LabelScan: a Nutritional Label Accessibility App

Third Year Project

Martín Cuesta Allende

MEng (Hons) in Computer Science (Human Computer Interaction)

Supervisor: Mary McGee Wood

MANCHESTER 1824
The University of Manchester

Abstract

This report discusses a project aimed to improve the accessibility of physical nutritional labels for people with dyslexia and visual impairments, creating an Android application for that purpose. Nutrition labels contain very important information about the foods we consume, but their design leaves behind those with dyslexia and those with visual impairments. Despite digital technology, being ubiquitous with modern life, no nutrition label standard aims to make information accessible through any digital means. Because of this, an important percentage of the general population is restricted access to nutritional information.

This report will first discuss the conception and design of the application using a user-centred design approach. This will be followed by a discussion of how the software was developed, including the technologies used and the challenges present in their implementation. Finally, we will reflect on the achievements of the project, its possible future development, its successes and failures, and what would have been done differently a second time around.

Acknowledgements

I wish to express my gratitude to my supervisor, Mary McGee Wood, for her continued support and insightful feedback that helped me complete and refine all aspects of this project. Additionally, I would like to thank Roger Broadbent for his valuable time giving feedback on my project.

I would also like to thank my parents and siblings for their patience, encouragement, and support, not only with this project, but life in general.

I also want to thank my friends for paying attention to my ramblings and supporting me in every crazy, life-changing project I start.

And finally, thank you to the countless other people that helped me get where I am.

Effect of lockdown and COVID-19 on the project

The COVID-19 pandemic and the several lockdowns it has provoked have significantly affected this project, both in its design and development. Indeed, this project consisted of developing an assistive app. As such, in a normal year, I would have conducted extensive user testing. However, due to the pandemic, this became impossible. While doing user-centred design without having access to users was feasible, the resulting design and application lacks user feedback, making user evaluation much more difficult. As such, developing effective user experience was extremely more difficult due to the lack of users caused by the COVID-19 pandemic.

Furthermore, COVID-19 moving the due dates for this project has affected the quality of the final product. Indeed, there were multiple planned features that I could not implement due to a lack of time. If the deadline had been the same as previous years, the final product would have had more features, creating a more polished, more complete, and overall better application. Moreover, having the deadline pushed forward made it clash with other important deadlines from other subjects. Indeed, despite the two automatic extensions, deadlines for this project coincided with those of other subjects regardless, leading to having to juggle too many deadlines at once. Due to COVID-19, deadlines for this project were moved forward, leaving less time to work on the project and making it clash with other subjects, ultimately leading to a less polished application.

Contents

Chapter 1 – Introduction 1
Chapter 2 - Background 2
Chapter 3 – Conception and Design 5
  3.1 - Conception 5
    3.1.1 – Finding an idea 5
    3.1.2 –Improving previous solutions 6
  3.2 – User Experience Design 7
    3.2.1 – Gathering user requirements 7
    3.2.2 – Modelling user requirements 9
    3.2.3 – Mockups and Prototyping 10
  3.3 – User Interface Design 12
    3.3.1 – Design language 13
    3.3.2 – An emphasis on typography 14
    3.3.3 –Ensuring accessibility 17
  3.4 – Iterative Design 18
Chapter 4 – Development 22
  4.1 – Management of work 22
  4.2 – Platforms 23
    4.2.1 – Deployment platform 23
    4.2.2 – Development environment 23
  4.3 – Programming Language 24
  4.4 – ML Kit 24
  4.5 – User Interface 27

4.5.1 – Building layouts........27

4.5.2 – Enhancing feedback........28

4.5.3 – Customisability and accessibility........29

4.6 – MPAndroidChart........29

Chapter 5 – Results........31

Chapter 6 – Evaluation........32

6.1 – Software Evaluation........32

6.2 – User Evaluation........33

Chapter 7 – Reflection, Future Work, and Conclusion........36

7.1 – Project Goals........36

7.2 – Possible future work........37

7.2.1 – Short-term objectives........37

7.2.2 – Long-term objectives........39

7.3 – Project Management........40

7.4 – Personal Achievements........41

7.5 – Overall Conclusions........41

References........43

Appendices........49

7.3 – Appendix 1 - Examples of Nutrition Labels Around the World.49

7.4 – Appendix 2 - Low Fidelity Mockups........53

7.5 – Appendix 3 – Other Design Artefacts........58

Chapter 1

Introduction

This project consisted of developing a simple assistive application, having complete control over the scope, design, development and evaluation of the app. As such, this project was open-ended in nature. After an initial conception stage, the project took shape as an Android app that would help improve the accessibility of nutrition labels.

Nutrition labels contain very important information, particularly for people with dietary restrictions. However, many have not been designed with accessibility in mind, or fail at delivering information in a format accessible for everyone. Indeed, nutrition labels are usually in a format that can be quite hard to read for many people. For instance, people with dyslexia might have trouble reading the small, cramped text, whereas the purely textual format can make it impossible for people with visual impairments to understand the information presented.

The aim of this project was therefore to create an application that could scan a label, process it, and extract the relevant nutrition information, presenting it in a more accessible manner. The application would be designed, developed, and evaluated using industry-wide user experience techniques. Accessibility would be the main focus of the application, designed to be compatible with a variety of assistive services.

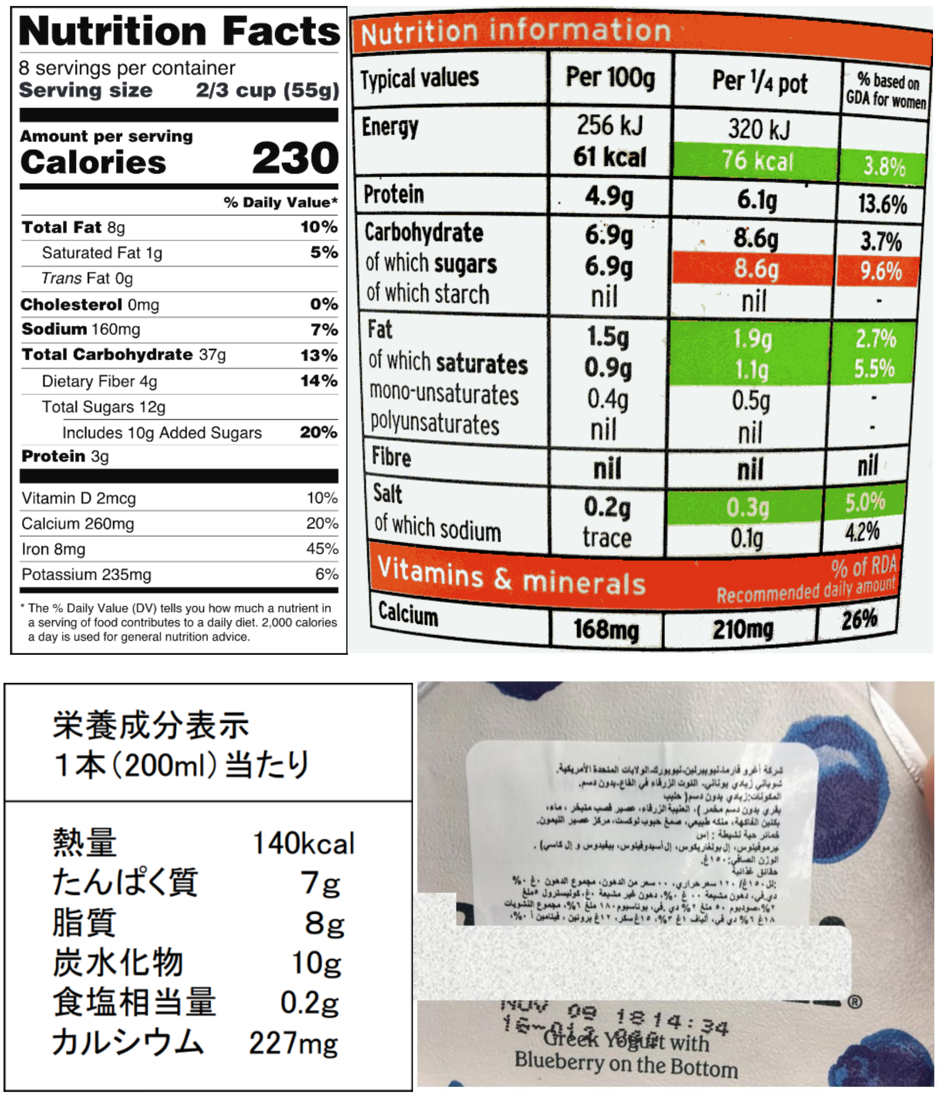
This report will explore the background for this application's motivations more in detail, the conception and design of the app, the code development, the results, the evaluation of the final product, and a reflection on work and the outcome of the project.

Chapter 2

Background

Nutrition labels provide very important information about the foods we consume, but there is no universal standard that all countries follow. There is no universally accepted standard for nutrition labels, even when accounting for language and culture differences. Consequently, all countries have nutrition labels that differ from each other, as each one implements a different solution to the problem of how to best convey the information. Figure 1 illustrates this point clearly, with more examples being shown in Annex 1. Furthermore, warning consumers about high contents of certain nutrients also varies by region. Some countries have adopted a front of packaging 'traffic light system', like the UK (Department of Health, 2016), while others use prominent labels warning of high contents of nutrients like sugar, like is the case in Chile (Ministerio de Salud, 2018, p.15). On the other hand, others do not warn consumers of high levels of nutrients in labels, like the United States (Food and Drug Administration, 2016). These differences reveal that an optimal solution has not yet been found for nutritional labels.

Furthermore, many regulations allow very cramped text in nutrition labels, making reading difficult, and creating inaccessible labels. For example, in Figure 2 displays a label with all its information in a single paragraph, with no structure or visual arrangement. This can make it difficult to find a nutrient we want information on, as we would have to scan the whole label to locate it. As the text is also restricted to a very small space, smaller values of inter-letter spacing are required to make the text fit, which can hinder word recognition for people with dyslexia (Perea et al., 2012). Furthermore, presenting the information in a purely textual manner can make comprehension of numerical values difficult when compared to using more visual information, like tables (Tufte, 1985). Some standards allow presenting the nutritional information in an inaccessible manner, creating a barrier of access for many people.



Nutrition Facts
8 servings per container
Serving size **2/3 cup (55g)**

Amount per serving
Calories **230**

	% Daily Value*
Total Fat 8g	**10%**
Saturated Fat 1g	**5%**
Trans Fat 0g	
Cholesterol 0mg	**0%**
Sodium 160mg	**7%**
Total Carbohydrate 37g	**13%**
Dietary Fiber 4g	**14%**
Total Sugars 12g	
Includes 10g Added Sugars	**20%**
Protein 3g	
Vitamin D 2mcg	10%
Calcium 260mg	20%
Iron 8mg	45%
Potassium 235mg	6%

* The % Daily Value (DV) tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

Nutrition information

Typical values	Per 100g	Per 1/4 pot	% based on GDA for women
Energy	256 kJ 61 kcal	320 kJ 76 kcal	 3.8%
Protein	4.9g	6.1g	13.6%
Carbohydrate	6.9g	8.6g	3.7%
of which sugars	6.9g	8.6g	9.6%
of which starch	nil	nil	-
Fat	1.5g	1.9g	2.7%
of which saturates	0.9g	1.1g	5.5%
mono-unsaturates	0.4g	0.5g	-
polyunsaturates	nil	nil	-
Fibre	nil	nil	nil
Salt	0.2g	0.3g	5.0%
of which sodium	trace	0.1g	4.2%
Vitamins & minerals			% of RDA Recommended daily amount
Calcium	168mg	210mg	26%

栄養成分表示
１本（200ml）当たり

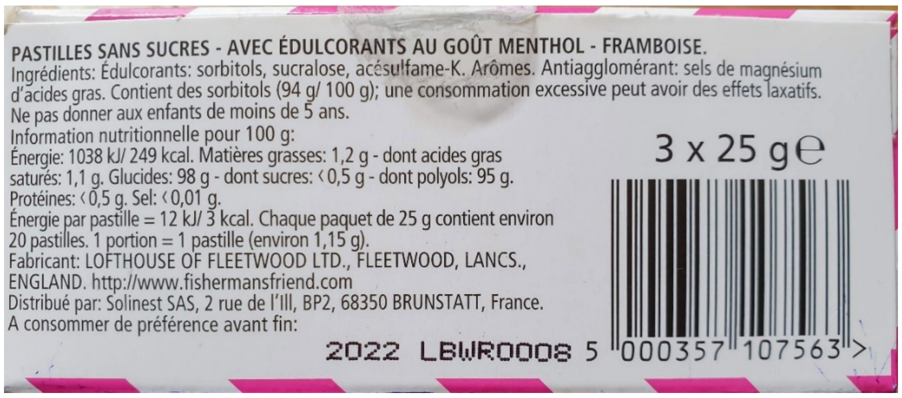
熱量	140kcal
たんぱく質	7g
脂質	8g
炭水化物	10g
食塩相当量	0.2g
カルシウム	227mg

شركة أغرو فارمـ-نيو بيرلين-نيويورك-الولايات المتحدة الأمريكية.
شوباني زبادي يوناني- التوت الزرقاء في القاع-بدون دسم.
المكونات:زبادي بدون دسم(حليب
بقري بدون دسم مخمر)، العنبية الزرقاء، عصير قصب متبخر ، ماء،
بكتين الفاكهة، منكه طبيعي، صمغ حبوب لوكست، مركز عصير الليمون.
خمائر حية نشيطة : إس
ثرموفيلوس، إل بولغاريكوس، إل أسيدوفيلوس، بيفيدوس و إل كاسي) .
الوزن الصافي: ١٥٠غ.
حقائق غذائية
:لل١٥٠غ/ ١٢٠ سعر حراري، ٠٠ سعر من الدهون، مجموع الدهون ٠غ ٠%
دي.في، دهون مشبعة ٠٠ غ ٠%، دهون غير مشبعة ٠غ، كوليسترول ٥ملغ
٢%،صوديوم ٥٠ ملغ ٢% دي .في، بوتاسيوم ١٨٠ ملغ ٦%، مجموع النشويات
١٨غ ٦% دي في، ألياف ١غ ٣%، ١٥غ سكر، ١٢غ بروتين ، فيتامين أ ٠%،

[redacted]

NOV 09 18 14:34
Greek Yogurt with
Blueberry on the Bottom

Figure 1: Clockwise from top left: FDA nutrition facts label, UK nutrition label using a traffic light system, Japanese standard nutrition label (Source: Consumer Affairs Agency, 2020), United Arab Emirates nutrition label for imported products (Source: United States Department of Agriculture, 2019)



PASTILLES SANS SUCRES - AVEC ÉDULCORANTS AU GOÛT MENTHOL - FRAMBOISE.
Ingrédients: Édulcorants: sorbitols, sucralose, acésulfame-K. Arômes. Antiagglomérant: sels de magnésium
d'acides gras. Contient des sorbitols (94 g/ 100 g); une consommation excessive peut avoir des effets laxatifs.
Ne pas donner aux enfants de moins de 5 ans.
Information nutritionnelle pour 100 g:
Énergie: 1038 kJ/ 249 kcal. Matières grasses: 1,2 g - dont acides gras
saturés: 1,1 g. Glucides: 98 g - dont sucres: ‹0,5 g - dont polyols: 95 g.
Protéines: ‹0,5 g. Sel: ‹0,01 g.
Énergie par pastille = 12 kJ/ 3 kcal. Chaque paquet de 25 g contient environ
20 pastilles. 1 portion = 1 pastille (environ 1,15 g).
Fabricant: LOFTHOUSE OF FLEETWOOD LTD., FLEETWOOD, LANCS.,
ENGLAND. http://www.fishermansfriend.com
Distribué par: Solinest SAS, 2 rue de l'Ill, BP2, 68350 BRUNSTATT, France.
A consommer de préférence avant fin:
2022 LBWR0008

3 x 25 g℮

5 000357 107563 >

Figure 2: French nutritional label where the nutritional information is not distinct from the rest of the information, and where line wrapping breaks up the names of the nutrients, like '*acides gras saturés*'

Moreover, even the more accessible nutritional labels prove to have accessibility problems for people with visual or reading impairments. Indeed, nutritional labels only use printed, visual information, which can exclude people with visual impairments. Indeed, concerns have been raised that nutrition labels are completely inaccessible for blind people (Southey, 2020). As around 285 million people in the world are visually impaired, from which 39 million are blind (World Health Organization, 2010), current standards exclude a large part or the global population from essential information. Even when the information is presented in a visually accessible manner, nutrition labels remain inaccessible for many people, requiring change.

In this section, it has been explained how there is no worldwide standard for nutrition labels, and how current ones remain inaccessible. This has created a gap in access where nutrition labels do not effectively provide the information they are supposed to for many people. The following chapters address how I attempted to fix this issue with my project by designing, developing, and evaluating an Android assistive application to scan nutrition labels and presenting them in a more accessible manner.

Chapter 3

Conception and Design

Due to the nature of this project, I could not start coding immediately. Instead, attention had to be paid at the conception and design of the app itself, including gathering and analysing user requirements, creating mockups and prototypes, and planning how it would work on a technical level.

3.1 Conception

3.1.1 Finding an idea

This project's only initial indication was to make a simple assistive piece of software, so coming up with a worthwhile, useful, and original idea was the first step of the process. While other third year projects available are much narrower in scope, this proposal offers much more freedom to the student in defining the project's ambitions and goals. However, this freedom also meant that I needed a clear idea of what I wanted to focus on, as a not well-defined enough concept would harm long-term development. As such, I started this project with multiple brainstorming sessions to determine what shape it should take.

The first ideas I came up with were all centred around improving the accessibility of digital products and devices. However, coming up with an original way to do so was proving to be a challenge. Many digital products still do not assess accessibility properly in their design or development phase. Indeed, Yan and Ramachandran (2019) found that only 1.7% out of 479 popular Android applications presented no accessibility issues. Accessibility issues persist in a variety of other digital products too, like online libraries (Spina, 2019), video games (Yuan et al., 2011), and MOOCs (Iniesto et al., 2016). Therefore, my first concepts attempted to tackle some of these issues. Some examples included making a browser extension that would allow users to change a website's layout to make it more accessible or advanced screen tinters to reduce eye strain. Despite having multiple ideas, there were already existing products that accomplished the same functions. Although I could have developed a complete and useful product based on these ideas, I wanted my project to be unique and not a simple imitation of other programs. Therefore, I shifted my focus away from improving the ease of use of software.

Instead of trying to improve the accessibility of technology, I changed my approach to target ways physical products and real-life situations could be made more accessible through technology. This change was inspired by my HCI course and general knowledge. Technology can help make the real world more accessible for people with disabilities and impairments. For instance, Google's Lookout app helps people with blindness explore the world around them by recognising objects using computer vision (Clary, 2019). I took inspiration from this example to try and find accessibility and access issues in real life that technology could provide a novel solution for. This approach of tackling real world problems without already existing, complete, and adequate solutions permitted me to create a project that was fully unique and original.

I finally settled on building an app that improves the accessibility of nutritional labels for people with dyslexia and visual impairments by scanning them and presenting them in a more accessible manner. Nutritional labels include important information about the food we eat every day, like calorie count, sugar quantity or proteins. Knowing these values can be crucial for a wide variety of people, including those on a diet, athletes, or people with diabetes. However, many nutritional labels have not been designed with accessibility in mind, as people with impairments like low vision and astigmatism may struggle reading the usually small and cramped text. If text legibility were not an issue, people with dyslexia could still have difficulties understanding purely textual information. As computer vision and text recognition has become feasible even on low-power devices, I decided to attempt to solve this issue through an app that scanned the label itself and presented the information present in an accessible style.

3.1.2 Improving previous solutions

There are many existing applications that can display a product's nutritional information, but their underlying design do not make them apt for making products accessible. For instance, many of these apps rely on scanning the barcodes to identify a product, and then access a database where the nutritional information has been stored beforehand. One of the most popular Android apps for calorie counting, MyFitnessPal (2021), uses this method to provide nutritional information to its users. However, for products that have not been logged by any user, a common issue in countries where the app is not as popular as in the United States or the UK, the app will not be able to display any nutritional information. Instead, it will ask the user to input the information themselves. Furthermore, as this information is mainly user-obtained, input errors may make the application less reliable. As such, this barcode- and database-oriented method does not work if we want to make all nutritional labels accessible.

Furthermore, accessing a database requires internet access, which creates another barrier to access. While 97% of the population in developed countries are covered by a fast speed

4G mobile network, only 40.5% of the population in the least developed countries enjoy the same level of coverage, with only 19% using the Internet due to unaffordability (International Telecommunication Union, 2020, pp. 4, 13). Furthermore, internet in developing regions is constrained by 'slow speed, low computational power, [and] reduced bandwidth', creating 'digital exclusion' (Harper, 2020). Therefore, requiring online access would create a barrier to usability for a large part of the world's population, which is to be avoided when developing an app that is focused on improving accessibility of food products.

To avoid these issues and improve the accessibility and effectiveness of current nutrition apps, I decided to have the app perform the nutritional label scan on-device. This ensured that the nutritional information was as accurate as possible, instead of relying on user-provided information. Doing this scanning offline also guaranteed that users in areas with no internet access could use the app and access the same information as all other users. With this design decision, a better and more efficient user experience could be built when compared to previous solutions.

3.2 User Experience Design

When building an app meant to improve accessibility, good user experience design is key. Indeed, following key principles of user experience will result in an app that is user-focused, effective, and accessible. However, great user experience cannot be retroactively fitted once the core development has finished, but it must be a consideration in the design process from the very start, and throughout the entire development. As such, special attention was given to the user experience design.

3.2.1 Gathering user requirements

Gathering user requirements was the first step of the design process, as is usual for projects using user-centred design. Indeed, building an assistive app requires understanding functional and non-functional user needs. Therefore, before developing an app, it is fundamental to understand what the app is trying to accomplish beyond a technical level (Rettig, 1992). This includes not only functional requirements, but also understanding who the target audience is, how the visual design should look like, how the app needs to behave, and how the interaction pathway should be like (Harper, 2020). Gathering this information allows to determine what elements and features will have a significant impact on improving the user experience and the usefulness of the product, and which ones can be set aside as they would not have a significant enough benefit compared to development costs. As such, gathering user requirements was the first step in the design phase of this product, allowing to determine what the next steps in development were, and how the final app should look and behave like.

Due to the COVID-19 pandemic, gathering user requirements was more difficult than in a normal year, as getting access to users was impossible. Indeed, many of the methods typically used in user experience design were impossible. Techniques like task analysis and participant observation are considered among the most effective for requirements elicitation (Harper, 2020; Norman, 2013, pp.221-224). Indeed, these allow a UX designer to better understand the user, their objectives, and the problems of the solution they currently use. In this project, participant observation would have been used extensively. However, this technique requires access to users. The COVID-19 pandemic made direct access to users impossible, so I could not use this observation technique. As such, gathering user requirements proved to be more difficult due to the lack of users caused by the pandemic.

Due to the lack of users, requirements were gathered by using both previously written formal and informal sources on how to design for people with dyslexia, but also about finding out what were the most common problems when it came to nutrition label accessibility. Indeed, this data reuse can be an effective method of requirement elicitation if there is a lack of users. By mixing both academic and other formal sources, like Nielsen (1996) and Graham et al. (2012), and more informal sources such as Guest (2016), Bjorn the UX Dog (2020), and Van den Rul (2019), this user requirements elicitation took into consideration empirical research and the subjectivity of user experience. While scientific journal articles and industry reports are extremely useful sources of information, direct user testimonials allow a UX designer to understand the personal feelings and thoughts a user has when dealing with a given problem. As UX uses both quantitative and qualitative approaches, it is therefore important to use both formal and informal sources. Consequently, previously written informal and formal sources were used, allowing to gather user requirements despite lacking a direct access to interview or observe users.

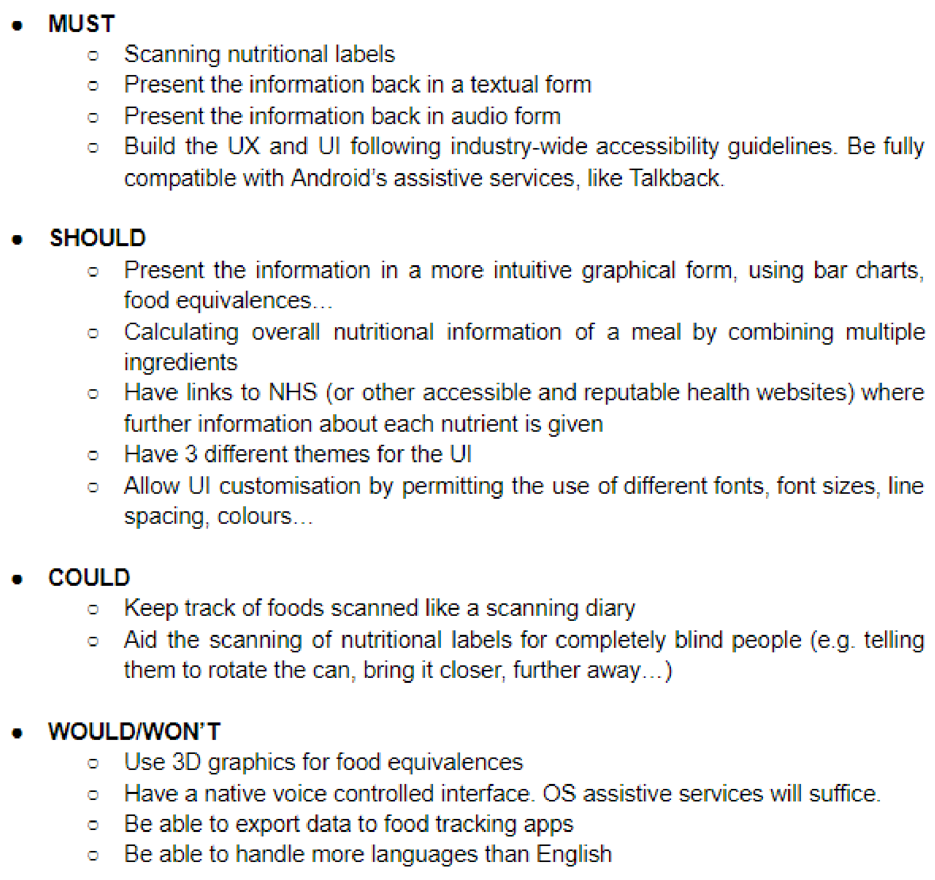
Some of the key requirements that were identified included using accessible fonts, including customisation options to respond to individual accessibility needs, and include a graphical representation of the nutritional information. Indeed, all of these were necessary elements to ensure that the application was as useable, useful, and accessible as required. For example, including font customisation was a planned feature as there is no single font preferred by users with dyslexia, as will be described further in section 3.3.2. In the case of including graphical representations of the nutritional information, some dyslexic users found that they were able to better understand information if it was also presented using charts. After reading and collating multiple sources, user requirements were determined to decide what features were significantly important and needed to be implemented.

3.2.2 Modelling user requirements

User requirements modelling is the next key step, allowing to gain a deeper understanding of the system while confirming that the user requirements we found are appropriate. This stage consists of analysing the previously gathered requirements and designing a system that responds to these requirements. While there are many different, parallel methods for modelling requirements, this project focused on ones that did not require users. Indeed, many modelling methods fully integrate the user, allowing them to make contributions in this stage too. However, due to the lack of users, as was explained earlier, the choice of which methods could be used was limited. For instance, using movable post-it notes to create storyboards is an effective method to encourage users to contribute to refining the system, as it is an unthreatening, familiar system to lay down thoughts (Harper, 2020). However, it becomes limited when done with no users or without physical access to a whiteboard, owing to the limited interaction in online systems. Consequently, as with requirements elicitation, requirements modelling was done using methods that did not require users.

Personas and scenarios were created to better imagine and understand the people using the system, and how they would use it. Personas are detailed descriptions of fictional persons relevant to a system. These are especially useful when 'constraints (...) exclude participatory design methods' (Matthews et al., 2012), as is the case for this project. On the other hand, scenarios are descriptions of a user using the system to accomplish one or multiple goals. Scenarios also include user's expectations of how a system will behave and how the interactions take place. These two user experience tools are useful not only in the design phase, but also in the testing phase, when validating if the built product fulfils the requirements. Therefore, creating personas and scenarios was a key step in modelling user requirements, as it helped create a better picture on what users looked like and how interactions with the app would take place.

Finally, using the MoSCoW system allowed me to sort and prioritise which features to focus on first based on their importance, allowing development to be organised and progress smoothly. Figure 3 shows how user requirements in this project were set into different categories. Those in the 'MUST' category were considered critical, meaning that they had to be developed first with the highest level of polish possible. On the other hand, those ranked at "WOULD/WON'T" were not deemed feasible to be completed in the timeframe of this project, either because they would not bring a significant improvement to the product, or because the time required to develop them would be better employed to implement and polish other, more important features. Organising and classifying user requirements into different categories based on their urgency and importance was the final step of the modelling user requirements stage, leading the way to start prototyping the system.



- **MUST**
  - Scanning nutritional labels
  - Present the information back in a textual form
  - Present the information back in audio form
  - Build the UX and UI following industry-wide accessibility guidelines. Be fully compatible with Android's assistive services, like Talkback.
- **SHOULD**
  - Present the information in a more intuitive graphical form, using bar charts, food equivalences…
  - Calculating overall nutritional information of a meal by combining multiple ingredients
  - Have links to NHS (or other accessible and reputable health websites) where further information about each nutrient is given
  - Have 3 different themes for the UI
  - Allow UI customisation by permitting the use of different fonts, font sizes, line spacing, colours…
- **COULD**
  - Keep track of foods scanned like a scanning diary
  - Aid the scanning of nutritional labels for completely blind people (e.g. telling them to rotate the can, bring it closer, further away…)
- **WOULD/WON'T**
  - Use 3D graphics for food equivalences
  - Have a native voice controlled interface. OS assistive services will suffice.
  - Be able to export data to food tracking apps
  - Be able to handle more languages than English

Figure 3: User requirements organised using MoSCoW analysis

The subsection that follows describes the final, and arguably most important, step of requirements modelling, consisting of building both low- and high-fidelity mockups and prototypes.

3.2.3 Mockups and prototyping

Mockups and prototypes are useful tools for user experience practitioners, allowing to get both qualitative data on the user experience of a product in its early stages. Indeed, creating and evaluating interactive prototypes allows to explore different ideas quickly (Buchenau and Suri, 2000) and obtain valuable qualitative data on the experience (Arhippainen and Tähti, 2003). Furthermore, high fidelity prototypes can have very high ecological validity. Indeed, the user can interact with this type of prototype on device, closely matching real world conditions. Therefore, building mockups and prototypes was a very important phase in the design stage of this project.

Several low fidelity mockups were made to quickly and iteratively explore different layout ideas. Low fidelity mockups, done using pen and paper, allow to quickly sketch out and compare layouts for one same screen. Compared to more high fidelity mockups, low fidelity mockups also would have allowed for users to input their ideas on how the layout should look like, but this did not take place due to the COVID-19 pandemic, as explained in the previous sections. Nonetheless, this type of mockups remained a very useful tool to investigate different solutions, as seen in Figure 4. In this case, being able to quickly create variations on the same screen permitted to analyse how element hierarchy was affected by the design, as well as finding flaws in the design. This saved time before building the more time consuming interactable prototypes, allowing to find the optimal solution faster. Further examples of these low fidelity mockups can be found in Appendix 2. The use of low fidelity mockups allowed to focus faster on what the final design would look like by finding the best visual solutions to the user requirements.

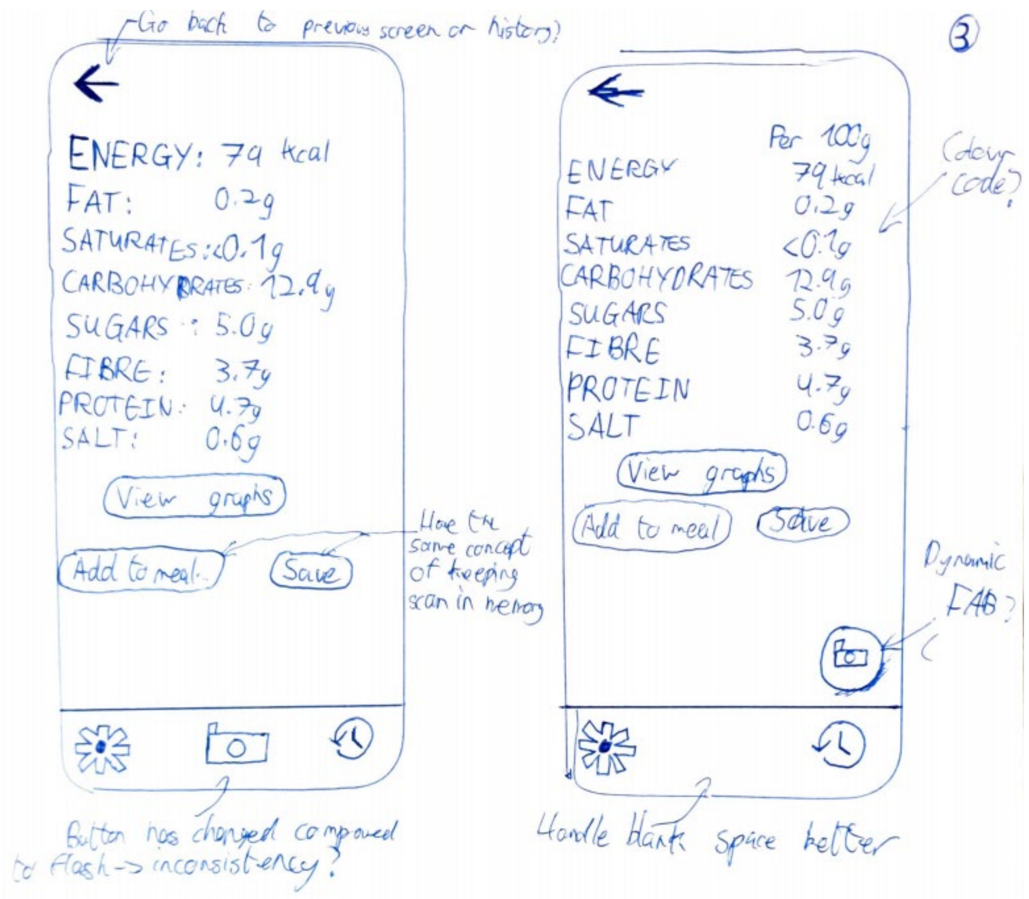


Figure 4: Low fidelity mockups of the detailed scan screen, comparing different solutions to element hierarchy and organisation

Finally, a high-fidelity prototype that could be run on device was created to simulate how physical interactions with the app would take place. Indeed, as this application would involve physically moving the phone around to scan a label, there is an additional degree of interaction that can only be simulated in a physical device. Because of this, building an interactive mobile prototype was necessary. Using Adobe XD (Adobe, 2021a), I built a prototype, partly shown in Figure 5, that implemented the features characterised as a

‘MUST’ previously. This prototype also implements one ‘COULD’ feature, namely keeping track of foods previously scanned, as the visual layout would be unbalanced without its icon in the bottom navigation bar. This prototype was then tested on the mobile version of Adobe XD (Adobe, 2021b), ensuring that the app would be easy to use on a mobile device. Creating a high-fidelity prototype helped make sure that the user interface and user experience design fully responded to the user requirements before starting code development.

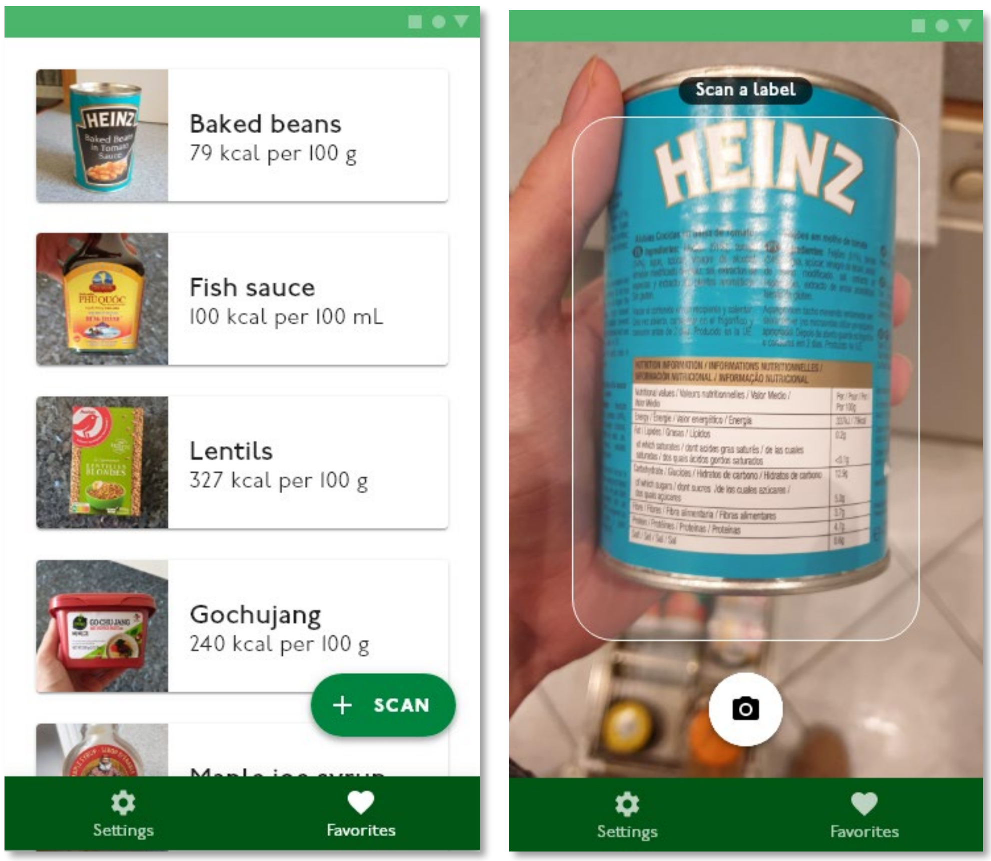


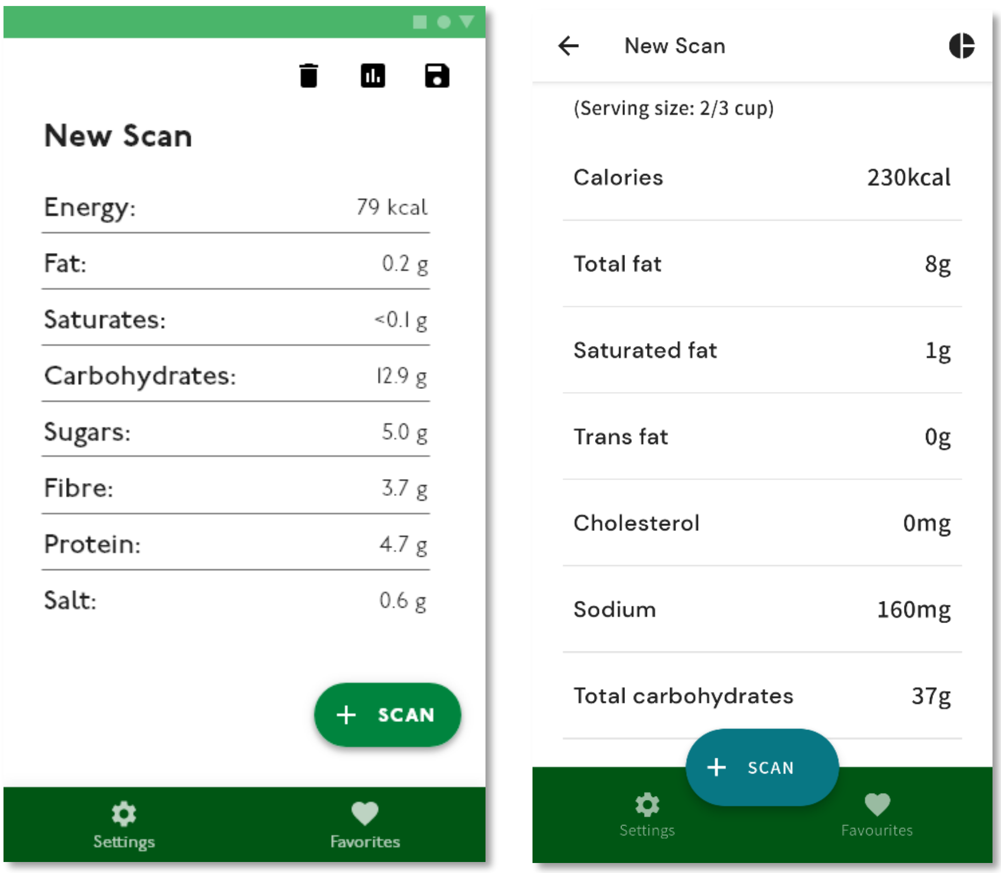
Figure 5: Examples of the initial Adobe XD prototype screens

3.3 User Interface Design

The user interface affects user experience, and vice versa. User interface elements, like typography, colours, or branding will affect the user’s emotion and inner state, as well as the app’s usability. An unappealing, bland interface will decrease the perceived quality of the app, resulting in a worse overall experience. An unreadable font will make the app difficult to use. Certain interactions will require specific interface elements. Although related to user experience design, user interface design is nonetheless distinct, as the latter focuses almost exclusively on the visual aspects described above, while the former pays attention to the workflow and overall experience when using a system. Therefore, proper user interface design was key in ensuring that the user experience was as polished and high-quality as possible.

3.3.1 Design language

As this project consisted of developing an app exclusively for Android, I decided to employ the Material design system (Google, 2021a). Material sets out a series of guidelines concerning components, layout, colour, sound, and type. While it would have been possible to design the user interface without following any already existing guidelines, Material is a robust and mature design language created with lots of research by a very established company. Having no previous experience doing interface design, following Material guidelines allowed to create a polished visual style faster than otherwise. Furthermore, using Material ensures that the application is visually consistent with other Android applications and with the overall Android system. This has a positive influence in helping users understand how to use the application, as it looks and feels similar to other applications they're familiar with. Following guidelines of the Material design system allowed to create a polished user interface that helped make the app feel familiar and effective for users.


New Scan

Energy:	79 kcal
Fat:	0.2 g
Saturates:	<0.1 g
Carbohydrates:	12.9 g
Sugars:	5.0 g
Fibre:	3.7 g
Protein:	4.7 g
Salt:	0.6 g

New Scan

(Serving size: 2/3 cup)

Calories	230kcal
Total fat	8g
Saturated fat	1g
Trans fat	0g
Cholesterol	0mg
Sodium	160mg
Total carbohydrates	37g

Figure 6: '+ SCAN' FAB in the first prototype (left) and the final application (right) not following Material guidelines at the time of development

Nonetheless, some user interface elements did not strictly follow Material guidelines at the time of development, as they were not fully adequate for the requirements of this application. Indeed, both in the interactive prototype and the final product, the placement of the 'Scan' floating action button (FAB) did not follow Material guidelines at the time of design and development. While Material guidelines state that FABs should not be placed outside of a bottom app bar, the first prototype did so, as can be seen in Figure 6. This was done like this as travelling to the 'new scan' screen was not simply travelling to a different part of the application, but also part of creating a new scan that could potentially be a persistent object if the scanning diary feature were implemented. Furthermore, at the time of development guidelines stated to not mix bottom navigation bars and floating action buttons, as they could confuse the user, but these recommendations have changed at the time of writing. Indeed, in a study they conducted 'participants liked the bottom navigation bar with an embedded, centered FAB because of the aesthetic and ergonomic benefits' (Google, 2021b), so the previous guidance has been removed. Going to the scanning screen remained as an embedded FAB within the bottom navigation bar in the final product as it was part of the navigation system, but also a way of potentially creating persistent objects if the diary feature was created. At the time of development, certain Material guidelines did not seem adequate, so I designed the user interface ignoring these. Recent changes in these guidelines suggest that the decision to do so was appropriate.

3.3.2 An emphasis on typography

Typography was a very important part of the design process, as one of the main objectives of this app was to render inaccessible nutritional label text accessible for people with dyslexia. Therefore, using a font that remained legible at even small font sizes was essential. As with other design requirements, a mix of formal and informal sources were used to discover which fonts help users with dyslexia, and which fonts to avoid.

Formal sources seem to suggest that sans-serif fonts are the best for screen readability, while fonts designed specifically for dyslexia like OpenDyslexic do not improve or worsen readability. For example, Boyarski et al. (1998) state that while Verdana was perceived as better than serif fonts on a computer screen, other factors like 'type size, line length, and line spacing' will also impact readability regardless of font. Rello and Baeza-Yates (2013) agree, stating that 'sans serif' font types 'increased significantly the reading performance' for people with dyslexia, recommending fonts like Helvetica, Arial, or Verdana, while 'italic fonts' decreased readability. They also suggest that OpenDyslexic 'did not lead to a faster reading', with participants preferring Verdana or Helvetica over this font specifically designed for dyslexia. Finally, Rello (2015) remarks that font size should be 'larger than the current recommendations' in WACG 2.0. Because of these studies, I gravitated towards sans-serif fonts, while keeping in mind how other factors, like font size, would impact readability and accessibility.

Informal sources generally agree with the more formal studies but also reveal how subjectivity plays an important role in font preference, further suggesting that customisability could be an important feature. Indeed, font readability seems to have a personal dimension, depending on each user's preferences. For example, multiple forum threads about dyslexia show clashing opinions on the OpenDyslexic font (Resorization, 2019; mano1990, 2018; xueli, 2015), as some users stated that it helped them whereas others argued that it made reading more difficult. While most users prefer sans serif fonts, Veroniiiica (2018) advises that cursive fonts like 'Lavanderia' can be appropriate at times, and Chen (2019) states that one should use a font that mixes serif and sans serif characteristics. These sources are a good illustration of how subjective font readability is, with each person having different personal preferences, despite seeming to be a consensus in the more formal sources. As such, font customisability could help significantly enhance the application's accessibility, responding to individual preferences.

The conflict between aesthetics and accessibility added to the challenge of finding an adequate font. Aesthetics play an important part in user experience, affecting the perceived quality of the product (Harper, 2020). Because of this, some fonts deemed accessible for people with dyslexia would not be appropriate for this project, as their aesthetics would hamper the quality of the application. For example, both Arial and Verdana are very prevalent on many systems, and it is a widely held view that they are both bland fonts (Rothstein, 2009). Due to this familiarity and little aesthetic value, using these fonts in the application would have created a product that feels bland and generic, diminishing the quality of the user experience. As such, the font used would have to not only be fully accessible for people with dyslexia, but also be visually pleasing to enhance further the user experience.

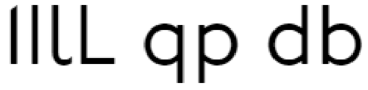


Figure 7: Accessibility issues with P22 Underground. The letter 'q' is a mirrored version of 'p', and so are 'd' and 'b', which can make words difficult to read when both letters are present in a word. The number '1' is also very similar to the capital letter 'I', which could cause problems when reading the value of a nutrient and its unit of measurement

The first choice of font was P22 Underground, but due to accessibility issues, licensing problems, and intentions to improve visual hierarchy, I switched to using both DM Sans and Noto Sans CJK as the primary fonts. I had already used P22 Underground in other projects, but upon testing its accessibility and utility for this application, some of its characters were not fully distinct from each other, as can be seen in Figure 7. Furthermore, licensing issues to use the font in Android applications prompted the switch to other fonts. After exploring multiple options, the DM Sans font was selected for headings and subheadings, while the Noto Sans CJK font was chosen for body text. While the DM Sans suffers from similar problems to P22 Underground, as the number ‘1’ is very similar to the capital letter ‘I’, this did not pose any issues, as no numbers would appear next to letters in headings or subheadings. The Noto Sans font family is a font developed by Google, with a wide adoption in Android, helping make the app feel more familiar to users. The Noto Sans CJK font, which covers East-Asian languages, was chosen over the regular Noto Sans font, as Latin characters in the former were less geometric, shown in Figure 8, helping character recognition and potentially improving readability. Furthermore, using different, distinct fonts with different weights helps create a visual hierarchy that follows the components hierarchy. As Figure 9 illustrates, DM Sans is used for the main dialog whereas Noto Sans CJK is used for the body text, which is less important and acts as helper text. The bigger weight used for the title also helps drive attention to it, helping the user understand faster the current state. After initially selecting P22 Underground, licensing and accessibility issues prompted the switch to the accessible DM Sans and Noto Sans CJK fonts.

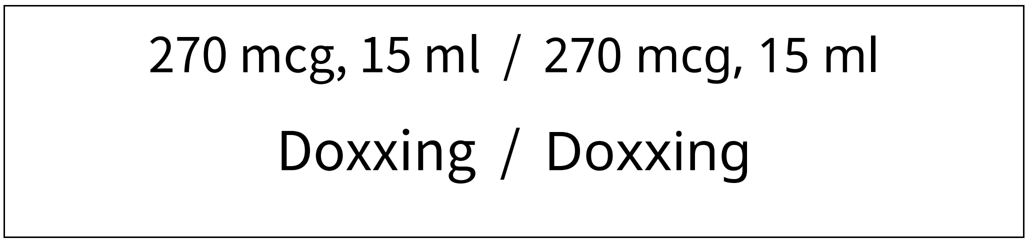
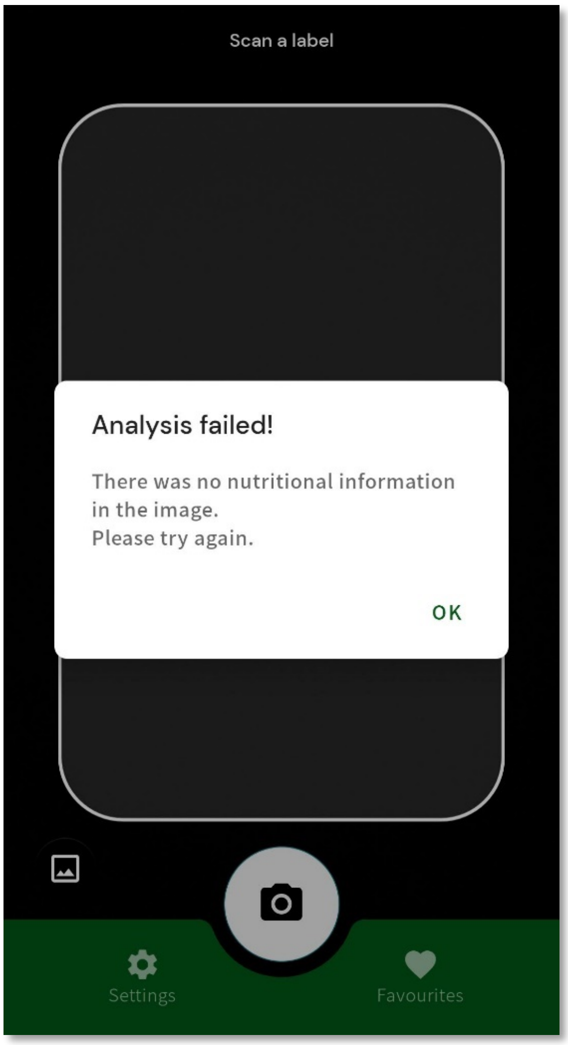


Figure 8: Comparison between Noto Sans CJK (left) and default Noto Sans (right). Noto Sans CJK is more humanist and less geometric than the default Noto Sans font. More geometric fonts can result in more difficult differentiation between letters. Noto Sans CJK also has a bigger x-height (height of a letter), which makes it easier to read at small font sizes



Analysis failed!

There was no nutritional information in the image.
Please try again.

Figure 9: Example of DM Sans and Noto Sans CJK being used create a visual hierarchy and help drive attention

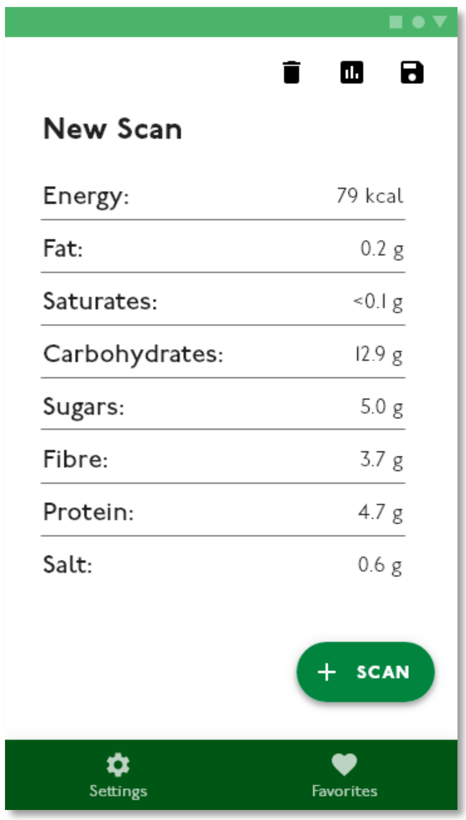
3.3.3 Ensuring accessibility

The previously mentioned high-fidelity prototype also served to validate the UI elements for their accessibility, including colour contrast and colour blindness checks. Indeed, the interactive prototype was the first time that colour was introduced into the design. Using a variety of tools ensured that the design was fully accessible for a maximum number of users. For instance, the WebAim colour contrast checker (WebAIM, 2021) was used in conjunction with the Stark plugin (Stark Lab, 2021) to ensure that text contrast conformed to WCAG AA standards. The Adobe colour picking tool (Adobe, 2021c) provided a way to check that colours used for different UI elements were colour-blind safe. During the user interface design process, a variety of tools were used to ensure that all elements remained accessible colour wise.

Three different themes were designed after identifying that having different theme options was an important user requirement, as it would be easier for an individual user to customise the application to their liking. The themes included the default light theme, a dark theme, and a pale theme. Different users will have different preferences on how they want their phone to look at based on functional requirements. For example, dark theme has gained popularity in these last years for many different reasons. For some users it may reduce eye strain, and while some users prefer it simply for aesthetic value, dark mode may also help people with low vision (Budiu, 2020). However, Budiu (2020) also notes that dark theme presents certain issues of its own, namely causing halation around letters for people with astigmatism, making text harder to read. Furthermore, some people might find light theme straining in the eyes, but do not wish to use a dark theme due to the aforementioned problems, so a third, pale theme was designed, shown in Figure X. This theme uses milder colours to reduce eye strain while still maintaining the feel and accessibility of a light theme. All three themes were verified to be colour contrast safe and colour-blind friendly. Light theme was chosen to be the default theme, with the dark and pale themes being selectable options in settings. Designing and creating different themes was a very important phase of ensuring that the user interface remained as accessible as possible for the widest set of users.

3.4 Iterative Design

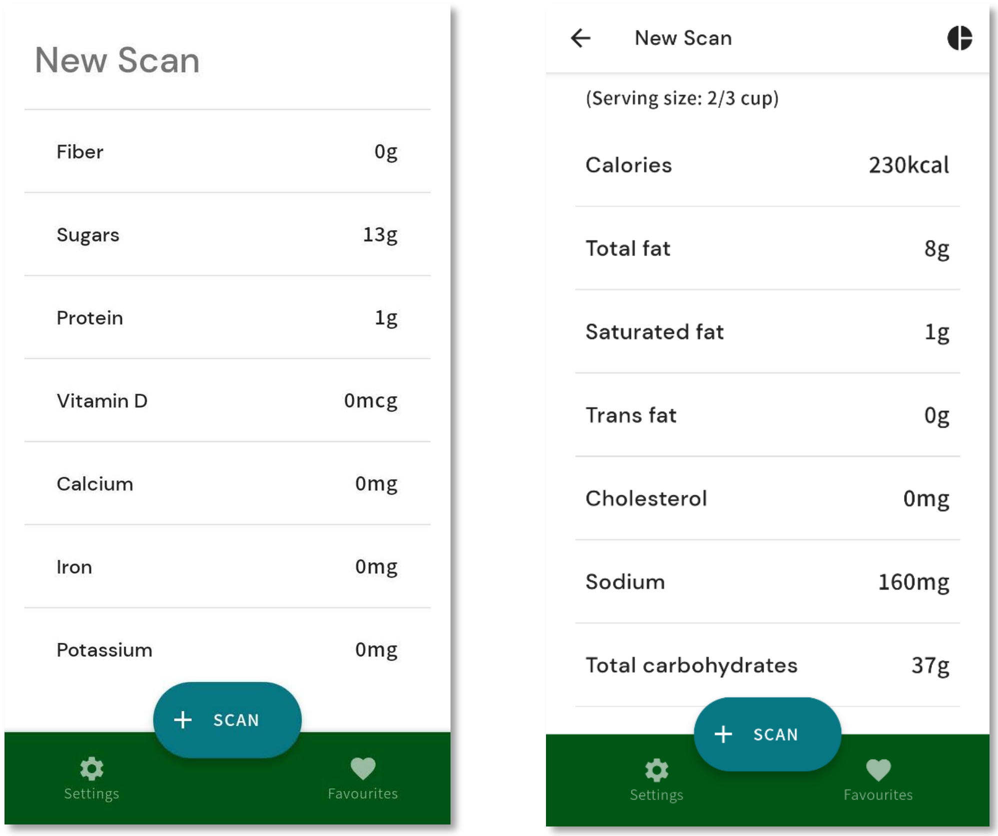
Iterative design was originally planned to take place by creating multiple prototypes throughout the year, but due to time constraints and practicality, the iterative design process was scaled down to a more simplified version. Iterative design is a staple of user experience design (Harper, 2020), usually done in 4 main, cyclic phases: Understanding users, Design, Prototyping, and Evaluating (Petrie and Bevan, 2009). Therefore, the creation of multiple prototypes was planned to take place throughout the development process to conduct iterative design. However, creating a prototype in Adobe XD took an almost equal amount of time compared to creating the layout directly in code. Consequently, these mid-stage prototypes were not created, instead opting for a faster version of the iterative design cycle without the prototyping stage. This is exemplified by the development and design of the detailed scan screen immediately after scanning a label. Figures 10 and 11 display how this screen changed from the first prototype to the final release version, with the design changing progressively directly in code without building any prototypes. While complete iterative design was planned to take place, it was cut down and shortened due to time constraints.



New Scan

Energy:	79 kcal
Fat:	0.2 g
Saturates:	<0.1 g
Carbohydrates:	12.9 g
Sugars:	5.0 g
Fibre:	3.7 g
Protein:	4.7 g
Salt:	0.6 g

Figure 10: Original Adobe XD prototype for the nutrient display screen



New Scan

Fiber	0g
Sugars	13g
Protein	1g
Vitamin D	0mcg
Calcium	0mg
Iron	0mg
Potassium	0mg

New Scan

(Serving size: 2/3 cup)

Calories	230kcal
Total fat	8g
Saturated fat	1g
Trans fat	0g
Cholesterol	0mg
Sodium	160mg
Total carbohydrates	37g

Figure 11: From left to right: mid-project design revision and final design for the nutrient display screen, both built directly in code

Nonetheless, despite the lack of a complete iterative design process, there were some small design stages during development, notably for the pie chart feature. As features were progressively implemented as lower priority ones were completed, not all the design phase was completed before code development started. Indeed, some features, like the pie charts to represent the daily recommended value percentage, were designed mid-development. Figure 12 shows how creating low fidelity mockups helped discover that using bar graphs to visually represent the nutrition information could lead to grouping, a process where different elements are grouped together perceptually due to shared characteristics (Goldstein, 2016). In this case, the bars seemed to be in competition to each other due to proximity and similarity. On the other hand, representing the daily recommended values as pie charts did not seem to create grouping from visual inspection. Small design phases during the development phase helped save time when developing new features.

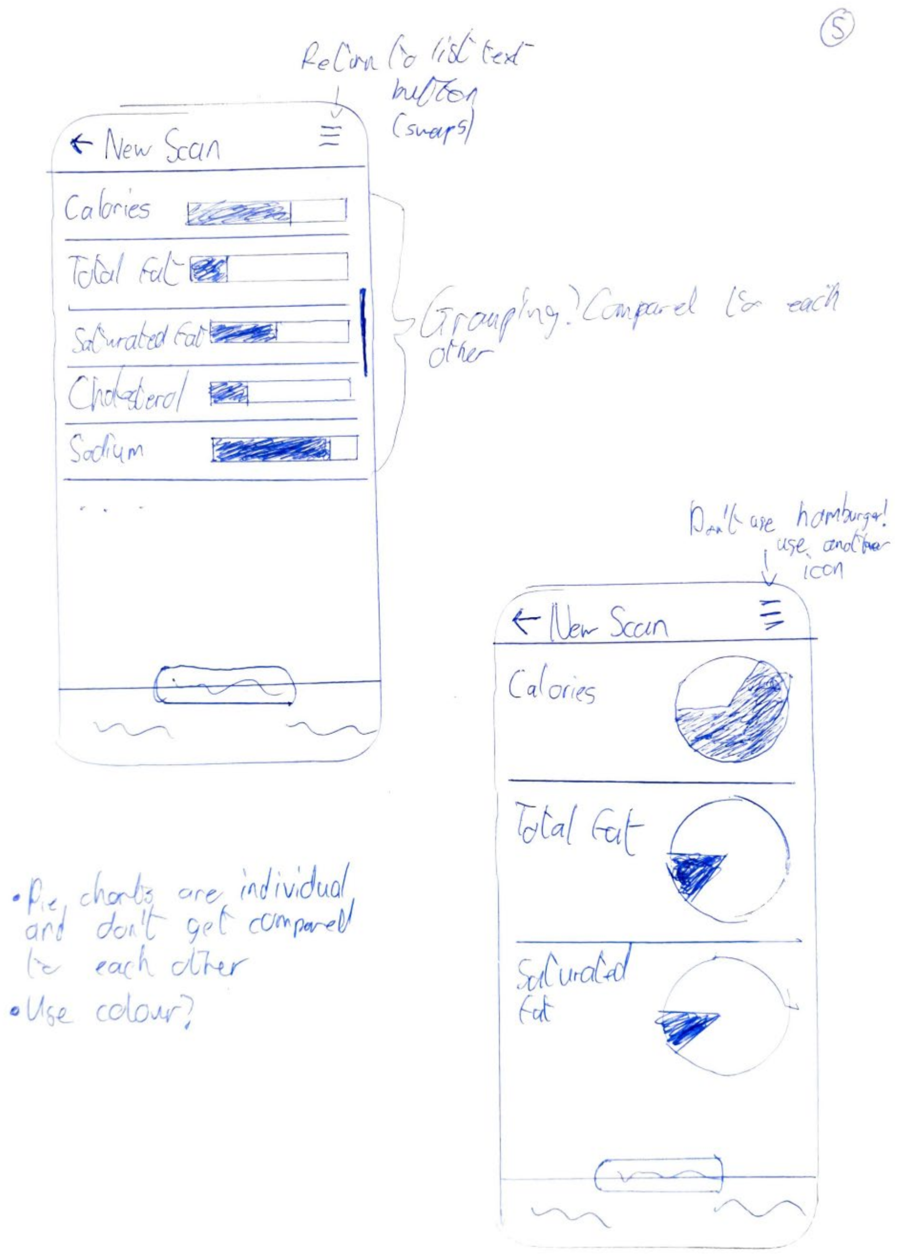

5
Return to list text button (swaps)
← New Scan
Calories
Total Fat
Saturated Fat
Cholesterol
Sodium
(Grouping) Compared to each other
Don't use hamburger! use another icon
← New Scan
Calories
Total Fat
Saturated Fat
• Pie charts are individual and don't get compared to each other
• Use colour?

Figure 12: Low fidelity mockups for the pie charts feature done mid-development

This section has explored how the design of the application took place, including iterative design. Further examples of artifacts created during the design period can be seen in Annex 3. In the chapter that follows, this report describes how development took place, and how the designs and prototypes were made into working code.

Chapter 4

Development

After an initial design and planning stage, the development phase started. This development phase can be separated in two main parallel parts, the logic section, including the creation of the label scanning algorithm, and the front-end development, encompassing both the implementation of UI and UX elements. Using a variety of tools and management techniques, work was organised efficiently to complete as many planned features as possible.

4.1 Management of Work

The work management style was heavily influenced by the development methodology I based my working on, Cowboy coding, which I integrated aspects of after originally planning to use Agile methods. Indeed, Agile methods were originally planned to be used, as they are considered appropriate for UX work (Harper, 2020; Kieffer et al., 2017; Peres et al., 2014), mainly due to their shared cyclical and iterative nature. As such, several small incremental releases were planned, between which UX and UI revisions would be done. However, as this was a one-person experimental project, having several small releases and UX revisions could be a distraction. Indeed, even if the app features increased, the changes in UX would not be big enough to justify doing a complete overhaul of the experience. Consequently, concepts of Cowboy coding were introduced into the working process. This methodology has been lauded as "useful for experimental (...) work because the coding team is often one person", allowing the developer to "rapidly change their algorithms and processes". (Harper, 2020). Most programming and development was done on technologies that I was unfamiliar with, requiring experimentation, trial and error, and quickly adapting the code to the most optimal solution. Therefore, integrating aspects of Cowboy coding allowed to manage time more efficiently than only Agile methods.

A variety of tools and techniques were used to keep track of deadlines and tasks to complete, aiding in organising my time and finishing the project. After having done requirements elicitation, the MoSCoW prioritization technique was employed to determine which features were the most important ones to implement, and which ones could be relegated for later development. GitLab's (GitLab, 2021) integrated issues and milestones tracker was also used to record bugs and tasks that needed to be taken care of. Using the labels feature allowed to group the tasks into the different MoSCoW prioritization stages, helping to faster determine which things should be worked on at any given stage. The use of these tools and techniques ensured that it was always clear what to work on and what needed to be done, even after breaks in development due to exams, for example.

4.2 Platforms

4.2.1 Deployment platform

The application was developed exclusively for the Android platform (Google, 2021c), as my familiarity using it, tools available, and a focus on a single platform would help develop a better product. Indeed, despite never having programmed for Android, I was familiar with the standards for user interfaces and experiences in that platform, as all my phones had Android as their operative system. For this reason, releasing for both iOS and Android would have proved to be a challenge, as more resources would have to be spent resources ensuring that the experience in both platforms was equivalent. In turn, this would have made less time available for development, ultimately deteriorating the quality of the product. Furthermore, Android integrates several useful APIs, such as CameraX or ML Kit, that simplify app development. Therefore, the application was deployed only on Android to ensure that the user experience was as polished as possible.

However, due to my inexperience programming for Android, a lot of development time was spent learning about fundamental Android development concepts. Indeed, my only previous experience developing for mobile was a simple augmented reality app using the Unity platform. As such, I had never developed a full-fledged, Android-first application, nor had I learned about Android development in any of my university courses. Therefore, before I started coding, I needed to learn important concepts like services, resources, intents, and activities, but also how layouts and themes were programmed to ensure complete screen compatibility. Thus, the choice of deployment platform required me to rapidly learn about important and fundamental Android development concepts.

4.2.2 Development environment

Android Studio (Google, 2021d) was chosen as the development IDE due to its native and dedicated support for Android mobile development, as well as for being the official and recommended IDE for Android. While there are many IDEs that support Android development, including one which I am familiar with, Eclipse, development was done exclusively with the official Android IDE developed by Google. This choice would ensure complete Android compatibility and faster development with the most up-to-date code recommendations available. Furthermore, features like its layout visualiser and emulator helped me develop the UI more quickly, even if the application still needed to be run natively on device to ensure that the UI displayed correctly. Due to being officially supported by Google, Android Studio was used as the IDE for this project.

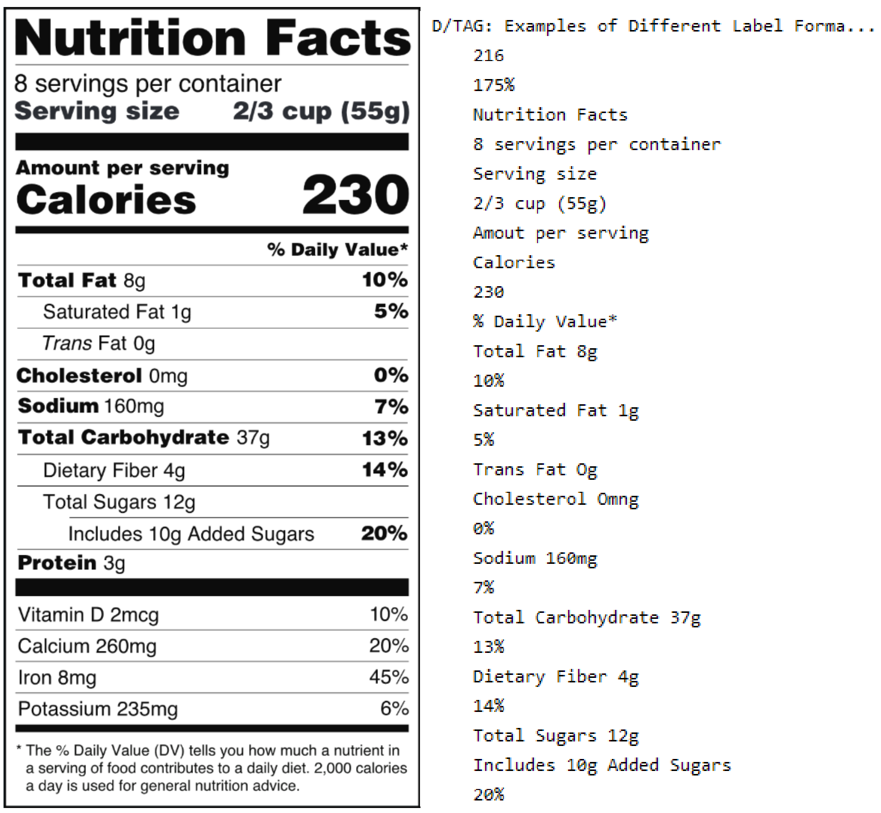
4.3 Programming Language

Android Studio supports writing applications in Kotlin, Java, and C++, but due to my previous experience in university, programming was started using Java. Indeed, I learned Java during the first year Object Oriented Programming with Java courses, and I became more comfortable using it during the second year Software Engineering course units. Because of this previous experience, Java was initially chosen as the programming language for this project, avoiding having to learn another programming language which would reduce the available development time.

However, Google's switch to Kotlin as a preferred language for Android has caused a lack of recent documentation in Java, prompting to switch development to Kotlin. Indeed, Google announced in 2019 that Kotlin was preferred over Java for Android development (Lardinois, 2019). This, from personal experience, made documentation in Java more difficult to find. For instance, the official CameraX starter code lab (Google, 2021e) is only available in Kotlin. Therefore, developed changed direction to use Kotlin to avoid problems finding documentation, despite having no previous experience using the language. Following Google's recommendations, Kotlin was used for the entire project, as finding official Android sometimes non-existent documentation for Java was much more difficult, adding unnecessary complexity and challenges to the project.

4.4 ML Kit

The main feature of the application, scanning nutritional labels and accurately extracting the information from them, was reliant on optical character recognition by design. I settled on using ML Kit's text recognition API (Google, 2021f), as it allowed to fulfil the design requirement of having the app fully run on device and not rely on cloud services. Indeed, ML Kit text recognition component works fully offline, allowing users with limited access to internet services to also use the app. Furthermore, as this API is also developed by Google, it ensures that performance is as fast as possible, enhancing the app's quality and the final user's experience. Using ML Kit allowed to accomplish the offline requirement for the app and make certain that performance was as smooth as possible.



Nutrition Facts

8 servings per container

Serving size	**2/3 cup (55g)**
Amount per serving	
Calories	**230**
	% Daily Value*
Total Fat 8g	**10%**
Saturated Fat 1g	**5%**
Trans Fat 0g	
Cholesterol 0mg	**0%**
Sodium 160mg	**7%**
Total Carbohydrate 37g	**13%**
Dietary Fiber 4g	**14%**
Total Sugars 12g	
Includes 10g Added Sugars	**20%**
Protein 3g	
Vitamin D 2mcg	10%
Calcium 260mg	20%
Iron 8mg	45%
Potassium 235mg	6%

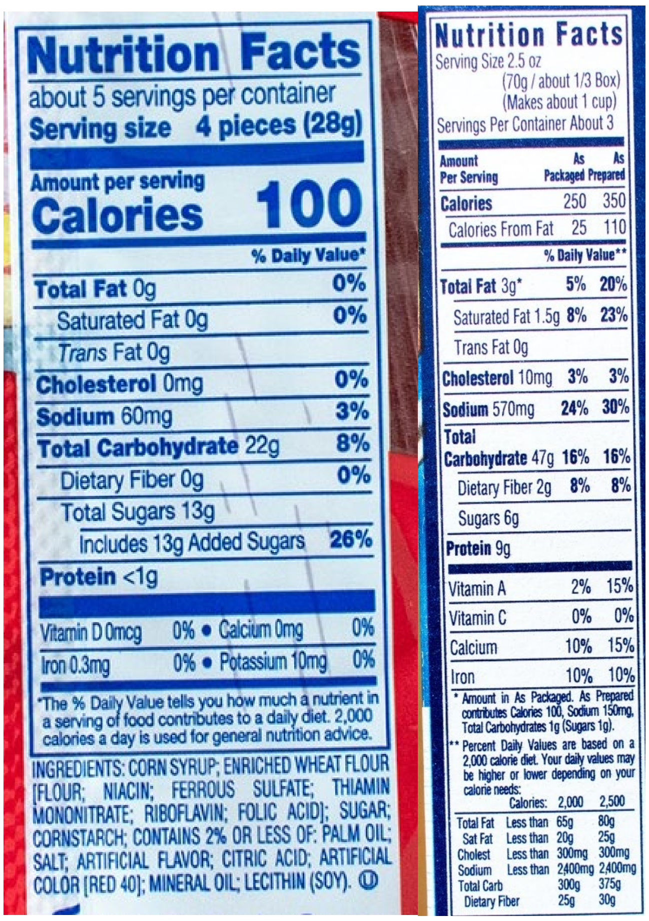
* The % Daily Value (DV) tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

```
D/TAG: Examples of Different Label Forma...
    216
    175%
    Nutrition Facts
    8 servings per container
    Serving size
    2/3 cup (55g)
    Amout per serving
    Calories
    230
    % Daily Value*
    Total Fat 8g
    10%
    Saturated Fat 1g
    5%
    Trans Fat 0g
    Cholesterol Omng
    0%
    Sodium 160mg
    7%
    Total Carbohydrate 37g
    13%
    Dietary Fiber 4g
    14%
    Total Sugars 12g
    Includes 10g Added Sugars
    20%
```

Figure 13: FDA nutrition label (left) and part output of ML Kit text recognition (right). Each line in the recognized text is recognised as an individual paragraph, which contains one line, which contains one or more words.

ML Kit's text recognition API organises a recognised text into separate paragraphs, lines, and words, which allowed to correctly extract the nutritional information from the image. After recognising the text, ML Kit creates a Text object, containing TextBlock, Line, and Element objects. Figure 13 illustrates how ML Kit separates the information of a standard FDA nutritional label. A nutrient's name and value are usually recognised within the same line. It is important to note that the calorie information is an exception to this rule, as its value gets recognised either in the line preceding or succeeding the line containing 'Calories'. This separation allowed to check if any lines contained a keyword corresponding to nutritional information, and if this keyword had an associated value. Extracting the nutritional information from the labels was possible thanks to ML Kit's organisation of the recognised text into distinct paragraphs, lines, and single words.

However, many real-life US nutritional labels do not follow the FDA standard perfectly, so manual testing was required to ensure that the code would work on the widest possible number of labels. For instance, while it was previously mentioned that the 'Calories' value is usually detected in a separate line, some thinner labels, shown in Figure 14, cause the API to recognise the value in the same line. Another example is that of labels containing multiple values and percentages on the same line, which can cause confusion as to which value should be read, also shown in Figure 14. Consequently, the algorithm had to be finetuned to also work on these edge cases. Labels that didn't follow the FDA standard perfectly made extracting the relevant information more difficult, requiring incremented testing and refinement of the code.



Nutrition Facts
about 5 servings per container
Serving size 4 pieces (28g)

Amount per serving
Calories 100

	% Daily Value*
Total Fat 0g	0%
Saturated Fat 0g	0%
Trans Fat 0g	
Cholesterol 0mg	0%
Sodium 60mg	3%
Total Carbohydrate 22g	8%
Dietary Fiber 0g	0%
Total Sugars 13g	
Includes 13g Added Sugars	26%
Protein <1g	

Vitamin D 0mcg	0%	Calcium 0mg	0%
Iron 0.3mg	0%	Potassium 10mg	0%

*The % Daily Value tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

INGREDIENTS: CORN SYRUP; ENRICHED WHEAT FLOUR [FLOUR; NIACIN; FERROUS SULFATE; THIAMIN MONONITRATE; RIBOFLAVIN; FOLIC ACID]; SUGAR; CORNSTARCH; CONTAINS 2% OR LESS OF: PALM OIL; SALT; ARTIFICIAL FLAVOR; CITRIC ACID; ARTIFICIAL COLOR [RED 40]; MINERAL OIL; LECITHIN (SOY). Ⓤ

Nutrition Facts
Serving Size 2.5 oz
(70g / about 1/3 Box)
(Makes about 1 cup)
Servings Per Container About 3

Amount Per Serving	**As Packaged**	**As Prepared**
Calories	250	350
Calories From Fat	25	110
	% Daily Value**	
Total Fat 3g*	5%	20%
Saturated Fat 1.5g	8%	23%
Trans Fat 0g		
Cholesterol 10mg	3%	3%
Sodium 570mg	24%	30%
Total Carbohydrate 47g	16%	16%
Dietary Fiber 2g	8%	8%
Sugars 6g		
Protein 9g		
Vitamin A	2%	15%
Vitamin C	0%	0%
Calcium	10%	15%
Iron	10%	10%

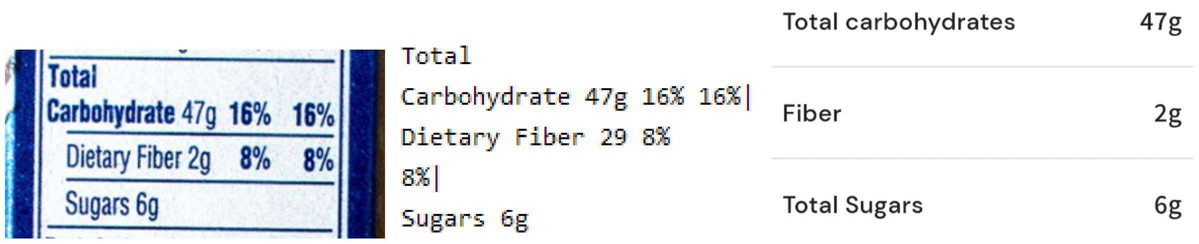
* Amount in As Packaged. As Prepared contributes Calories 100, Sodium 150mg, Total Carbohydrates 1g (Sugars 1g).
** Percent Daily Values are based on a 2,000 calorie diet. Your daily values may be higher or lower depending on your calorie needs:

	Calories:	2,000	2,500
Total Fat	Less than	65g	80g
Sat Fat	Less than	20g	25g
Cholest	Less than	300mg	300mg
Sodium	Less than	2400mg	2400mg
Total Carb		300g	375g
Dietary Fiber		25g	30g

Figure 14: From left to right: label where the 'Calories' value is read in the same line as the nutrient name, label where multiple values and percentages are displayed

Furthermore, ML Kit's text recognition API is not perfect, so manually testing the algorithm with edge cases allowed me to build a more robust product. Indeed, as the optical character recognition runs fully offline, it uses less computational power, being more prone to error than a cloud-based system. A notable example of this is how ML Kit sometimes recognises two separate values as being part of one individual bigger value, as it does not recognise the space separating the two values. Another important example, shown in Figure 15, is how due to the similarity between the lower case 'g' and the digit '9' in an FDA label, ML Kit will sometimes recognise the former as the latter, resulting in an erroneous reading. These issues could not be fully fixed as it was the text recognition API that failed and not my produced code, but they can both be mitigated by checking that the extracted value is not too large, truncating the value to the first digits if required, while also verifying if the line were the value is contained a unit. If not, this meant that the unit had been interpreted as a number and had to be removed from the recognised value. As ML

Kit can error when reading the labels due to cramped text, additional checks were included to ensure that the parsed values were correct and coherent.



```
Total
Carbohydrate 47g 16% 16%|
Dietary Fiber 29 8%
8%|
Sugars 6g
```

Total carbohydrates	47g
Fiber	2g
Total Sugars	6g

Figure 15: Clockwise from top left: detail of nutrition label, ML Kit incorrectly reading the 'Dietary Fiber' value as '29' instead of '2g', value correctly displaying in the application

4.5 User Interface

4.5.1 Building layouts

The core of UI programming was done through XML, one of the main ways to code layouts and interfaces in Android. Indeed, each Activity, or app screen, has an associated XML layout specifying the elements that should be present when running said activity, including buttons, text fields, and app bars. Despite never having written a layout in XML before, I was able to quickly learn how to do so thanks to my skills creating CSS layouts for websites. Indeed, both systems share some common concepts. For instance, using ConstraintLayout in Android to define an element's positioning relative to another element is like using the position attribute in CSS. Furthermore, as the app needed to be as versatile as possible, I built separate vertical and horizontal layouts, as seen in Figures 16 and 17, creating a more pleasant and accessible user experience. Using XML, layouts were coded that adapted to screens of all forms and sizes, ensuring a consistent and efficient user experience.

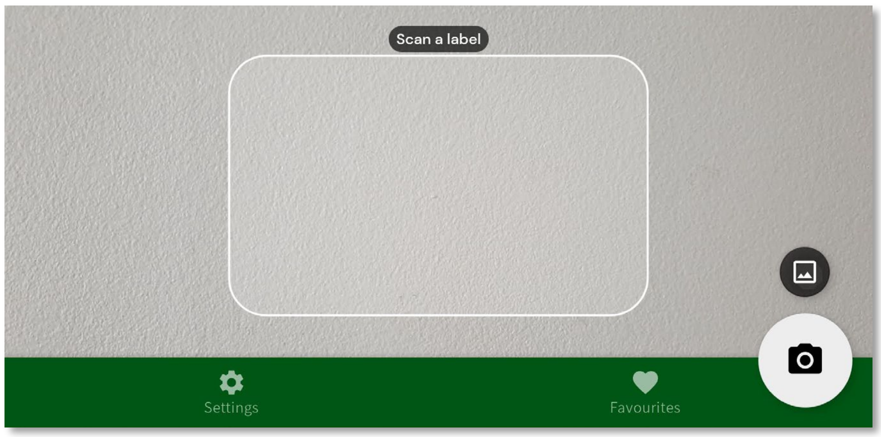


Figure 16: Landscape layout of the main scanning screen, with the buttons being shifted to an easier to reach position

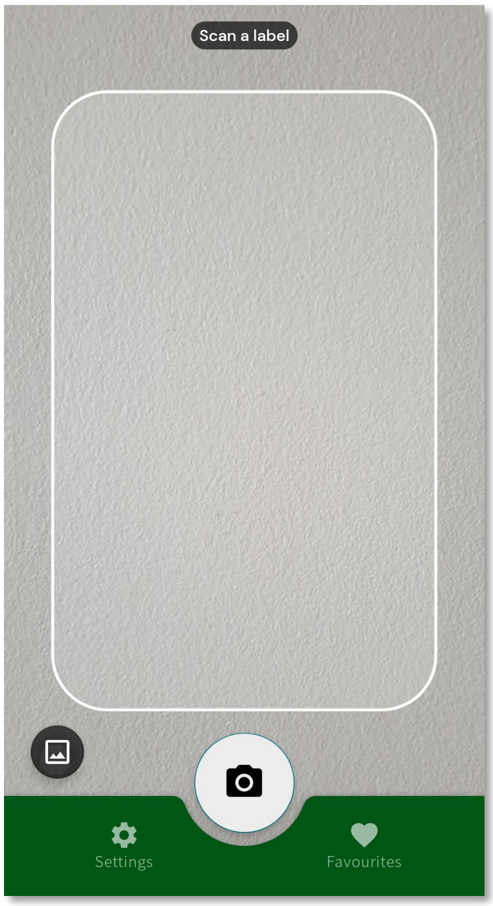


Figure 17: Vertical layout of the main scanning screen

4.5.2 Enhancing feedback

Nonetheless, using solely XML was limiting, as it is impossible to define some indispensable elements necessary for fluid interaction and feedback. Indeed, Norman (2013, p.72) details that feedback is necessary in any interactive system, and that good feedback is present if, 'after an action has been executed, it is easy to determine the new state'. For example, in early builds of the app, the camera preview did not freeze or stop after pressing the camera button, giving the impression that the button press had no effect. Using additional Kotlin code on top of the XML code, I the camera preview froze after pressing the camera button. This way, the user has the impression that the app is analysing the current photo shown frozen on screen. This feedback is further enhanced by system dialogs, also controlled with Kotlin code. Moreover, so that the user gets information in both a visual and auditory manner, sound feedback was added using earcons, long recognised for being an effective method to improve accessibility and information communication (Brewster et al., 1993). Using additional Kotlin code on top of the XML layout code, feedback was enhanced, providing a more responsive user experience following industry-wide guidelines.

4.5.3 Customisability and accessibility

No two users are the same, and as it is impossible to create a single experience that is adequate for everyone, multiple customisability options were implemented to build a flexible and malleable app that could adapt to the user's preferences. User experience is subjective by nature, as it is dependent on the user's inner state, including their sense of aesthetics, perception, and cognition. For example, as described on Chapter 3, changing the system theme to a dark theme can be beneficial to many users, whereas for some, including people with astigmatism, dark theme hinders reading. Because of this issue, the user can choose their preferred theme. Other options included toggling earcons on and off, and reducing animations, useful for people with vestibular disorders. These options were implemented using a separate and dedicated preferences activity for easy access and creating an interaction hierarchy. Indeed, coding a separate activity for customisation permitted to separate this feature from the others. Granting the ability of choice can greatly improve the user's experience, as the app can better adapt to their personal preferences.

Special care was taken to ensure that screen readers worked perfectly with the application, including creating screen reader labels programmatically. One of the most popular and useful Android assistive services is TalkBack, a screen reader that also provides spoken feedback and enhanced interaction for people who are blind or partially sighted. TalkBack uses special 'contentDescription' XML tags, which describe what the purpose of an element, like a button, is. For text content, TalkBack automatically reads out the text, but this could cause comprehension issues. For instance, while the service correctly reads out 'g' as 'grams', it does not understand that 'mcg' stands for 'micrograms'. Moreover, certain text descriptions appropriate for visual users are not for those that rely on audio feedback. For example, the application presents the daily value of a nutrient as 'Calories DV: 5%', which can be confusing to hear. Therefore, contentDescription tags are changed programmatically to a more natural spoken structure, like 'Calories: 5% of the recommended daily value'. Other steps were taken to ensure that navigation using TalkBack was as smooth as possible, such as marking decorative elements not important for accessibility, making TalkBack ignore these when the user navigates the application, improving the user experience. Compatibility with Android assistive services was done by creating XML contentDescription tags, as well as programmatically changing these depending on the shown content for a more natural experience.

4.6 MPAndroidChart

The implementation of the visual representation through pie charts of the nutrient information was done with the third-party library MPAndroidChart (Jahoda, 2019), although its limitations required some workarounds to display the pie charts as desired. Despite Material design including guidelines on how to design data visualisations, there is no official library to create these on Android, so this project had to use a third-party library, MPAndroidChart. This library included some useful features, like automatically calculating percentage values and integrated animations, but additional work was needed to make the charts visually consistent with the rest of the application. For example, the pie charts are colour-coded, as seen in Figure 18, but it employs different colours depending on the chosen theme. Before displaying the charts, the theming value had to be retrieved from settings, and then colours had to be applied individually depending on the value of the nutrient. Aside from small workarounds, using MPAndroidChart to implement the pie charts feature was relatively straightforward and without any major complications.

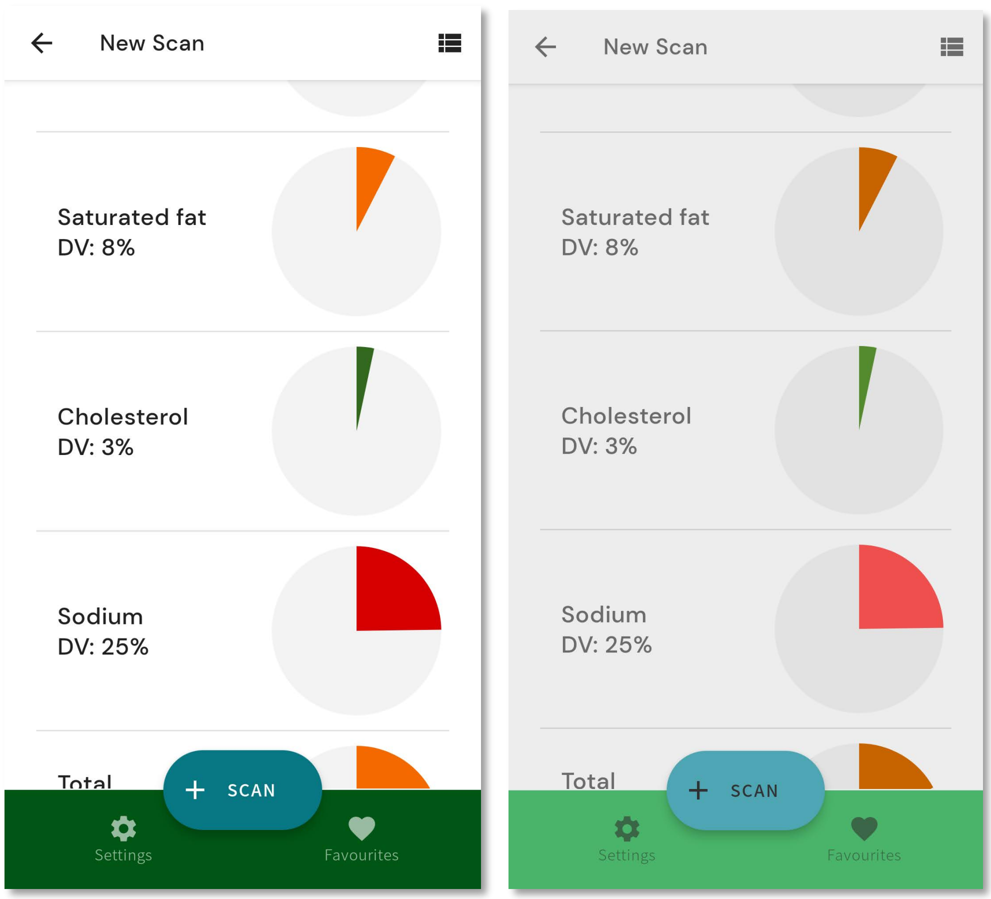


Figure 18: When the pale theme is selected, the pie charts will also use more muted colours

Chapter 5

Results

The final application successfully makes American FDA nutrition labels more accessible. Indeed, it can scan a label and successfully extract the relevant nutrition information, displaying it in a more accessible manner. The application is fully accessible and integrated with Android's assistive services, opening nutritional information for many people that could not easily read printed nutrition labels.

The features in the final version of the application include:

- Be able to scan FDA nutrition labels and extract the relevant information.
- Give the user useful feedback of the application's state through animations, alert dialogs, and sound.
- Present the nutritional information using a visual representation with pie charts, helping the user understand more quickly how much of a nutrient a food product contains.
- Be able to choose an image from gallery instead of scanning it directly through the application.
- Be able to receive a shared image and scan it for nutritional information.
- 3 full-fledged UI themes.
- Fully designed for and compatible with Android's native accessibility services, like TalkBack or Voice Access, as well as with other third-party screen readers.
- Fully accessible UI following the latest Material guidelines.
- Customisation options to adapt to the user's personal preferences. These include UI theming, enabling reduced motion for people with vestibular disorders, toggling double tap back to avoid accidentally exiting the app, and toggling sound feedback.

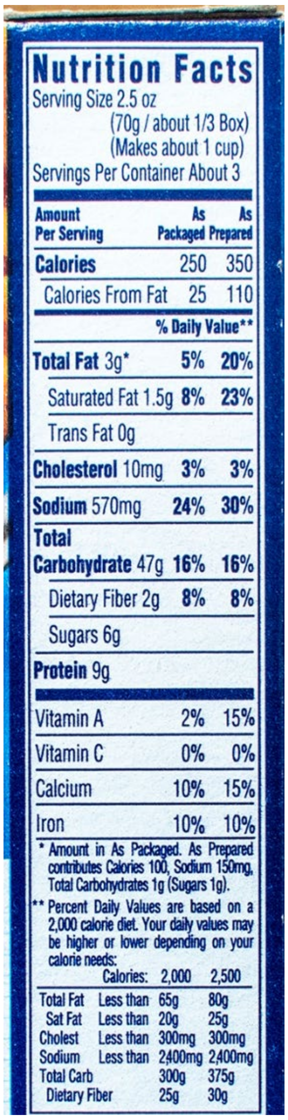
Chapter 6

Evaluation

The application evaluation was not done in one single stage, but throughout the entire project development. As it was briefly touched upon on Chapter 3, a version of iterative design was used in this project, which, despite being simplified, maintained the 'Evaluation' stage in the process. As such, evaluation took place multiple times as development progressed. Doing both software and user continuous evaluation allowed to create a more polished and robust application.

6.1 Software Evaluation

Software evaluation was performed manually using edge cases, nutrition labels that had singular and non-standard characteristics that produced errors when scanning them. Indeed, after the application worked with the most common FDA standard nutrition label, I bought multiple American products with labels that the application struggled with. As was mentioned in Chapter 4, some labels had characteristics that made it necessary to refine the label parsing algorithm, like misinterpreting the 'g' symbol standing for grams for the digit '9'. Moreover, one of the labels tested, shown in Figure 19, did not follow the new standards, applicable from 2020. However, that product was purchased in February 2021, but the label was nonetheless used to test the application to ensure maximum compatibility even among non-standard labels. Furthermore, as these were real, physical products, the software evaluation had a better ecological validity, as the application was tested using real-life use cases. Testing these labels allowed to catch errors and bugs in the label scanning algorithm, as well as extending the application's compatibility to a bigger number of labels. Software evaluation was done by trying to find use cases that the application could not properly handle, figure the reason why, and fixing the related issue, making the application more robust, reliable, and accurate overall.



Nutrition Facts
Serving Size 2.5 oz
(70g / about 1/3 Box)
(Makes about 1 cup)
Servings Per Container About 3

Amount Per Serving	**As Packaged**	**As Prepared**
Calories	250	350
Calories From Fat	25	110
	% Daily Value**	
Total Fat 3g*	**5%**	**20%**
Saturated Fat 1.5g	**8%**	**23%**
Trans Fat 0g		
Cholesterol 10mg	**3%**	**3%**
Sodium 570mg	**24%**	**30%**
Total Carbohydrate 47g	**16%**	**16%**
Dietary Fiber 2g	**8%**	**8%**
Sugars 6g		
Protein 9g		
Vitamin A	2%	15%
Vitamin C	0%	0%
Calcium	10%	15%
Iron	10%	10%

* Amount in As Packaged. As Prepared contributes Calories 100, Sodium 150mg, Total Carbohydrates 1g (Sugars 1g).
** Percent Daily Values are based on a 2,000 calorie diet. Your daily values may be higher or lower depending on your calorie needs:

	Calories:	2,000	2,500
Total Fat	Less than	65g	80g
Sat Fat	Less than	20g	25g
Cholest	Less than	300mg	300mg
Sodium	Less than	2400mg	2400mg
Total Carb		300g	375g
Dietary Fiber		25g	30g

Figure 19: Nutrition label bought in February 2021 still following pre-2016 FDA standards, which became enforceable in 2020

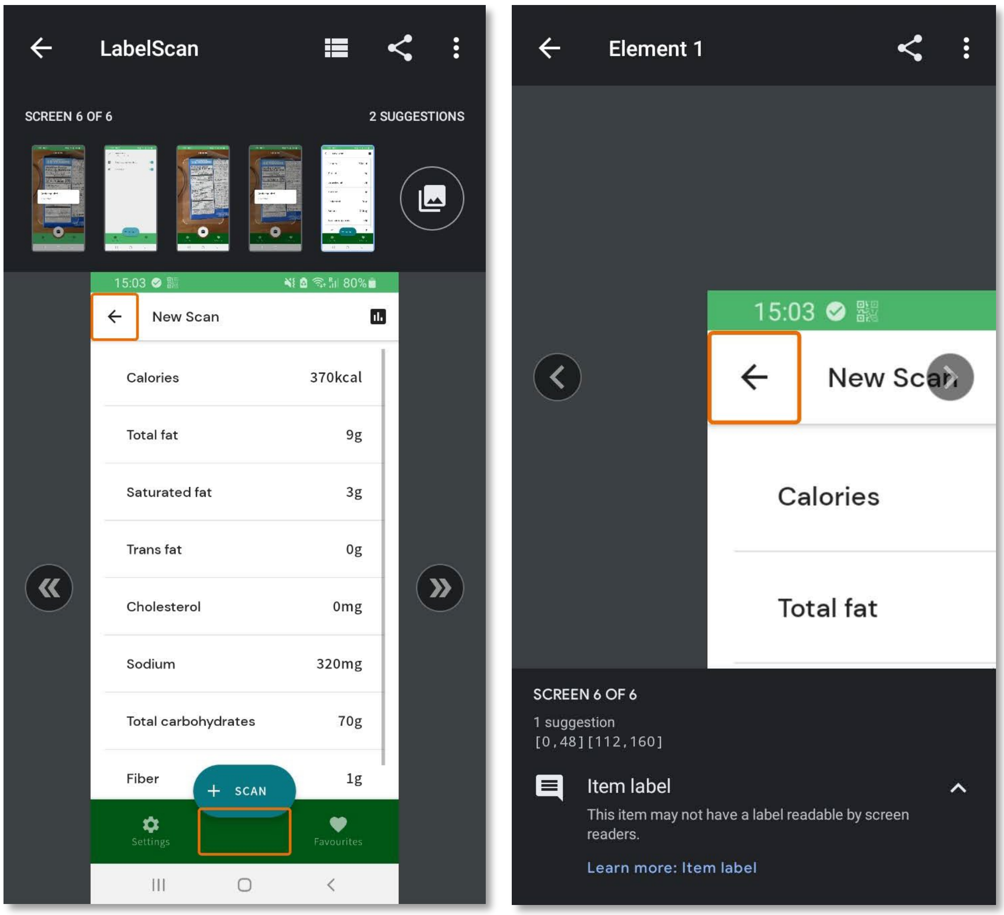
6.2 User Evaluation

User evaluation was marked by the COVID-19 pandemic and the impossibility of receiving direct feedback from users, resulting in a limited user evaluation where I was the only participant. User evaluations are usually done in collaboration with users, using interviews, surveys, but also walkthrough observations and think-aloud processes, where both qualitative and quantitative data can be gathered. However, this was impossible to do due to the pandemic, as has been explained in previous chapters. As a result of this, user evaluation had to be done by me. As a result of the COVID-19 pandemic and its lockdowns, user evaluation had to be done individually by me, as traditional methods involving users were not possible.

Using my personal HCI and UX knowledge in combination with other methods, I was able to judge the product and evaluate how accessible and effective it was, making appropriate changes. Indeed, my previous knowledge HCI and UX knowledge proved not only beneficial during the design and development phases, but also to perform user evaluation lacking users. For instance, I repeatedly tested the application using Android's accessibility services to verify that the user experience was as accessible as required, taking notes on how attention was driven by the layout and the use of colour, finding ways of how to improve it. Other, less conventional tests were also used, like using the application without my glasses or contacts to see how accessible it was for people with low vision, or using the application with my non-dominant hand to verify the UI layout was accessible, a method borrowed from Epicurious (2020). It is important to note that this method of individual evaluation was more prone to bias, as there was only one point of view involved, and there was personal attachment to the application due to having also developed it. Because of this risk steps were taken to ensure that bias did not cloud the final judgement in all instances of user evaluation. User evaluation was done multiple times using HCI and UX skills gained in previous university course, employing a diverse number of methods while trying to adjust for bias as much as possible.

Although there was no user testing, Roger Broadbent, head of the Dyslexia Institute UK and a respected expert on accessibility, kindly gave feedback on the gathered user requirements, planned features, and other more general aspects of the application, helping validate certain choices. Indeed, mid-way through the development phase, Roger gave feedback through a Zoom call, in which I did a short demonstration of the application and gave a rundown of the features that were planned to be implemented. Roger gave praise to the overall concept of the project and suggested that a history of previously scanned labels was saved, which was one of the requirements found during the design phase. Feedback from Roger Broadbent, head of the Dyslexia Institute UK, helped ensure that the application's concept, design, and planned features would help make nutrition labels more accessible for people with dyslexia.

Finally, different automated tools were used to confirm that the application was overall accessible. While automated tools cannot capture the qualitative nature of user experience, they are useful in obtaining quantitative data. Indeed, multiple tools are available that evaluate different aspects that affect the accessibility of a product, like colour contrast or layout. For this project, the Accessibility Scanner application (Google, 2021g) was used to identify issues in the design, as seen in Figure 20. The tool takes screenshots while one uses the application, creating a report listing all the found issues. Indeed, Google (2021g) revealed problems in text contrast and missing item labels used by screen readers for some items. The final version of the application was also tested on all three of the themes, with no suggestions for improvement found for either of them. Automated tools helped make certain that all accessibility errors made during the design and development stages were found, helping create a better final product.



Calories	370kcal
Total fat	9g
Saturated fat	3g
Trans fat	0g
Cholesterol	0mg
Sodium	320mg
Total carbohydrates	70g
Fiber	1g

SCREEN 6 OF 6
1 suggestion
[0,48][112,160]

Item label
This item may not have a label readable by screen readers.
Learn more: Item label

Figure 20: Accessibility Scanner pointing out that certain user interface elements were not accessible by assistive services like TalkBack

This chapter has described the methods used for both user and software evaluation, explaining the impact COVID and the particularity of the project had on these. In the chapter that follows, the results and conclusions of these evaluations will be presented.

Chapter 7

Reflection, Future Work, and Conclusion

Looking back at the project and the final version of the application, there are many reasons to be satisfied. Notably, the scanning feature works with a wide array of American FDA labels, while the pie chart feature provides an effective way of understanding the information in the label visually. However, some errors and mistakes were also made, notably in project organisation and the lack of certain planned features. Nonetheless, I believe that the application serves as a robust base for future development, as well as a good demonstration of user-centred design while lacking users.

7.1 Project Goals

Many of the original project goals were successfully implemented, while some could not make it in due to time constraints. Indeed, all the features ranked as 'MUST' features are present. The application is fully accessible, as it is compatible with Android's assistive services, and uses accessible fonts and colours. The user experience has also been built following industry-wide guidelines, using animations, layout, and sound to enhance feedback. However, some planned features categorised as 'SHOULD' features in the design stage are missing, most notably font customisation. Indeed, the lack of this feature can be seen as the biggest failure of the project. Nonetheless, the implementation of all the most important requirements, as well as some of secondary importance, makes this project overall successful.

Some of the originally planned features were scrapped after development started, with other requirements not originally found in the first design stage being implemented instead. Indeed, some requirements were deemed as not necessary or not having substantial benefit midway through development. For instance, feature of linking to health websites where users could gain more information about nutrients was scrapped, as it would create visual clutter for a feature that a simple internet search already accomplishes. Moreover, as more user evaluation was completed, it became evident that some additional features needed to be implemented. For example, the ability to select an image from gallery or share an image to the application was added to make the application work more like other Android applications that use the camera. While some requirements found during the first design stage were deemed as not necessary by further user evaluation, other features that were not part of the original project goals were added as important requirements.

This application's main requirement, scanning labels, works well for a variety of FDA labels, but flaws in its technical design may limit future development. Despite technically fulfilling the original requirement, a technical rework would be needed to make scanning even more robust and versatile.

7.2 Possible Future Work

7.2.1 Short-term objectives

The most important action to do in the short-term would be to conduct extensive user testing. Indeed, as it has been pointed out multiple times in this paper, it was not possible to conduct user testing. As such, I cannot say with complete confidence how useful the final application is, or how adequately it responds to user requirements. User testing would be used not only to re-explore user requirements, but also to evaluate the user interface and features of the final application. User testing would therefore be the most significant short-term change possible to truly make this a project built using user-centred design.

Implementing more of the planned features would also be an improvement to do in the short-term, most notably the scanning diary and switching fonts features. Indeed, I was not able to create these features due to time constraints, preferring to focus on polishing the already existing functionality during the last two weeks rather than implementing barely functioning features. Being able to switch fonts would be the best way to improve accessibility for people with dyslexia, as was indicated previously in Chapter 3, while the scanning diary could help speed up and simplify the process of finding nutritional information. Implementing these planned features would significantly improve the quality and usefulness of the product.

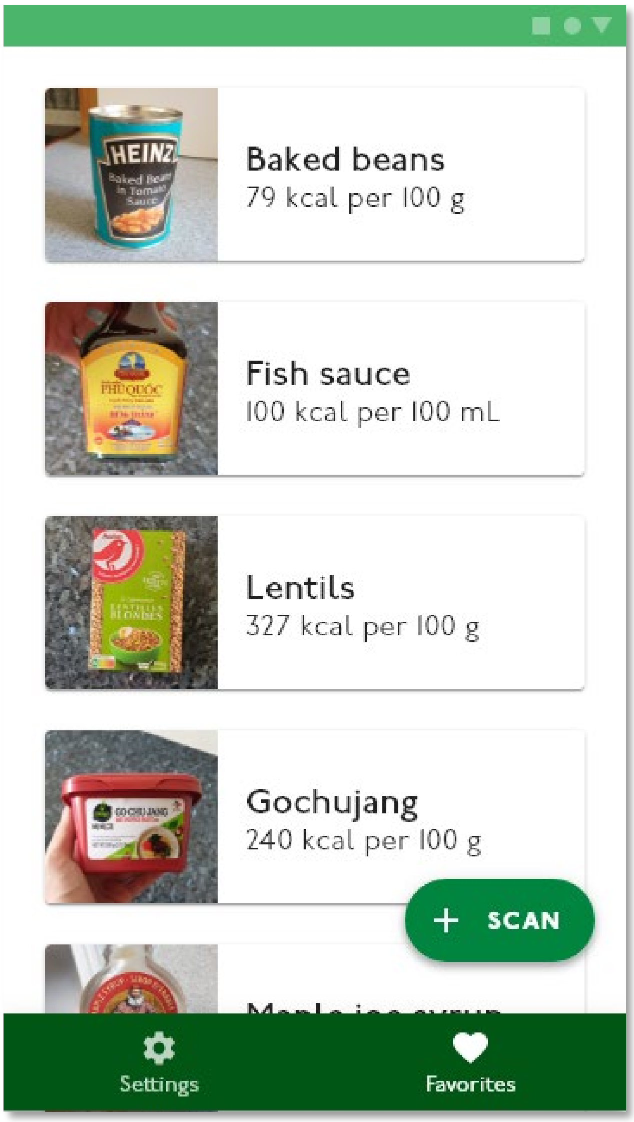
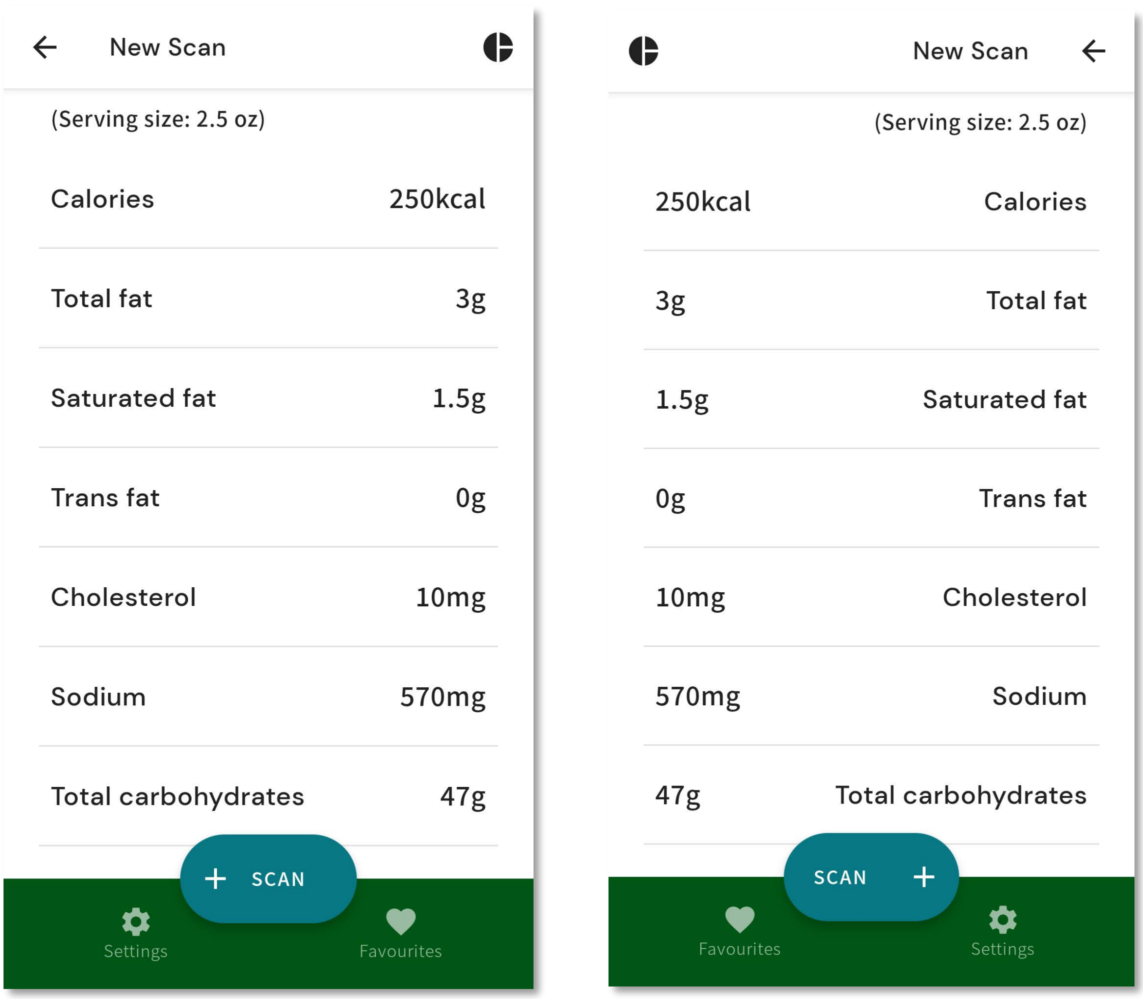


Figure 21: Prototype of the scanning diary/favourites screen done in the initial design stage

Finally, small fixes and refinements would help make the app more polished, creating a better user experience. For instance, when animations are enabled after scanning a label, the shown progress indicator at the top of the screen might freeze for around 50 frames, giving the impression that the app lags. This is caused by CameraX passing the captured image to ML Kit, as this slowdown does not appear when the image is chosen from gallery. A possible solution to this problem could be creating two separate threads for the UI and the text recognition. Another potential change could be to increase the use of Fragments instead of Activities, which would always maintain the bottom navigation present instead of having it disappear when going to a different Activity, creating a faster and more polished interaction. Small code changes and fixes could significantly improve the overall user experience.

7.2.2 Long-term objectives

Long-term objectives include features that were originally ranked as ‘WOULD/WON’T’ features using the MoSCoW system, mainly scanning labels other than FDA ones, but also new features that were not part of the original requirements, like complete localisation. Indeed, some of these features could help make the product completer and more useful. For instance, parts of the code have been written to support other languages than English. In fact, the order of the elements in the detailed scan screen are based on the language locale the user employs. For example, a user whose phone is configured in English will see ‘Calories 370kcal’, while a user whose phone is configured in Arabic would see ‘370kcal Calories’ as Arabic uses right-to-left script. An illustration of this can be seen in Figure 22. Of course, there would need to be a complete localization overhaul for any supported language, like translating UI text, changing the colours used in pie charts to adapt to regional differences in interpreting colour, and being able to analyse other labels than American FDA ones.



(Serving size: 2.5 oz)

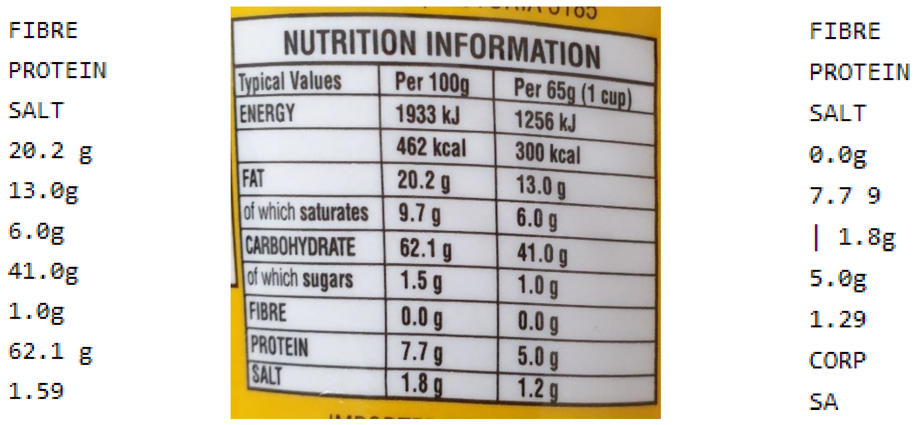
Calories	250kcal
Total fat	3g
Saturated fat	1.5g
Trans fat	0g
Cholesterol	10mg
Sodium	570mg
Total carbohydrates	47g

(Serving size: 2.5 oz)

250kcal	Calories
3g	Total fat
1.5g	Saturated fat
0g	Trans fat
10mg	Cholesterol
570mg	Sodium
47g	Total carbohydrates

Figure 22: Comparison of current behaviour when the phone language is set to English (left), and when it is set to a language with right-to-left script (right). Localization would also include text translation of UI elements

However, the most important long-term objective would be completely changing the way nutrition labels are scanned to widen compatibility with different labels, including ones that do not follow set standards. While the current method works well for FDA labels, it is not sufficient for other labels, like Australian ones. Indeed, the order in which the text is recognised is not consistent, as shown in Figure 23. As such, a more technically advanced method, like recognising text based on its physical position, would be necessary. This method would be possible to do with just ML Kit's optical character recognition capabilities, but it would have trouble with non-standard labels, as the position of the nutrients in the label could not be predicted. Moreover, optical character recognition sometimes fails and misreads values. A potential solution to this issue could be taking multiple photos of the label automatically when the user presses the scan button and displaying the most frequent value for each nutrient. To build an application that is truly compatible with all nutrient labels, regardless of country of origin or standardisation, advanced machine learning methods outside of my field of expertise may be required. Further testing would be required to find a feasible way to expand this app's capabilities to all nutrition labels.



```
FIBRE
PROTEIN
SALT
20.2 g
13.0g
6.0g
41.0g
1.0g
62.1 g
1.59
```

NUTRITION INFORMATION

Typical Values	Per 100g	Per 65g (1 cup)
ENERGY	1933 kJ	1256 kJ
	462 kcal	300 kcal
FAT	20.2 g	13.0 g
of which saturates	9.7 g	6.0 g
CARBOHYDRATE	62.1 g	41.0 g
of which sugars	1.5 g	1.0 g
FIBRE	0.0 g	0.0 g
PROTEIN	7.7 g	5.0 g
SALT	1.8 g	1.2 g

```
FIBRE
PROTEIN
SALT
0.0g
7.7 9
| 1.8g
5.0g
1.29
CORP
SA
```

Figure 23: Two consecutive partial readings of the same UK nutrition label by ML Kit, with no predictable element order

7.3 Project Management

While I was able to manage my time efficiently for the most part, in retrospect the start of development could have been more focused, which could have in turn made it possible to complete at least one extra feature. As explained in Chapter 4, I used a variety of tools like GitLab's issue tracker combined with the MoSCoW prioritisation technique that helped me organise my work efficiently. However, it is true that during the early stages of development I did not manage my time properly. Indeed, I started development after the design phase, around the same time that I needed to start studying for semester exams and complete other important assignments. After the winter exams and assignments were done, I experienced burnout. This made it difficult to focus on development, leading to an extremely slow initial progress, which ultimately resulted in development being hampered. Indeed, during the middle and last phases of development, I was playing catch-up on development time. Had I managed work and burn out better, it is probable that more features could have been completed and thereby making a more complete product. Even if I was able to get organised during development, the start slowed down due to burnout, restraining the rest of development. In retrospect, a better management of time at the start of code development could have yielded a more polished application.

7.4 Personal Achievements

This project has been a big learning opportunity and a useful playground to further develop my user experience skills. I am currently in an HCI degree, which focuses on multiple, interdisciplinary aspects of user experience. However, despite having multiple course units surrounding these aspects, I had never had the opportunity to apply my skills practically. This project has allowed me to not only use the knowledge gained in the user experience course unit, but also those from other subjects, like psychology, useful for knowing how to drive attention and perception, and anthropology, which helped me account for cultural differences in the use of technology. As such, I gained valuable practical experience on designing and creating good UX that will be extremely useful in the future.

I have also gained valuable experience and knowledge on Android development, a field that I had never tackled before. Indeed, all my front-end and back-end development before this project had been done for web applications, and even if I had already created some mobile-facing interfaces, Android applications work completely differently. Therefore, I had to learn many fundamental aspects of Android development, like Activities, Fragments, RecyclerViews, Shared Preferences, and building screen-responsive layouts using XML. As such, I can now confidently say that I am able to design and develop applications for Android.

Finally, this project has been an opportunity to learn the Kotlin language. Indeed, almost the entirety of the codebase was built using Kotlin, a programming language that I had never used before. Building this project has been a good method of learning the language and its functional aspects.

7.5 Overall Conclusions

In conclusion, I cannot help but have mixed feelings about this project. On one hand, this project covers several aspects of computer science, and many of the project goals have been achieved. On the other hand, better organisation at the start of development could have yielded a better product overall. Nonetheless, I do believe that this application succeeds in achieving what it was meant to do, making nutrition labels more accessible, and that it constitutes a good base to continue development in the future.

References

Adobe (2021a). *Adobe XD* (Version 39.0.12). [Windows program]. Available at: https://www.adobe.com/uk/products/xd.html (Accessed: 26 April 2021).

Adobe (2021b). *Adobe XD* (Version 39.0.0). [Android program]. Available at: https://play.google.com/store/apps/details?id=com.adobe.sparklerandroid (Accessed: 26 April 2021).

Adobe (2021c). *Accessible color palette generator | Adobe Color*. Available at: https://color.adobe.com/create/color-accessibility (Accessed: 27 April 2021).

Arhippainen, L. & Tähti, M. (2003). 'Empirical evaluation of user experience in two adaptive mobile application prototypes', *MUM 2003. Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia*, December 2003. Citeseer. pp. 27-34. Available at: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.99.7147&rep=rep1&type=pdf (Accessed: 26 April 2021).

Budiu, R. (2020). 'Dark Mode vs. Light Mode: Which Is Better?', *Nielsen Norman Group*, 2 February [Online]. Available at: https://www.nngroup.com/articles/dark-mode/ (Accessed: 27 April 2021).

MyFitnessPal, Inc. (2021). *MyFitnessPal* (Version 21.5.1). [Android program]. Available at: https://play.google.com/store/apps/details?id=com.myfitnesspal.android (Accessed: 25 March 2021).

Bjorn the UX Dog (2020). 'Accessibility: how to involve dyslexic users into your design', *UX Collective*, April 12. Available at: https://uxdesign.cc/accessibility-how-to-involve-dyslexic-users-into-your-design-aab031ee588d (Accessed: 24 April 2021).

Boyarski, D., Neuwirth, C., Forlizzi, J. & Regli, S. H. (1998). 'A study of fonts designed for screen display', *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Los Angeles, California, USA. 18th – 23rd April 1998. Los Angeles, California, USA: ACM Press/Addison-Wesley Publishing Co., pp. 87–94.

Brewster, S. A., Wright, P. C. & Edwards, A. D. N. (1993). 'An evaluation of earcons for use in auditory human-computer interfaces', *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, Amsterdam, The Netherlands. 24-29 April 1993. Association for Computing Machinery. pp. 222–227. Available at: https://doi.org/10.1145/169059.169179 (Accessed: 07 April 2021).

Buchenau, M. & Suri, J. F. (2000) 'Experience prototyping', *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*, New York City, New York, USA. August 2000. Association for Computing Machinery. pp. 424–433. Available at: https://doi.org/10.1145/347642.347802 (Accessed: 26 April 2021).

Chen, A. (2019). 'The controversy of accessible type', *Queer Design Club*, 12 October. Available at: https://medium.com/queer-design-club/the-controversy-of-accessible-type-8def04eb8808 (Accessed: 27 April 2021).

Clary, P. (2019). 'With Lookout, discover your surroundings with the help of AI', *The Keyword*, 12 March. Available at: https://www.blog.google/outreach-initiatives/accessibility/lookout-discover-your-surroundings-help-ai/ (Accessed: 21 March 2021).

Consumer Affairs Agency (2020). *栄養成分表示*. Tokyo, Japan. Available at: https://www.caa.go.jp/policies/policy/food_labeling/health_promotion/assets/food_labeling_cms206_20210318_01.pdf (Accessed: 26 April 2020).

Department of Health (2016). *Guide to creating a front of pack (FoP) nutrition label for pre-packed products sold through retail outlets*. Available at: https://www.gov.uk/government/publications/front-of-pack-nutrition-labelling-guidance (Accessed: 10 April 2021).

Epicurious (2020). *5 Egg Kitchen Gadgets Tested by Design Expert | Well Equipped | Epicurious* [YouTube]. Available at: https://www.youtube.com/watch?v=byiVjrJaOnc (Accessed: 28 April 2021).

Food and Drug Administration (2016). *Food Labeling: Revision of the Nutrition and Supplement Facts Labels*. Available at: https://www.regulations.gov/document/FDA-2012-N-1210-0875 (Accessed: 10 April 2021).

GitLab (2021). *Iterate faster, innovate together | GitLab*. Available at: https://about.gitlab.com/ (Accessed: 06 April 2021).

Goldstein, E.B. (2016). 'Perceiving Objects and Scenes', in Brockmole, J.R. (eds.) *Sensation and perception*. 10th edn. *Cengage* [Online], pp.92-123. Available at: https://read.kortext.com/reader/epub/282538 (Accessed: 27 April 2021).

Google (2021a). *Homepage – Material Design*. Available at: https://material.io/ (Accessed: 25 April 2021).

Google (2021b). *Bottom navigation – Material Design*. Available at: https://material.io/components/bottom-navigation#research (Accessed: 27 April 2021).

Google (2021c). *Android | The platform pushing what's possible*. Available at: https://www.android.com/ (Accessed: 07 April 2021).

Google (2021d). *Download the official Android IDE and developer tools to build apps for Android phones, tablets, wearables, TVs, and more*. Available at: https://developer.android.com/studio (Accessed: 07 April 2021).

Google (2021e). *Getting Started with CameraX*. Available at: https://codelabs.developers.google.com/codelabs/camerax-getting-started#0 (Accessed: 04 April 2021).

Google (2021f). *Text Recognition | ML Kit | Google Developers*. Available at: https://developers.google.com/ml-kit/vision/text-recognition (Accessed: 06 April 2021).

Google (2021g). *Accessibility Scanner* (Version 2.2.1.351925010) [Android program]. Available at: https://play.google.com/store/apps/details?id=com.google.android.apps.accessibility.auditor (Accessed: 29 April 2021).

Graham, D. J., Orquin, J. L. & Visschers, V. H. M. (2012). 'Eye tracking and nutrition label use: A review of the literature and recommendations for label enhancement', *Food Policy*, 37(4), pp. 378-382. Available at: https://doi.org/10.1016/j.foodpol.2012.03.004 (Accessed: 24 April 2021).

Guest (2016). 'Accessibility and me: Dealing with dyslexia', *Accessibility in government*, 15 November. Available at: https://accessibility.blog.gov.uk/2016/11/15/accessibility-and-me-dealing-with-dyslexia/ (Accessed: 24 April 2021).

Harper, S. (2020). *UX from 30,000ft. Leanpub*. [Online]. Available at: https://leanpub.com/UX/read (Accessed: 26 March 2021).

Iniesto, F., McAndrew, P., Minocha, S. & Coughlan, T. (2016). 'The current state of accessibility of MOOCs: What are the next steps?', *Proceedings of Open Education Global 2016: Convergence Through Collaboration*, Krakow, Poland. 12-14 Apr 2016. Available at: https://conference.oeglobal.org/2016/presentation/the-current-state-of-accessibility-of-moocs-what-are-the-next-steps/ (Accessed: 20 March 2021).

International Telecommunication Union (2020). *Measuring digital development: Facts and figures 2020*. Available at: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/FactsFigures2020.pdf (Accessed: 26 March 2021).

Jahoda, P. (2019). *MPAndroidChart* (Version 3.1.0) [Android library]. Available at: https://github.com/PhilJay/MPAndroidChart/releases/tag/v3.1.0 (Accessed: 26 April 2021).

Kieffer, S., Ghouti, A., Macq, B. (2017). 'The Agile UX Development Lifecycle: Combining Formative Usability and Agile Methods', *Proceedings of the 50th Hawaii International Conference on System Sciences*, Hilton Waikoloa Village, Hawaii. 4th-7th January 2017. Available at: https://aisel.aisnet.org/hicss-50/cl/hci/7/ (Accessed: 29 March 2021).

Lardinois, F. (2019). 'Kotlin is now Google's preferred language for Android app development', TechCrunch, 7 May [Online]. Available at: https://techcrunch.com/2019/05/07/kotlin-is-now-googles-preferred-language-for-android-app-development/ (Accessed: 06 March 2021).

mano1990 (2018). 'This sub should be designed with Opendyslexic.', *r/Dyslexia* [Reddit] 1 December. Available at: https://www.reddit.com/r/Dyslexia/comments/a23yoz/this_sub_should_be_designed_with_opendyslexic/ (Accessed: 27 April 2021).

Matthews, T., Judge, T. & Whittaker, S. (2012). 'How do designers and user experience professionals actually perceive and use personas?', *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Austin, Texas, USA: Association for Computing Machinery. pp. 1219–1228. Available at: https://doi.org/10.1145/2207676.2208573 (Accessed: 24 April 2021).

Ministerio de Salud (2018). *Manual de Etiquetado Nutricional de Alimentos.* Santiago, Chile. Available at: https://www.minsal.cl/wp-content/uploads/2018/01/Manual-Etiquetado-Nutricional-Ed.-Minsal-2017v2.pdf (Accessed: 10 April 2021).

MyFitnessPal, Inc. (2021). *MyFitnessPal* (Version 21.5.1). [Android program]. Available at: https://play.google.com/store/apps/details?id=com.myfitnesspal.android (Accessed: 25 March 2021).

Nielsen, J. (1996). Accessible Design for Users With Disabilities. *Nielsen Norman Group*, 30 September [Online]. Available at: https://www.nngroup.com/articles/accessible-design-for-users-with-disabilities/ (Accessed: 24 April 2021).

Norman, D. (2013). *The Design of Everyday Things: Revised and Expanded Edition*. New York: Basic Books.

Perea, M., Panadero, V., Moret-Tatay, C., Gómez, P. (2012). 'The effects of inter-letter spacing in visual-word recognition: Evidence with young normal readers and developmental dyslexics', *Learning and Instruction*, 22(6), pp. 420-430. Available at: https://doi.org/10.1016/j.learninstruc.2012.04.001 (Accessed: 11 April 2021).

Peres, A. L., Da Silva, T., Silva, F. S., Soares, F. F., Rosemberg, C. & Romero, S. (2014). 'Agileux model: Towards a reference model on integrating ux in developing software using agile methodologies', *2014 Agile Conference*, Kissimmee, FL, USA. 28 July-1 August

2014. IEEE. pp. 61-63. Available at: https://doi.org/10.1109/AGILE.2014.15 (Accessed: 29 March 2021).

Petrie, H. & Bevan, N. (2009). 'The evaluation of accessibility, usability, and user experience', *The universal access handbook*, 1, pp. 1-16.

Rello, L. & Baeza-Yates, R. (2013). 'Good fonts for dyslexia', *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*, Bellevue, Washington. October 2013. Association for Computing Machinery. Article 14, pp. 1-8. Available at: https://doi-org.manchester.idm.oclc.org/10.1145/2513383.2513447 (Accessed: 27 April 2021).

Rello, L. (2015). 'Dyslexia and web accessibility: Synergies and challenges', *Proceedings of the 12th International Web for All Conference*, Florence, Italy. 18-20 May 2015. Association for Computing Machinery, Article 9, pp. 1-4. Available at: https://doi.org/10.1145/2745555.2746655 (Accessed: 27 April 2021).

Resorization (2019). 'What do you think about the OpenDyslexic font? Did it help you? I consider changing my Linux-PC 's fonts to it.', *r/Dyslexia* [Reddit] June 19. Available at: https://www.reddit.com/r/Dyslexia/comments/c2dyf1/what_do_you_think_about_the_opendyslexic_font_did/ (Accessed: 27 April 2021).

Rettig, M. (1992). 'Hat racks for understanding', *Commun. ACM*, 35(10), pp. 21–24. Available at: https://doi.org/10.1145/135239.135247 (Accessed: 20 April 2021).

Rothstein, E. (2009). 'Typography Fans Say Ikea Should Stick to Furniture', *The New York Times*, 4 September [Online]. Available at: https://www.nytimes.com/2009/09/05/arts/design/05ikea.html (Accessed: 27 April 2021).

Southey, F. (2020). 'Nutrition labels overlooking the blind: "I am massively disadvantaged when it comes to food choices"', *Food Navigator*, 20 February [Online]. Available at: https://www.foodnavigator.com/Article/2020/02/19/Are-food-labels-excluding-the-visually-impaired (Accessed: 11 April 2021).

Spina, C. (2019). 'WCAG 2.1 and the Current State of Web Accessibility in Libraries', *Weave Journal Of Library User Experience*, 2(2) [Online]. Available at: https://doi.org/10.3998/weave.12535642.0002.202 (Accessed: 20 March 2021).

Stark Lab (2021). *Stark* [Computer program]. Available at: https://www.getstark.co/ (Accessed: 27 April 2021).

Tufte, E. R. (1985). 'Aesthetics and Technique in Data Graphical Design' in *The Visual Display of Quantitative Information*. 2nd edn. Cheshire, CT: Graphics Press.

United States Department of Agriculture (2019). *Example of Sticker Label* [Illustration]. Available at: https://apps.fas.usda.gov/newgainapi/api/report/downloadreportbyfilename?filename=Food%20and%20Agricultural%20Import%20Regulations%20and%20Standards%202018%20_Dubai_United%20Arab%20Emirates_5-1-2019.pdf (Accessed: 26 April 2021).

Van den Rul, C. (2019). 'On the Dyslexic Mind', *The Ascent*, September 3. Available at: https://medium.com/the-ascent/on-the-dyslexic-mind-f6ddb43915d (Accessed: 24 April 2021).

Veroniiiica (2018). 'My Eight Favorite Free Fonts for Print Disabilities', *Perkins eLearning*, 14 August. Available at: https://www.perkinselearning.org/technology/blog/my-eight-favorite-free-fonts-print-disabilities (Accessed: 27 April 2021).

WebAIM (2021). *WebAIM: Contrast Checker*. Available at: https://webaim.org/resources/contrastchecker/ (Accessed: 27 April 2021).

World Health Organization (2010). *Global Data on Visual Impairments 2010*. Geneva: World Health Organization. Available at: https://www.who.int/blindness/publications/globaldata/en/ (Accessed: 11 April 2021).

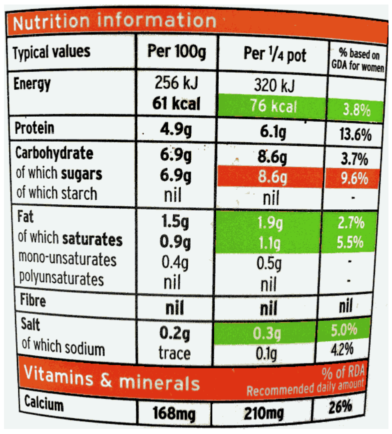
xueli (2015). 'I downloaded the opendyslexic font and found that it was much easier for me to read', *r/Dyslexia* [Reddit] 21 July. Available at: https://www.reddit.com/r/Dyslexia/comments/3e2fip/i_downloaded_the_opendyslexic_font_and_found_that/ (Accessed: 27 April 2021).

Yan, S. & Ramachandran, P. G. (2019). 'The current status of accessibility in mobile apps', *ACM Trans. Access. Comput.*, 12(1), p. Article 3 [Online]. Available at: https://dl.acm.org/doi/abs/10.1145/3300176 (Accessed: 20 March 2021).

Yuan, B., Folmer, E. & Harris, F. C. (2011). 'Game accessibility: A survey', *Universal Access in the Information Society*, 10(1), pp. 81-100 [Online]. Available at: https://doi.org/10.1007/s10209-010-0189-5 (Accessed: 20 March 2021).

Appendices

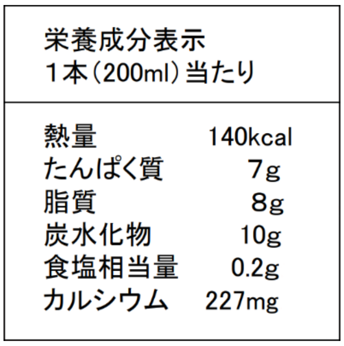
Appendix 1 – Examples of Nutrition Labels Around the World



Nutrition information

Typical values	Per 100g	Per 1/4 pot	% based on GDA for women
Energy	256 kJ 61 kcal	320 kJ 76 kcal	 3.8%
Protein	4.9g	6.1g	13.6%
Carbohydrate	6.9g	8.6g	3.7%
of which sugars	6.9g	8.6g	9.6%
of which starch	nil	nil	-
Fat	1.5g	1.9g	2.7%
of which saturates	0.9g	1.1g	5.5%
mono-unsaturates	0.4g	0.5g	-
polyunsaturates	nil	nil	-
Fibre	nil	nil	nil
Salt	0.2g	0.3g	5.0%
of which sodium	trace	0.1g	4.2%
Vitamins & minerals			% of RDA Recommended daily amount
Calcium	168mg	210mg	26%

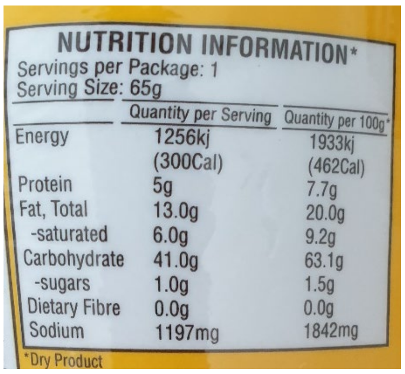
UK nutrition label for a cup of yogurt



栄養成分表示
1本（200ml）当たり

熱量	140kcal
たんぱく質	7g
脂質	8g
炭水化物	10g
食塩相当量	0.2g
カルシウム	227mg

Japanese standard for nutrition labels (Source: Consumer Affairs Agency, 2020)

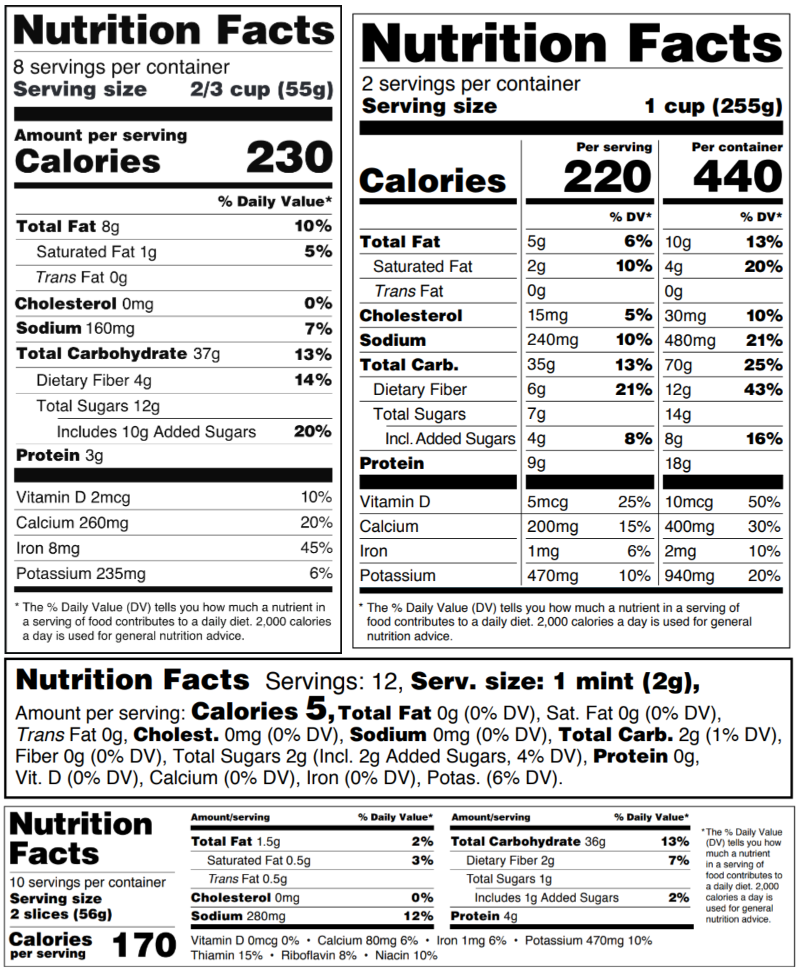


NUTRITION INFORMATION*
Servings per Package: 1
Serving Size: 65g

	Quantity per Serving	Quantity per 100g*
Energy	1256kj (300Cal)	1933kj (462Cal)
Protein	5g	7.7g
Fat, Total	13.0g	20.0g
-saturated	6.0g	9.2g
Carbohydrate	41.0g	63.1g
-sugars	1.0g	1.5g
Dietary Fibre	0.0g	0.0g
Sodium	1197mg	1842mg

*Dry Product

Australian nutrition label



Nutrition Facts
8 servings per container
Serving size 2/3 cup (55g)

Amount per serving
Calories 230

	% Daily Value*
Total Fat 8g	**10%**
Saturated Fat 1g	**5%**
Trans Fat 0g	
Cholesterol 0mg	**0%**
Sodium 160mg	**7%**
Total Carbohydrate 37g	**13%**
Dietary Fiber 4g	**14%**
Total Sugars 12g	
Includes 10g Added Sugars	**20%**
Protein 3g	
Vitamin D 2mcg	10%
Calcium 260mg	20%
Iron 8mg	45%
Potassium 235mg	6%

* The % Daily Value (DV) tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

Nutrition Facts
2 servings per container
Serving size 1 cup (255g)

	Per serving		Per container	
Calories	**220**		**440**	
		% DV*		% DV*
Total Fat	5g	**6%**	10g	**13%**
Saturated Fat	2g	**10%**	4g	**20%**
Trans Fat	0g		0g	
Cholesterol	15mg	**5%**	30mg	**10%**
Sodium	240mg	**10%**	480mg	**21%**
Total Carb.	35g	**13%**	70g	**25%**
Dietary Fiber	6g	**21%**	12g	**43%**
Total Sugars	7g		14g	
Incl. Added Sugars	4g	**8%**	8g	**16%**
Protein	9g		18g	
Vitamin D	5mcg	25%	10mcg	50%
Calcium	200mg	15%	400mg	30%
Iron	1mg	6%	2mg	10%
Potassium	470mg	10%	940mg	20%

* The % Daily Value (DV) tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

Nutrition Facts Servings: 12, **Serv. size: 1 mint (2g),**
Amount per serving: **Calories 5, Total Fat** 0g (0% DV), Sat. Fat 0g (0% DV), *Trans* Fat 0g, **Cholest.** 0mg (0% DV), **Sodium** 0mg (0% DV), **Total Carb.** 2g (1% DV), Fiber 0g (0% DV), Total Sugars 2g (Incl. 2g Added Sugars, 4% DV), **Protein** 0g, Vit. D (0% DV), Calcium (0% DV), Iron (0% DV), Potas. (6% DV).

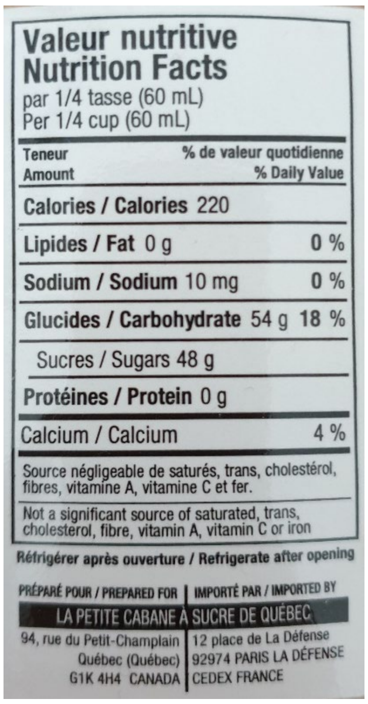
Nutrition Facts
10 servings per container
Serving size
2 slices (56g)
Calories per serving 170

Amount/serving	% Daily Value*	Amount/serving	% Daily Value*
Total Fat 1.5g	**2%**	**Total Carbohydrate** 36g	**13%**
Saturated Fat 0.5g	**3%**	Dietary Fiber 2g	**7%**
Trans Fat 0.5g		Total Sugars 1g	
Cholesterol 0mg	**0%**	Includes 1g Added Sugars	**2%**
Sodium 280mg	**12%**	**Protein** 4g	

Vitamin D 0mcg 0% • Calcium 80mg 6% • Iron 1mg 6% • Potassium 470mg 10%
Thiamin 15% • Riboflavin 8% • Niacin 10%

*The % Daily Value (DV) tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

Examples of FDA standard nutrition labels (Source: Food and Drug Administration, 2016)



Valeur nutritive
Nutrition Facts
par 1/4 tasse (60 mL)
Per 1/4 cup (60 mL)

Teneur / Amount	% de valeur quotidienne / % Daily Value
Calories / Calories 220	
Lipides / Fat 0 g	0 %
Sodium / Sodium 10 mg	0 %
Glucides / Carbohydrate 54 g	18 %
Sucres / Sugars 48 g	
Protéines / Protein 0 g	
Calcium / Calcium	4 %

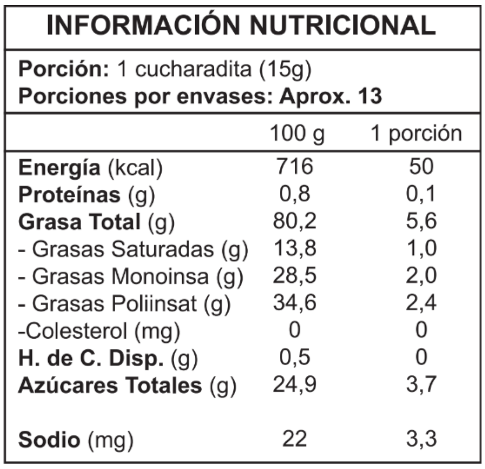
Source négligeable de saturés, trans, cholestérol, fibres, vitamine A, vitamine C et fer.

Not a significant source of saturated, trans, cholesterol, fibre, vitamin A, vitamin C or iron

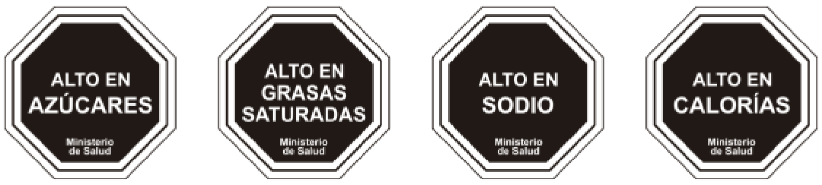
Réfrigérer après ouverture / Refrigerate after opening

PRÉPARÉ POUR / PREPARED FOR	IMPORTÉ PAR / IMPORTED BY
LA PETITE CABANE À SUCRE DE QUÉBEC	
94, rue du Petit-Champlain Québec (Québec) G1K 4H4 CANADA	12 place de La Défense 92974 PARIS LA DÉFENSE CEDEX FRANCE

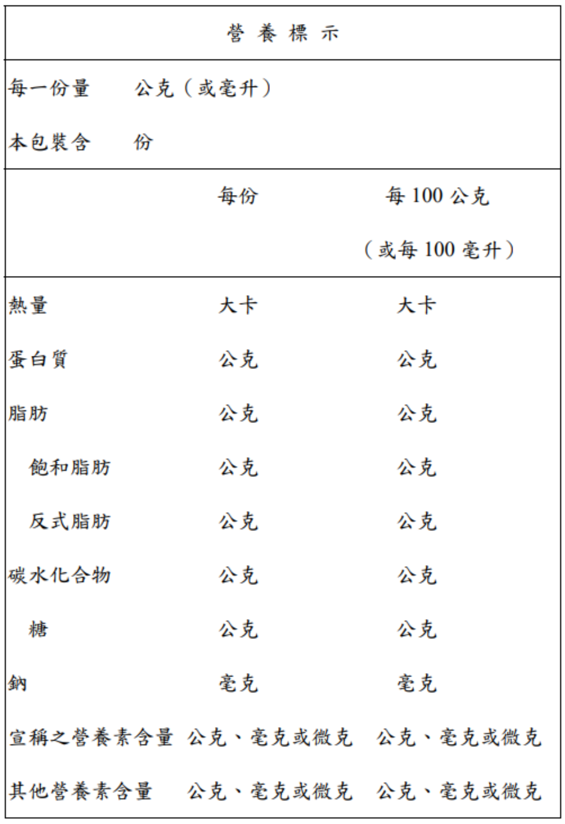
Canadian bilingual nutrition label from the Québec region



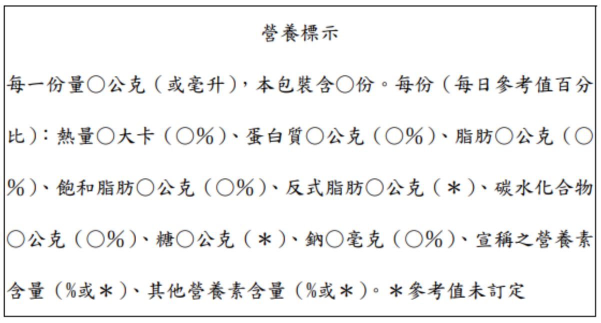
INFORMACIÓN NUTRICIONAL		
Porción: 1 cucharadita (15g)		
Porciones por envases: Aprox. 13		
	100 g	1 porción
Energía (kcal)	716	50
Proteínas (g)	0,8	0,1
Grasa Total (g)	80,2	5,6
- Grasas Saturadas (g)	13,8	1,0
- Grasas Monoinsa (g)	28,5	2,0
- Grasas Poliinsat (g)	34,6	2,4
-Colesterol (mg)	0	0
H. de C. Disp. (g)	0,5	0
Azúcares Totales (g)	24,9	3,7
Sodio (mg)	22	3,3



Nutrition label in Chile and packaging using warning signs for high contents of given nutrients (Source: Ministerio de Salud, 2018)



營 養 標 示		
每一份量	公克（或毫升）	
本包裝含	份	
	每份	每 100 公克 （或每 100 毫升）
熱量	大卡	大卡
蛋白質	公克	公克
脂肪	公克	公克
飽和脂肪	公克	公克
反式脂肪	公克	公克
碳水化合物	公克	公克
糖	公克	公克
鈉	毫克	毫克
宣稱之營養素含量	公克、毫克或微克	公克、毫克或微克
其他營養素含量	公克、毫克或微克	公克、毫克或微克



營養標示

每一份量○公克（或毫升），本包裝含○份。每份（每日參考值百分比）：熱量○大卡（○%）、蛋白質○公克（○%）、脂肪○公克（○%）、飽和脂肪○公克（○%）、反式脂肪○公克（＊）、碳水化合物○公克（○%）、糖○公克（＊）、鈉○毫克（○%）、宣稱之營養素含量（%或＊）、其他營養素含量（%或＊）。＊參考值未訂定

Examples of Taiwanese nutrition label standards (Source: Taiwan Food and Drug Administration, 2014)

Appendix 1 References

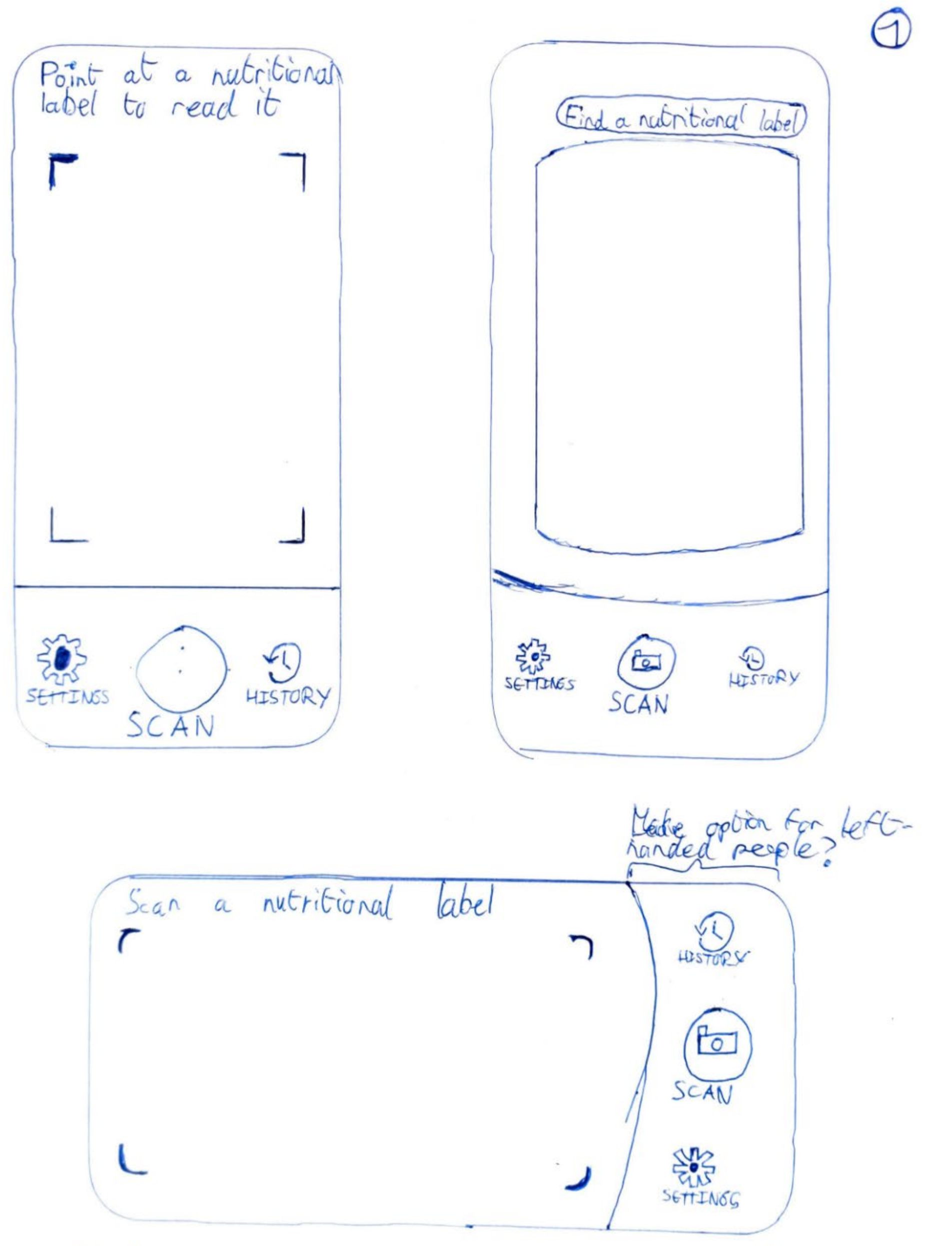
Consumer Affairs Agency (2020). *栄養成分表示*. Tokyo, Japan. Available at: https://www.caa.go.jp/policies/policy/food_labeling/health_promotion/assets/food_labeling_cms206_20210318_01.pdf (Accessed: 26 April 2020).

Food and Drug Administration (2016). *The New Nutrition Facts Label – Examples of Different Label Formats*. Available at: https://www.regulations.gov/document/FDA-2012-N-1210-0875 (Accessed: 27 April 2021).

Ministerio de Salud (2018). *Manual de Etiquetado Nutricional de Alimentos*. Santiago, Chile. Available at: https://www.minsal.cl/wp-content/uploads/2018/01/Manual-Etiquetado-Nutricional-Ed.-Minsal-2017v2.pdf (Accessed: 10 April 2021).

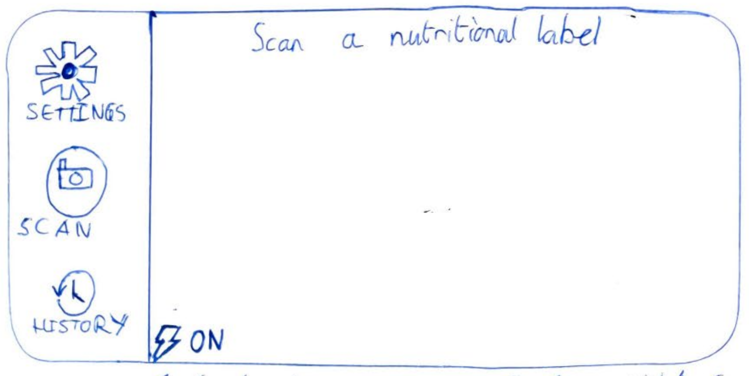
Taiwan Food and Drug Administration (2014). *包裝食品營養標示應遵行事項總說明*. Republic of China. Available at: file:///C:/Users/User/Downloads/%E5%8C%85%E8%A3%9D%E9%A3%9F%E5%93%81%E7%87%9F%E9%A4%8A%E6%A8%99%E7%A4%BA%E6%87%89%E9%81%B5%E8%A1%8C%E4%BA%8B%E9%A0%85%E7%B8%BD%E8%AA%AA%E6%98%8E%E5%8F%8A%E9%80%90%E9%BB%9E%E8%AA%AA%E6%98%8E.pdf (Accessed: 26 April 2021).

Appendix 2 – Low Fidelity Mockups

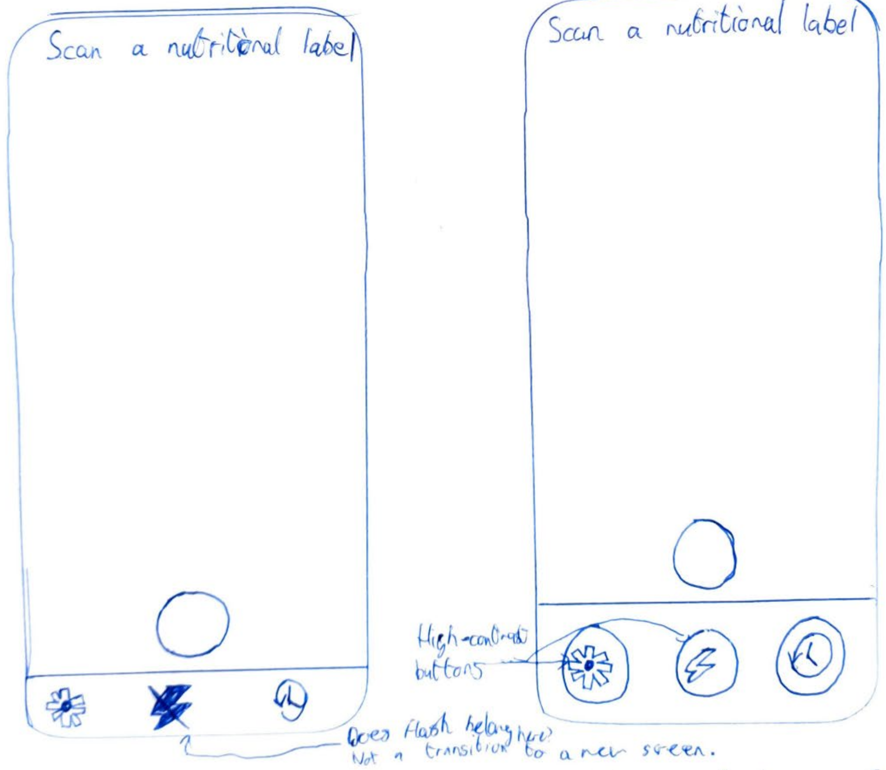


- The pointing frame is here just for helping the user put all the nutritional label in the frame. Information outside of the overlay, but in the image will be analysed even if not in the overlay frame
- HISTORY will be more used than SETTINGS. Put it closer to the hand (bottom in horizontal mode)

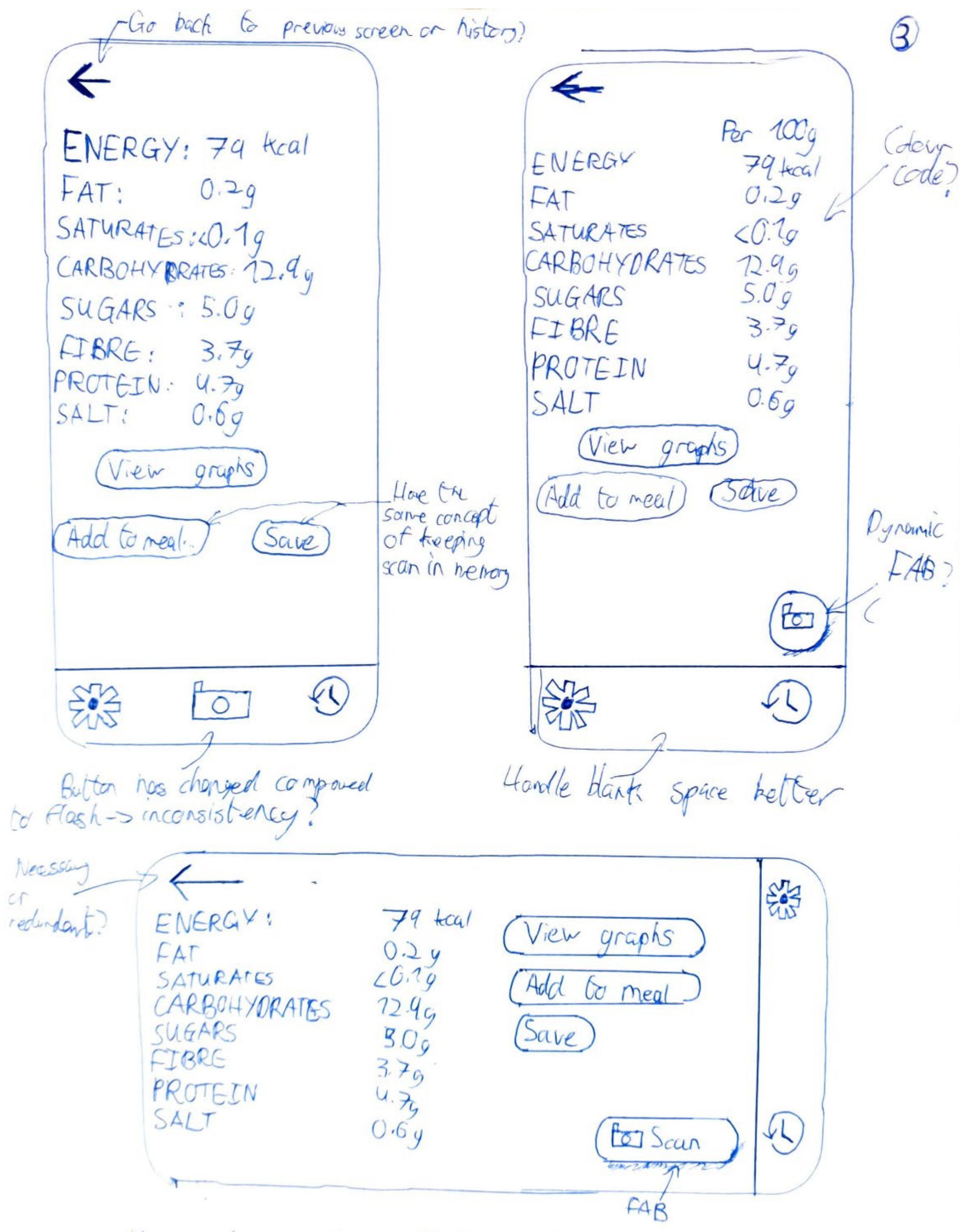
②



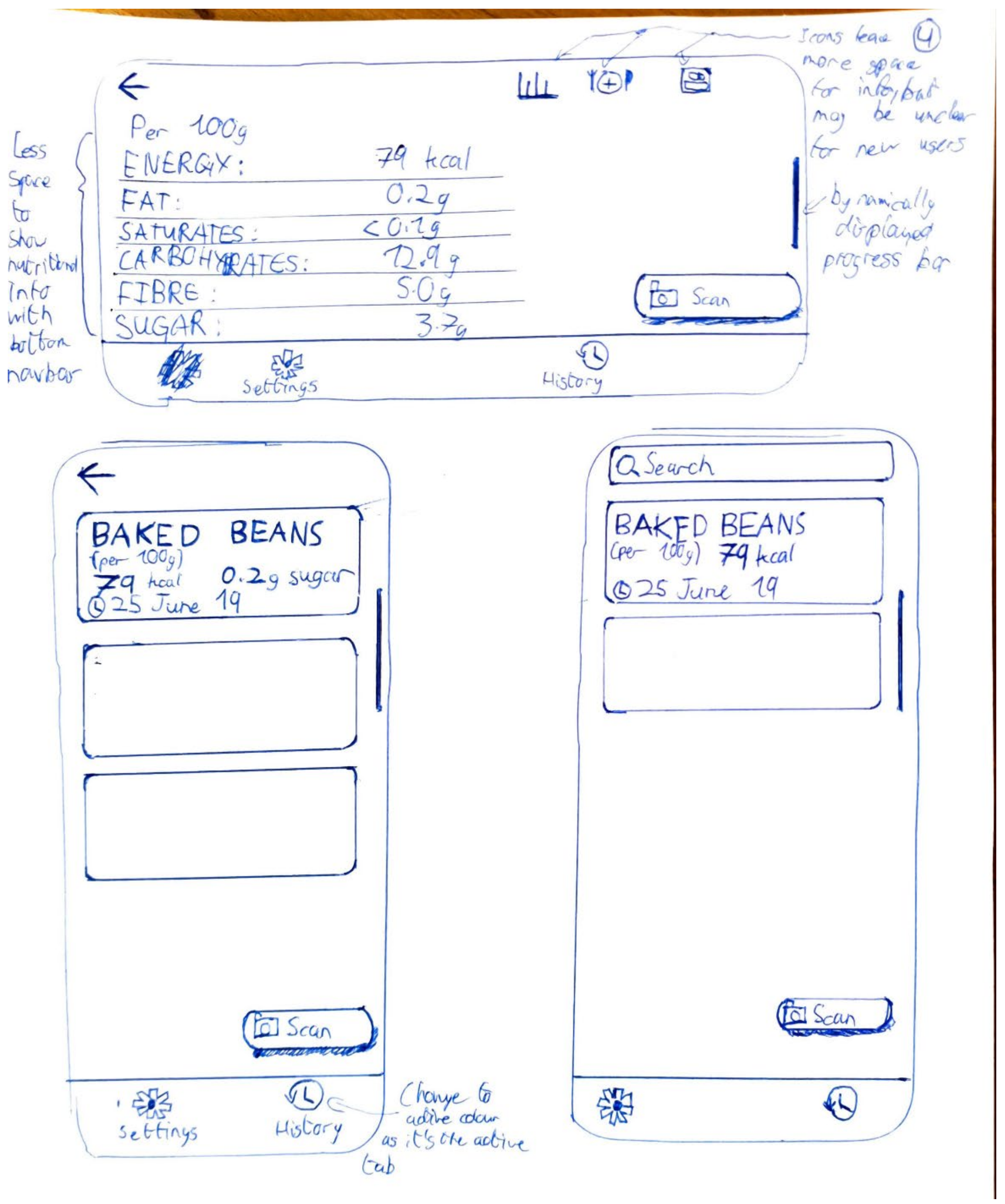
Should it stay here or move to the right for "breathing space"?



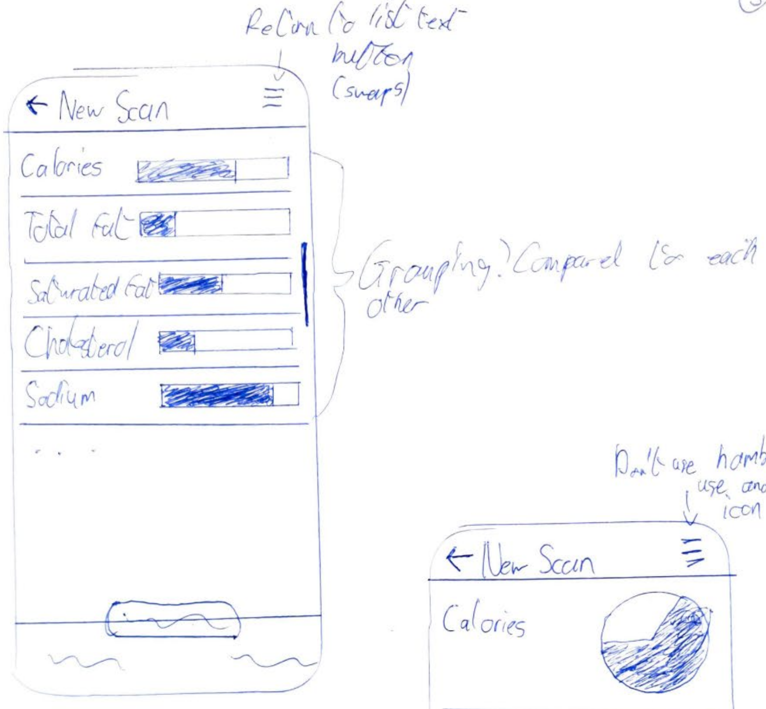
- Is the overlaying frame necessary? Google Translate doesn't have a frame when scanning text. Bixby vision does it depending on context + dynamic
- Tapping the screen should focus the camera, similar to other apps.
- Getting rid of labels allows for more of the screen to be used for pointing at the label

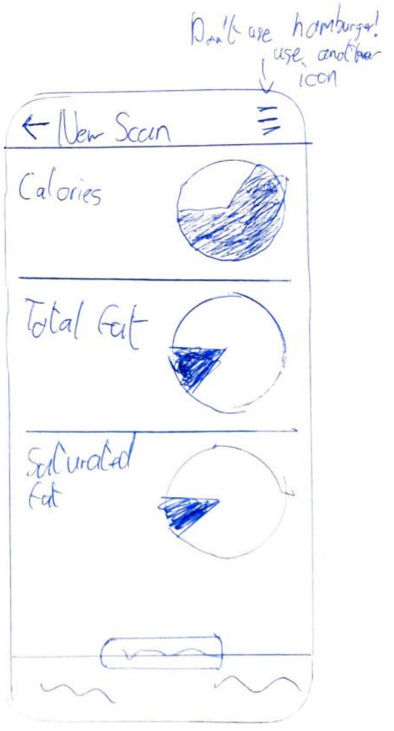


- I could use elements from Material design like FABs.
- Go back button seems redundant, but without it users might think they can't go back to the previous screen.


Icons leave more space for infobar may be unclear for new users
4
Per 100g
ENERGY: 79 kcal
FAT: 0.2g
SATURATES: <0.1g
CARBOHYDRATES: 12.9g
FIBRE: 5.0g
SUGAR: 3.7g
Scan
Settings
History
Less space to show nutritional info with bottom navbar
dynamically displayed progress bar
BAKED BEANS
(per 100g)
79 kcal 0.2g sugar
25 June 19
Scan
Settings
History
Change to active colour as it's the active tab
Search
BAKED BEANS
(per 100g) 79 kcal
25 June 19
Scan

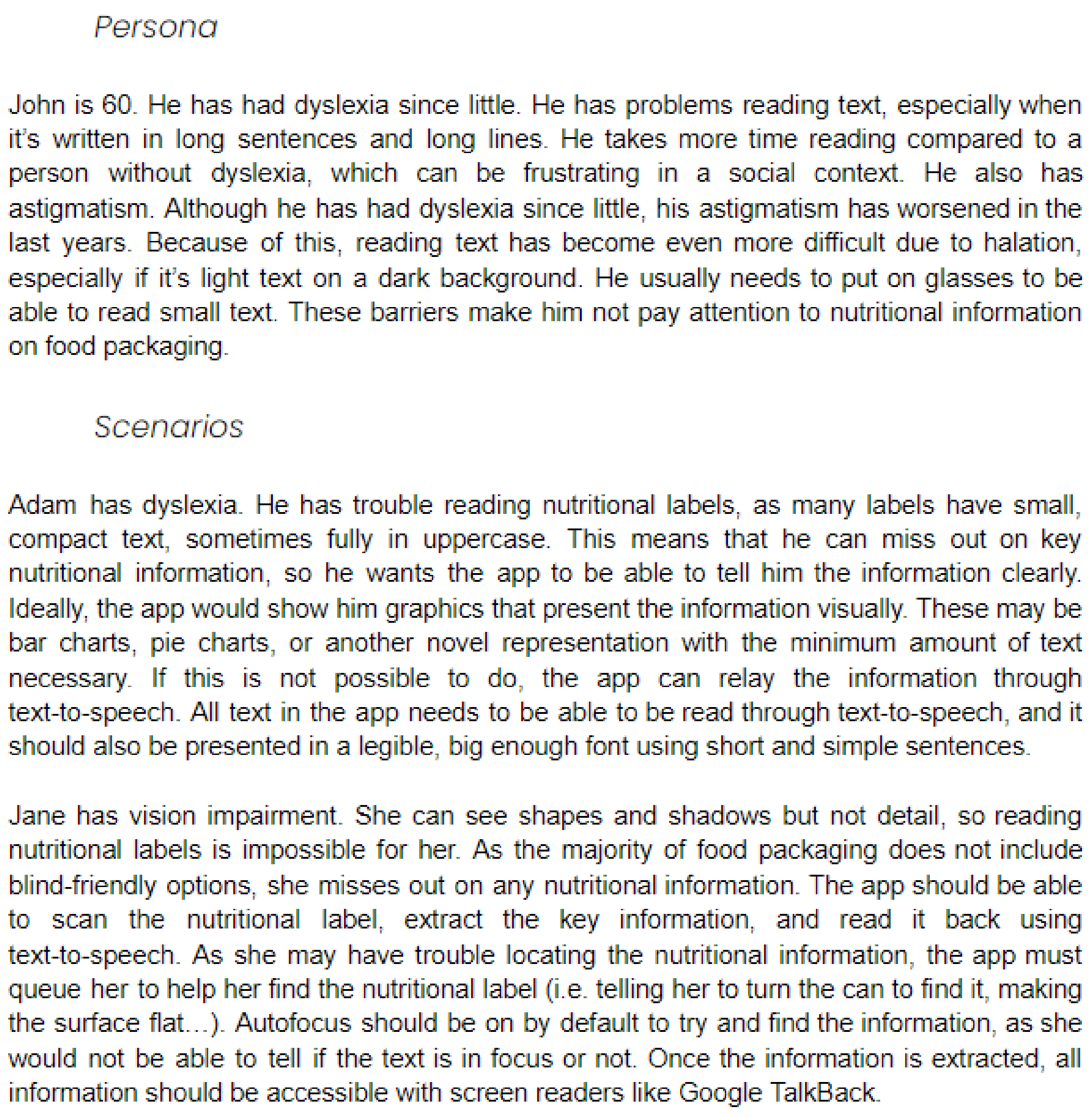
S


Return to list text button (swaps)
← New Scan
Calories
Total Fat
Saturated Fat
Cholesterol
Sodium
. . .
(Grouping) Compared to each other



- Pie charts are individual and don't get compared to each other
- Use colour?

Appendix 3 – Other Design Artefacts



Persona

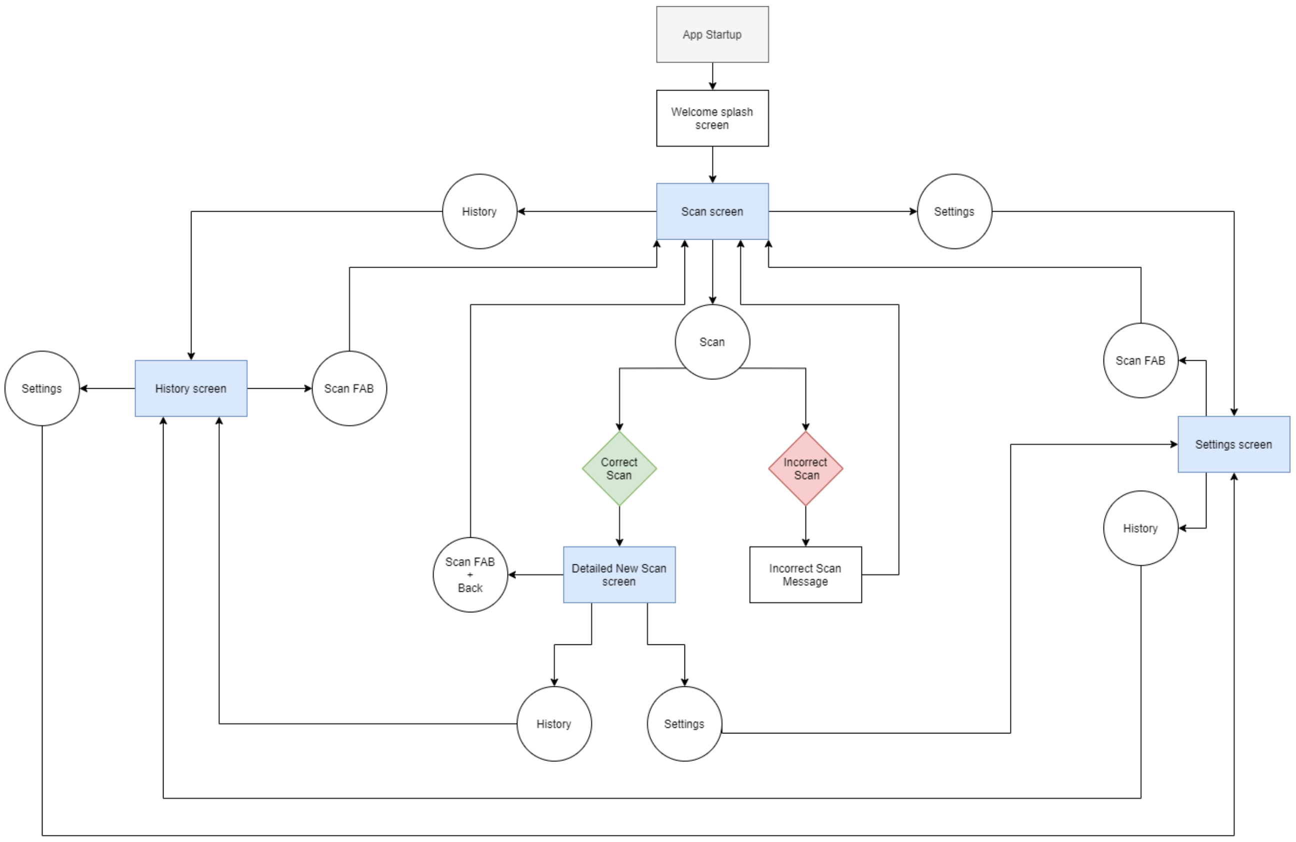
John is 60. He has had dyslexia since little. He has problems reading text, especially when it's written in long sentences and long lines. He takes more time reading compared to a person without dyslexia, which can be frustrating in a social context. He also has astigmatism. Although he has had dyslexia since little, his astigmatism has worsened in the last years. Because of this, reading text has become even more difficult due to halation, especially if it's light text on a dark background. He usually needs to put on glasses to be able to read small text. These barriers make him not pay attention to nutritional information on food packaging.

Scenarios

Adam has dyslexia. He has trouble reading nutritional labels, as many labels have small, compact text, sometimes fully in uppercase. This means that he can miss out on key nutritional information, so he wants the app to be able to tell him the information clearly. Ideally, the app would show him graphics that present the information visually. These may be bar charts, pie charts, or another novel representation with the minimum amount of text necessary. If this is not possible to do, the app can relay the information through text-to-speech. All text in the app needs to be able to be read through text-to-speech, and it should also be presented in a legible, big enough font using short and simple sentences.

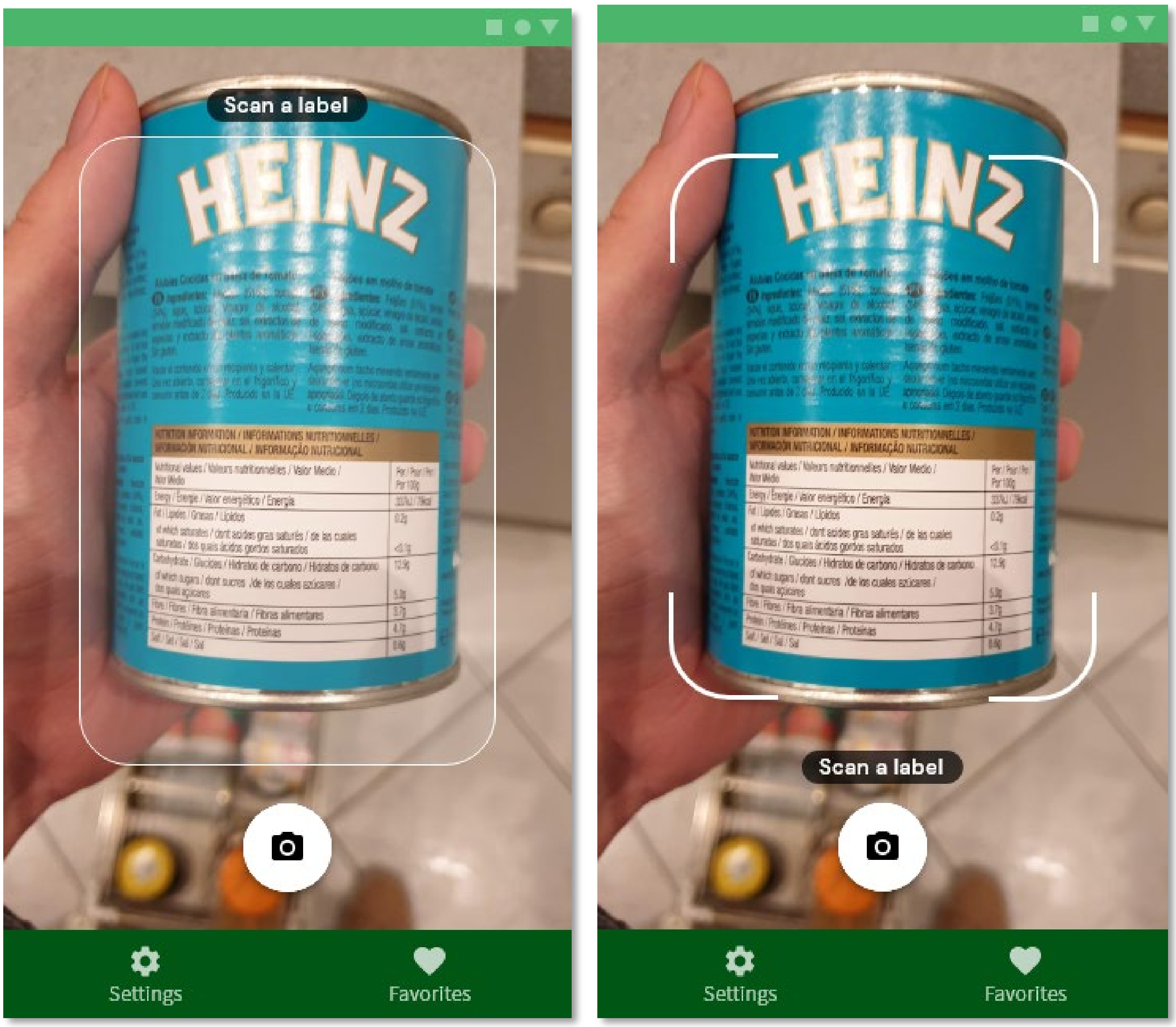
Jane has vision impairment. She can see shapes and shadows but not detail, so reading nutritional labels is impossible for her. As the majority of food packaging does not include blind-friendly options, she misses out on any nutritional information. The app should be able to scan the nutritional label, extract the key information, and read it back using text-to-speech. As she may have trouble locating the nutritional information, the app must queue her to help her find the nutritional label (i.e. telling her to turn the can to find it, making the surface flat…). Autofocus should be on by default to try and find the information, as she would not be able to tell if the text is in focus or not. Once the information is extracted, all information should be accessible with screen readers like Google TalkBack.

Personas and scenarios created to better understand the target users

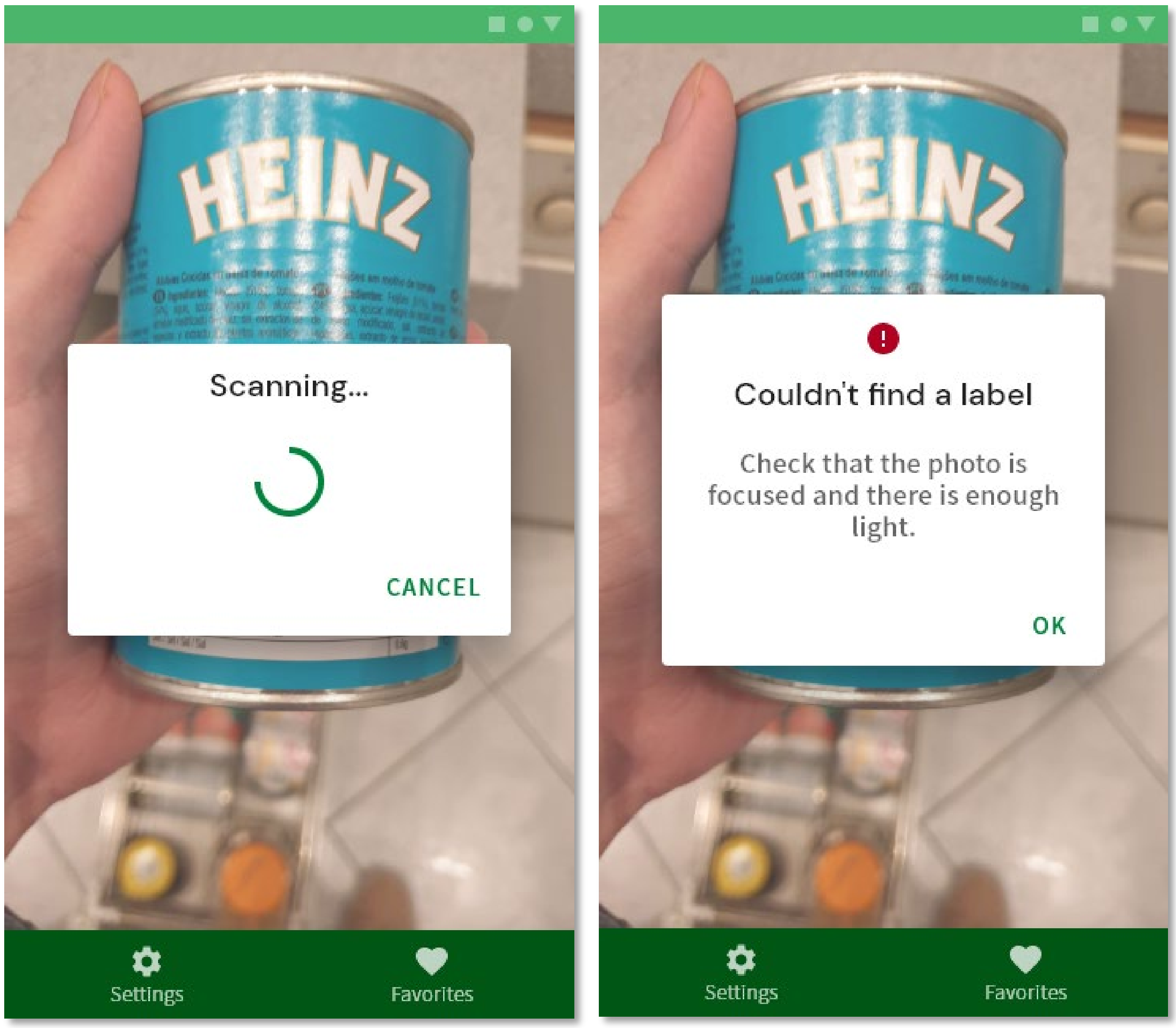


Interaction flowchart

Prototype splash screen, scrapped to make the application feel faster at start up



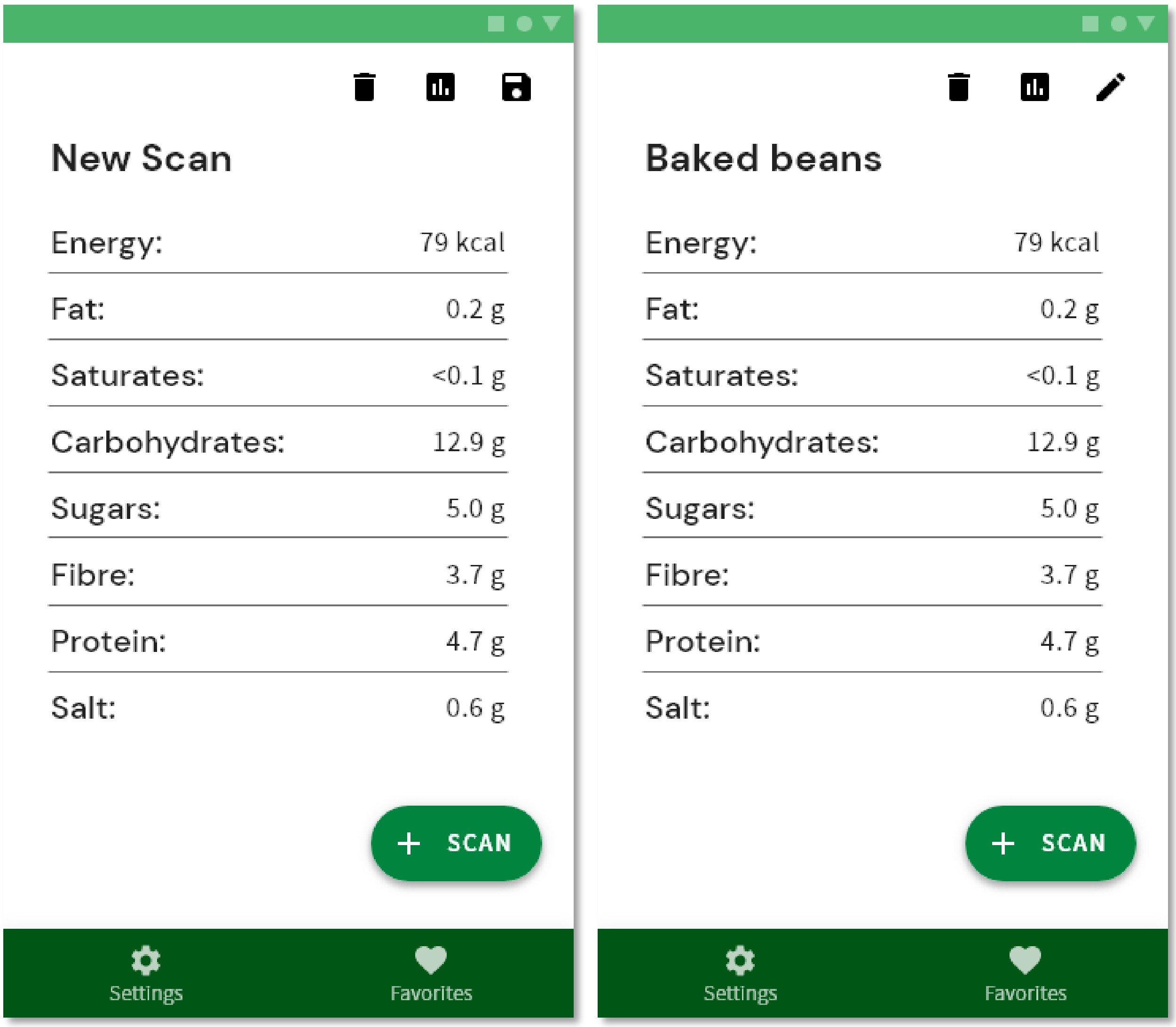
Two prototype concepts for the scanning screen, using different camera frames to drive the attention of the user and get them to focus closely on the label


Scanning...

Couldn't find a label

Check that the photo is focused and there is enough light.

Prototype feedback dialogs. The final version of these dialogs have significant changes to better conform to Material guidelines



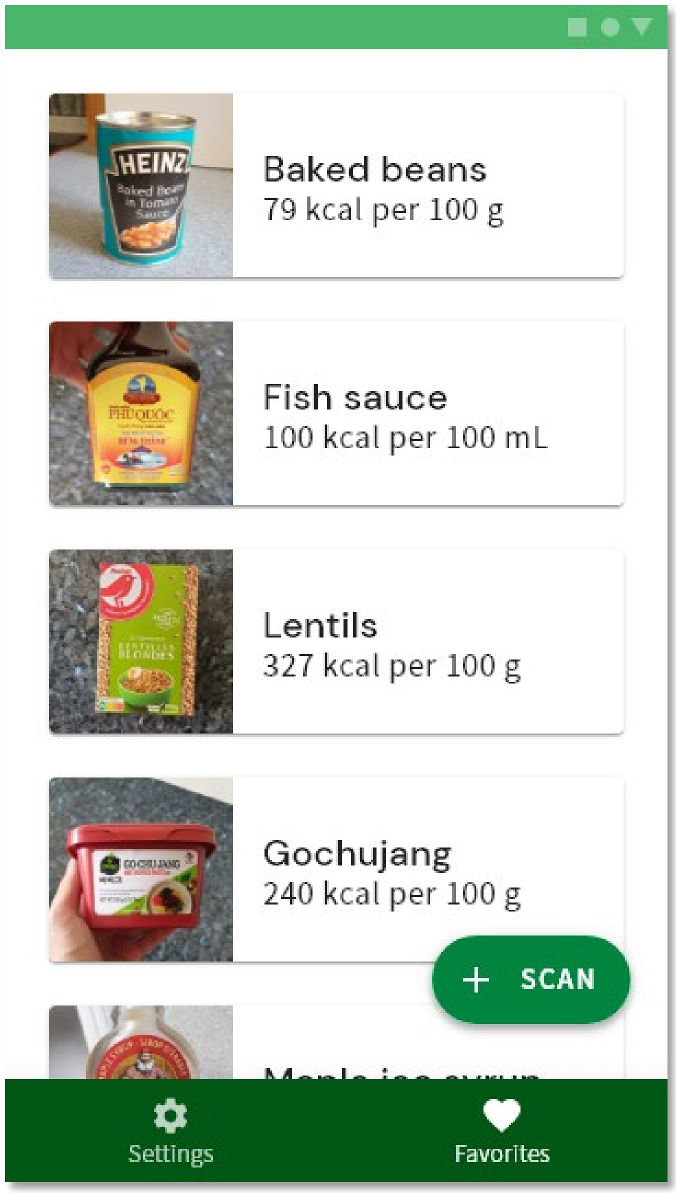
New Scan

Energy:	79 kcal
Fat:	0.2 g
Saturates:	<0.1 g
Carbohydrates:	12.9 g
Sugars:	5.0 g
Fibre:	3.7 g
Protein:	4.7 g
Salt:	0.6 g

Baked beans

Energy:	79 kcal
Fat:	0.2 g
Saturates:	<0.1 g
Carbohydrates:	12.9 g
Sugars:	5.0 g
Fibre:	3.7 g
Protein:	4.7 g
Salt:	0.6 g

Prototype screens for a new scan on the left, and a saved scan on the right



Prototype screen for the history of previously scanned labels. Users would have been able to set custom photos of the product to faster recognise it when browsing the list, with the default photo being that of the scanned nutrition label.