Imperial College London

Imperial College London

Department of Computing

Alter Eco: An Aid for a Greener Lifestyle

Author:
Michele Asara

Supervisor:
Dr. Anandha Gopalan

Second marker:
Dr. Timothy Kimber

Submitted in partial fulfillment of the requirements for the MSc degree in M.Sc. Computing Science of Imperial College London

September 2020

Abstract

In recent years the public has been exposed to the problem of global warming and its troubling consequences for the environment. The root of the issue is the release of greenhouse gases into the atmosphere as a byproduct of human, industrial and technological complexes. Part of the general public has grown conscious of this problem, but even today understanding and improving our own environmental impact can be a considerable challenge. To fill this gap, we propose Alter Eco, an iOS app that combines gamification and automatic computation of a user's carbon footprint to guide towards a more eco-friendly behaviour. By employing custom-made algorithms and natural language processing techniques, Alter Eco is capable of estimating, aggregating and visualizing user's travelling and dietary emissions. This data then acts as the basis for a point system which can be spent to build a 3D virtual forest. Continuous feedback in the form of surveys and the last evaluation by our beta testers seem to confirm the effectiveness of Alter Eco as an aid for a greener lifestyle.

Acknowledgments

To my girlfriend Deli, for her continuous support. To my sister, Irene, for her inputs and help in redesigning the logo. And last but not least, to Dr. Anandha Gopalan for his supervision and suggestions.

Contents

List of Figures

Chapter 1

Introduction

1.1 Global warming

A vital component for the thermal regulation of the planet are the greenhouse gases (GHG). These are gases which absorb and release energy within the infrared spectrum. Since the industrial revolution, their concentration in the atmosphere has been alarmingly increasing, and it is predicted to keep its current trend (Figure 1.1). Under normal circumstances the amount of gases would be naturally regulated, but human civilization has been exponentially incrementing its greenhouse production beyond what is sustainable [2]. This led to a global warming effect, with consequences such as the melting of glaciers [3], the rising of sea levels, and numerous struggles for the wildlife and vegetation [4, 5]. Therefore, we are now facing a threat to the balance of all life on the planet, humans included.

Being the biggest contributor to overall emissions (Figure 1.2a), carbon dioxide is used as a reference unit for greenhouse gases, which are often converted to a *carbon dioxide equivalent* (also written as CO_2e, CO_2eq or similar abbreviations) [6]. A *carbon footprint* is then the amount of GHG associated with an activity or product and expressed in terms of carbon dioxide [7].

1.2 Lack of technological aids

A closer look at the causes of GHG emissions in the world reveals the critical role of the energy, transport, and agricultural sectors. As an example, in the European Union of 2017, up to 53% of all emissions were due to fuel combustion and fugitive emissions (Figure 1.2b). Nevertheless, this kind statistics is more within the realm of countries as a whole rather than normal citizens. A single individual would perhaps find more interest in the fact that up to 25% of all emissions was due to transportation, and that food production alone accounts for 26% (Figure 1.2b and Figure 1.3, respectively). As such, it follows there are elements in the life of an individual which can be modified to

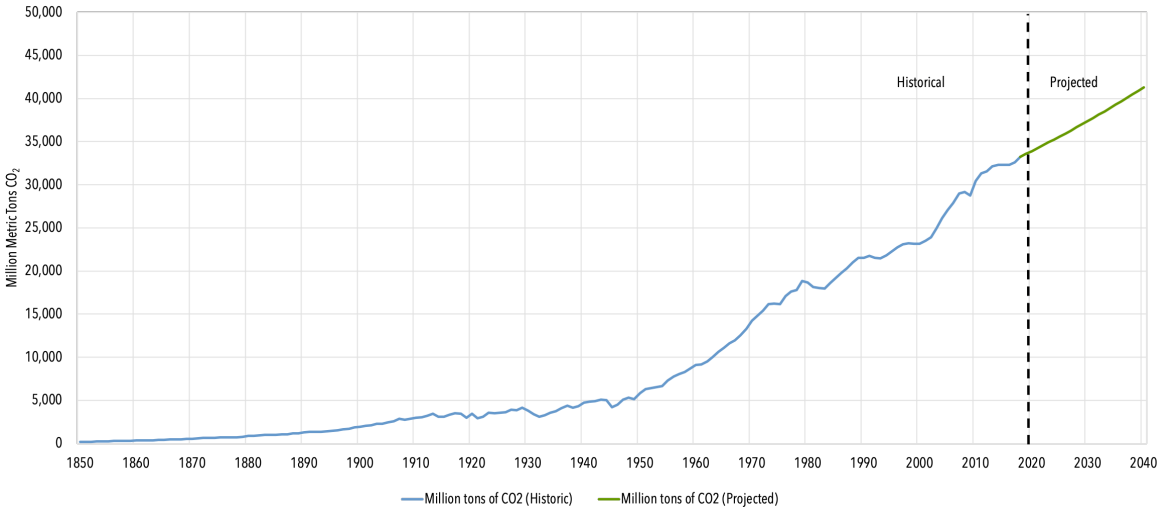
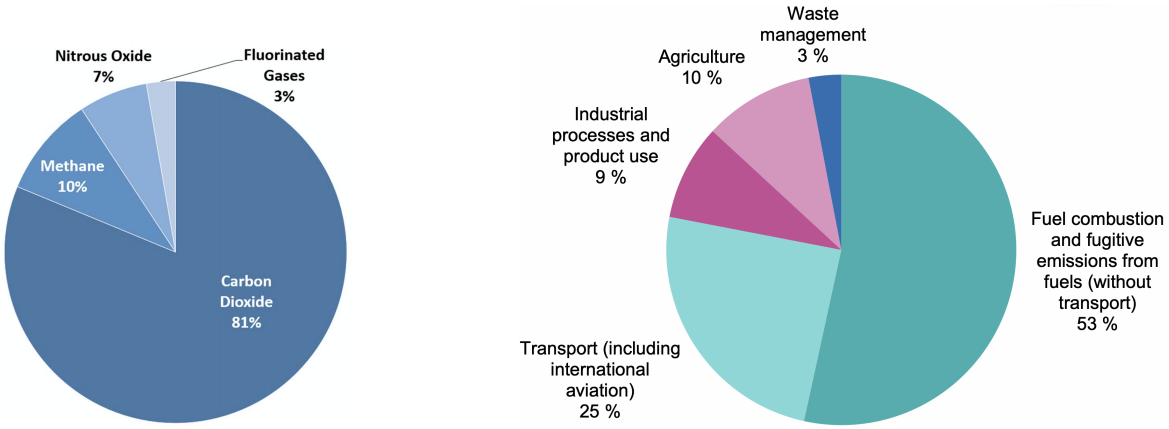


Figure 1.1: Global increase of carbon dioxide in the atmosphere starting from 1850, with forecast up to 2040. Image from [1].



(a) (b)

Figure 1.2: (**a**) Overview of greenhouse emissions in 2018. Percentages do not add up to 100% due to independent rounding. Image from [8]. (**b**) Greenhouse gas emissions by source sector, EU-28 2017. Image from [9].

reduce global warming significantly.

Due to the lack of proper instruments, individuals are left to reason about their emissions in a qualitative, rather than quantitative way. Albeit carbon footprint calculators exist, they rely on continuous inputs from the user, which can be erroneous. Perhaps more importantly, requiring users to repeatedly input their data into a multi-variable calculator comes at a non-trivial interaction cost [10]. That is, the effort which the design requires the user to put is likely to be unsustainable in the long run. Therefore, even if one wishes to take action to reduce their carbon footprint, there are no easy technological aids that can help them quantitatively analyse their own habits. This can lead to false beliefs around one's own emissions or what contributes to them. For example, contrary to popular belief, it has been the finding of a study by Wynes et al. that lifestyle changes such as avoiding transatlantic flights or switching to a plant-based diet can impact an individual's carbon footprint as much as, if not more, than four times what they would get by only recycling [11].

1.3 Aims and objectives

It is now clear there is a gap between what technology should provide to reduce global warming and what is currently available. Even if we employ national or worldwide policies to contain our emissions, we are still missing the means for a stronger individual action. The general aim of this project is to introduce a system that can offer support in the formation of greener, long-lasting habits. In order to achieve this, we need something people can easily acquire and regularly use. Furthermore, there must be a way to direct a user towards our target behaviour. In other words, it is our aim to build a system that is:

- **Accessible**, because without users there is nothing to change.
- **Able to orient behaviour**, to fulfill our aim of a greener lifestyle.
- **Engaging, intuitive and easy to use**, so as to form lasting habits.

We particularly believe in a data-driven approach, as it allows to infer patterns and make decisions based on evidence, rather than intuition. Therefore, the main objectives of the system we envision are to:

- **Collect relevant data** in an automated, or semi-automated fashion.
- **Allow visualization** of this data.
- **Provide an evaluation mechanism** to determine how well or how bad a user is doing.

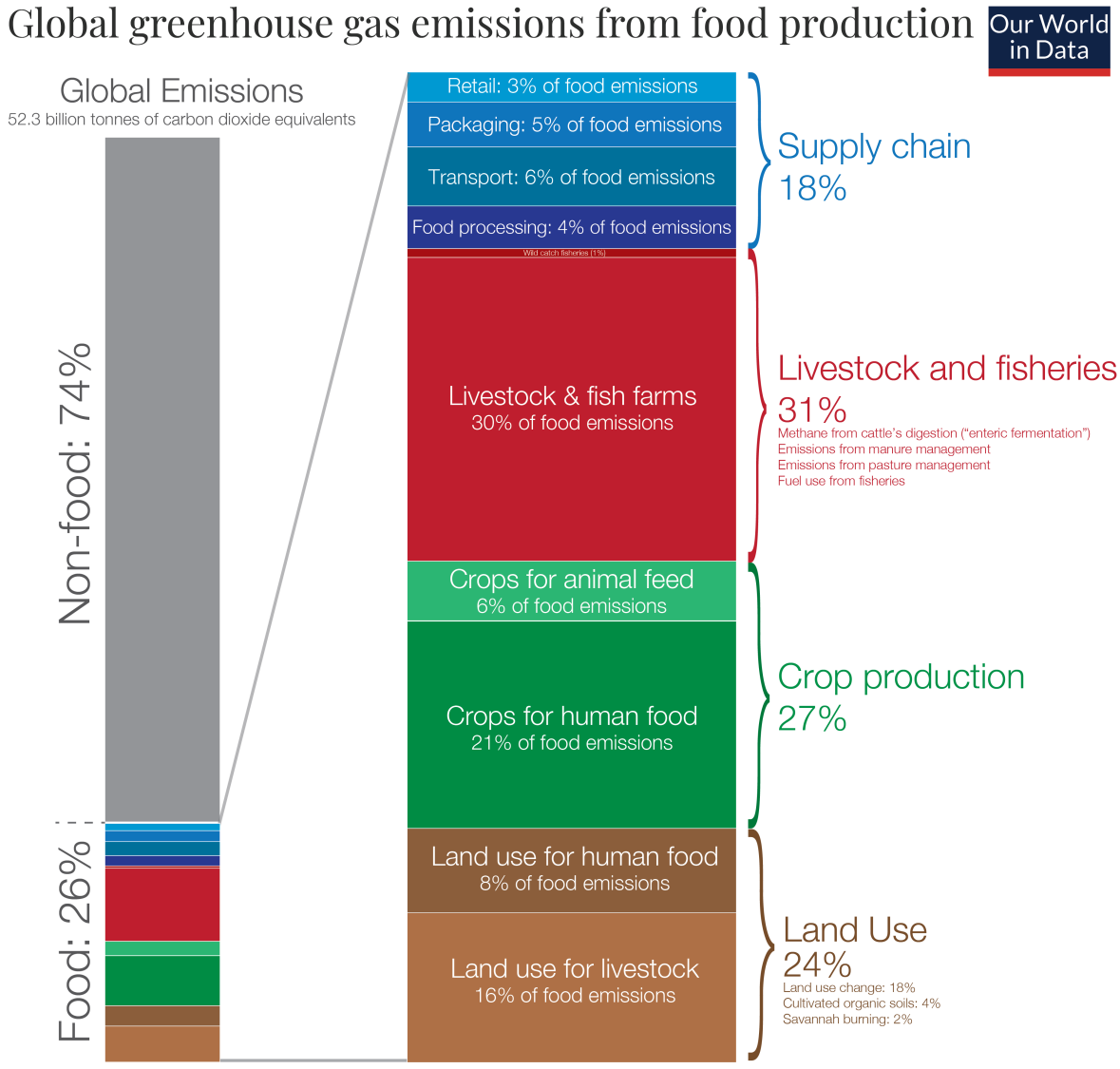


Figure 1.3: Visualization of the impact food production has on global greenhouse emissions. Data from [12]. Image from [13].

1.4 Contribution

To fill this gap, we present Alter Eco. It is an application for systems running iOS/iPadOS 13 and above, which aims to be a technological aid for users wishing to gain insight about their carbon footprint and how to reduce it. It was originally developed as a student group work at Imperial College London [14, 15, 16], but it has been considerably improved and extended in the duration of this individual project.

Alter Eco was designed to be run on smartphones to solve the first accessibility requirement. Indeed, in the last decade smartphones have conquered the digital communication market all over the world, so much that almost 80% of people in the UK own a personal device [17]. The accessibility is not just a matter of numbers, but it is also physical, as users can access the system by literally reaching for their pocket. Furthermore, we as engineers can rely on the device's sensors to capture relevant data, thus minimizing the interaction-cost issue previously described. Finally, users were involved in the design and development of the application since the early stages of the project both via surveys and a beta version, which helped building an intuitive interface and ensuring an overall pleasing user experience.

In conclusion, the current functionalities of Alter Eco include:

- Automatic computation of a user's transport emissions.
- Estimation of the carbon equivalent of a food product by scanning its barcode.
- Data visualization and aggregation in either bar or pie charts.
- Comparison between the user's daily emissions and the UK average for food and transport, together with quantitative and visual feedback.
- A gamification system that rewards continuous use of the app and eco-friendly behaviours with points and achievements. The score can then be spent to build a personalized 3D forest.

Given the relevance of the food and the transport sectors in the context of GHG emissions (Figure 1.3 and Figure 1.2b), Alter Eco is able to estimate a considerable amount of a user's carbon footprint. Furthermore, the use of visual and quantitative cues provides the individual with a mechanism to evaluate their pollution. Finally, the gamification system acts as a guide to orient users towards our target behaviour and as a form of entertainment to keep them engaged.

Chapter 2

Background

Here we provide the necessary background theory to understand the origin and the current features of Alter Eco.

2.1 Estimation of carbon footprint and traditional calculators

To gain a better appreciation of the usefulness of Alter Eco, we will briefly discuss the current methodologies for estimation of carbon footprint. We can identify four main methods:

- *Emission coefficient method* (ECM), in which mathematical methods are employed to derive the average emissions per relevant unit. The data could be taken from historical statistics either at the national or at the household level, but it would be constrained to one particular sector only (e.g. transport or energy).

- *Life cycle assessment* (LCA), which is a technique to estimate the environmental impact associated with all the stages of a product's lifetime. It starts from raw material extraction and includes data related to processing, manufacture, distribution, use, maintenance, and disposal. The International Organization for Standardization provides guidelines for conducting a Life Cycle Assessment [18].

- *Consumer Lifestyle Approach* (CLA), which is an interdisciplinary framework to study consumer-related issues of carbon emissions and energy use [19]. The key idea behind this approach is to put the focus on the individual, their external cultural or technological variables, their attitudes, purchases and expenditures, and their household characteristics (such as size, income and location).

- *Carbon Emissions Calculator* (CEsC), which is usually comprehensive of all the methods mentioned above and requires direct inputs. The main advantage is that it simplifies the process by hiding most of the complexity from the user.

Some of these methodologies are better suited for environmental policy analysis at a macroscopic level, more within the realm of countries or large organisations rather than individuals. Furthermore, some of them are long, complex processes which require knowledge and understanding of concepts that are unlikely to be found outside of professionals of the field. In particular, ECM, LCA and CLA are usually employed by policy makers, rather than everyday people. As such, carbon calculators are likely to be the only true candidate for individual users.

Nevertheless, modern calculators do not follow a standard procedure. Some of them directly ask the user to input detailed measurements regarding their energy, food, waste and transportation (see [20] for an example). In these cases, they ask about the specific energy conversion factor for a country, or the car mileage, or average expenditure in a pre-determined food category. All information which the user might not even be aware of, and that is likely to divert them from a second (let alone frequent!) use. Alternatively, some calculators are lengthy questionnaires which do not require detailed information, but are still time consuming and very much vulnerable to incorrect inputs (as is, for instance, the one made by WWF [21]). The purpose of Alter Eco is then to simplify the calculation of a user's carbon footprint by hiding unnecessary details and by performing the computations as quickly, reliably and automatically as possible.

2.2 Word embeddings

The computation of the carbon footprint of a food product from its barcode was achieved by essentially connecting two different databases: one for the general product information (such as weight, category tags and nutritional values) and one associating food types to a carbon footprint density. As it will be discussed in Chapter 4, we were able to connect these two databases by relying on a *Natural Language Processing* (NLP) technique concerning the semantics of words and known as *Word Embeddings*. As such, in this section we will give a brief introduction to the purposes and key concepts behind word embeddings.

2.2.1 The elusiveness of word semantics

Building a computational model capable of understanding the meaning of text is one of the fundamental challenges of the branch of Natural Language Processing. When we look up a word in a dictionary, we immediately notice the same lemma can have different meanings or *word senses*. Generally, word sense disambiguation is based on the context in which a word appears due to the fact that appearing in similar contexts leads to similar meanings. This connection between how words are distributed and their semantics is what is known in linguistics as distributional hypothesis and was initially formulated in the 1950s [22].

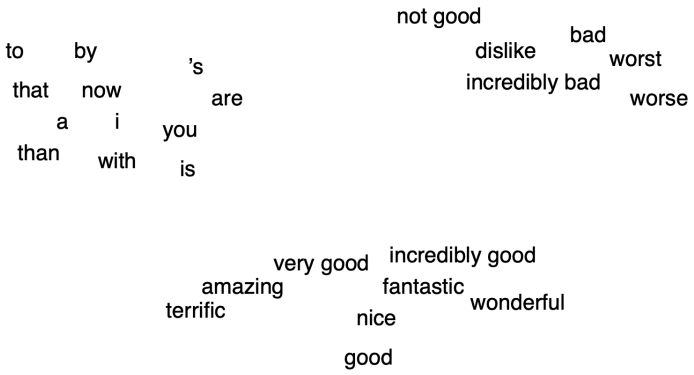


Figure 2.1: A 2D projection of 60-dimensional word embeddings trained for sentiment analysis. Words of similar meaning are spatially close. Image from [23].

In linguistics, semantics is not limited to assigning meaning to a sequence of symbols. One important aspect of the meaning of a word is the relationship between word senses. For example, if, given a sentence, we can substitute a word for another without altering the truth conditions of the sentence, we call the two word senses *synonyms* [23]. Nevertheless, words can be *similar* or *related* even though they do not have any matching word senses. Consider the words 'mouse' and 'cat': although these share a similarity (both describe animals), they have no common word senses. Something analogous could be said about 'glasses' and 'sight': they are not the same, nor are they similar, and yet have related meanings.

2.2.2 Words as vectors

It is then clear how understanding the meaning of a word and how it relates to others is no trivial task. The need for a quantitative way of reasoning about word semantics becomes fundamental if one wishes to build a computational model capable of such a feat. Thankfully, as early as 1957, Osgood et al. [24] realized that a *word connotation* (i.e. how the meaning of a word relates to the reader's emotions) could be identified by a numerical quantity on three dimensions: dominance, valence and arousal. Thus, for the first time, a word could be represented and visualized as a point on a three dimensional space.

The combination of Osgood et al.'s idea and the distributional hypothesis is what lies underneath's the modern *vector semantics*. A word is represented as a vector in a semantic space of N-dimensions, where each vector component is calculated based on counts of neighbouring words [23]. These vectors are generally called *embeddings* as the word they represent is embedded in a vector space (see Figure 2.1). The main advantage of this kind of representation is that it is great for encoding similarities, inferring relations or deducing synonymity. We can perform semantic operations on embeddings as if they were algebraic operations on any other vector, so that for instance the following

equivalences hold:

$$king - queen + woman \approx man$$

$$london - england + italy \approx rome$$

$$car - cars \approx apple - apples$$

Importantly, we can quantify similarity between two vectors $\mathbf{A}$ and $\mathbf{B}$ by considering the angle between them. Re-arranging the dot product formula leads to the derivation of the *cosine similarity* (Equation 2.1), which ranges from -1 (for opposite embeddings) to 1 (for identical ones).

$$cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{2.1}$$

Also commonly used is the *cosine distance* (Equation 2.2), which instead maps similarity to a value between 0 (for identical vectors) and 2 (for opposite vectors).

$$D_C(\mathbf{A}, \mathbf{B}) = 1 - cos(\theta) \tag{2.2}$$

Although we have now acquired an understanding of the properties of word embeddings, how the numerical value of each dimension is calculated has not been discussed yet. In truth, it depends on the methodology adopted, but we can identify two state-of-the-art algorithms in the literature: *GloVe* [25] and *word2vec* [26].

2.2.3 GloVe and word2vec: a brief overview

Both GloVe and word2vec are machine learning algorithms which take a text corpus and output a word embedding. Nevertheless, the way in which this is done is completely different. For instance, word2vec is a neural network with a single hidden layer that can be trained in two ways:

- Continuous Bag of Words (CBOW): learning word embeddings by predicting a word given its context (e.g. given '*The king of the jungle is the*' we would expect 'lion' as the most likely answer).

- Skip-Gram: learning word embeddings by predicting the context given a word, where by context we mean a probability distribution over the whole vocabulary.

Although these two methods are conceptually different, the real purpose behind the training is to obtain the weights of the hidden layer. Indeed, once the model has been trained, we can extract an embedding by selecting the relevant weights of the hidden layer.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Table 2.1: Encoding of meaning via ratio of co-occurrence probabilities in GloVe. The notation $P(A|B)$ is to be intended as the probability of A given B. Data from [27].

On the other hand, GloVe is completely unsupervised. Although the mathematical intricacies of the model are beyond the purposes of this paper, the main idea is that ratios of co-occurrences between words probabilities can encode meaning [28, 29]. As an example, consider the data relative to the keywords 'ice' and 'steam' displayed in Table 2.1. The word 'solid' is related to 'ice' but not to 'steam', which gives $P(solid|ice)/P(solid|steam)$ a high value. Conversely, 'gas' is related to 'steam' but not 'ice', and indeed the ratio is small. Finally, 'water' (which is related to both keywords) and 'fashion' (which is not related to any of them) obtain a ratio close to 1. In practice, GloVe learns from a co-occurrence matrix based on the whole corpus, and trains word vectors so their differences predict ratios of co-occurrence probabilities [27].

Which model effectively performs better is highly dependent on the task and training set at hand. That being said, the advantage of GloVe over word2vec is that it does not rely on local statistics only (i.e. local context information), but incorporates global statistics to infer embeddings. For example, this means word2vec cannot tell if the word 'the' in '*the cat is on the table*' has any special relation to 'cat' or if it is simply a very common word. In conclusion, it should be noted that Pennington et al. [25] compared GloVe and word2vec in a word analogy task and found that GloVe consistently outperformed its counterpart for the same parameters. It should then not come as a surprise that we preferred GloVe as our embedding model in Alter Eco.

2.3 Alter Eco group project

As mentioned earlier, Alter Eco started as a group project at Imperial College London. Therefore, it is appropriate to discuss what features were implemented by the end of it and how these were modified during the individual project stage.

2.3.1 Transport tracking

By the end of the group project, Alter Eco was an application only capable of tracking and displaying the data relative to a user's transportation methods (i.e. car, train, airplane and walking). Due to the COVID-19 emergency [30], most countries around the world forbid non-fundamental travelling and employed strict lockdown policies. This meant the inability to verify the software's behaviour extensively outside of computer-aided simulations and unit testing. By the start of the individual project, these lockdown policies were relaxed and Alter Eco was released to beta testers around the globe. Although the main algorithm for transport detection stayed the same, the first weeks of the individual project were spent collecting feedback from the users, polishing the existing software and fixing bugs which could not be detected before.

A detailed description of the algorithm is beyond the scope of this report, but the keen reader may refer to Appendix B. In summary, a user's position is tracked over time, and using both location and speed data we infer the type of transportation they are likely to have taken. Location data is obtained via *Core Location* [31], the official Apple framework for location tracking which relies on GPS, Wi-Fi and other built-in sensors. The algorithm is instead based on the concept of activities, which are structures containing the start and the end times of a movement, the distance covered and the inferred transportation mode. There are two types of activities:

- **Speed-based activities**, which are determined on the basis of how fast the user has travelled. Speeds equal or greater than a given threshold are considered car activities, while those below are classified as walking or cycling. A list of speed activities is kept in memory until a significant change occurs, where by significant change we mean a specified number of activities in a row which follow activities of a different type. When this happens, the list is synthesized into one average activity and stored in the database.

- **Region-of-interest (ROI) activities**, which are created when the user visits locations such as airports or underground stations. These are not stored in a list. Rather, once two ROIs of the same type are visited (such as going from a station to another), the distance is inferred and the activity is stored in the database.

Albeit the fundamental functioning of the transport algorithm remained the same as the group project version, we did make some changes that will be discussed in Chapter 4.

2.3.2 The old gamification model

Gamification was considered since the beginning, and thus users were able to gain points which then translated into textually-represented *leagues* (Figure 2.2). A league was intended as a level the user could obtain by continuous use of the app. To be specific, every 3000 points the user went up to the next league. There were three leagues: seedling, medium plant and tree, all represented by an emoji. Once the user reached the '*tree*' league, a tree counter would increase, their points would drop to zero and the league system would restart from seedling.

This was the origin of the 3D virtual forest which is currently implemented in Alter Eco. Many users complained the point system was too strict, meaning that accessing the next league took way too long, and the textual representation was not stimulating. Although the concept is similar, using 3D rather than the old model offers stronger visual effects. The addition of a spatial dimension also allows greater personalisation, as the user is now able to move the items within the forest as they please. Furthermore, as it will be discussed later, the current 3D forest introduced the concept of a '*shop*' where users can spend their points to buy items. As such, we are now able to offer different items for different prices, thus giving more flexibility in the difficulty of the game. Finally, a 3D world inherently lends itself to more future extensions, such as the addition of in-game quests or multiplayer features.

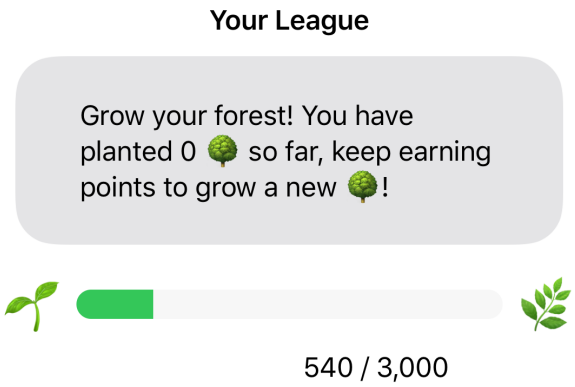


Figure 2.2: The old gamification screen, showing progress from seedling to medium plant.

2.3.3 The rewrite of the old bar chart

The earliest engineering efforts of the individual project were directed towards the rewrite of the bar chart employed to display transport data. There were several reasons for this:

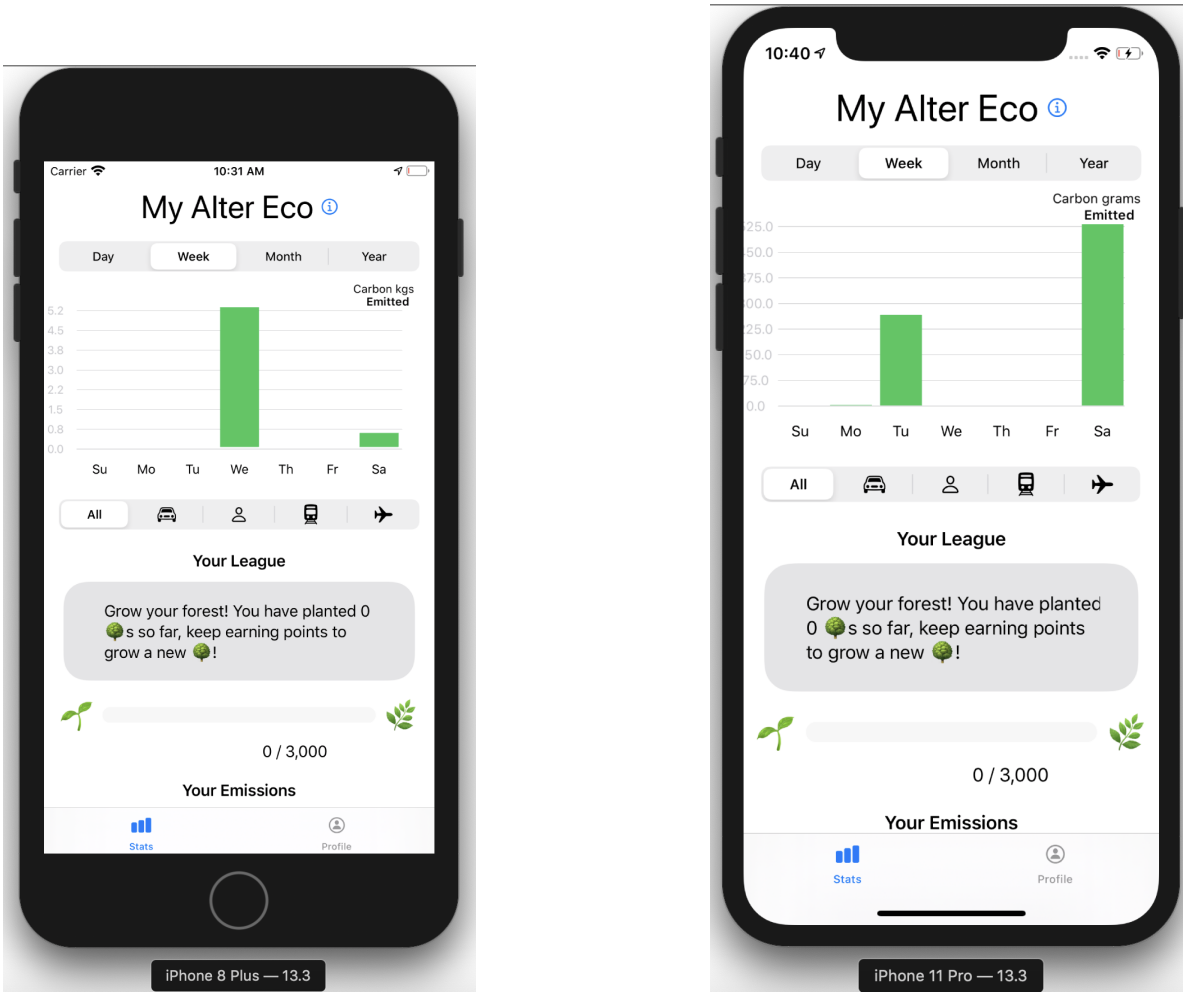
- The original bar chart did not have any kind of architecture, rather it used global variables and global functions to retrieve its internals. At the moment, we instead employ a Model-View-ViewModel pattern (see Chapter 3 for a detailed discussion), which offers a much greater separation of concerns and handling of state.

- A careful reader might notice how in Figure 2.3a, the ticks of the y-axis are inconsistent, and oscillate between 0.7 and 0.8. In Figure 2.3b, we can instead see how the labels go out of bounds. These and numerous other issues were due to poor design choices in the code structure.

- The software behind the chart was overly complex, and was a prime example of the copy-and-paste anti-pattern. As such, it was fragile and hard to modify or extend. As an example, the reader might appreciate that the old chart's core was a nested dictionary of dictionaries of over 350 lines, with hardcoded calls to the database manager to retrieve the relevant data. If any of these calls failed, the whole application crashed. By rewriting it, together with a better error handling, we are now able to manipulate the axis, the gridlines or other elements of the chart with ease.

- Due to the poor architecture of the chart, additional features were added to the backend (specifically to the database management system) essentially as a patch to allow the correct functioning of the app. These features are not needed, and were removed once the chart was rewritten.

In conclusion, by rewriting this part of Alter Eco we improved the organisation of the code and enhanced most of the older features. Therefore, we believe this work to provide an overall better user experience both directly, by having a more reliable system, and indirectly, by facilitating future updates with a clearer architecture and separation of concerns.



(a) On iPhone 8 Plus. (b) On iPhone 11 Pro.

Figure 2.3: Main view for the group project version of Alter Eco. Images from [16].

2.3.4 Summary

To summarize what we just discussed, Alter Eco was born as a group project. By the end of its initial phase, it offered an unpolished transport detection, a somewhat unreliable bar chart for data visualization and a first attempt at gamification. During this individual project, we have focused on improving the existing software whenever possible. When this could not be done, the functionality was preserved but the software component was rewritten. Furthermore, extra functionalities, such as food carbon estimation, have been added. Table 2.2 provides a compact view of the changes between the group and the individual project versions.

Feature	**Differences**
Transport detection	Adjustments and bug fixes. Updated tests.
Gamification	Replaced by 3D virtual forest.
Bar chart	Rewritten to improve architecture and remove bugs. Changed color scheme.
Pie charts	Not present in old version.
Food carbon estimation	Not present in old version.
Profile page	Fixed distortions to the profile picture layout.
User interface	Changed logo. General restyling to include new features, allow device rotation and based on user feedback.
Database	Improved database hygiene and minor refactoring.

Table 2.2: List of differences in terms of features between the group project and the individual project versions of Alter Eco.

2.4 Ethical and legal considerations

This project is based upon collection of personal data, such as locations and dietary habits. As will be discussed in Chapter 3 and Chapter 4, everything is stored locally on the device and it is secured with Apple's default encryption. It is also important to note that no exact coordinates are ever kept, rather they are processed and saved as generic transportation data which only includes timestamps, distance and transport mode. Furthermore, when communicating with third party services (e.g. via API calls) we do not explicitly provide any sensitive user details. We believe Alter Eco to be conforming to GDPR [32], as we give users complete control over their information. The right to be forgotten can be claimed by simply uninstalling the application, causing all data to be erased.

In addition, during beta testing, people external to the project were involved to try the app for free. Invitations were made with a public link, accessible through Apple's *TestFlight* platform [33]. A summary explaining the purpose of Alter Eco and its privacy implications was given both on TestFlight and within the app, so as to inform the users about data collection and removal procedures. Examples of privacy related messages displayed during normal usage of Alter Eco are shown in Appendix C. Finally, in conformance with the departmental policies [34], we included an ethics checklist in Figure 2.4. The reader may notice we stated this project involves collection of information which could be used for discriminatory purposes. Although we believe this to be technically possible, it would be a very unlikely outcome. As previously mentioned, each user has total control over their data. Furthermore, we cannot identify any violent

movements that could make use of Alter Eco. As such, we believe there is no need to worry about any potential misuses.

	Yes	No
Section 1: HUMAN EMBRYOS/FOETUSES		
Does your project involve Human Embryonic Stem Cells?		✓
Does your project involve the use of human embryos?		✓
Does your project involve the use of human foetal tissues / cells?		✓
Section 2: HUMANS		
Does your project involve human participants?	✓	
Section 3: HUMAN CELLS / TISSUES		
Does your project involve human cells or tissues? (Other than from "Human Embryos/Foetuses" i.e. Section 1)?		✓
Section 4: PROTECTION OF PERSONAL DATA		
Does your project involve personal data collection and/or processing?	✓	
Does it involve the collection and/or processing of sensitive personal data (e.g. health, sexual lifestyle, ethnicity, political opinion, religious or philosophical conviction)?		✓
Does it involve processing of genetic information?		✓
Does it involve tracking or observation of participants? It should be noted that this issue is not limited to surveillance or localization data. It also applies to Wan data such as IP address, MACs, cookies etc.	✓	
Does your project involve further processing of previously collected personal data (secondary use)? For example Does your project involve merging existing data sets?		✓
Section 5: ANIMALS		
Does your project involve animals?		✓
Section 6: DEVELOPING COUNTRIES		
Does your project involve developing countries?		✓
If your project involves low and/or lower-middle income countries, are any benefit-sharing actions planned?		✓
Could the situation in the country put the individuals taking part in the project at risk?		✓
Section 7: ENVIRONMENTAL PROTECTION AND SAFETY		
Does your project involve the use of elements that may cause harm to the environment, animals or		✓
Does your project deal with endangered fauna and/or flora /protected areas?		✓
Does your project involve the use of elements that may cause harm to humans, including project staff?		✓
Does your project involve other harmful materials or equipment, e.g. high-powered laser systems?		✓
Section 8: DUAL USE		
Does your project have the potential for military applications?		✓
Does your project have an exclusive civilian application focus?	✓	
Will your project use or produce goods or information that will require export licenses in accordance with legislation on dual use items?		✓
Does your project affect current standards in military ethics – e.g., global ban on weapons of mass destruction, issues of proportionality, discrimination of combatants and accountability in drone and autonomous robotics developments, incendiary or laser weapons?		✓
Section 9: MISUSE		
Does your project have the potential for malevolent/criminal/terrorist abuse?		✓
Does your project involve information on/or the use of biological-, chemical-, nuclear/radiological-security sensitive materials and explosives, and means of their delivery?		✓
Does your project involve the development of technologies or the creation of information that could have severe negative impacts on human rights standards (e.g. privacy, stigmatization, discrimination), if misapplied?	✓	
Does your project have the potential for terrorist or criminal abuse e.g. infrastructural vulnerability studies, cybersecurity related project?		✓
SECTION 10: LEGAL ISSUES		
Will your project use or produce software for which there are copyright licensing implications?		✓
Will your project use or produce goods or information for which there are data protection, or other legal implications?	✓	
SECTION 11: OTHER ETHICS ISSUES		
Are there any other ethics issues that should be taken into consideration?		✓

Figure 2.4: Ethics checklist provided by the Department of Computing at Imperial College London.

Chapter 3

Design

Here we describe the overall features and characteristics of Alter Eco, outlining why they are the way they are. In addition, we discuss our software design principles and architectures.

3.1 Design model

One of the ideas behind the design of Alter Eco is BJ Fogg's behaviour model [35]. According to this model, behaviour is product of three factors which must occur at the same moment. These are:

- **Motivation**, described as the degree of willingness to behave in a certain way. Examples of motivators are pleasure, fear, social acceptance or rejection.
- **Ability**, intended as the capability to perform the behaviour. Sometimes also referred to as *Simplicity*.
- **Triggers**, defined as prompts for the user to take action.

Therefore, a system which is simple or easily accessible, offers continuous triggers and leverages motivators has better chances of altering a user's actions. This is why Alter Eco redirects the user towards its target behaviour by using pleasure as a motivator and a point system as a trigger. Furthermore, triggers can and should be visual: displaying information intuitively can go a long way when it comes to catching a user's attention. For example, Alter Eco changes the colour of some of its charts from blue to red to signify the user is going beyond the average daily emissions. For the same reason, the virtual forest provides a smog effect as to give a feel of what impact the user's actions have on the real world.

It should be noted that Alter Eco chooses to rely mostly on positive reinforcement by the means of more points, unlocked achievements and a richer virtual forest. It does not use point penalties, nor does it directly encourage competition. This is related to

the *ability* described in Fogg's model, as we recognise not all individual actions are a matter of choice: sometimes, work duties or other external factors are beyond a user's control. As such, punishing for what we consider to be 'incorrect' behaviour or directly comparing users' performance against one another is not only likely to backfire, but also morally questionable. On the other hand, positive reinforcement is likely to be an effective strategy when the goal is altering behaviour, as it has been shown in several studies [36, 37].

3.2 System overview

The main purpose of Alter Eco is to orient users towards a more environmentally sustainable lifestyle. At its core, this is done by providing an ecosystem of features that collects relevant data, informs the user and encourages greener habits. As stated in Figure 1.2b and Figure 1.3, transport and food correspond for 25% and 26% of global GHG emissions, respectively. Furthermore, it is clear from our previous discussion of BJ Fogg's behaviour model that we need an engaging system capable of offering triggers which can leverage the right motivators. For these reasons, Alter Eco's ecosystem can be divided into three topics:

- **Transport**: a user's position is tracked over time and their mode of transport (i.e. car, train, airplane, pedestrian and cycling) is inferred based on speed and locations visited. The relevant data is then displayed on a bar chart and on a pie chart, while points are awarded depending on the activity. A user's score acts as a trigger, with more points being given for less polluting transport modes. Points are paired with achievements, which can be unlocked when completing predetermined challenges. Finally, visual clues, such as the bars turning red when the total pollution is above the average UK daily emissions, are employed to direct behaviour.

- **Food**: the user is able to scan a food product's barcode and retrieve relevant information such as quantity and type. Based on this information (which the user can always change if deemed inaccurate), a carbon equivalent measure is calculated and presented both in raw units and as a car ride equivalent in kilometers. If the product is not found in the database, the user can add it from inside the app. All the data is aggregated in a pie chart and points are awarded for the first scan of the day. Users were keen on this feature, with 85% approval rates (Figure 3.2).

- **Virtual forest**: the user can enter their own virtual forest where points can be spent to shop for items (e.g. different types of trees). These can be arranged in space as to allow for personalization. Visual clues are employed to direct behaviour when the user surpasses the average UK daily emissions, with a smog effect changing the looks of the forest. The main idea behind the forest is to

use pleasure as a motivator, by letting users express their creativity and gain something tangible through their efforts.

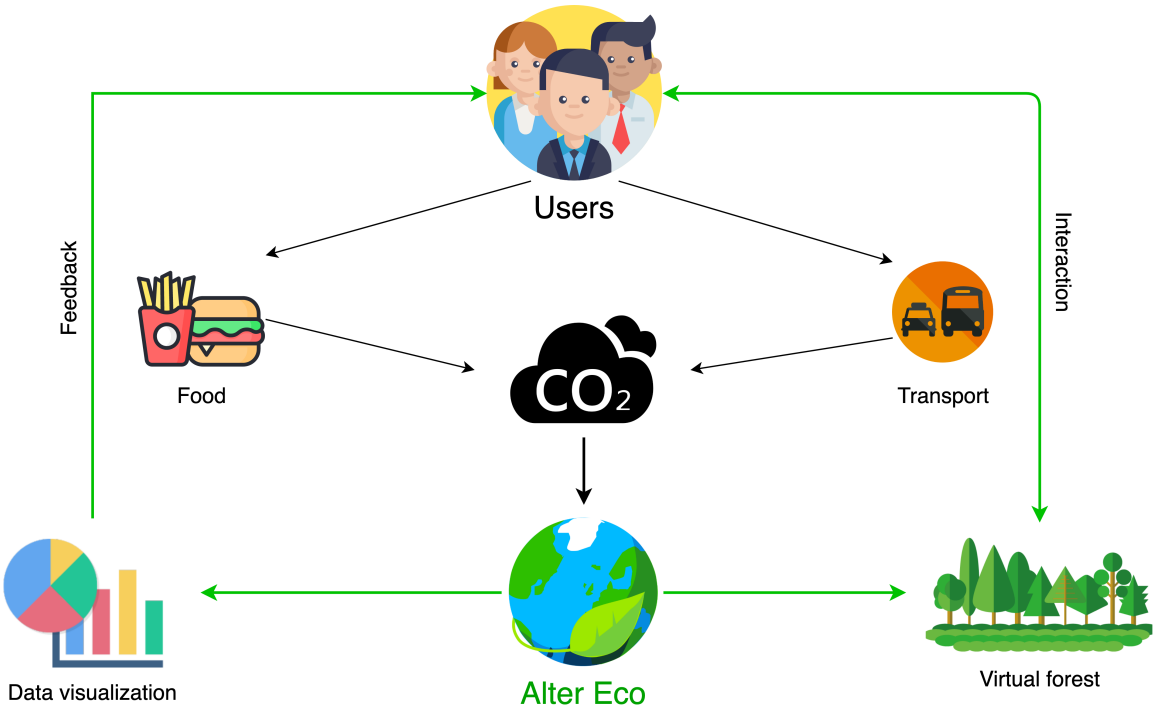


Figure 3.1: Illustration of a typical user interaction with Alter Eco's ecosystem.

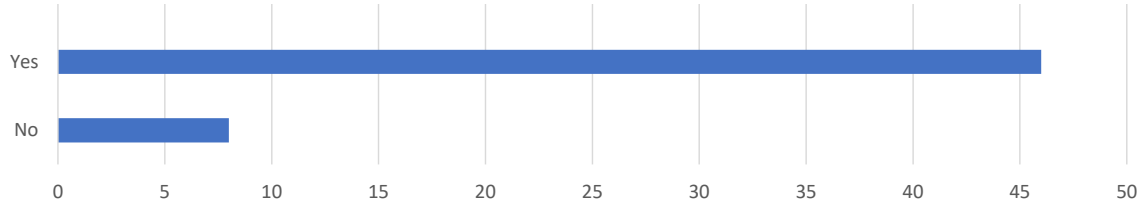


Figure 3.2: Answer distribution in a survey asking '*Would you like to know how much CO2 you produce when you go grocery shopping?*'. A total of 46 out of 54 people replied '*Yes*'.

The Alter Eco's ecosystem incorporates data of different origin, translates it into a factor of interest (i.e. carbon emissions) and unifies the experience with a gamification side (Figure 3.1). In addition, together with the use of visual clues, users are informed in a quantitative manner of how their daily emissions compare with respect to the individual UK average.

3.3 Software design

Since the very beginning, Alter Eco's paradigm has been mostly Object Oriented Programming (OOP), with minimal elements of Functional Programming. There were two reasons for this. First and foremost, the team that originally started the project had the most familiarity with OOP, and there was no incentive to do rewrites or change this approach going forward. Secondly, OOP languages and solutions are widely used and can benefit from a plethora of literature. Nevertheless, although we made strong use of key OOP concepts such as polymorphism, encapsulation and abstraction, we avoided inheritance whenever possible. This was a design choice motivated by the fact that inheritance creates a very tight coupling between subclass and superclass, so that any change on a superclass' members is likely to impact its subclasses as well. Furthermore, favouring composition over inheritance results in fewer implementation dependencies, making the codebase more flexible [38].

In general, and in decreasing order of importance, our aim was to build software that behaved correctly, minimized duplication, maximised clarity and had as few elements as possible. In order to achieve this, we relied on the SOLID design principles described by Robert C. Martin in the early 2000 [39]:

1. ***S**ingle responsibility principle*: each module should do only one thing, or in other words have a single purpose.
2. ***O**pen-closed principle*: software entities should be open for extension, but closed for modification.
3. ***L**iskov substitution principle*: objects should be replaceable with instances of their derived types without altering the correctness of the program.
4. ***I**nterface segregation principle*: creating many client-specific interfaces is better than a big general purpose one, as a module should not be forced to have methods it does not use.
5. ***D**ependency inversion principle*: modules should depend on abstractions, rather than implementation details.

The advantages of SOLID are numerous. For example, to ensure a correct behaviour, unit and integration tests are fundamental. Dependency inversion and interface segregation served the purpose of increasing testability (specifically via a pattern known as dependency injection) by making mocking and decoupled testing easier. Similarly, the Open-closed and Liskov substitution principles reduced duplication, as there is no need for repeated boilerplate codes that checks what method to call depending on the object's runtime type or the arguments provided. This kind of check is either taken care of by polymorphism, or done only once (typically within the internals of the class). Finally, the single responsibility principle makes it easier to reason about the code and pushes towards a greater separation of concerns.

3.3.1 No data centralisation

Alter Eco was designed to process and store sensitive data within the device. In today's mobile market, this is an unusual choice. It is common for apps to be nothing more than a user interface connected to a remote server, which is where most storage and computation efforts occur. Although this kind of design can have advantages such as battery saving and easier updates, it does come with a price.

Consider the issue of privacy. During the design of Alter Eco, care was taken to ensure the app would be GDPR compliant [32]. To do so, no location data is ever stored, and all other information is saved locally on the device. This means we ensure the right to be forgotten by simply uninstalling the app. Similarly, most of the computations occur on the smartphone or tablet. The only exception to this is retrieval of information which is too large to be stored locally. Namely, geographical information (such as underground stations or airports nearby) or food details (like quantity and name of the product) are extracted from third party services, as this type of data needs to be up to date and is likely to be orders of magnitudes greater than what a smartphone can contain. Still, it should be noted that when requests to a remote server are executed, Alter Eco does not explicitly upload any information which identifies the user.

Another potential issue with remote storage and processing is the cost of maintenance. Ideally, Alter Eco should strive to work autonomously for as long as possible, even if the creators are no longer taking care of it. This idea clashes with the concept of a remote server, which has associated costs of financial and technical nature.

3.3.2 Architecture

By far, the most common architecture employed in Alter Eco is the Model-View-ViewModel (MVVM) pattern. It separates objects into three distinct groups:

- **Models**, which are an implementation of the application's domain model. In other words, models provide methods for data access, validation and processing.
- **Views**, which are responsible for displaying data to the user.
- **ViewModels**, which are responsible for most of the business logic concerning the presentation and formatting of data.

Perhaps the most important and distinguishing feature of MVVM is how data flows (Figure 3.3). One ViewModel can incorporate several Models, which it uses to retrieve data when needed or when it becomes available. It then formats the data appropriately and forwards it to the Views. What is innovative is that the engineer does not have to include any explicit references to the Views within the ViewModel. This is made possible by the mechanism of data binding. That is, there is an automatic connection between the data to be displayed and the UI. Instead of holding and managing data flow

among multiple references, Views subscribe to the relevant ViewModels, which in turn simply publish the data. Data binding is not the only way Views have to communicate with ViewModels, as it is common for the latter to offer helper functions, but it is by far the main way in which information is exchanged. This is different to what we find in more traditional architectures such as the Model-View-Controller (MVC) pattern. For example, in MVC, we cannot avoid boilerplate code required to coordinate data flow across models and views, and this can often translate into unforeseen bugs and maintenance overheads. Furthermore, the lack of explicit references within a ViewModel means greater separation of concerns and ease of testing for the presentation layer.

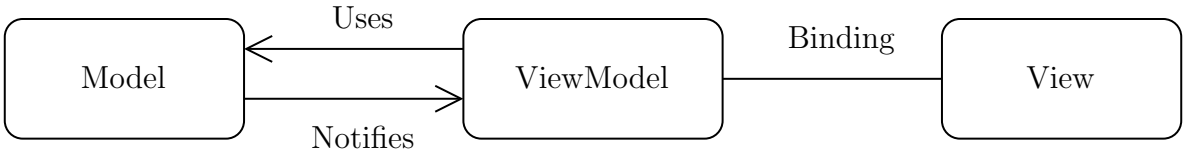


Figure 3.3: Representation of data flow in the MVVM architectural pattern.

Nevertheless, in some circumstances that will be discussed in Chapter 4, we needed to interact with older software components which were designed to work with an MVC pattern. To be precise, we developed some solutions which employed a variation of MVC commonly referred to as Model-View-Presenter (MVP). Here, we have Models and Views as described earlier, but instead of a ViewModel we use a Presenter, which acts as a middle-man (Figure 3.4). Compared to MVVM, the main difference is the lack of data binding mechanisms and the need for the Presenter to hold references to all Views, which are then explicitly updated when changes occur.

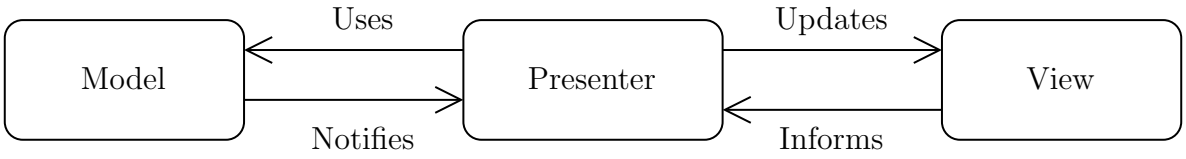


Figure 3.4: Representation of data flow in the MVP architectural pattern.

3.4 Platform

Alter Eco was developed on iOS for several reasons. As explained in Chapter 1, a mobile system is appropriate as it is convenient in terms of accessibility. Having chosen the hardware, the choice then is between the operating systems. Apple's systems tend to have one strong advantage: their uniformity. Contrary to, say, Android devices, smartphones running iOS are limited in hardware and layout variations. This makes it easier to design a user interface and to plan around what features will be available on a given machine. Furthermore, Apple users are known to update their devices at a much faster rate than Android users, with iOS 13 reaching 92% adoption in less than

a year since its release [40]. This is a great advantage for iOS developers, as there is less need to worry about backward compatibility and more space for state-of-the-art technologies. Having discussed the technical advantages behind choosing Apple as our

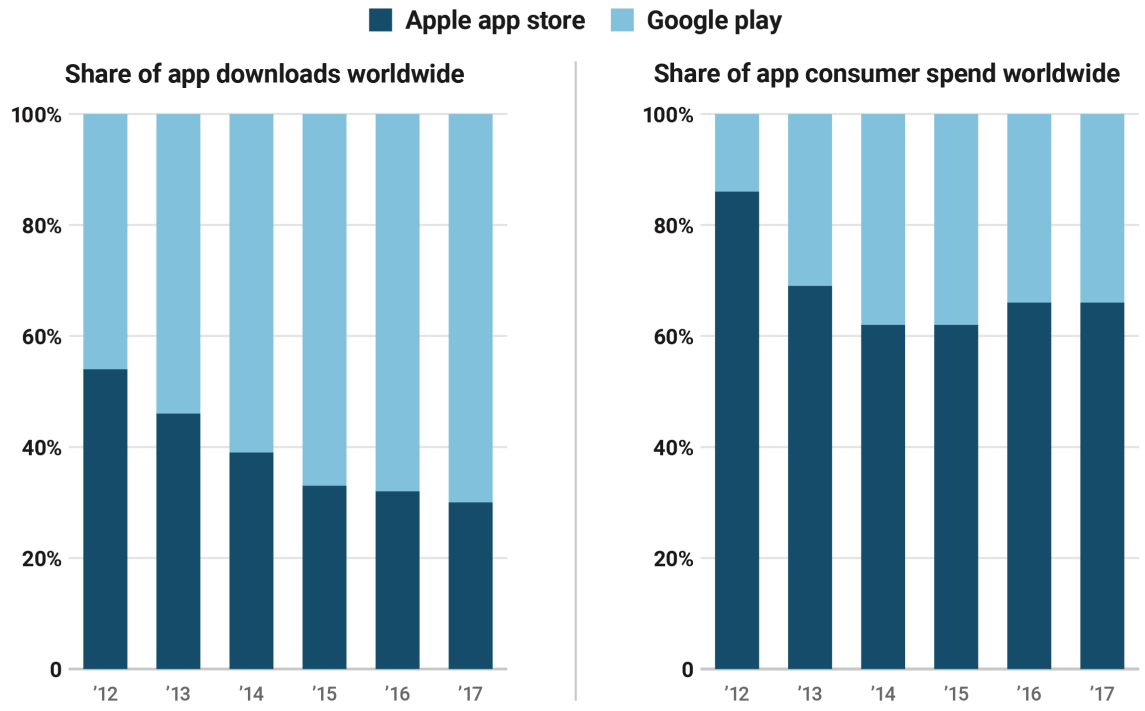


Figure 3.5: Market share for Apple App Store and Google Play (left) and consumer spending (right) from 2012 to 2017. Image from [41]. Original data from [42].

main platform, the reader might be wondering about the huge gap in market share between Android and iOS. It is a fact that Android controls around 75% of the mobile market, compared to the 25% of iOS [43]. From this data alone, it would appear as if Android would be the ideal commercial platform to choose for a new product. Nevertheless, Apple users generate considerably more revenue than Android users [44, 45] (Figure 3.5). This has two major implications. The first is of purely financial nature, meaning that if Alter Eco were to be monetised in the future, concentrating our efforts on iOS is likely to be a more profitable choice. The second is instead socioeconomic, and has much to do with the primary mission of Alter Eco. According to a 2018 study [46] from the National Bureau of Economic Research:

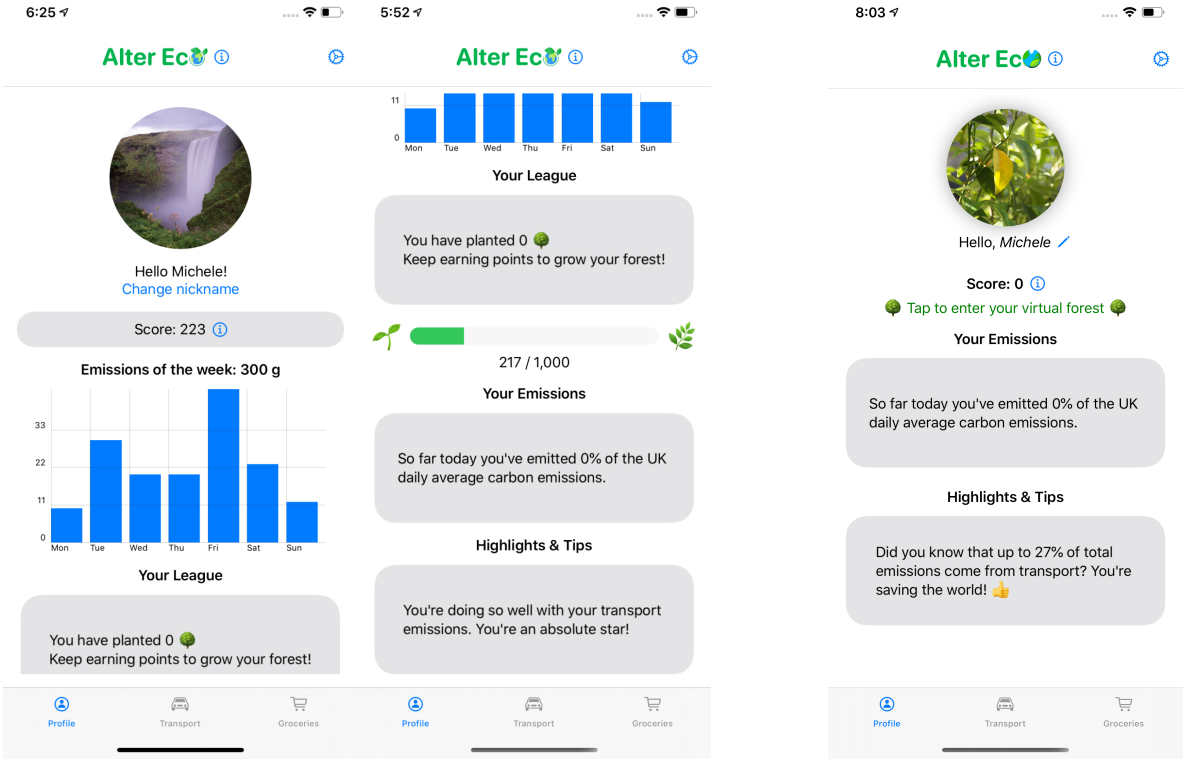
'*Knowing whether someone owns an iPad in 2016 allows us to guess correctly whether the person is in the top or bottom income quartile 69 percent of the time. Across all years in our data, no individual brand is as predictive of being high-income as owning an Apple iPhone in 2016*'.

In other words, owning an iPhone acts as a predictor of status. This is of particular relevance for Alter Eco, as its main goal is to orient the individual towards a more eco-friendly lifestyle. Indeed, changing the behaviour of a single person can have repercussions on others by shifting what is perceived as 'normal'. Importantly, this becomes especially true when the change comes from a high-profile person [47]. It is then clear that targeting a wealthy demographic has a higher chance of triggering a domino-like effect, which makes Apple an ideal platform choice not just financially, but also in terms what Alter Eco is trying to achieve.

3.5 User Interface

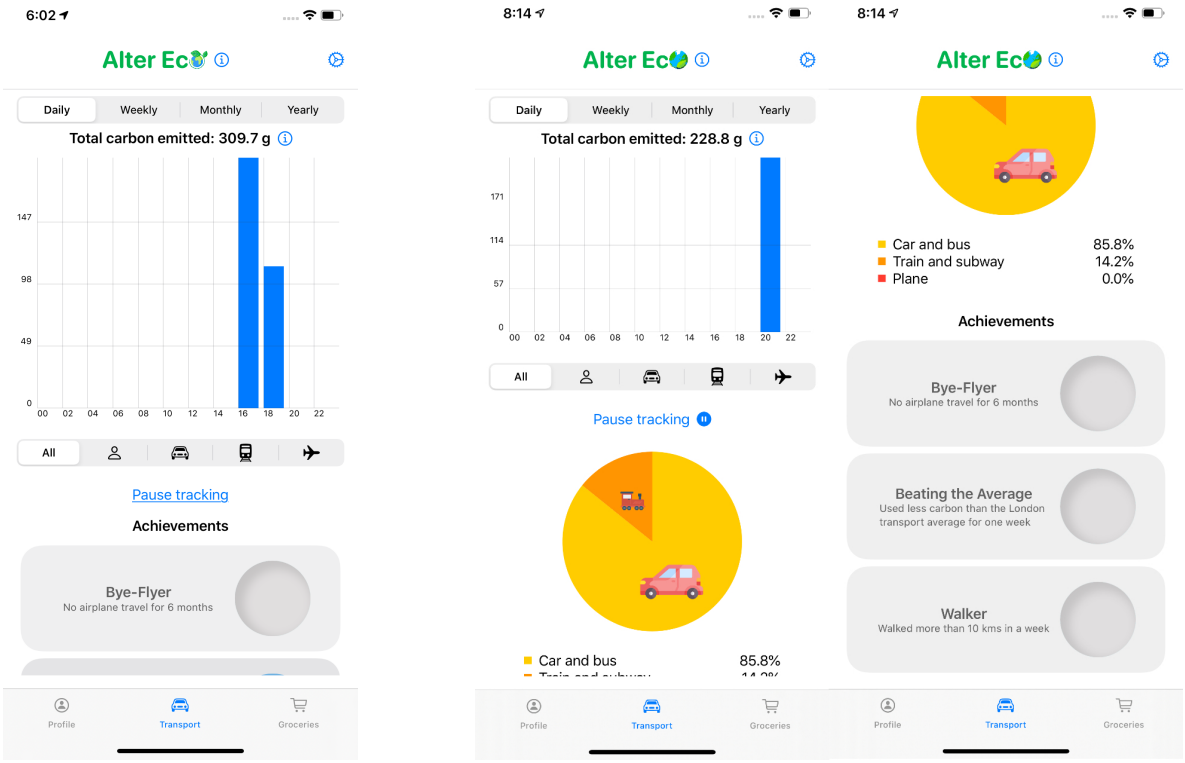
As a significant component of what constitutes the overall user experience is the graphical interface, we built ours by referring to Apple's *Human Interface Guidelines* [48]. We made sure to support all existing layouts and rotation modes, and that users would be able to switch between light and dark mode. We opted for existing UI elements from Apple whenever these were available, so as to create a traditional iOS experience. Furthermore, we used existing apps such as the *Health App* [49] by Apple to draw inspiration about the layout and general organisation of the interface. Nevertheless, our primary source of visual design were inputs from the users, often in the form of surveys.

For example, in one of our surveys we posted screenshots of the profile and transport page, asked users whether they liked them and if they had any suggestions. For both, the approval rate was around 90% from a total of 54 respondents. Nevertheless, some users complained about the crowded and cold look of the UI. Initially we focused on finding a different colour pattern, but then we changed strategy: instead of focusing on a colour theme, we would make use of emojis and introduce a pie chart. Indeed, when asked if a pie chart would be helpful for data visualization, around 95% of the respondents replied with either *'Yes'* or *'Maybe'*. Furthermore, pie charts lend themselves to be colorful to differentiate among slices, which helps reducing the issue of a 'cold' look. A side-by-side comparison of the old and new Profile views can be seen in Figure 3.6, while a comparison for the Transport view is illustrated in Figure 3.7.
Something similar was done to guide the design of the food scanning process. We initially sketched what the views were supposed to look like, and then asked the users to judge whether the design looked intuitive and appealing. Out of 44 people, 63.64% said it was extremely or somewhat easy, while 34.09% said it was neither easy nor difficult. Only one person stated they found it somewhat difficult, and no one described it as extremely difficult. Finally, to further confirm whether the scanning process was intuitive, we asked them to write how they would delete an item from the list shown. The vast majority of people correctly responded with *'by sliding'* or similar answers. An instance of the scanning process is shown in Figure 3.8.



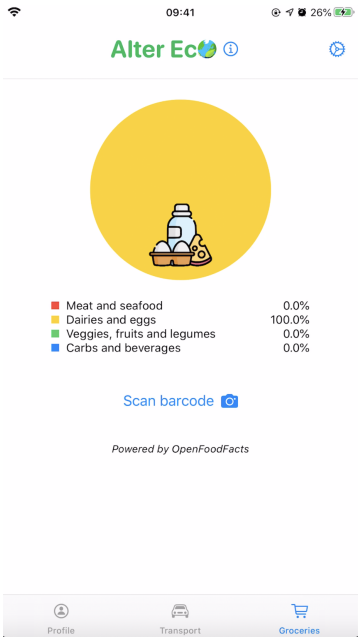
(a) Old version. (b) New version.

Figure 3.6: Comparison between old and new Profile views.



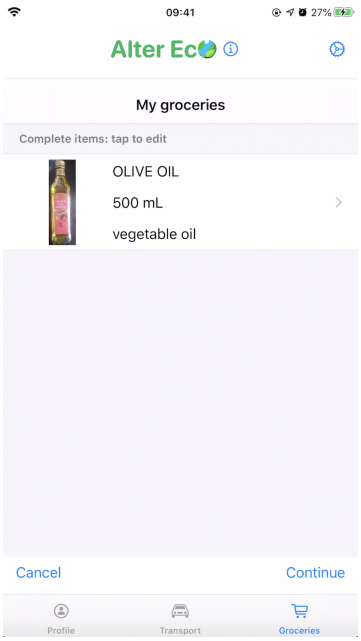
(a) Old version. (b) New version.

Figure 3.7: Comparison between Transport views.

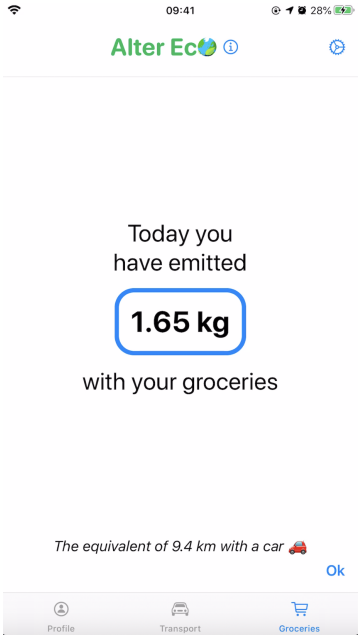


(a) Initial screen.

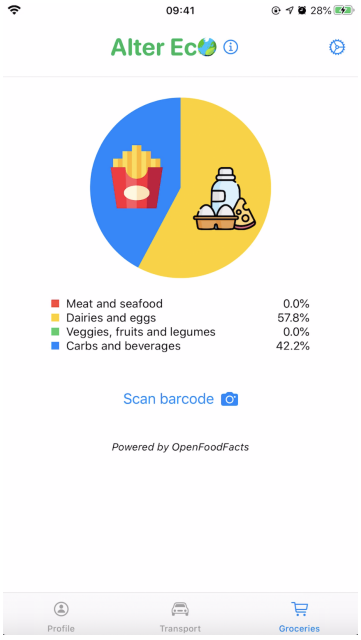
(b) Barcode scanning.



(c) List of scanned items.



(d) Carbon equivalence.



(e) End result.

Figure 3.8: The conversion of a bottle of olive oil into its carbon equivalent. Notice how (**d**) shows the equivalent of a car ride in kilometers.

Chapter 4

Implementation

Here we discuss the implementation details of the app, mostly with regard to the specific technologies employed and the communication among objects.

4.1 Language and UI framework

Our language of choice was *Swift*. According to its official page [50], it is up to 2.6x faster than *Objective-C*. Furthermore, its adoption rate is now considerably greater than its older counterpart [51]. As such, we believe that this was the only reasonable choice for developing Alter Eco. The same cannot be said about which UI framework should be used. As of today, Apple provides two official libraries to build interfaces on their platforms: *SwiftUI* [52] and *UIKit* [53]. Although their aim is the same, there are substantial differences among the two. For Alter Eco, we predominantly make use of SwiftUI. Nevertheless, there are pros and cons for both which need to be addressed.

UIKit is the traditional framework for user interfaces on iOS, and has been around for more than ten years. It is extremely widespread, and thus it is a well tested, documented and reliable technology. Fundamentally, it is based upon an event-driven paradigm, which means it manages user inputs and the life-cycle of the application by triggering specific callback functions known as events. Importantly, UIKit has been designed to follow the MVC pattern. Therefore, we generally have a subclass of *UIViewController* [54] which manages and holds references to a hierarchy of *UIViews* [55], just like we would expect a controller to do with views in MVC.

On the other hand, SwiftUI was announced in June 2019 [56]. Compared to UIKit, it lacks experience, documentation and it is only supported by iOS 13 and above. Nevertheless, it is no coincidence that most code examples of Apple's last developer conference (WWDC20 [57]) were written in SwiftUI: it is the technology Apple is pushing as the new standard. As years go by, most new updates are likely to focus on it rather than on its older counterpart. In other words, SwiftUI is an investment for the

future. That being said, there are fundamental technical differences that made us prefer this framework over UIKit:

- It makes use of **declarative programming**, which makes the code easier to read. Indeed, in declarative programming we only need to state *what* we need, rather than *how* to achieve it. This is in contrast with imperative programming, where algorithms are instead described in explicit steps that depends on the language constructs.
- It hugely **facilitates** the management of **layout and dark/light mode** issues. Most of the logic regarding these two can be trivialized by the use of native framework tools.
- It offers a **live preview** of what we are building, which can automatically generate different layouts for UI testing purposes.
- It makes **views immutable**. That is, views depend on a *source of truth* which identifies the state. Whenever the state changes, the view must be rebuilt to match it. In this approach, what constitutes state and what can cause a change is naturally highlighted.
- Views are structs, which cannot use inheritance but can adhere to protocols (i.e. Swift interfaces). They generally do not violate the interface segregation principle, as they are not forced to inherit unused members. Also, structs are value types, which means they are allocated on the stack rather than on the heap. This translates into a **performance increase** during allocation and read/write operations.
- It is based upon the **MVVM** pattern, which comes with all the advantages discussed earlier.

Some components, such as the libraries used to build the virtual forest or to interact with the camera, required the use of UIKit. Nevertheless, our choice was SwiftUI for the majority of the UI logic, mostly due to the reasons highlighted above. In conclusion, Alter Eco was developed favouring the most recent technologies Apple has to offer. We believe this choice came at the cost of a more reliable documentation, but offered technical and prospective advantages which could not be neglected.

4.2 Permanent storage

4.2.1 Technologies and architecture

As mentioned in Chapter 3, all user's information is saved locally on the device. In particular, app settings are stored with *UserDefaults* [58], Apple's dedicated system to deal with user preferences. UserDefaults uses caching and other optimizations, but is

currently limited to 1 MB. For this reason, heavy storage and retrieval operations are handled via *Core Data* [59], which is Apple's official framework for object graphs and persistent storage. Under the hood, I/O operations are based on *SQLlite* [60] and queries are executed similarly to other relational database management systems. Following the dependency inversion and interface segregation principles, we have abstracted Core Data's functionalities from the rest of the system. The idea is that if, in the future, we wanted to change the technology we use to store data, all of our components would be depending on the interfaces we have created, rather than on Core Data's details. As shown in Figure 4.1, our database management system is divided into *DBReader* for pure reading, *DBWriter* for pure writing, and *DBManager* for a mix of both.

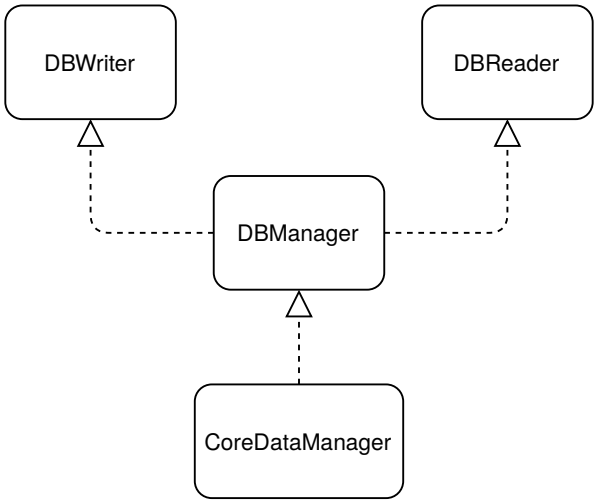


Figure 4.1: UML diagram representing the architecture of the database management system.

4.2.2 Database hygiene

Modern smartphones tend to have several gigabytes available for permanent storage. Although each database entry is less than 1 kilobyte, continuous use of Alter Eco tends to accumulate data. In particular, transport tracking can write several entries per day if the user travels long distances with a car or bicycle. To limit this issue, we periodically schedule background cleaning of the database. For each transport motion, we collect all entries from a month before to the beginning of the very same month. For example, if today is February 22nd, we consider all activities from January 1st to January 22nd. These are replaced by an activity of the same type, having as distance the cumulative distance travelled and representing the overall entry for that time period. Further iterations of the cleaning procedure would then consider this average activity like any other belonging to that month. Thanks to this system we can potentially summarise thousands of older records into a single entry, therefore reducing waste of storage resources.

4.2.3 Security

Albeit no external encryption is applied, Apple devices are secured by default with a technology called *Data Protection* [61], which is implemented with a hierarchy of keys, and builds on the hardware encryption technologies already present on the device. The lack of a centralised and remote data storage, together with the encryption found in each physical device, gives Alter Eco users a very high level of security when it comes to potential data breaches. Finally, as mentioned earlier, if for whatever reason the user wishes to claim their right to be forgotten, all data can be permanently deleted by simply uninstalling the app.

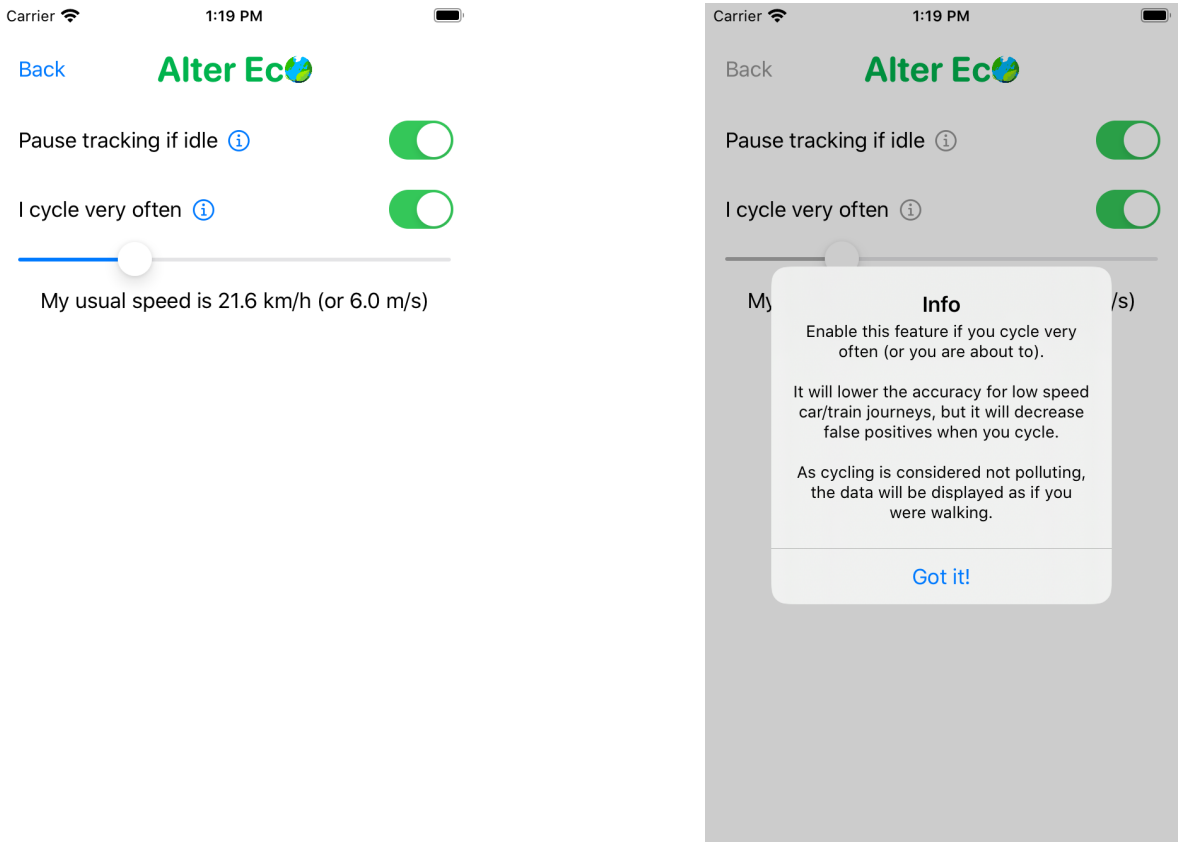
4.3 Transport tracking changes

The modifications to the transport tracking system served both the purpose of ensuring its correct functionality and fine-tuning its capabilities. Here we discuss some of the main changes between the group work and the individual project version. In general, bugs have been identified and corrected, existing functionalities have been polished and new features have been added.

4.3.1 New features

We can identify two main additional features: the pausing functionality and the support for cycling. As will be detailed in Chapter 5, location tracking tends to be high in energy demand. Therefore, we released a feature that allows users to pause and resume it as they wish from the main screen by simply tapping a button (which can be seen in Figure 3.7). This gives users direct control over how much energy Alter Eco can consume. Nevertheless, remembering to open the app can be inconvenient. As such, we have also employed a system that automatically pauses tracking whenever it detects the user has not moved for a while (i.e. approximately 20 minutes on most phones according to our testing). The actual time varies as it depends upon heuristics which are internal to iOS, together with the inputs from several built-in sensors [31].

While the pausing feature was something we decided to implement by inspecting testing data, the addition of support for cycling was instead due to popular demand from our users. This is based upon modification of the speed threshold the app uses to differentiate between automotive and pedestrian activities. The new threshold is set by the user, instead of just being a hidden constant. All data relative to cycling and walking is shown together, as cycling is considered not polluting. To enable this feature, the user must select the appropriate setting in the options, as modifying the threshold comes with the downside of a higher chance of false negatives for car rides at low speeds (Figure 4.2).



(a) (b)

Figure 4.2: (**a**) The options menu of Alter Eco showing the bicycle setting. A user can select their average speed as to minimize the risk of false positives. (**b**) The info alert informing the user of the potential downsides.

4.3.2 Fine-tuning

We can then identify other changes of minor nature, but that are important contributors to the overall user experience. For example, we have improved the way in which we estimate the distance of those activities that are based on visits to specific regions of interest (e.g. stations and airports). Similarly, we changed the way in which we detect airplane flights to be more accurate.

In the past, Alter Eco used to estimate the distance of a ROI activity by relying on average speeds for each type of motion. So that, as expected, airplane flights and tube journeys would have differing velocities. Then, given the timestamps of the two ROI visits and the corresponding average speed, we could compute an estimate of the distance travelled. This particular system is still in use, but with an important difference: we also compute the point-to-point distance between the coordinates of the regions of interest. In particular, this is done by tracing a line between the two points and

measuring the length of its arc. The line follows the curvature of the Earth, and it is assumed to be the shortest possible path not accounting for altitude changes. Finally, this measure is compared to the speed-based estimate. If the point-to-point distance is greater, we take this to mean that the constant underestimated the actual average speed. Conversely, if the speed-based estimate is greater, we take this to mean that the actual route taken was longer than the shortest possible distance.

Finally, as previously mentioned, we have made some noteworthy changes in the way we determine if an airplane flight has occurred. Detection of ROI activities is triggered when two ROIs are visited in a row. Nevertheless, to avoid false positives (such as a user driving past a train station), the system forgets about the first visit if enough activities of the right kind have occurred in a row. Deactivation of the airplane ROI flag is based on detection of a sufficient number of car rides. This has been made more sophisticated by the addition of a speed limit after which speed activities are not recorded. As airplane average speed is considerably greater than our walking threshold, not employing a speed limit could have inappropriately turned off the plane flag due to fake car activities being recorded. To further limit this issue, we have also employed an altitude limit of 6000m, above which locations are not recorded.

4.4 Date handling

4.4.1 Why it is necessary

One of the most noteworthy changes in the overlapping features between the group and the individual project versions is how dates are handled. The older version was mostly developed in London with the team working in a GMT timezone (UTC +0). This, combined with a general inexperience with the iOS environment, led us to mistakenly assume Swift's *Date* [62] would be working with the local time as set on the user's smartphone. In truth, Date is based on UTC, an international standard which has no practical time differences with GMT.

It was only at the beginning of the individual project that London switched to British Summer Time (BST), which is one hour ahead of GMT. Coincidentally, the first beta version of Alter Eco was released outside of the UK in this time period. The issue then became obvious when transport dates started being incorrectly displayed on the bar chart. For example, a user in BST might go for a car ride at midnight local time, which should be displayed on the present day in the appropriate time slot. Instead, the old system would incorrectly interpret midnight BST as 11PM GMT, thus assigning the ride to the day before. The problem also extended to any other edge case of sufficiently big scale, be it weeks, months or years.

Time conversions can be considered a specific instance of a more generic localisation

problem. Sometimes, like in the situation just described, we want to go from a standard to a user's locale, but some other times we want to do the opposite. For example, it is our desire to maintain Alter Eco in English for simplicity reasons. Therefore, care needs to be taken when retrieving the name of days or months, or when dates are processed as strings, so as not to let localisation introduce bugs or inconsistent labelling. Clearly, in an application as data-centric as Alter Eco, incorrect or inconsistent representation of data is a concerning issue which needs to be addressed.

4.4.2 Current system

We could identify two potential solutions to the timezone issue previously described:

1. Convert all dates to the local timezone and store them as such.
2. Store and process all times as UTC, and only convert them to the local timezone when about to present them.

The first solution would work only if we assumed no change in a user's timezone. Nevertheless, this is not always the case: countries change their clock during the year and people travel across time zones. Consequently, we chose the second option. We extended Swift's Date to include some helper functions that allow conversion from the local timezone to UTC and vice versa. These also offer ways to retrieve or process date information as strings, mostly via a thread-safe singleton [63] we named *FixedDateFormatter*. There were several reasons behind the choice of this pattern:

- Converting between dates and strings is a costly operation, and Apple recommends caching a raw formatter object [64].
- A singleton can ensure all operations are handled with UTC.
- A singleton can provide a fixed *en_US_POSIX* locale, which guarantees US English results regardless of both user and system preferences. In addition, this locale is invariant in time, meaning that if the US one day were to change how they format dates, no change would occur in the system [65].

In conclusion, even though handling dates and localisation is a critical and surprisingly complex topic, our tests seem to confirm we have achieved a robust solution that will behave correctly regardless of specific system settings.

4.5 Food carbon footprint

4.5.1 Overview

Associating a food item to a carbon equivalent value in a semi-automated fashion requires a considerable amount of data, which is not easily available. Indeed, at the

time of writing there is no free database of food carbon footprints. Nevertheless, several studies have been done that estimate the average carbon density of common foods and beverages in $kgCO_2eq/kg$. Furthermore, there is an open database named *Open Food Facts* (OFF) [66] which links a barcode to information such as the quantity of the food item, its nutritional value and other useful details. New entries can be added from users and are moderated by the community and by a dedicated team of maintainers. This makes it extremely valuable, as it can be updated with missing products as soon as they are released or discovered. Overall, the way in which Alter Eco computes the carbon footprint of a food product is straightforward:

- A list of barcodes is obtained by communicating with the device camera.
- Relevant food information is acquired from Open Food Facts. Quantity and what categories the product belongs to are of particular importance. Indeed, we use Word Embeddings to link the OFF categories to a list of custom chosen food types stored within the app. If no such information is available, the user can enter it manually.
- The food types are associated to carbon density values taken from the literature and other resources [67, 68, 69, 70, 71, 72]. Having carbon density and quantity of the product, the GHG emissions are calculated and displayed, while the item is saved in the database.

4.5.2 Barcode scanning

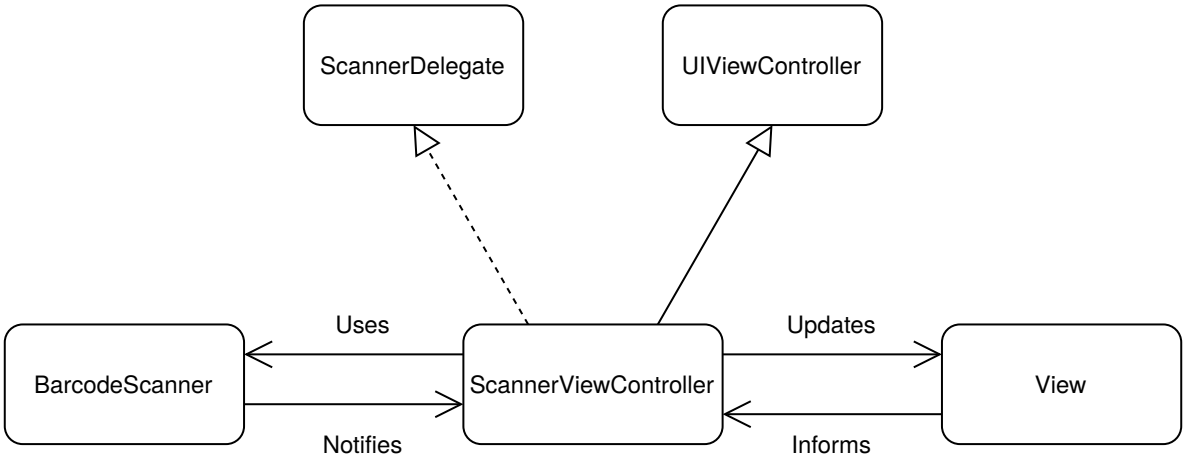


Figure 4.3: UML diagram of the barcode scanner. Note the MVP architecture.

Most of the logic regarding barcode scanning is based upon *AVFoundation* [73], which is Apple's framework to work with audiovisual media. It offers the tools to interact with the hardware and to process a camera feed, such as to translate barcodes from a

video format into its corresponding string representation. Following the single responsibility principle, we created different components to fulfill different purposes: some are responsible for barcode scanning and its subtasks, while others query Open Food Facts. The reason behind the split is not only easier maintenance, but also re-usability of the scanner, which could be, say, employed to scan non-food products (e.g. clothes) in a future release.

The barcode scanner follows an MVP pattern. We have a *ScannerViewController* which coordinates the interaction between the model (*BarcodeScanner*) and the views overlaying the camera feed (Figure 4.3). ScannerViewController extends the protocol *ScannerDelegate* so as to apply dependency inversion and facilitate unit tests. Finally, the user can scan more than one barcode at once, essentially by creating a set of strings which are later passed on and used to query the remote food database.

4.5.3 Remote retrieval and embeddings

At the moment, AVFoundation cannot be used directly with SwiftUI. Thus, to integrate the scanner with the rest of the interface, we built a bridge between UIKit and SwiftUI. This was done by making a *FoodScannerView* which extends *UIViewControllerRepresentable* [74], Apple's given protocol for integration of older components into the newer UI framework. Just like in the rest of our interface, we employed an MVVM pattern. As such, the food scanner has an associated *FoodScannerViewModel* which is responsible for taking the barcodes and handling the queries to Open Food Facts. That is, given a set of barcodes, we make as many API calls as necessary and either process and then pass the information forward or display an error message. To be precise, the formal declaration of the ViewModel includes a generic type which has to conform to ScannerDelegate. This is because our ViewModel initializes the controller when necessary, and because we can then inject a mock during testing to ensure the correct error messages are being reported. A UML diagram of this architecture is available in Figure 4.4.
Queries to Open Food Facts are made via a dedicated object which extends, as usual, the appropriate protocols. In particular, we request quantity and taxonomy information. Due to the open nature of Open Food Facts, quantities come in different units and formats. For example, a user might have input '10lb' for a product and '120.3 g' for another. To avoid most of these formatting issues, we employed the regular expression shown below. It detects a numeric value and a string-represented unit between one and three characters. We determine if the extrapolated unit is correct by making a case-insensitive comparison with a list of allowed units. The regex is as follows:

```
(?<value>[0-9]+(?:[.,,][0-9]+)?)\s*(?<unit>[A-z]{1,3})
```

On the other hand, language issues regarding the taxonomy are taken care of by the OFF API, which offers standardised english categories in a machine readable format.

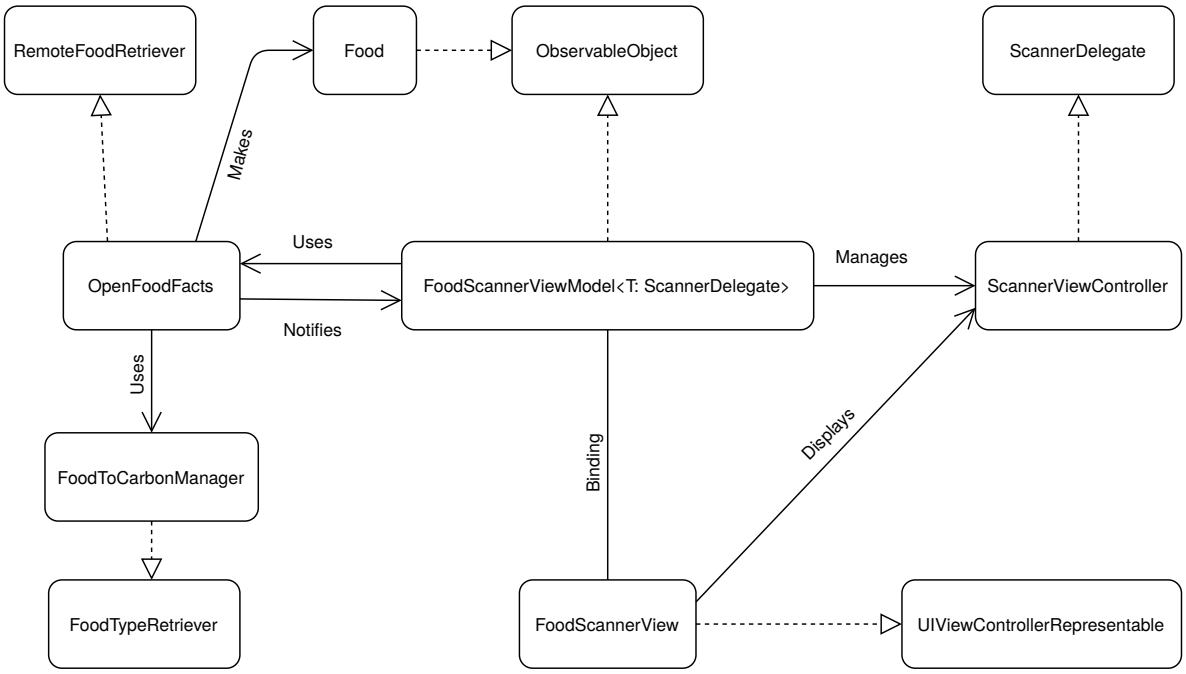


Figure 4.4: UML diagram of the food retrieval architecture.

As mentioned earlier, to connect our internal database of carbon densities to the data we get from Open Food Facts we make use of Word Embeddings. To be precise, we use Apple's english model [75] whenever this is available. Sometimes, for instance if autocorrect is not enabled on the phone, this is not the case. When that happens, and as discussed earlier, we rely on a 50-dimensional GloVe model which was trained on *Wikipedia 2014* and *Gigaword 5* [27]. We give priority to Apple's model, rather than the static alternative, as it is likely to be updated with future releases of iOS. Finally, we link the two databases with the following steps:

1. All categories are preprocessed by removing unwanted substrings (e.g. language prefixes, dashes), duplicates and stopwords (such as 'product' or 'food', which add no valuable information).

2. All words are lemmatized via Apple's *NLTagger* [76] for english.

3. An average vector is computed by using only the words for which an embedding is available.

4. This average vector is compared to the embeddings corresponding to the food types for which a carbon density is available via cosine distance (Equation 2.2). If a food type contains a space (e.g. 'peanut butter'), each word embedding is taken individually and their average is used instead.

5. All food types are sorted based on the cosine distance, so that they are in decreasing order of semantic similarity.

The first step is handled by an *OpenFoodFacts* object, while the rest is responsibility of a *FoodToCarbonManager* (refer to Figure 4.4 for their role in the overall architecture). More importantly, at the end of this process we have obtained the representation of a food product from its barcode. In particular, we have associated its specific characteristics (e.g. quantity, name etc.) to a hierarchy of available food types and their respective likelihoods. As such, unless not enough data was available in the remote database, we now have most of the information necessary to compute the carbon equivalent associated with the given product.

4.5.4 Food to carbon conversion

At the end of the retrieval stage, we are left with one or more *Food* objects that are likely to contain information such as quantity and type. These are fed into a *FoodListViewModel*, which groups them appropriately depending on whether they have all, part of, or none of the information necessary to compute GHG emissions. The user can then input any missing or incorrect information via a *FoodListView*. Products which were not found in the remote database can be added as well, and are then uploaded to Open Food Facts for later use. Technically, quantity and type of food are essential to compute a carbon footprint. Nevertheless, if no quantity is available, Alter Eco defaults to 100 grams and highlights the weight in orange to draw the user's attention and potentially let them enter the correct amount.

Once all information has been verified, Alter Eco can start the conversion with the same FoodToCarbonManager we mentioned earlier. By itself, this is a very easy step: given the carbon density for the most likely type, we multiply by the weight of the food and obtain the estimated emissions. Nevertheless, there are still two minor complications:

1. The units retrieved from OFF can vary considerably, but all data relative to the carbon densities is in $kgCO_2eq/kg$.

2. The product might be a liquid, and its quantity a unit of volume, rather than a unit of mass (e.g. 100 mL for a bottle of water).

We solved the first issue by relying on Apple's *Measurement* [77], which is capable of converting among units of the same dimension (e.g. from a mass unit, like lb, to another, like kg). The second issue was resolved by creating a small list of representative densities for common liquid foods and beverages in kg/L (i.e. water, alcohol and vegetable oil). Once again, Word Embeddings are employed to associate the food type to one of the densities. All units of volume are converted to liters, and the computation can proceed as expected. Finally, a UML diagram of the overall architecture can be seen in Figure 4.5.

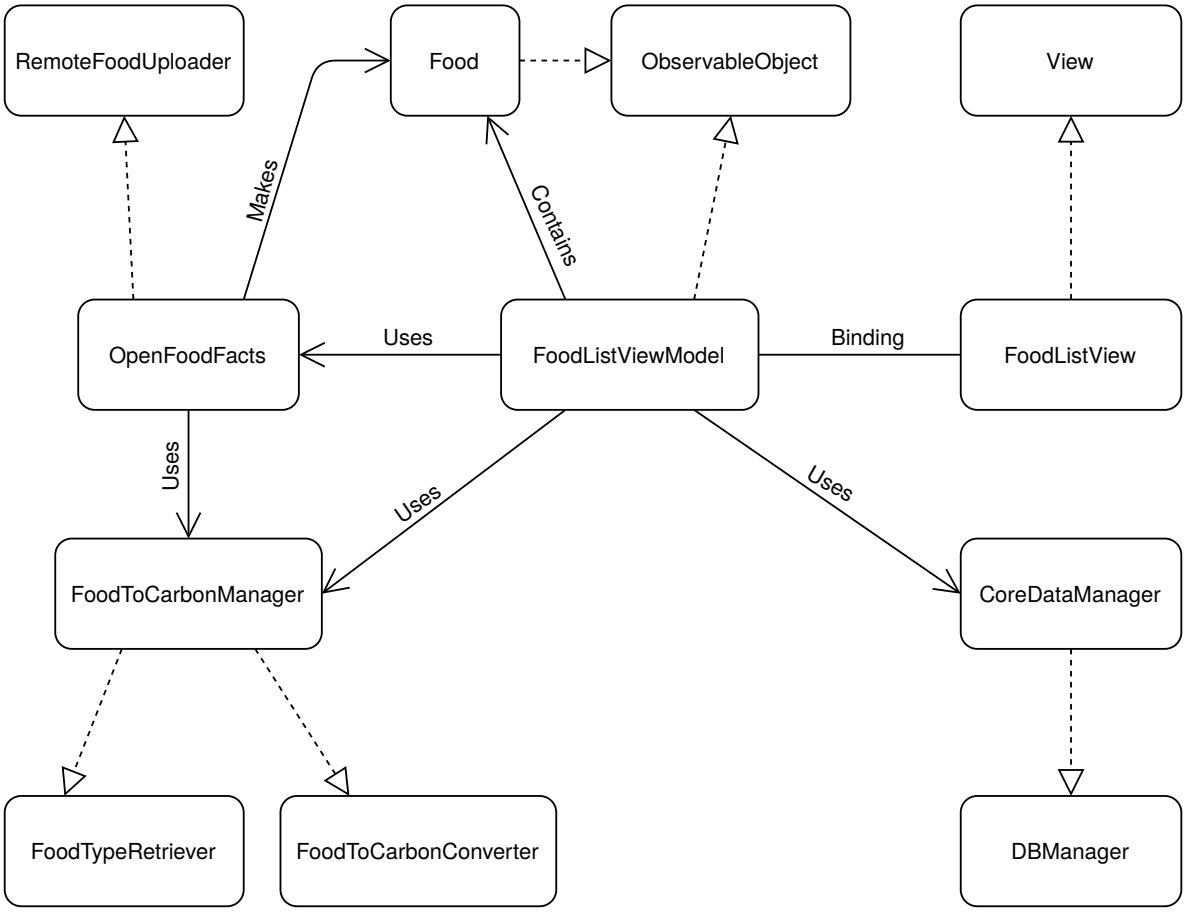


Figure 4.5: UML diagram of the architecture responsible for food to carbon conversion. Sub-views of FoodListView are not shown.

4.6 Virtual forest

4.6.1 Main technologies and architecture

The gamification side of Alter Eco is concentrated in the virtual forest. This is a small 3D world in which a user can place and organize forest items such as trees. The main idea is to allow the user to earn and spend points in a shop of forest items. As such, we can immediately identify the need for a view, which renders the world, and a model, which interacts with the database.

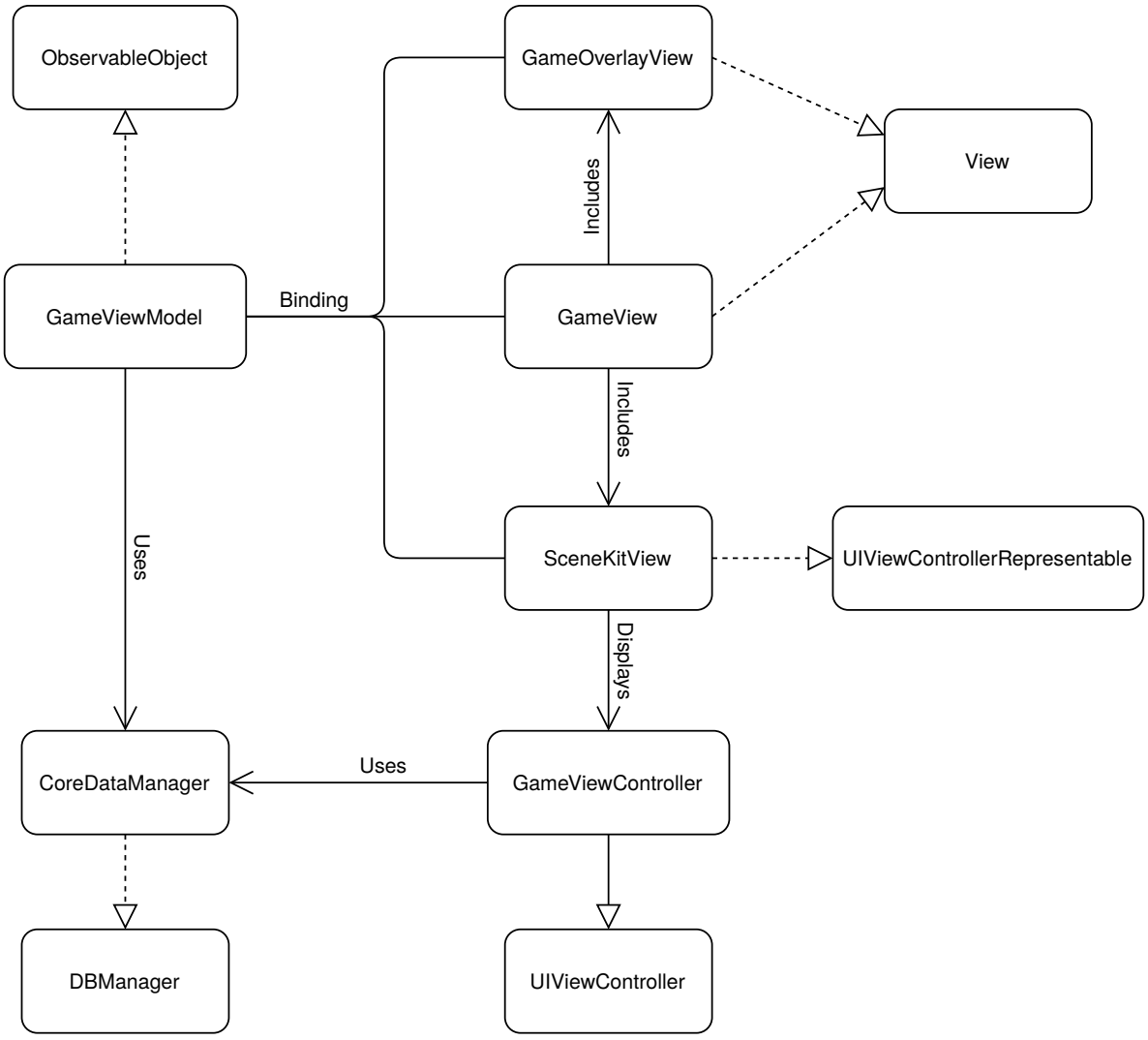


Figure 4.6: UML diagram of the architecture of the virtual forest. Data objects representing shop or forest items are not shown.

We used Apple's *SceneKit* [78] as our 3D graphics framework. It provides an API to perform rendering and handle physics. It can be combined with Xcode's editor

[79] to create basic 3D resources and organize them into a scene. Furthermore, the framework supports COLLADA files [80], which makes it extendable to more detailed assets generated with other popular design softwares. At the time of writing, SceneKit is only directly supported by SwiftUI on iOS14, which is currently in beta and will be released later during the year. As such, we have once again relied on UIKit and an MVP architecture to handle a *GameViewController*, which is then integrated into the rest of the UI through a *SceneKitView*. The menu overlaying the 3D world is generated as usual in SwiftUI and, together with SceneKitView, it is embedded into a compact *GameView*. Finally, the latter is linked to a *GameViewModel*, which coordinates user interactions and handles some database operations. Please refer to Figure 4.6 for a graphical representation of the architecture.

4.6.2 SceneKit organization and physics

Content in SceneKit is implemented as a hierarchical tree of nodes, which are structural elements that contain position, orientation and graphical information. Each scene has a root node that defines a coordinate space, and the rest of the nodes populating the world. The purpose of GameViewController is now clearer: it coordinates user interaction with the scene by adding, removing or changing the state of nodes. In particular, it converts *ForestItem* data objects, which are loaded from the database, into the appropriate nodes.

SceneKit is a powerful framework. When configured correctly, it requires little effort to manage details relative to its internal objects. It comes equipped with a physics engine that automatically responds to collisions, application of forces, torques and similar events. Although the virtual forest is mostly a static world at the moment, the physics engine comes in handy as it allows us to avoid invalid renderings. For example, enabling collisions between certain nodes and the floor prevents trees from falling due to gravity. Similarly, enclosing the visible world inside an invisible cage prevents the user from accidentally placing items into the void. The physics engine essentially acts as the backbone structure which holds the virtual forest together. Achieving the correct rendering of each node without it would be exponentially harder, as we would have to manually check for each potential surface intersections in a 3D space. Furthermore, it lends itself to interesting features (such as the smog effect which will be discussed later) and potential expansions.

It is then worth mentioning how this engine handles collisions within Alter Eco. All nodes can be optionally equipped with a physics body. SceneKit allows us to specify parameters such as friction, mass and other physical quantities. In addition, we can decide whether the body should be static (i.e. it does not react to collisions or forces), kinematic (i.e. collisions and applications of forces are not rendered, but we get notified with an event) or dynamic (i.e. all physical phenomena are rendered according to the engine simulation). SceneKit determines which node can collide with what by

using bitmasks. To be specific, each body is assigned to a numeric category. When two bodies come into contact with each other, a bitwise AND operation is executed. If the result is nonzero, then a collision has occurred and the physics engine takes care of the simulation. For example, given a node with category 1, another node with category 3, and assuming 4 bits for illustration purposes:

$$1_{10} \,\&\, 3_{10} = 0001_2 \,\&\, 0101_2$$

$$0001_2 \,\&\, 0101_2 = 0001_2 \neq 0_{10}$$

As the result is nonzero, a collision has occurred. On the other hand, given a node with category 1 and another with category 2:

$$1_{10} \,\&\, 2_{10} = 0001_2 \,\&\, 0010_2$$

$$0001_2 \,\&\, 0010_2 = 0000_2 = 0_{10}$$

Since the result is zero, there was no collision. This system gives great flexibility for handling physical interactions. Nevertheless, in Alter Eco we only needed to assign the same category number to the forest items, the floor and the invisible walls. Floor and walls are static, meaning they themselves do not move when collided with, but forest items can indeed be moved and must be prevented from overlapping with other nodes.

4.6.3 Shop and Edit Mode

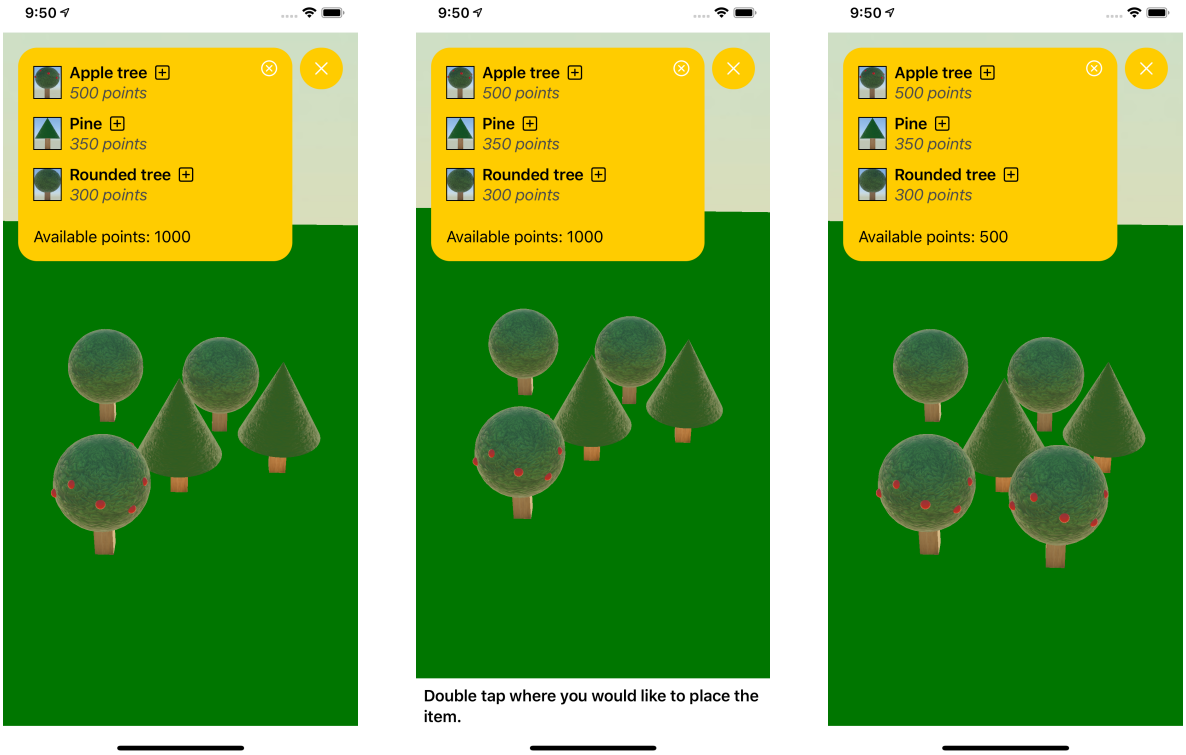
Forest items can be acquired with points through a shop. This requires a verification system to ensure the user has enough points to continue, and to handle the rest of the transaction. Fundamentally, the whole process is constituted of several steps:

1. The user selects an item by tapping on GameOverlayView (Figure 4.7a). This confirms with the ViewModel that there are enough points in the database, and if so passes a *ShopItem* to the ViewModel.

2. GameViewModel then publishes the selected item. SceneKitView is notified of the change and tells the controller to allow item placement.

3. Once the user double taps where he would like to position the item (Figure 4.7b), the controller checks once more if there are enough points, and if so subtracts the cost of the item from the total. In case of database failure, the transaction is cancelled. Finally, a ForestItem is created, added to the 3D scene and stored in the database (Figure 4.7c).

At the end of these steps, the user can interact with a new node within the virtual world. Essential information such as node internal name, unique identifier and coordinates are stored in the database so as to give permanence to the virtual forest and its organization.



(a) (b) (c)

Figure 4.7: Series of screenshots showing the purchase of an apple tree within the virtual forest.

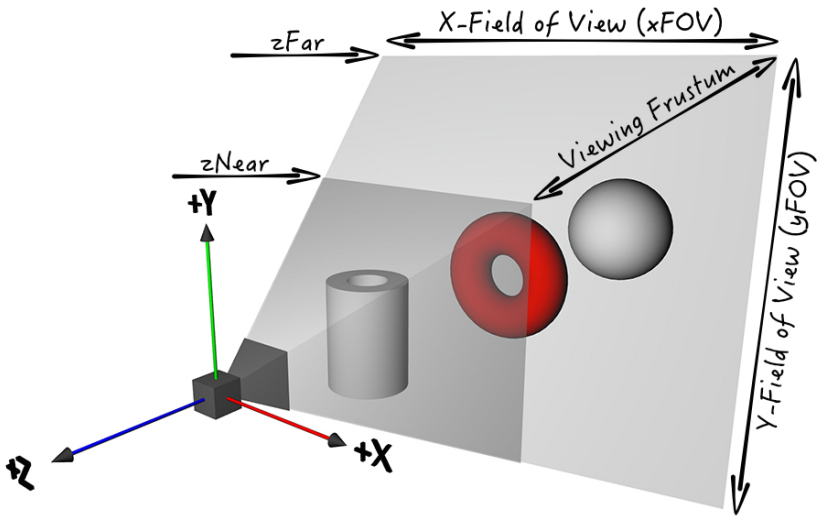


Figure 4.8: Representation of the camera view in SceneKit. The depth (Viewing Frustum) can be regulated by changing the zNear and zFar parameters, and determines which nodes the user can interact with. Image from [81].

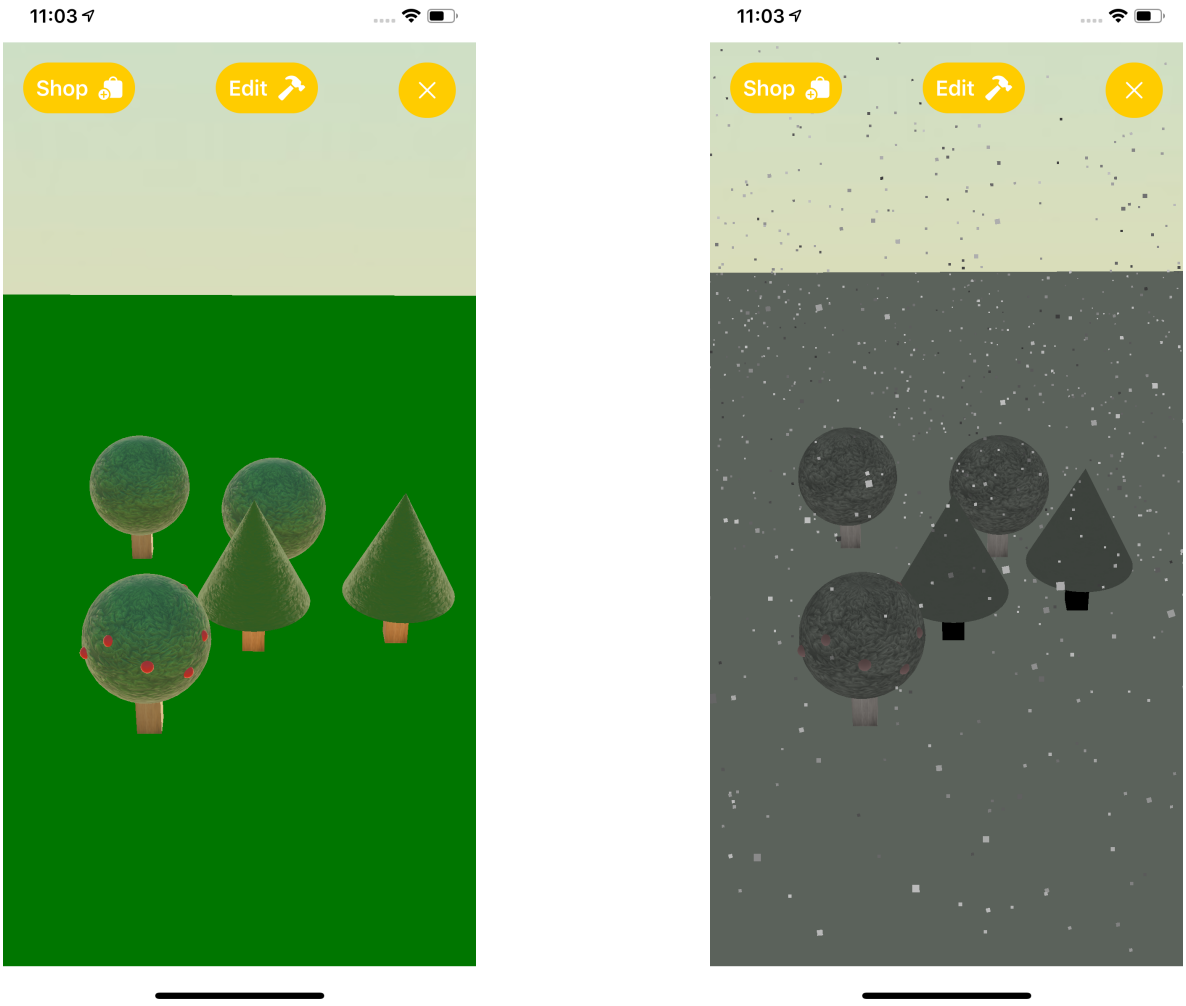
The user is not only allowed to purchase new items, but they can also interact with them. In particular, they can enter *Edit Mode* and change the position of one or more nodes. As mentioned before, the virtual forest is a 3D world. Nevertheless, screens are two-dimensional. To allow the user to select an item by simply tapping on it, we rely on a mechanism internally handled by SceneKit and known as *hit test*. It consists of projecting an imaginary ray from the object to the screen, effectively mapping 3D coordinates into 2D (Figure 4.8). Technically, it is possible for multiple items to be mapped to the same plane coordinates. For this and for performance concerns, we limit our search to the first visible match. Given the coordinates, all we have to do is track the change in finger positioning across the screen to determine a variation in 2D location. If a change has indeed occurred, we unproject the 2D point back to 3D and move the node by the difference between the old and the new position. Once the finger leaves the screen, the controller updates the database accordingly. As a final note, the physics engine and the collision mechanism explained before act as a safety net against incorrect coordinates, ensuring that problems related to nodes going out of bounds or being misplaced are minimised.

4.6.4 Graphics and smog effect

Most of the available nodes where created using Xcode's built-in editor. Each of them is placed within its own scene file and has an internal name. Every forest item is associated to a unique identifier, and it is then linked to its graphical representation by its internal name. This allows us to create multiple instances of visually identically, but ontologically different, nodes. Graphically speaking, nodes are characterised by rendering properties such as geometry and shading. We drew all nodes with a cartoonish style for two main reasons:

1. We wanted the forest to appear friendly.
2. We wanted to keep the design overhead as small as possible.

As such, Alter Eco nodes are geometrically and visually simple (Figure 4.9a). Their textures and light reflection properties are based upon free-to-use images and normal maps which were kindly made available online [82]. Most of the time, the same node has the same appearance. Nevertheless, there is one particular instance where this is not the case: the smog effect. This is triggered when the user emissions reach 100% or more of the UK daily average. When that happens, GameViewModel sets a dedicated flag, which is then detected by SceneKitView and visually shown in GameViewController. A smog effect is then recreated by employing a particle system and an overlaying gray filter upon the whole scene (Figure 4.9b). The smog only goes away if in the last 24 hours the user's emissions have gone below the average, and it was intended as a trigger to alter behaviour.



(a) Normal view.

(b) Smog effect.

Figure 4.9: Comparison between the virtual forest under normal circumstances and with the smog effect on.

Chapter 5

Evaluation

Here we discuss if and how Alter Eco achieved the aims described in Chapter 1. The evaluation is considered both from a technical point of view and according to the beta testers' feedback.

5.1 User evaluation

5.1.1 Methods

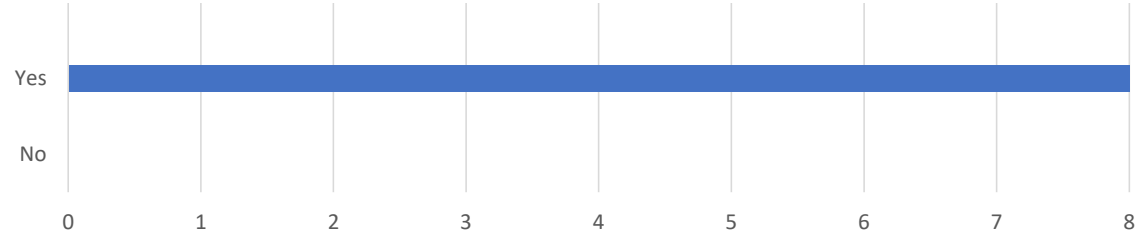
As previously mentioned throughout this paper, users have been involved in the development of Alter Eco since the very beginning [16]. During the individual project stage, we released a preliminary version on TestFlight [33], Apple's official beta testing platform. We were able to receive feedback in the form of private communications (e.g. through e-mails or in person), TestFlight reports and surveys. The latter method was especially useful because it provided quantitative information for targeted questions. We mainly developed two types of surveys:

1. An opinion poll open to everybody, be them Android or iOS users, which mainly focused on interface and perceived user experience.

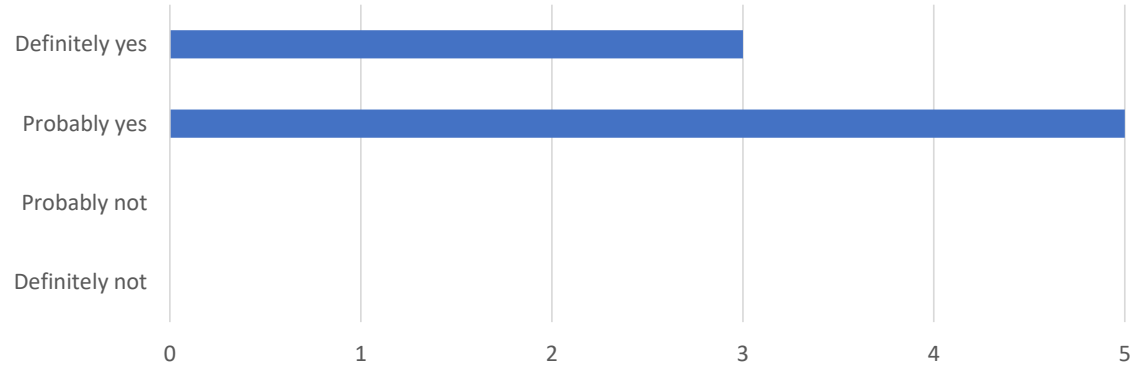
2. A closed questionnaire only for regular beta testers, as to get the opinions of those who actually used Alter Eco.

Not surprisingly, the first type lent itself to a much wider sample size, often composed of around 50 respondents. Chapter 3 provides a detailed discussion of how we used the users' opinions to validate and improve both our interface and general design decisions. On the other hand, the second survey was considerably more limited, including the answers of only 8 people. Nonetheless, these are iOS users who tested Alter Eco regularly during the summer, and as such we take their opinions into consideration. In this section we will focus on the beta testers' feedback and how it relates to our original aims.

5.1.2 Results



(a) *'Do you think that using Alter Eco made you more aware of your carbon emissions?'*



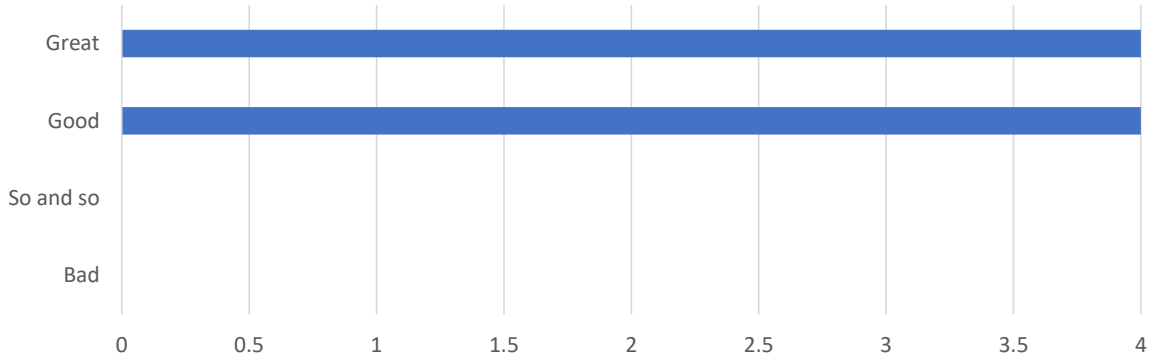
(b) *'Can you identify any concrete eco-friendly behaviours that you have adopted thanks to Alter Eco?'*

Figure 5.1: Survey results related to changes in behaviour and carbon awareness.

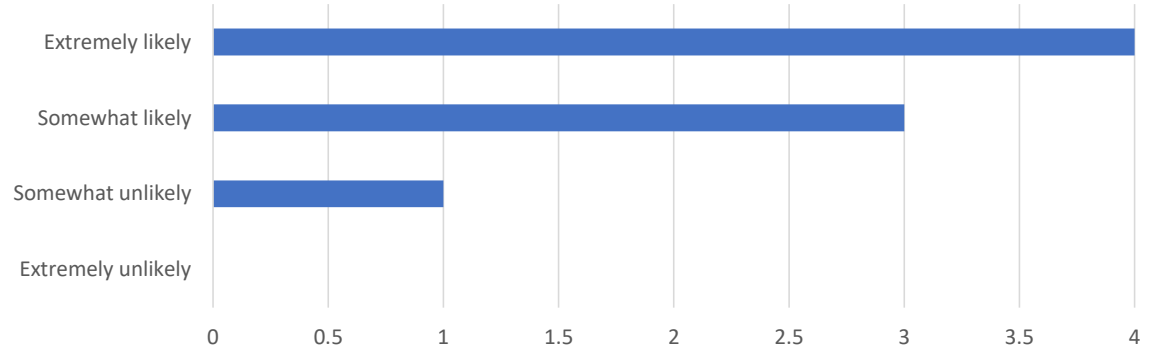
As the reader might remember from Chapter 1, the main aim of Alter Eco is to create habits that are long-lasting and environmentally friendly. We determined this would only be possible through an accessible, engaging and intuitive system that was able to orient a user's behaviour. Consequently, the survey was set to investigate if these desirable characteristics were indeed achieved.

All testers replied affirmatively when asked if Alter Eco was overall accessible, while 7 out of 8 said they found it also intuitive. Perhaps more importantly, all users declared Alter Eco made them more aware of their carbon emissions (Figure 5.1a). In particular, when asked if they could identify any concrete and eco-friendly behaviours adopted thanks to the app, 5 replied with 'probably yes' and the remainder with 'definitely yes' (Figure 5.1b). Furthermore, all users described their experience as either 'great' or 'good' (Figure 5.2a). We believe these results to corroborate the correctness of the design, especially in regard to the aims previously outlined. Interestingly, when asked how likely they were to continue using the app, 7 replied with either 'somewhat likely' or 'extremely likely' (Figure 5.2b), a result that would seem to suggest Alter Eco is on the right path in terms of user retention.

Finally, we asked some open ended questions, mostly enquiring about what they liked of the current functionalities, what extensions they would like to see and what extra features they might want. A list of all the answers and questions can be found in Appendix A. Generally, people stated appreciation and interest for all features, with the transport tracking being considered the most useful. In terms of improvements, most of the suggestions were about the virtual forest. Our beta testers expressed desire for additional items, quest and achievements. In particular, different users claimed to wish for animals to be added. Some interesting ideas were also proposed as new features: some people stated they would like it if they could use the app to buy eco-friendly products, or to make donations and investments. Even though partnership with a third party would be necessary, this is something to consider nonetheless, and that we will continue discussing in Chapter 6. In conclusion, we believe our testers shared a general good opinion of Alter Eco. They identified areas with potential room for improvement, and confirmed that our initial aims have indeed been fulfilled.



(a) *'Overall, how would you describe your experience using the app?'*



(b) *'How likely are you to continue using Alter Eco in the future?'*

Figure 5.2: Survey results related to user experience and retention.

5.2 Technical evaluation

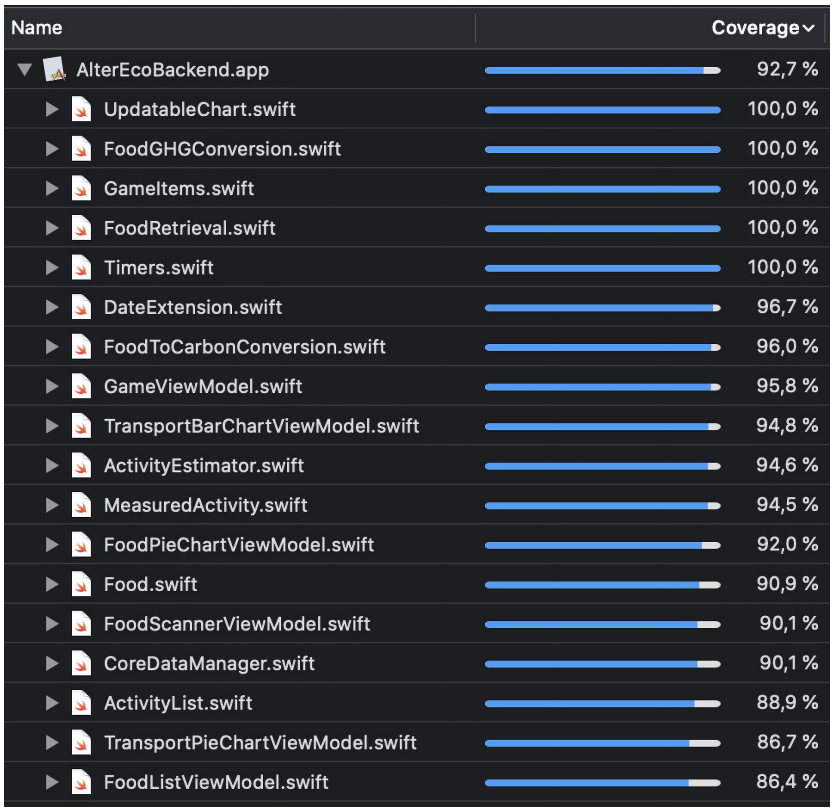


Figure 5.3: Code coverage generated using Xcode's Test Navigator. Average coverage from over 80 unit tests is 92.7%.

5.2.1 Unit tests

To ensure the system behaved correctly, we made conspicuous use of automatic unit testing. The application of dependency inversion facilitated the testing process by allowing a technique known as mocking. Generally, software components rely on each other, thus creating dependencies. Mocking consists in simulating these dependencies via so-called mocks. A mock is a custom-made object or structure which is embedded into the component we wish test. Its purpose is to collect information regarding data flow and communication between the tested object and its dependencies. At the present day, Xcode does not offer any mocking framework. As such, we created mock classes for pure testing, and we included them in the component to test via dependency injection [83]. The mock would then record which methods were called, how many times and with what arguments. So, dependency inversion served the purpose of decoupling software components from their dependencies, while mocking allowed the collection of relevant parameters. Our unit testing mostly concerned the model and business logic of

the application, excluding views and other purely graphical components. We obtained an overall 92.7% coverage (Figure 5.3).

Although 100% coverage can be beneficial, sometimes it is not necessary. For instance, Swift is a language that natively supports optional types. That is, types which can be nil at runtime and that have special language constructs and compile-time checks. In particular, one of the main reasons we did not aim for 100% coverage is the use of the nil-coalescing operator [84]. This language construct allows to specify a default value to use when an optional type is indeed nil (e.g. an empty string could be used to coalesce an optional string retrieved from a database). As such, we made sure to provide default values for some variables even where the program logic would dictate they are never supposed to be nil. This is essentially a safety net for technically impossible situations, and having a dedicated unit test for every instance of the coalescing operator would reduce to testing the constructs of the language, more than the behaviour of the program. In conclusion, we believe our test coverage confirms the correctness of Alter Eco to a more than adequate degree.

5.2.2 End-to-end tests and energy consumption

Nevertheless, testing in isolation is not sufficient. A system needs to behave correctly not only in terms of its parts, but also and mostly as a whole. For this reason, we continuously performed end-to-end tests whenever a new update was released. These tests were conducted by a human, who would manually provide inputs and verify outputs. A special note should be reserved for the testing of the transport tracking system, which was conducted via Xcode's simulator [85]. Our simulations were mostly concerned with emulating different speeds and transport types, such as walking, airline flights and car rides. All test journeys are files encoded with the *GPS eXchange Format* (GPX), which is an XML-based protocol that includes details such as locations and timestamps. The reader may visualize their content as what appears on the screen any modern GPS navigator. To create these files there are many possible methods, but in our case we:

- relied on Google's *My Maps* [86] to generate a route by specifying mode of transport, origin and destination;
- forwarded the output to *GPS Visualizer* [87], which converted the route into a GPX file containing coordinate-based waypoints;
- used *GOTOES* [88] to add timestamps based on an average speed.

Xcode uses these files to simulate motion by interpolating the rate of speed according to the locations and timestamps provided. An additional layer of realism comes in as an extra, since the interpolation causes the average speed recorded on each simulation to vary slightly.

Generally, the simulations made us spot and correct a non-trivial amount of mistakes and allowed us to identify edge cases we had not considered. That being said, nothing can completely replace real life usage. This is why we also personally verified that everything worked as expected. For example, we checked plane detection and estimation by using Alter Eco in our international flights. Figure 5.4 shows the output we obtained for July. During that month, we did a round trip from London (STN) to Cagliari (CAG), and another round trip from London (STN) to Tenerife (TFS). The results shown can be confirmed to be correct by the use of other available calculators. For example, [89] and [90] give a total of 1.58t and 1.4t, respectively, which average to 1.49t.

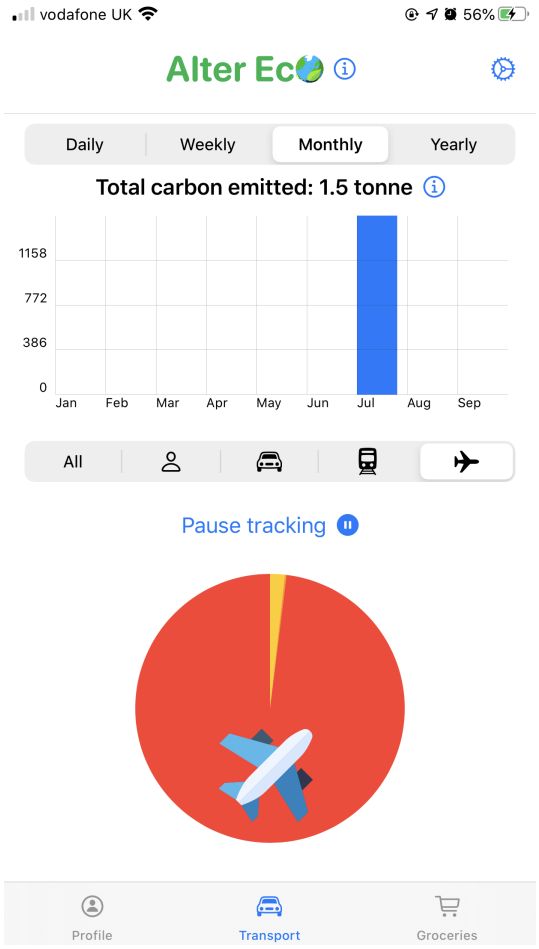
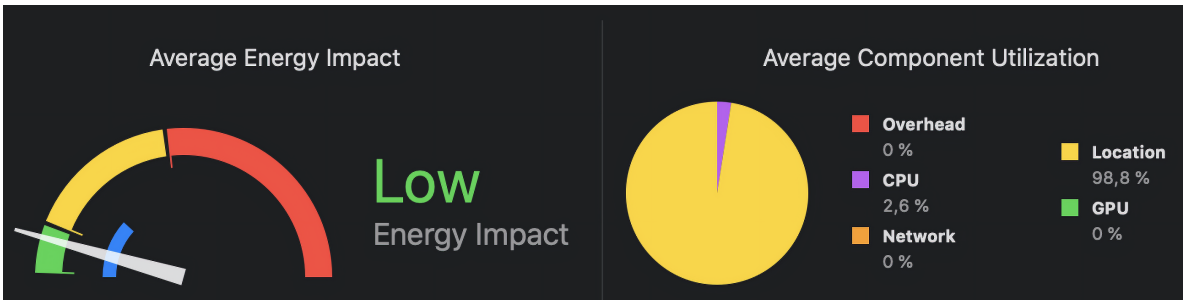


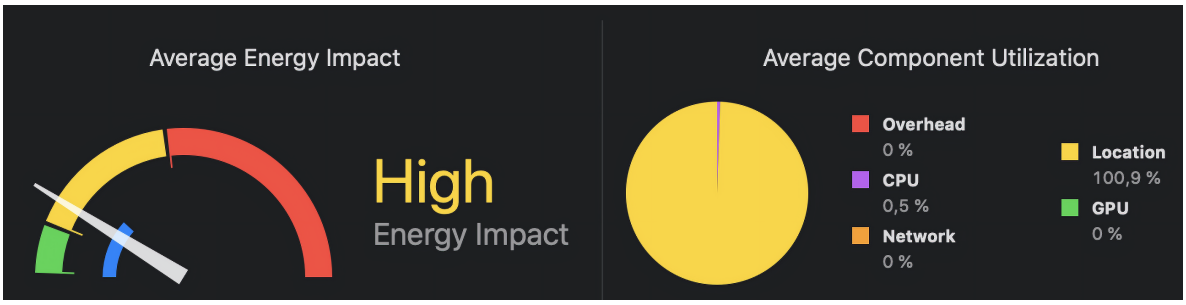
Figure 5.4: Flight output for our testing in July 2020.

Finally, we investigated energy expenditure. Xcode comes equipped with an energy gauging tool [91]. It displays a live chart about energy usage at runtime, and provides a resource breakdown. According to this tool, Alter Eco requires a low amount of energy when in the foreground with transport tracking enabled (Figure 5.5a). Energy demands then increase when in the background (Figure 5.5b), and are maximal when paired with

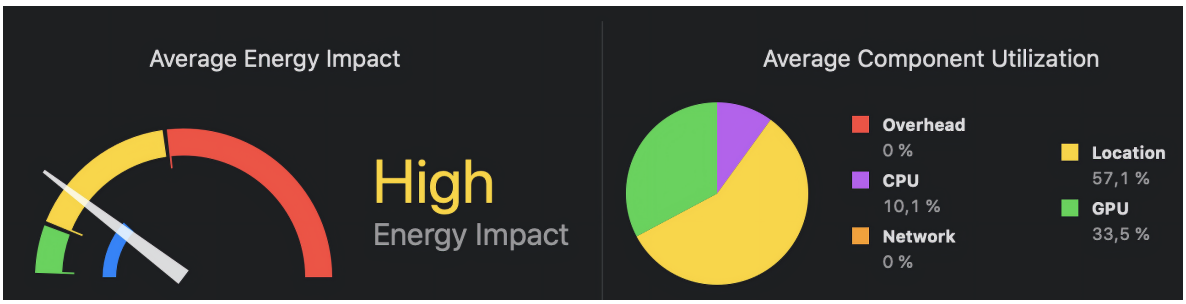
continuous moving of the camera within the virtual forest (Figure 5.5c). Even at its peak, Alter Eco remains at a relatively low area of the 'high' section. Nevertheless, in order to limit waste, and as discussed in Chapter 2, we have released a feature which allows the user to pause tracking. Furthermore, we have employed a system that suspends tracking automatically whenever iOS determines the user has been stationary for a reasonable time. In conclusion, we believe this level of energy consumption and the strategies employed to mitigate it to be sufficient for the tasks at hand.



(a) In the foreground with tracking.



(b) In the background with tracking.



(c) With tracking and rapid camera activity in the virtual forest.

Figure 5.5: Xcode's energy gauge outputs for different tasks run on an iPhone 8 Plus.

5.2.3 UI tests

Although Xcode does come with a tool to generate tests for the user interface [92], its results are notoriously unreliable and it is difficult to setup correctly [93]. As such, we decided to avoid automatic testing and instead rely on human inspection. The device simulator was used to emulate all available smartphones ranging from iPhone 8 to iPhone SE (2nd Generation). In addition, SwiftUI provided a very useful feature that can automatically generate a UI preview according to different layouts and colour settings (e.g. light and dark modes). This tool, known as *SwiftUI Preview* [94], was used to test all views, particularly to ensure a correct appearance for different parameters. As an example, the reader might refer to Figure 5.6, where we highlight the use of Preview to simulate various setups of the bar chart. Finally, by pressing the arrow button shown in Figure 5.6, a human tester can interact with the UI in its preset conditions, so as to verify that tapping controls, alerts and other interactive features are working as intended.

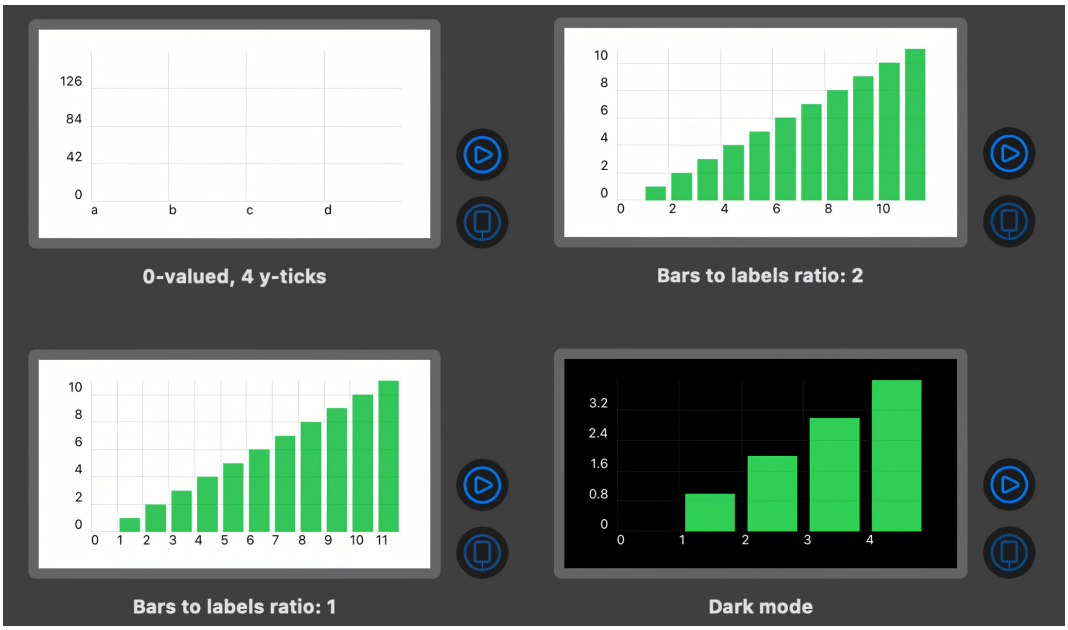


Figure 5.6: An example of SwiftUI Preview being employed to test the appearance of the bar chart with different parameters.

Chapter 6

Conclusion

6.1 Overview

In this project report we have outlined the issue of global warming and the lack of easy technological aids for individuals. To help minimise this problem, we have introduced Alter Eco, a recent app for iOS systems which we developed to help users gain a better appreciation for their emissions and to aid them in the development of eco-friendly, long-lasting habits.

We have seen how Alter Eco uses software engineering and natural language processing techniques to estimate, aggregate and visualize the contribution of transportation and food to the individual's carbon footprint. We have also seen how the results of these techniques have been paired with a gamification side, which includes a 3D virtual forest, to maintain users engaged. The use of extensive testing, both in the form of unit and end-to-end tests, confirmed the correctness of the methods employed. Furthermore, preliminary results from beta testers who have had the chance to try Alter Eco over the summer confirmed its effectiveness.

6.2 Future work

Although we believe Alter Eco to be an overall useful aid for a greener lifestyle, there are certainly a lot of extensions which are possible and worth exploring. For example, being able to port our app to Android would be useful to increase our userbase. That being said, there is also a lot that could be done just in terms of features. Many of our beta testers expressed the desire for more content within the virtual forest. This is definitely a valid suggestion, as at the present time there are still very little forest items one can buy. Furthermore, the interactivity with them is somewhat lacking: they are static and serve a mostly decorative purpose, but they could be expanded to include in-game quests. Adding animated models representing animals and designing a more varied landscape would also go a long way in terms of user engagement.

Moreover, we can identify room for improvement in the precision of the transport tracking system too. Some users have expressed interest in expanding the typologies of cars which one can use. For example, differentiating between electric and diesel engines or accounting for other characteristics would lead to a more accurate estimation. Since most of the data is already available in the primary source of our GHG emissions constants [95], this is a very feasible and reasonable suggestion.

The inclusion of a social aspect is also something to consider. As the reader might remember from Chapter 3, we have deliberately chosen to avoid competition-based games as we deemed them to be unfit for an application that reflects a very personal side of the user's life, over which they might not always have complete control. Nevertheless, it is true that Alter Eco at its current stage cannot take into consideration if, for instance, two users are sharing a car. Even though the emissions should technically be split in two, two different phones would each record one ride. Similarly, having a way to visualize the contribution of the whole household, rather than just of an individual, could help in assisting decision-making at the family level. Furthermore, we could aim for a better integration with social media by, for instance, letting users share screenshots of their forests or newly unlocked achievements. Although we had thought about this since before the beginning of the individual project, recent news articles [96, 97] have reported the Facebook SDK as the cause of apps crashing on iOS, which led us to postpone its integration until the situation stabilises or better alternatives become available.

Finally, as some of our beta testers pointed out in their comments, having a partnership would go a long way for Alter Eco. If we allowed conversion of points or other similar mechanics into concrete effects or products in real life, this could act as an additional trigger to alter behaviour. For instance, planting a real tree as an achievement whenever your virtual forest is thriving, or getting discounts for eco-friendly products. These are all ideas with a lot potential, but they do nonetheless require support from a third party that is unlikely to come without an mutual benefit agreement. Although this is not an option that we can materialize any time soon, it is something to consider for the future.

Bibliography

[1] Center for Climate and Energy Solutions. *Global Emissions*. URL: https://www.c2es.org/content/international-emissions/ (visited on 01/06/2020).

[2] Amnesty International. *CLIMATE CHANGE*. URL: https://www.amnesty.org/en/what-we-do/climate-change/ (visited on 01/06/2020).

[3] Stephen Leahy. *Greenland's ice is melting four times faster than thought—what it means*. URL: https://www.nationalgeographic.com/environment/2019/01/greeland-ice-melting-four-times-faster-than-thought-raising-sea-level/ (visited on 01/06/2020).

[4] Royal Society for the Protection of Birds. *Effects of climate change on wildlife*. URL: https://www.rspb.org.uk/get-involved/campaigning/climate-change-effects-on-nature-and-wildlife/effects-of-climate-change-on-wildlife/ (visited on 01/06/2020).

[5] Raúl Cassia, Macarena Nocioni, Natalia Correa-Aragunde and Lorenzo Lamattina. 'Climate change and the impact of greenhouse gasses: CO2 and NO, friends and foes of plant oxidative stress'. In: *Frontiers in plant science* 9 (2018), p. 273.

[6] eurostat. *Glossary:Carbon dioxide equivalent*. URL: https://ec.europa.eu/eurostat/statistics-explained/index.php/Glossary:Carbon_dioxide_equivalent (visited on 01/06/2020).

[7] Noelle Eckley Selin. *Carbon footprint*. URL: https://www.britannica.com/science/carbon-footprint (visited on 01/06/2020).

[8] United States Environmental Protection Agency. *Overview of Greenhouse Gases*. URL: https://www.epa.gov/ghgemissions/overview-greenhouse-gases (visited on 02/06/2020).

[9] eurostat. *Greenhouse gas emission statistics - emission inventories*. URL: https://ec.europa.eu/eurostat/statistics-explained/index.php/Greenhouse_gas_emission_statistics#Trends_in_greenhouse_gas_emissions (visited on 31/05/2020).

[10] Raluca Budiu. *Interaction Cost*. URL: https://www.nngroup.com/articles/interaction-cost-definition/ (visited on 31/05/2020).

[11] Seth Wynes and Kimberly A Nicholas. 'The climate mitigation gap: education and government recommendations miss the most effective individual actions'. In: *Environmental Research Letters* 12.7 (2017), p. 074024.

[12] Joseph Poore and Thomas Nemecek. 'Reducing food's environmental impacts through producers and consumers'. In: *Science* 360.6392 (2018), pp. 987–992.

[13] Hannah Ritchie. *Food production is responsible for one-quarter of the world's greenhouse gas emissions*. URL: `https://ourworldindata.org/food-ghg-emissions` (visited on 01/06/2020).

[14] Eloise Withnell, Hannah Kay, Benedetta Delfino, Michele Asara, Maurice Zard and Maxime Redstone Leclerc. *Carbon Footprint Application - Alter Eco - Report One*. 29th Jan. 2020.

[15] Eloise Withnell, Hannah Kay, Benedetta Delfino, Michele Asara, Maurice Zard and Maxime Redstone Leclerc. *Carbon Footprint Application - Alter Eco - Report Two*. 26th Feb. 2020.

[16] Eloise Withnell, Hannah Kay, Benedetta Delfino, Michele Asara, Maurice Zard and Maxime Redstone Leclerc. *Carbon Footprint Application - Alter Eco - Report Three*. 10th May 2020.

[17] Ofcom. *A decade of digital dependency*. 2018. URL: `https://www.ofcom.org.uk/about-ofcom/latest/features-and-news/decade-of-digital-dependency` (visited on 12/08/2020).

[18] Matthias Finkbeiner, Atsushi Inaba, Reginald Tan, Kim Christiansen and Hans-Jürgen Klüppel. 'The new international standards for life cycle assessment: ISO 14040 and ISO 14044'. In: *The international journal of life cycle assessment* 11.2 (2006), pp. 80–85.

[19] Shui Bin and Hadi Dowlatabadi. 'Consumer lifestyle approach to US energy use and the related CO2 emissions'. In: *Energy policy* 33.2 (2005), pp. 197–208.

[20] Carbon Footprint Ltd. *Carbon Footprint Calculator For Individuals And Households*. URL: `https://www.carbonfootprint.com/calculator.aspx` (visited on 17/08/2020).

[21] World Wide Fund for Nature. *WWF Footprint Calculator*. URL: `https://footprint.wwf.org.uk/` (visited on 17/08/2020).

[22] Martin Joos. 'Description of language design'. In: *The Journal of the Acoustical Society of America* 22.6 (1950), pp. 701–707.

[23] Dan Jurafsky and James Martin. *Speech and Language Processing (3rd ed. draft)*. 2019. Chap. 6.

[24] Charles Egerton Osgood, George J Suci and Percy H Tannenbaum. *The measurement of meaning*. 47. University of Illinois press, 1957.

[25] Jeffrey Pennington, Richard Socher and Christopher D Manning. 'Glove: Global vectors for word representation'. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[26] Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 'Efficient estimation of word representations in vector space'. In: *arXiv preprint arXiv:1301.3781* (2013).

[27] Jeffrey Pennington, Richard Socher and Christopher Manning. *GloVe: Global Vectors for Word Representation*. URL: `https://nlp.stanford.edu/projects/glove/` (visited on 15/08/2020).

[28] Keita Kurita. *Paper Dissected: "Glove: Global Vectors for Word Representation" Explained*. 29th Apr. 2018. URL: `https://mlexplained.com/2018/04/29/paper-dissected-glove-global-vectors-for-word-representation-explained/` (visited on 14/08/2020).

[29] Sciforce. *Word Vectors in Natural Language Processing: Global Vectors (GloVe)*. 14th Aug. 2018. URL: `https://medium.com/sciforce/word-vectors-in-natural-language-processing-global-vectors-glove-51339db89639` (visited on 14/08/2020).

[30] World Health Organization. *WHO Director-General's opening remarks at the media briefing on COVID-19 - 11 March 2020*. 11th Mar. 2020. URL: `https://www.who.int/dg/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020` (visited on 15/08/2020).

[31] Apple Inc. *Core Location*. URL: `https://developer.apple.com/documentation/corelocation` (visited on 02/06/2020).

[32] Paul Voigt and Axel von dem Bussche. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. 1st. Springer Publishing Company, Incorporated, 2017. ISBN: 3319579584. DOI: `10.5555/3152676`.

[33] Apple Inc. *TestFlight*. URL: `https://developer.apple.com/testflight/` (visited on 03/06/2020).

[34] *Guide to MSc Individual Projects - Legal and Ethical Considerations*. URL: `https://www.doc.ic.ac.uk/lab/msc-projects/ProjectsGuide.html#ethics` (visited on 30/08/2020).

[35] Brian J Fogg. 'A behavior model for persuasive design'. In: *Proceedings of the 4th international Conference on Persuasive Technology*. 2009, pp. 1–7.

[36] J Philippe Rushton and Goody Teachman. 'The effects of positive reinforcement, attributions, and punishment on model induced altruism in children'. In: *Personality and Social Psychology Bulletin* 4.2 (1978), pp. 322–325.

[37] Joseph S Lalli, Timothy R Vollmer, Patrick R Progar, Carrie Wright, John Borrero, Dency Daniel, Christine Hoffner Barthold, Kathy Tocco and William May. 'Competition between positive and negative reinforcement in the treatment of escape behavior'. In: *Journal of Applied Behavior Analysis* 32.3 (1999), pp. 285–296.

[38] Erich Gamma. *Design patterns: elements of reusable object-oriented software*. Pearson Education India, 1995. Chap. 1.

[39] Robert C Martin. *Agile software development: principles, patterns, and practices*. Prentice Hall, 2002.

[40] Ed Hardy. *Adoption of iOS 13 by iPhone users is nearly total*. 19th June 2020. URL: `https://www.cultofmac.com/714641/adoption-of-ios-13-by-iphone-users-is-nearly-total/` (visited on 15/08/2020).

[41] Prachi Bhardwaj and Shayanne Gal. *Despite Android's growing market share, Apple users continue to spend twice as much money on apps as Android users*. 6th July 2018. URL: `https://www.businessinsider.com/apple-users-spend-twice-apps-vs-android-charts-2018-7?r=US&IR=T` (visited on 26/08/2020).

[42] App Annie. *The Most Popular iOS Apps of All Time*. 2nd July 2018. URL: `https://www.appannie.com/en/insights/market-data/popular-ios-apps-time/` (visited on 17/08/2020).

[43] Statista. *Mobile operating systems' market share worldwide from January 2012 to December 2019*. URL: `https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/` (visited on 17/08/2020).

[44] Felix Richter. *Apple Users More Willing to Pay for Apps*. 6th July 2018. URL: `https://www.statista.com/chart/14590/app-downloads-and-consumer-spend-by-platform/` (visited on 17/08/2020).

[45] Stephanie Chan. *Global App Revenue Reached $50 Billion in the First Half of 2020, Up 23% Year-Over-Year*. 20th June 2020. URL: `https://sensortower.com/blog/app-revenue-and-downloads-1h-2020` (visited on 17/08/2020).

[46] Marianne Bertrand and Emir Kamenica. *Coming apart? Cultural distances in the United States over time*. Tech. rep. National Bureau of Economic Research, 2018.

[47] Steve Westlake. 'A Counter-Narrative to Carbon Supremacy: Do Leaders Who Give Up Flying Because of Climate Change Influence the Attitudes and Behaviour of Others?' In: *Available at SSRN 3283157* (2017).

[48] Apple Inc. *Human Interface Guidelines*. URL: `https://developer.apple.com/design/human-interface-guidelines` (visited on 04/06/2020).

[49] Apple Inc. *A more personal Health app. For a more informed you.* URL: `https://www.apple.com/ios/health/` (visited on 03/09/2020).

[50] Apple Inc. *Swift Official Page*. URL: https://www.apple.com/swift/ (visited on 18/08/2020).

[51] Jonny Evans. *Swift is again replacing Objective-C, report claims*. 5th Feb. 2020. URL: https://www.computerworld.com/article/3519437/swift-is-again-replacing-objective-c-report-claims.html (visited on 18/08/2020).

[52] Apple Inc. *SwiftUI*. URL: https://developer.apple.com/documentation/swiftui (visited on 18/08/2020).

[53] Apple Inc. *UIKit*. URL: https://developer.apple.com/documentation/uikit (visited on 18/08/2020).

[54] Apple Inc. *UIViewController*. URL: https://developer.apple.com/documentation/uikit/uiviewcontroller (visited on 18/08/2020).

[55] Apple Inc. *UIView*. URL: https://developer.apple.com/documentation/uikit/uiview (visited on 18/08/2020).

[56] Apple Inc. *SwiftUI - News - Apple Developer*. 3rd June 2020. URL: https://developer.apple.com/news/?id=06032019b (visited on 18/08/2020).

[57] Paul Hudson. *WWDC20: Wrap up and recommended talks*. 28th June 2020. URL: https://www.hackingwithswift.com/articles/222/wwdc20-wrap-up-and-recommended-talks (visited on 18/08/2020).

[58] Apple Inc. *UserDefaults*. URL: https://developer.apple.com/documentation/foundation/userdefaults (visited on 30/08/2020).

[59] Apple Inc. *Core Data*. URL: https://developer.apple.com/documentation/coredata (visited on 02/06/2020).

[60] D. Richard Hipp. *SQLite*. URL: https://www.sqlite.org (visited on 02/06/2020).

[61] Apple Inc. *Apple Platform Security*. URL: https://support.apple.com/en-gb/guide/security/secf6276da8a/web (visited on 02/06/2020).

[62] Apple Inc. *Date*. URL: https://developer.apple.com/documentation/foundation/date (visited on 25/08/2020).

[63] Norbert Horvát. *Swift GCD part 1: Thread safe singletons*. 1st July 2019. URL: https://blog.autsoft.hu/thread-safe-singletons-swift-gcd/ (visited on 03/09/2020).

[64] Apple Inc. *Data Formatting Guide*. URL: https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/DataFormatting/Articles/dfDateFormatting10_4.html#//apple_ref/doc/uid/TP40002369-SW10 (visited on 25/08/2020).

[65] Apple Inc. *Technical Q&A QA1480*. URL: https://developer.apple.com/library/archive/qa/qa1480/_index.html (visited on 25/08/2020).

[66] *Open Food Facts - World*. URL: https://world.openfoodfacts.org/ (visited on 24/08/2020).

[67] Stephen Clune, Enda Crossin and Karli Verghese. 'Systematic review of greenhouse gas emissions for different fresh food categories'. In: *Journal of Cleaner Production* 140 (2017), pp. 766–783.

[68] Helen Harwatt, Small World Consulting and Blueberry Consultants. *Carbon Food Calculator*. URL: https://www.vegansociety.com/take-action/campaigns/plate-planet/carbon-calculator (visited on 24/08/2020).

[69] Mustafa Özilgen. 'Energy utilization and carbon dioxide emission during production of snacks'. In: *Journal of Cleaner Production* 112 (2016), pp. 2601–2612.

[70] Mustafa Özilgen. 'Assessment of the energy utilization and carbon dioxide emission reduction potential of the microbial fertilizers. A case study on "farm-to-fork" production chain of Turkish desserts and confections'. In: *Journal of Cleaner Production* 165 (2017), pp. 564–578.

[71] Lindsey Partos. *Walkers crisps cut carbon footprint to retain carbon label*. 17th Feb. 2009. URL: https://www.bakeryandsnacks.com/Article/2009/02/17/Walkers-crisps-cut-carbon-footprint-to-retain-carbon-label?utm_source=copyright&utm_medium=OnSite&utm_campaign=copyright (visited on 24/08/2020).

[72] Ben & Jerry's. *A Life Cycle Analysis Study of Some of Our Flavors*. URL: https://www.benjerry.com/values/issues-we-care-about/climate-justice/life-cycle-analysis (visited on 24/08/2020).

[73] Apple Inc. *AVFoundation*. URL: https://developer.apple.com/av-foundation/ (visited on 24/08/2020).

[74] Apple Inc. *UIViewControllerRepresentable*. URL: https://developer.apple.com/documentation/swiftui/uiviewcontrollerrepresentable (visited on 24/08/2020).

[75] Apple Inc. *NLEmbedding*. URL: https://developer.apple.com/documentation/naturallanguage/nlembedding (visited on 24/08/2020).

[76] Apple Inc. *NLTagger*. URL: https://developer.apple.com/documentation/naturallanguage/nltagger (visited on 24/08/2020).

[77] Apple Inc. *Measurement*. URL: https://developer.apple.com/documentation/foundation/measurement (visited on 24/08/2020).

[78] Apple Inc. *SceneKit*. URL: https://developer.apple.com/documentation/scenekit (visited on 25/08/2020).

[79] Apple Inc. *Xcode Overview: Adding 3D Scenes*. URL: https://developer.apple.com/library/archive/documentation/ToolsLanguages/Conceptual/Xcode_Overview/Adding3DScenes.html (visited on 25/08/2020).

[80] Khronos Group. *COLLADA Overview*. URL: https://www.khronos.org/collada/ (visited on 25/08/2020).

[81] Chris Language. *Scene Kit Tutorial with Swift Part 2: Nodes*. 22nd Mar. 2016. URL: https://www.raywenderlich.com/1260-scene-kit-tutorial-with-swift-part-2-nodes (visited on 25/08/2020).

[82] *3D TEXTURES*. URL: https://3dtextures.me/.

[83] Martin Fowler. *Inversion of Control Containers and the Dependency Injection pattern*. 23rd Jan. 2004. URL: https://www.martinfowler.com/articles/injection.html (visited on 15/08/2020).

[84] *Basic Operators - The Swift Programming Language*. URL: https://docs.swift.org/swift-book/LanguageGuide/BasicOperators.html (visited on 29/08/2020).

[85] Apple Inc. *Set a location or route - Simulator Help*. URL: https://help.apple.com/simulator/mac/current/#/devc6eac95d6 (visited on 28/08/2020).

[86] Google LLC. *My Maps*. URL: https://www.google.com/maps/d/ (visited on 29/08/2020).

[87] Adam Schneider. *GPS Visualizer: Do-It-Yourself Mapping*. URL: https://www.gpsvisualizer.com/convert_input (visited on 29/08/2020).

[88] GOTOES. *Add Time Stamps to GPX Files*. URL: https://gotoes.org/strava/Add_Timestamps_To_GPX.php (visited on 29/08/2020).

[89] myclimate. *Offset your flight emissions!* URL: https://co2.myclimate.org/en/flight_calculators/new (visited on 02/09/2020).

[90] Carbon Footprint Ltd. *Flight carbon footprint calculator*. URL: https://calculator.carbonfootprint.com/calculator.aspx?tab=3 (visited on 02/09/2020).

[91] Apple Inc. *Measure Energy Impact with Xcode*. URL: https://developer.apple.com/library/archive/documentation/Performance/Conceptual/EnergyGuide-iOS/MonitorEnergyWithXcode.html#//apple_ref/doc/uid/TP40015243-CH34-SW1 (visited on 28/08/2020).

[92] Apple Inc. *Recording UI Tests*. URL: https://developer.apple.com/library/archive/documentation/ToolsLanguages/Conceptual/Xcode_Overview/RecordingUITests.html (visited on 28/08/2020).

[93] Paul Hudson. *How to test your user interface using Xcode*. 29th Mar. 2019. URL: https://www.hackingwithswift.com/articles/83/how-to-test-your-user-interface-using-xcode (visited on 28/08/2020).

[94] Apple Inc. *Previews*. URL: https://developer.apple.com/documentation/swiftui/previews (visited on 29/08/2020).

[95] Energy & Industrial Strategy Department for Business. *Greenhouse gas reporting: conversion factors 2019*. URL: https://www.gov.uk/government/publications/greenhouse-gas-reporting-conversion-factors-2019 (visited on 28/08/2020).

[96] Lisa Eadicicco. *How to workaround a Facebook bug causing popular apps like Spotify, Pinterest, and Tinder to crash*. 10th July 2020. URL: https://www.businessinsider.com/facebook-sdk-bug-causes-apps-spotify-crash-how-to-fix-2020-7?r=US&IR=T (visited on 28/08/2020).

[97] Benjamin Mayo. *For the second time this year, the Facebook SDK is causing apps like TikTok and Spotify to crash on launch*. 10th July 2020. URL: https://9to5mac.com/2020/07/10/app-crash-facebook-sdk/ (visited on 28/08/2020).

[98] Michele Asara. *Alter Eco beta*. URL: https://testflight.apple.com/join/PkPIn9Xs (visited on 30/08/2020).

[99] Apple Inc. *Running Your App in the Simulator or on a Device*. URL: https://developer.apple.com/documentation/xcode/running_your_app_in_the_simulator_or_on_a_device (visited on 30/08/2020).

Appendix A

Survey results

Here we list the open ended questions asked in the survey for the beta testers, with all individual answers being enclosed in quotes. The first and the last question were mandatory, while the second was optional.

A.1 What do you think of the main three features of the app (transport tracking, food input and virtual forest)?

'I think they are interesting and mostly accurate'

'Love the virtual forest, transport tracking is pretty cool, and I haven't used the food input much yet but I like the idea!'

'Virtual forest can be improved. Transport super useful'

'All very engaging!'

'It's not easy to get into a habit of inputting things manually / enabling tracking. The virtual forest is nice and makes the experience more engaging'

'I liked the transport tracking most.'

'The virtual forest should maybe give more rewards, like earning points for every ten trees or so. The transportation part is quite accurate while for the food I think maybe it could be made better showing also a list of the daily consumption?'

'I like them. I think they are really useful, especially the transport tracking, and the fact that I can now have trees is a really good feature'

A.2 Is there anything that you would like to see as an extension to these three features?

'Funky trees, maybe animals in the virtual forest.'

'Better graphics'

'Transport tracking by car type. Possibility to choose car model.'

'Could get recipes based on scanned ingredients for food input - planted trees in virtual forest could grow over time if carbon consumption is overall better than uk average'

'An expansion of the forest idea, maybe with some rewards or goals to achieve'

'Maybe a avatar (a cat or dog maybe) in the virtual forest'

A.3 Are there any other features that you would like to see implemented?

'Extending the virtual forest'

'A way to offset the carbon I've created by doing other things, like investments/donations. A partnership would go a long way for this app!'

'No'

'Highlights & Tips could be displayed via notification (maybe once a day to not make it too invasive)'

'More eco friendly tips? Recommendations -> possibility to buy eco friendly products within app?'

'Weekly challenges'

'Maybe a bit more infos about food and average production of co2'

'Not really'

Appendix B

Transport tracking

Here we discuss how the algorithm for transport tracking implemented in Alter Eco works in its most recent version.

B.1 How it works

B.1.1 Overview

The transport tracking feature was developed as part of the group stage of Alter Eco, but has been thoroughly tested and fixed during the individual project. It is based on the GPS functionalities of modern iPhones. When reading the official documentation [31], we discovered that requesting location updates based on distance, rather than time, would save battery and obtain a greater accuracy. To be precise, we heuristically chose a threshold of 50m to trigger an update, as this allowed for a good trade-off in granularity and accuracy. A user's position is then tracked over time, and using both location and speed data we infer the type of transportation they are likely to have taken. The algorithm is based on the concept of activities, which are structures containing the start and the end times of a movement, the distance covered and the inferred transportation mode. There are two types of activities:

- **Speed-based activities**, which are determined on the basis of how fast the user has travelled.
- **Region-of-interest (ROI) activities**, which are created when the user visits two locations such as airports or underground stations in a row.

A summary of the whole algorithm in the form of a flowchart is available in Figure B.1.

B.1.2 Speed-based activities

Speed-based activities are determined according to a threshold. Speeds equal or greater than a constant (currently 4m/s) are considered car activities, while those below are

classified as walking or cycling. A list of speed activities is kept in memory until one of three possibilities happens:

- A significant change is detected, where by significant change we mean a sequence of activities which follow others of a different type (e.g. 2 walking activities and then 2 car rides).
- We get a certain amount of activities of the same kind in a row (currently 3).
- Enough time has passed without any updates or significant changes (at the moment 5 minutes).

When any of these happens, the list is averaged into one activity and stored in the database. We use a weighted average, where cars have a higher weight to account for traffic situations (in which several walking events could have been erroneously recorded). The mean activity is then set to have begun with the start of the first activity and terminated with the end of the last.

B.1.3 ROI activities

The estimation of ROI-based activities is of higher priority, meaning that if the algorithm believes the user has taken the tube, then previously collected speed activities will be ignored. The program determines whether a journey between ROIs has occurred with a system of flags. If the user visits a ROI, the corresponding flag is set. Once the user visits another ROI of the same type (e.g. station to station, or airport to airport) with the flag still on, an activity is recorded if there is a sufficient distance among the two. The reason behind this extra distance check is to avoid issues related with airports terminals, which are often close in space but are identified as different ROIs. These flags can be turned off by a certain number of speed activities in a row, which must have occurred after the original ROI visit. Similarly, we use a timer which automatically turns the flag off for inactivity.

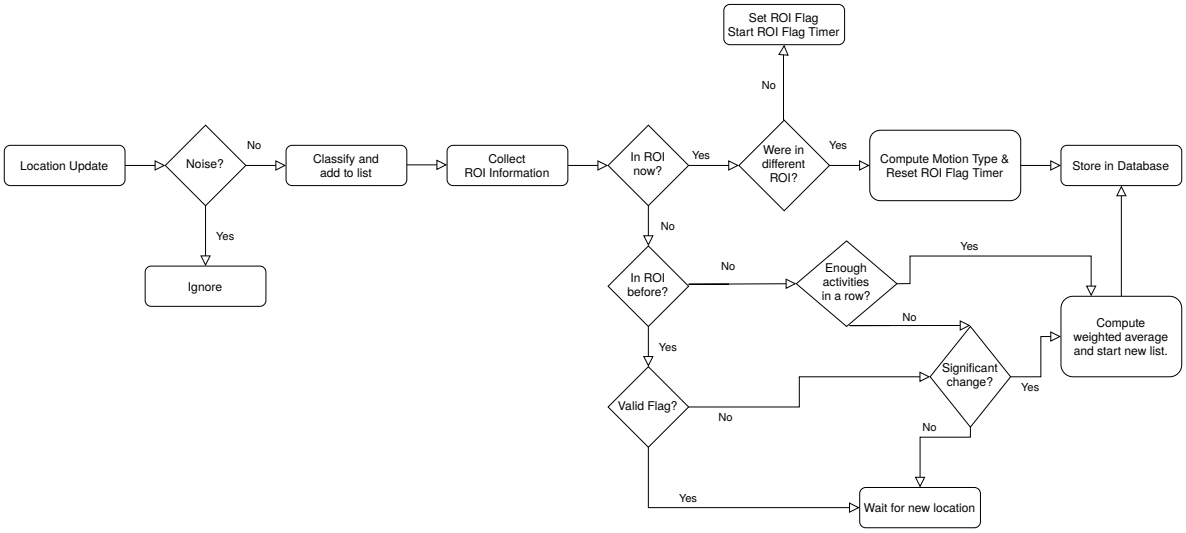


Figure B.1: Flowchart of the transport tracking algorithm.

Appendix C

User guide

Here we show how to install and use Alter Eco in its current version.

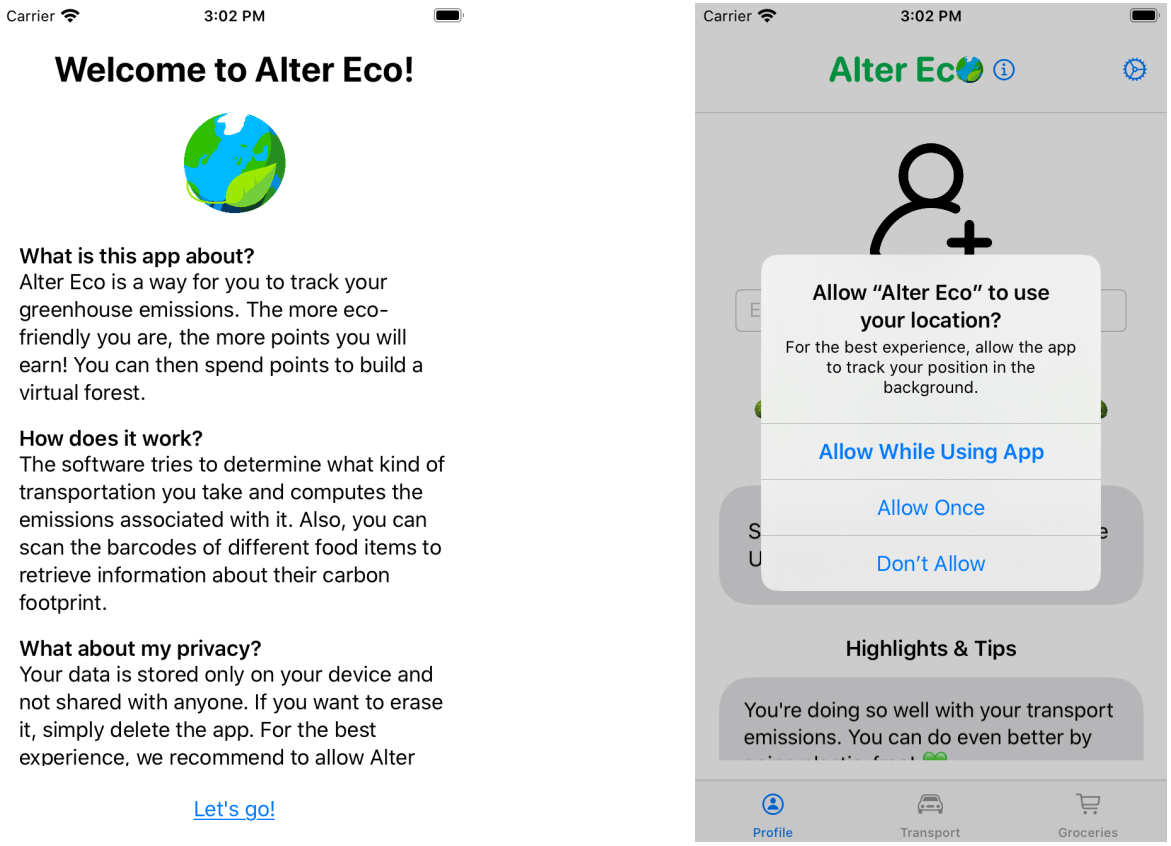
C.1 Installation and first launch

Alter Eco requires iOS/iPadOS 13 or above. It is currently not available on the App Store. Consequently, there are only two ways to install it:

1. Through the open link to the beta testing version.
2. With direct compilation of the source code.

The beta version can be downloaded from [98] by following the instructions on screen. If, alternatively, direct installation from the source code is preferred, Xcode 11 or above is necessary. Simply open the *xcodeproj* file, connect your device to the computer and proceed to compile as normal [99]. Make sure to have selected your device as the installation target. As a final note, please keep in mind that installing from the open link will give you access to the most recent updates, while direct compilation will be static and relative to the source code you have.

During your first launch only, you will be greeted with a welcome screen explaining the purpose of the app, its privacy policies and other information (Figure C.1a). Similarly, the first time you access some features (such as the virtual forest) you will receive a brief introductory message. Please make sure to grant Alter Eco the permissions it requires (Figure C.1b), or functionality might be affected. Also, if something is not clear, you may try clicking the info buttons which can be found all around Alter Eco (symbolised by a blue i), as they provide a general overview of the available features.



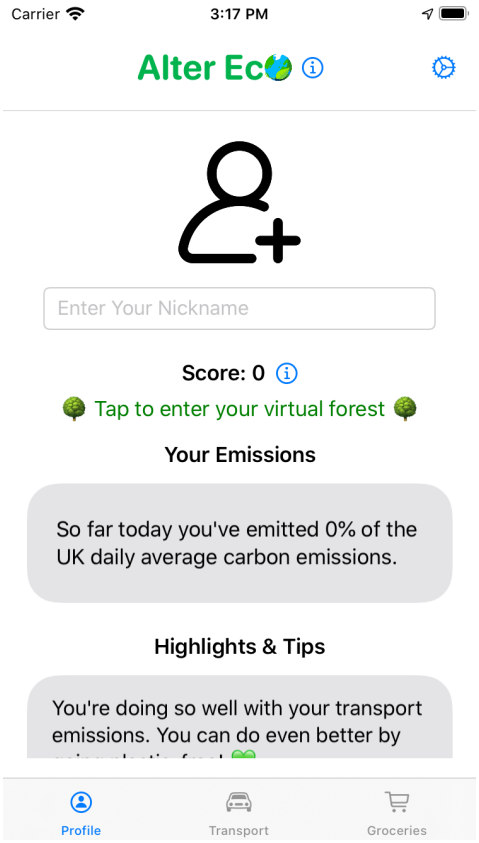
(a) Welcome screen.

(b) Permission request.

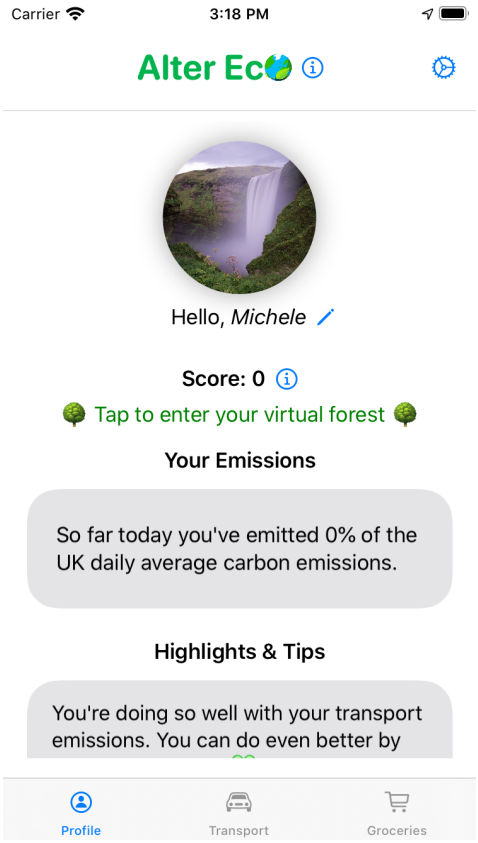
Figure C.1: Welcome screen and an example of permission request.

C.2 Profile view

The Profile view (Figure C.2) displays information such as an indicator of how you compare to the average daily carbon footprint in the UK, your score, and some highlights and tips which change according to your current emissions. In general, you gain more points for more eco-friendly behaviours (e.g. walking gives more than riding a car). You can personalise the Profile view by selecting a picture as your avatar. Simply tap on the plus sign and select a picture from your device. Similarly, you can enter a nickname by tapping the text field on screen. Make sure to press enter to save your nickname. You can then change either of them at any time by tapping the image you selected or the blue pencil next to your name C.2b.



(a) Standard Profile view.



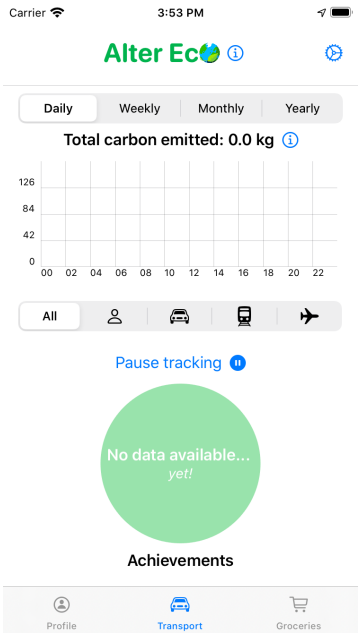
(b) Personalised Profile view.

Figure C.2: Examples of standard and personalised Profile views.

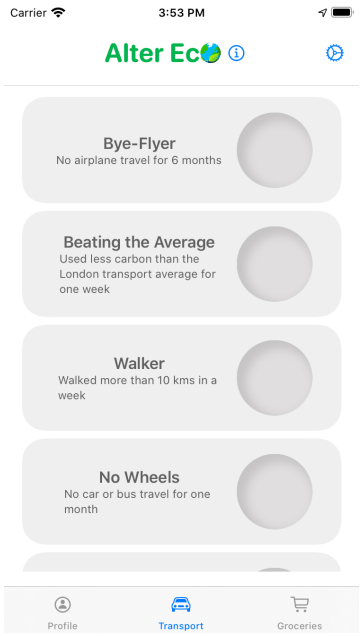
C.3 Transport view

In the transport view you will be able to see transport specific information. Initially, the view will be mostly empty (Figure C.3a), but it will be populated as you use Alter Eco. The data will be available through a bar chart and through a pie chart (Figure C.3c). The bar chart uses blue for polluting activities and green for those that are not. If your daily emissions go beyond the UK average, the blue bars turn red. If you wish to see the exact carbon associated with an activity in the bar chart, simply tap on the corresponding bar and an alert will appear (Figure C.3d).

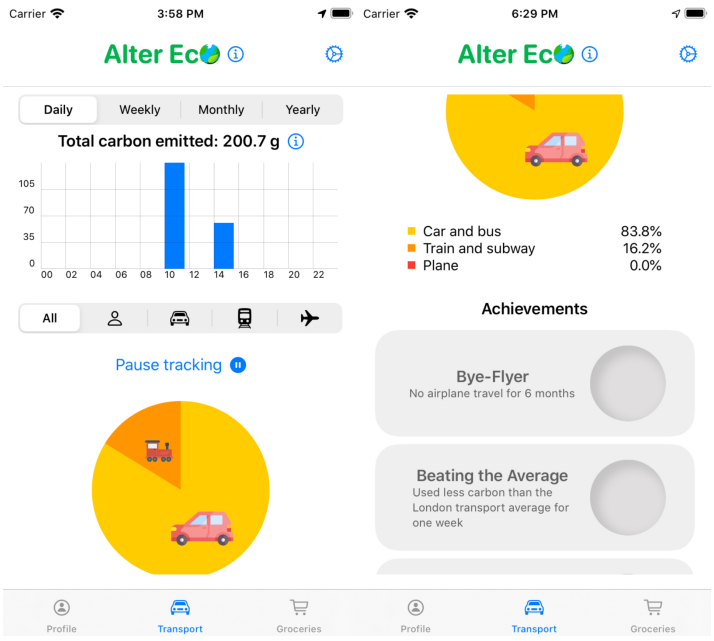
Furthermore, below the bar chart, there is a button which you can use to pause and resume transport tracking. Finally, a series of achievements are available and can be unlocked with specific actions (Figure C.3b). When you unlock one, it will gain colour and a badge will be displayed.



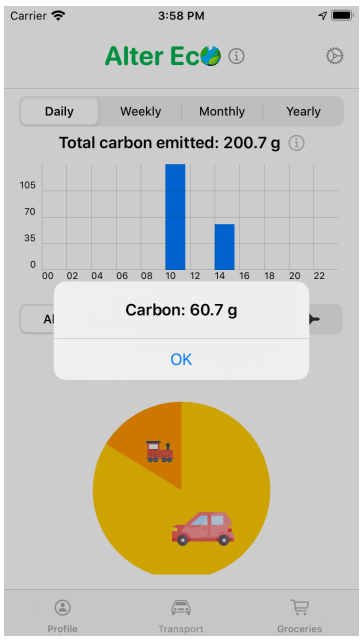
(a) Standard view.



(b) Achievements.



(c) View with data.

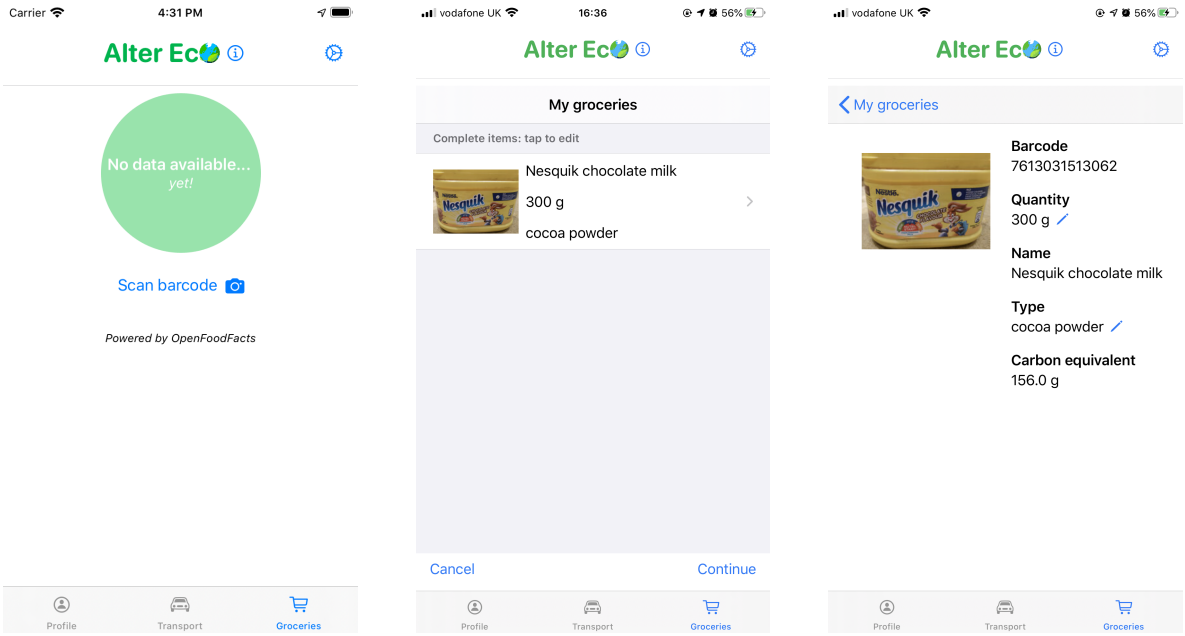


(d) Bar alert.

Figure C.3: Examples of what is available in the Transport view.

C.4 Food view

The first time you open the Food view no data will be available (Figure C.4a), but, just like in the Transport view, the pie chart will gain colour and information as you use Alter Eco. To scan one or more products, tap *'Scan barcode'* and proceed to scan. It is important to make sure that the right weight and food type are selected, so we recommend inspecting the list of items (Figure C.4b). If they are not, click on the item on the list and proceed to input the correct information (Figure C.4c). When you are done, press *'Continue'* to finish. You will gain points for each first scan of the day.



(a) Standard view. (b) Food list. (c) Item editing.

Figure C.4: Examples of what is available in the Food view.

C.5 Virtual forest

To access the virtual forest, go to the Profile view and press *'Tap to enter your virtual forest'* (Figure C.2). Once it finishes loading, you will be presented with a small 3D world that you can populate with items. You can pan the camera by sliding with two fingers, rotate it with one and zoom in/out with a pinch motion. In order to get items you need points, which you can spend in the shop (Figure C.5). If you wish to reorganize them, press *'Edit'*, then tap the item and move it where you want. Be careful of your emissions, as when they reach or go beyond the average UK value your forest gets polluted (Figure C.5d).

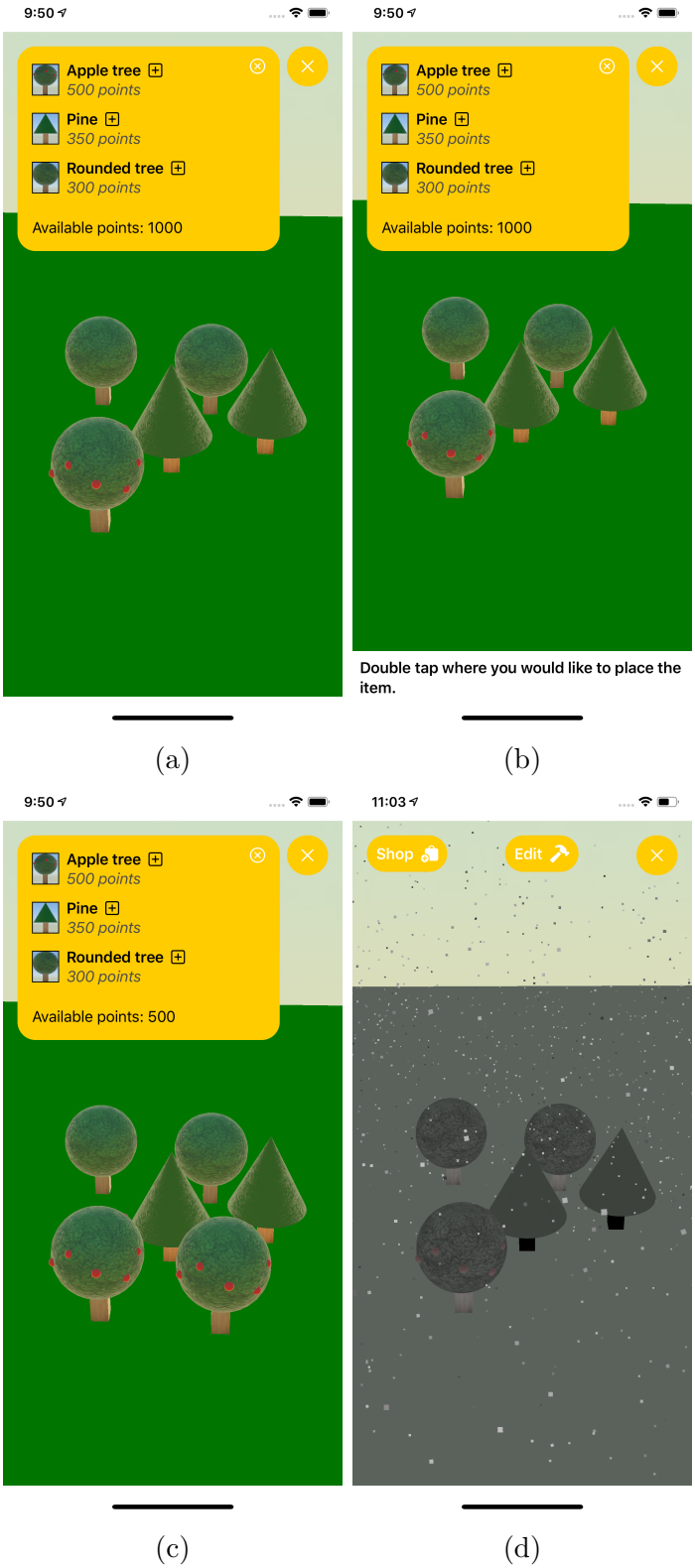


Figure C.5: (**a**) to (**c**) show how to use the shop. (**d**) shows the pollution.

C.6 Settings

To access the settings, tap on the blue gear icon next to the title (Figure C.2, Figure C.4, Figure C.5). Alter Eco comes equipped with a mechanism that automatically stops tracking your location when it determines you have not been moving in a while, which can be disabled or enabled here. Furthermore, if you are more likely to ride a bicycle than a car, we recommend toggling the appropriate option and selecting your average speed. Doing so will increase the likelihood of low speed car/train rides being misclassified as cycling, but will ensure cycling does not get misclassified as car or train rides. Any cycling journey is regarded as non-polluting and will be aggregated with the rest of the walking data. A screenshot of the settings is available in Figure C.6.

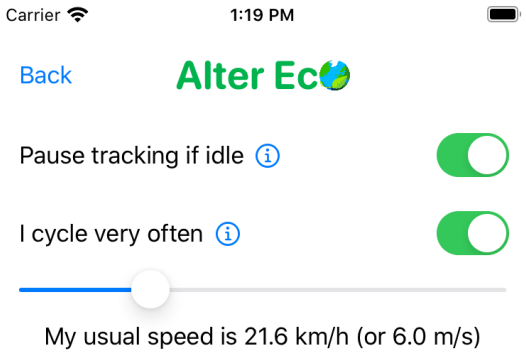


Figure C.6: Screenshot of the settings.