckay, B. (2024) *CO2 emissions per person in the UK - over the last 200 years!: Greenmatch*, *GreenMatch.co.uk*. Available at: https://www.greenmatch.co.uk/blog/uk-co2-emissions (Accessed: 07 May 2024).
17. Mohammad, S.M. (2013) Even the abstract have colour: Consensus in word-colour associations, arXiv.org. Available at: <https://arxiv.org/abs/1309.5391> (Accessed: 08 April 2024).
18. Montag, C. *et al.* (2015) *Smartphone usage in the 21st century: Who is active on WhatsApp? - BMC research notes*, *SpringerLink*. Available at: <https://link.springer.com/article/10.1186/s13104-015-1280-z> (Accessed: 05 April 2024).
19. Nielsen, J. (2005) Semanticscholar, Ten Usability Heuristics. Available at: https://pdfs.semanticscholar.org/5f03/b251093aee730ab9772db2e1a8a7eb8522cb.pdf (Accessed: 14 April 2024).
20. Panzone, L.A. *et al.* (2021) ‘Sustainable by design: Choice architecture and the carbon footprint of Grocery Shopping’, *Journal of Public Policy &amp; Marketing*, 40(4), pp. 463–486. doi:10.1177/07439156211008898.
21. Pargaonkar, S. (2023) ‘A comprehensive research analysis of Software Development Life Cycle (SDLC) Agile & Waterfall Model Advantages, disadvantages, and application suitability in software quality engineering’, International Journal of Scientific and Research Publications, 13(8), pp. 120–124. doi:10.29322/ijsrp.13.08.2023.p14015.
22. Philippe, F.-X. and Nicks, B. (2015) ‘Review on greenhouse gas emissions from pig houses: Production of carbon dioxide, methane and nitrous oxide by animals and manure’, *Agriculture, Ecosystems &amp; Environment*, 199, pp. 10–25. doi:10.1016/j.agee.2014.08.015.
23. Rigden, C. (1999) Designing for colour-blind users - accessible Techcomm, The Eye of the Beholder’— Designing for Colour-Blind Users. Available at: https://accessible-techcomm.org/wp-content/uploads/colours-eye-of-the-beholder.pdf (Accessed: 14 April 2024).
24. *Shinywidgets overview* (2019) *shinyWidgets*. Available at: <https://shinyapps.dreamrs.fr/shinyWidgets/> (Accessed: 06 May 2024).
25. *Tabsets* (2014) *Shiny*. Available at: <https://shiny.posit.co/r/gallery/application-layout/tabsets/index.html> (Accessed: 06 May 2024).
26. Tiseo, I. (2023) *UK: Per capita co₂ emissions 1970-2022*, *Statista*. Available at: https://www.statista.com/statistics/1299198/co2-emissions-per-capita-united-kingdom/ (Accessed: 07 May 2024).
27. Tiseo, I. (2023b) *UK: Travel Carbon Footprint by transport mode 2023*, *Statista*. Available at: https://www.statista.com/statistics/1233337/carbon-footprint-of-travel-per-kilometer-by-mode-of-transport-uk/ (Accessed: 10 May 2024).
28. *Version Information* (2008) *Version information*. Available at: https://www.carbonindependent.org/25.html (Accessed: 10 May 2024).
29. Visual Studio, V.S. (2021) *Get started with visual studio code*, *RSS*. Available at: <https://code.visualstudio.com/learn> (Accessed: 17 April 2024).
30. Wickham, H. (2021) *Mastering shiny: Build interactive apps, reports, and dashboards powered by R*. Sebastopol, CA: O’Reilly, $c 2021.
31. Wiedenhofer, D. *et al.* (2018) ‘Household time use, carbon footprints, and urban form: A review of the potential contributions of everyday living to the 1.5 °C climate target’, *Current Opinion in Environmental Sustainability*, 30, pp. 7–17. doi:10.1016/j.cosust.2018.02.007.

# Appendix A

Undergraduate and PGT Application: ID 63878

A screenshot of a computer

Description automatically generated

A close-up of a text

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A questionnaire with text on it

Description automatically generated

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a medical survey

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A close-up of a document

Description automatically generated