Mobile Application That Uses Retrieve Augment Generate with Computer Vision to Detect Potentially Harmful Cosmetic Ingredients for People with Skin Conditions

Laura Hofmanova

N00222003

Supervisor: Mohammed Cherbatji

Second Reader: Catherine Noonan

Year 4 2025/26

Dl836 BSc (Hons) In Creative Computing

Contents

Declaration of Authorship

The incorporation of material without formal and proper acknowledgement (even with no deliberate intent to cheat) can constitute plagiarism.

If you have received significant help with a solution from one or more colleagues, you should document this in your submitted work, and if you have any doubt as to what level of discussion/collaboration is acceptable, you should consult your lecturer or the Course Director.

WARNING: Take care when discarding program listings lest they be copied by someone else, which may well bring you under suspicion. Do not leave copies of your own files on a hard disk where they can be accessed by others. Be aware that removable media, used to transfer work, may also be removed and/or copied by others if left unattended.

Plagiarism is an act of fraud and an offence against the Institute's discipline.

Alleged plagiarism will be investigated and dealt with appropriately by the Institute. Please refer to the Institute Handbook for further details of penalties.

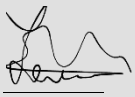
The following is an extract from the B.Sc. in Creative Computing (Hons) course handbook. Please read carefully and sign the declaration below

Collusion may be defined as more than one person working on an individual assessment. This would include jointly developed solutions as well as one individual giving a solution to another, who then makes some changes and hands it up as their own work.

DECLARATION:

I am aware of the Institute's policy on plagiarism and certify that this thesis is my own work.

Student: Laura Hofmanova N00222003

Signed 

Date: 30/04/26

Failure to complete and submit this form may lead to an investigation into your work.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Mohammed Cherbatji, for his continuous guidance, support, and valuable feedback throughout the development of this project. His insights and encouragement were instrumental in shaping both the technical and academic aspects of this work.

I would also like to thank my lecturers and the faculty for providing the knowledge and resources necessary to complete this project. Their teaching has played a key role in developing the skills required to undertake this project.

Special thanks are extended to those who participated in testing and provided feedback during the development process. Their input was essential in refining the application and improving its overall usability and performance.

Finally, I would like to acknowledge my friends and family for their ongoing support and encouragement throughout this project.

Abstract

This project presents the design and development of a mobile application that uses artificial intelligence to analyse cosmetic product ingredients based on personalised user profiles. The aim is to support users in making informed decisions by identifying potential risks associated with allergens, skin conditions, and personal preferences.

The system combines computer vision for ingredient extraction with a Retrieval-Augmented Generation (RAG) approach to produce clear and context-aware evaluations. A profile-based decision system allows results to be tailored to individual users, enabling more accurate and meaningful analysis compared to generic ingredient checkers.

The application was developed using a client–server architecture, with a React Native mobile frontend and a Node.js backend supported by PostgreSQL and Prisma. An admin panel was also implemented to manage core data entities through CRUD operations, reducing reliance on external APIs and improving data reliability.

Testing was conducted across functionality, performance, and usability, including API validation, role-based access control, and consistency of analysis results. Challenges such as API rate limits, network configuration issues, and performance constraints were addressed through optimisation and system design decisions.

The final system demonstrates how AI can be applied to improve consumer awareness and decision-making in cosmetic product usage, while highlighting the importance of personalised, transparent, and explainable outputs.

Table Of Abbreviations

Abbreviation	**Meaning**
RAG	Retrieval-Augmented Generation
AI	Artificial Intelligence
LLM	Large Language Model
ERD	Entity Relationship Diagram
JWT	JSON Web Token
CRUD	Create, Read, Update, Delete
UI	User Interface
UX	User Experience
ORM	Object-Relational-Mapper
CSS	Cascading Style Sheets
HTTP	HyperText Transfer Protocol

1. Introduction

In recent years, there has been a rapid advancement in AI-powered mobile applications, particularly within the well-being sector. As these applications have become increasingly mainstream, the market has grown oversaturated, often with limited regulation regarding how user data is collected and handled. At the same time, the continued expansion of the cosmetics industry and the widespread daily use of its products have contributed to a decline in consumer understanding and awareness of product ingredients. Additionally, many cosmetic products do not address, or even mention, common skin conditions on their packaging or how these conditions may affect consumers.

This causes issues with purchasing decisions, especially if the consumer has underlying dermatological conditions that can be triggered by products containing harmful ingredients. While there are minor tools that can help consumers purchase the correct products, they're often limited, offer little to no control over the data output, do not provide ways to input users' conditions, and mostly rely on LLMs' pre-trained data.

This project aims to develop a mobile application that addresses the limitations of existing cosmetic ingredient analysis tools, while improving awareness and supporting informed decision-making for users with skin conditions. Using computer vision and text extraction, we can simplify this process for consumers by developing a simple on-hand tool. When it comes to the output data, we'll apply RAG for analysis, providing a clear, user-friendly explanation of ingredients and a clear UI/UX outlook. This ensures that less technically experienced consumers are not excluded from purchasing cosmetic products due to concerns about triggering skin flare-ups.

By delivering this solution through a mobile application, users can conveniently access the service without needing to conduct in-depth ingredient research, utilising a device that is already widely owned: a mobile phone. Using AI, we can formulate personalised evaluations while maintaining full control over the output through prompt engineering and RAG, thereby mitigating the risk of AI hallucinations.

While this software is not a medical tool, it is based on detailed medical research into various dermatological conditions, consumers' current understanding of ingredients, and in-depth research into effective methods for preventing flare-ups.

This thesis explores the relevant literature surrounding AI in well-being applications and cosmetic ingredient analysis, followed by an investigation into user needs. It then outlines the design and development of both client-side and server-side system architecture, including database design and UI considerations. Finally, the implementation process is evaluated, along with challenges encountered, the role of AI in development, and testing across functionality, usability, and performance.

2. Literature Review and Technical Research

2.1 Introduction

Within the European Union, a large proportion of the population suffers from various chronic skin diseases, including eczema, psoriasis, and acne, which require long-term care and ongoing skincare management. For many individuals with complex or chronic skin conditions (including trichological disorders), skin health is closely linked to both physical well-being and overall psychological quality of life.

Although cosmetic and dermatological remedies are readily available, consumers often encounter challenges with ingredient lists, vague marketing claims, and a lack of transparency about ingredient compliance and ethical standards. This can make it hard for individuals to make informed decisions, especially if they have pre-existing health conditions, sensitivities, or moral concerns. In recent years, AI-powered ingredient analysis apps have emerged as helpful tools, offering quick, personalised skincare evaluations.

The trustworthiness, transparency, and regulatory compliance of such tools remain uncertain, especially with outputs from LLM's that may not provide clear sources. Misuse of AI-generated content can lead to ineffective or even harmful product applications. The article explores the convergence of dermatology, formulation regulation, and AI analysis systems in cosmetics. It reviews common skin conditions, risk factors, regulatory challenges, and ethical considerations before evaluating current ingredient analysis tools. The discussion concludes with technical approaches like computer vision, natural language processing, and factchecking, aimed at improving the accuracy, security, and credibility of AI-driven skincare analysis.

2.2 Analysis of Harmful Skin Care Ingredients and Their Effects on Chronic Dermatological Conditions

-Introduction

This subchapter focuses on the causes and effects of various dermatological skin conditions and their overall impact on skincare management. Focused on factors influencing chronic dermatological conditions: their causes and triggers. Continuing from this, analysing common symptoms from all the skin conditions/diseases, including their beneficial ingredients used in treatment. Referring to research studies focusing on the impact of skin diseases on European populations and the prevalence of conditions.

Additionally, examining common misunderstood ingredients in dermatological products, consumer interpretation of ingredients, and alternative naming of harmful ingredients. Briefly discussing the ethics of commonly used ingredients in cosmetic products, and their associated risk factors to everyone.

We aim to effectively describe, research, and analyse the most common conditions, assess their associated risk factors across multiple datasets. Whilst discussing methods in which we can relieve symptoms and advocate potential treatment solutions.

2.2.1 Understanding Dermatological Conditions: Causes and Evidence-Based Skincare Management

I. Common Skin Conditions and Their Causes

There is a wide variety of skin diseases, conditions, and allergens affecting many individuals in Europe. It is essential to establish a basic understanding of common skin conditions and their underlying causes. Although multiple skin conditions and diseases exist, most share a common factor. Some researchers believe that we can partially attribute these cases to Climate Change, diet, family genetics, and much more. Not only can these changes cause skin conditions, but they may also be a direct link to various skin cancers.

A study conducted in 2022, focusing on mainland Europe, found that over 43.35% of individuals aged 18 and above suffer from some form of skin disease [14]. With around 23 million people (5.5% overall) having some form of Atopic dermatitis (eczema), another 17 million (3.9% overall) with Psoriasis, and finally 23 million individuals (5.4% overall) with chronic acne. With a further 6 conditions falling within alopecia (5.1%), rosacea (2.0%), chronic urticaria (1.1%), non-melanoma skin cancers (1.1%), and vitiligo (0.8%). While these may not be the highest in the recorded figures (that falls to fungal infections around

8.9%), they all share one common trait: the chronic nature of these conditions. Oxford Dictionary defines a chronic skin condition as “lasting a long time, especially for a disease or problem,” meaning incurable in most cases.

Caring for and managing these conditions is vital to preventing further impact on the affected individual, both physically and psychologically. [14][17]

II. Skin Conditions Symptoms and Beneficial Ingredients

Although each condition has unique symptoms, we can identify a common complaint across all previously mentioned diseases. These vary from skin tightness/excessive dryness, Itching or pruritus (pruritus: uncomfortable sensation that creates an urge to scratch), skin scaling (across purely topical skin conditions such as Psoriasis) usually due to dryness of the skin, and inflammation (seen across all conditions), redness, and burning sensations [3][19].

Although many modern laboratory chemicals can alleviate some of these symptoms, there remains a high risk that they can cause various adverse reactions. Continuous use can lead to tolerance to these chemicals, reducing their effectiveness. As a result, plant-based ingredients, or ‘natural’ ingredients, can provide additional long-term benefits. These commonly found ingredients are listed below:

- Salicylic Acid

- Allium sativum (Garlic)

- Aloe vera (reduces redness)

- Mangifera indica (Mango, high anti-inflammatory properties)

- Portulaca oleracea (Purslane, for itching and burning sensations)

This reinforces the importance of selecting treatments that balance effectiveness with long-term skin tolerance and advocating for naturally sourced ingredients.[16]

III. Why aren’t products labelled as unsuitable for specific skin conditions?

In the European Union, many products are allowed to advertise for specific skin types. These kinds of labels include ‘for oily skin,’ or ‘for dry skin’, while others claim to be suitable for all.

As mentioned earlier, individual ingredients have distinct properties that can benefit specific skin issues but may also worsen others. This leads into a

discussion about current EU regulations: why are cosmetic products not required to indicate if they might be unsuitable for certain skin conditions, especially since marketing often uses generalised terms to describe ingredient effects?

Product labels are a crucial tool for communication between consumers and the cosmetic manufacturer; a lack of standardised naming conventions can cause confusion between advertised benefits and the practical application of ingredients.

A study was conducted based on false advertising of labels, promoted by healthcare professionals, stating, "While many claims align with scientific methods and regulations, several terms remain vague or lack contextualization. Phrases such as ‘natural origin,’ ‘safe,’ or ‘clinically evaluated’ were often inconsistent and presented without clear methodological references. This lack of standardisation may hinder consumer interpretation and limit healthcare professionals’ ability to assess product relevance." [3].

The conclusion drawn from this is that cosmetic manufacturers avoid directly targeting specific skin conditions in their product claims, as doing so may raise regulatory concerns under European Union guidelines, mainly due to the presence of already false claims created by the product manufacturers.

2.2.2 Harmful Ingredients and Their Dermatological Impact

I. Common Skincare Ingredients and Their Associated Risk Factors

The Scientific Committee on Consumer Safety (SCCS) is the European Union's advisory body responsible for evaluating the safety of cosmetic ingredients under Regulation (EC) No 1223/2009. Ingredients identified as hazardous are classified according to criteria outlined in Regulation (EC) No 1272/2008, and are as follows: [20]

- Skin Sens 1: The ingredient is a skin sensitiser (may cause allergic skin reaction).
- Skin Sens 1A: Strong (high potency) sensitiser considered a more severe hazard.
- Skin Sens 1B: Moderate to weak sensitiser.
- Carcinogenic, meaning ingredients are linked to or can cause cancer.
- Mutagenic (causes genetic mutations)
- CMR (toxic for reproduction)

While there are many commonly found ingredients in beauty products, such as preservatives like hydroxyisohexyl 3-cyclohexene carboxaldehyde (HICC), formaldehyde stands out the most.

Formaldehyde, as its own standalone ingredient, is classified as carcinogenic, mutagenic, and a Skin Sensitiser Category 1 substance, posing a significant toxicological risk at elevated concentrations. While it is a prohibited substance under Annex II of Regulation (EC) No 1223/2009, it is still allowed to be used in the form of 'releasers', provided that the total free formaldehyde concentration in the finished product does not exceed 0.2%.

Formaldehyde has been proven to cause sensitisation, allergic reactions, and may also induce dermatitis. If the free formaldehyde content in a cosmetic product exceeds 0.05% (500 ppm), the product must be labelled with the warning "contains formaldehyde."

The distinction becomes more complex in the context of formaldehyde-releasing preservatives.

Releasing formaldehyde means that a certain ingredient, over time, releases formaldehyde as a byproduct, such as DMDM Hydantoin, but cannot bypass the 0.2% overall content.

This causes ethical concern and potential harmful reactions to individuals who are allergic to formaldehyde, as they are unaware of the ingredient being present.

These examples demonstrate how commonly used cosmetic ingredients can pose sensitisation risks, reinforcing the need for strict regulatory oversight and transparent ingredient disclosure. [19][3]

II. Use of Alternative Names for Harmful Ingredients

As previously mentioned, the Scientific Committee on Consumer Safety (SCCS) provides scientific guidance that informs the regulation and prohibition of certain cosmetic ingredients. Individual substances may be classified under the Skin Sens 1 hazard category based on their sensitisation potential. These lists are more commonly referred to as negative lists. The EU defines these lists as follows:

- ‘’Negative Lists: Official lists established under Regulation (EC) No. 1223/2009 that identify substances which are either prohibited or restricted from use in cosmetic products.
- Prohibited substances are those banned entirely due to safety concerns.
- Restricted substances are those allowed under specific conditions, such as limits on concentration, product types, or labelling requirements to ensure safe use.’’ [22]

This implies that the ban applies to the substance itself, regardless of the name used, thereby preventing manufacturers from using alternative names.

Though it satisfies the regulatory limitations, it doesn’t apply to ingredients that fall within the scope of ethical concerns. Ingredients may be renamed by producers, which may end up causing consumers to unwittingly buy things that violate personal or moral beliefs, thus hampering the ability to ethically make decisions, like the occasion with releasers of formaldehyde.

The difference illustrates a divide between regulatory compliance with safety and the ethical transparency that consumers increasingly demand. [2]

III. Consumer Interpretation of Ingredients

While manufacturers can use alternative names for ingredients, they must maintain a standard set by the International Nomenclature of Cosmetic Ingredients (INCI). Such standardised, recognised names for cosmetic ingredients are provided by this system for transparency and consistency, which was seen with DMDM Hydantoin. Accordingly, safe or regulated ingredients must be reported under the appropriate INCI name, which they cannot change to avoid compliance or mislead consumers or regulators. [6]

But the major issue arises when consumers aren't informed about the frequent changes to the listed ingredient names. Ethical behaviour is not observed if products, contrary to people's personal or ethical beliefs, are bought through unawareness due to insufficient transparency. This leads to the question of why consumers are not informed enough.

That's because there has been little or no communication with consumers so far, or oversight over companies that routinely undertake these renaming practices. Such data implies that where ingredient renaming practices hinder transparency, oversight and enforcement are inadequate. This indicates a need and scope for greater regulatory oversight and enforcement.

IV. Use of Ingredients Considered Ethically Problematic

The use of certain ingredients in skincare products has been the subject of ongoing controversy, with palm oil being notable. Palm oil is a cosmetic ingredient (and is also used in food) derived from the fleshy fruit of oil palm trees, primarily sourced from Indonesia and Malaysia. [1]

A study conducted in 2022 notes that in Nigeria, palm oil production is largely characterised by traditional small-scale wild grove harvesting systems, which are associated with relatively low yields. As a result, large-scale land purchases usually involve corruption, displacement, poor labour conditions, and many breaches of human rights. Due to its high demand in the cosmetic industry, palm oil causes significant damage to the environment in which it is produced. Contributing to deforestation, forced relocation of wildlife, pollution of the ecosystem, and the quality of life for individuals in the surrounding areas contribute to this challenge to sustainable development. [5]

Despite these concerns, the ingredient is fully licensed by the SCCS and is usually valued as a moisturising/nourishing property for the skin, and has no limitation on amount allowed, like seen with Formaldehyde.

Unfortunately, palm oil is a commonly used natural ingredient in many dermatological products, usually found in shampoos, conditioners, and many other products within the broader skincare ecosystem.

Palm oil remains widely used in skincare products due to its functional properties, despite ongoing concerns regarding its environmental and social impacts.

2.3 Limitations Of Current Analysis Tools

-Introduction

Recently, there has been a rise in AI-based applications, predominantly those that deliver personalised outputs tailored to each user. As dependence on these applications grows, users become increasingly influenced by them. Such applications are often assumed to be reliable sources, without any evidence or citations. In this section, we compare these applications based on factors such as the source of information, reliability, and evidence of accuracy.

Additionally, we will examine the application at the consumer level, analysing components of the UI, user security, and overall assurance that individuals with pre-existing skin conditions can manage them effectively.

2.3.1 Reliability of Information

As ingredient-scanning applications gain in popularity, they come with important implications for consumers regarding the reliability and accuracy of the information they present to consumers. An examination of several applications was conducted, focusing on key factors such as the origin of information, potential bias arising from trained data, adherence to SCCS regulations, and transparency in how results are generated.

Whilst many apps provide rapid feedback once a product is scanned; there is no clear evidence for the sources of their information. Although these tools are frequently marketed as 'AI-based applications,' they offer limited evidence of the accuracy of their results. This raises concerns about the biased data returned to the consumer. Assuming these applications rely on computer vision and prompt engineering, it is reasonable to believe that most of their information is based on the pre-trained data, derived from the LLMs they are using. [23]

However, another issue arises from how the results are interpreted; oversimplified outputs fail to account for each consumer's needs, increasing the likelihood of errors occurring whilst limiting meaningful consumer understanding.

These factors can range from the consumer's ethical consideration regarding certain ingredients to underlying conditions that may be aggravated by continued use. As mentioned before, Formaldehyde, commonly found in trace amounts, can worsen the effects of dermatitis, causing further discomfort to the consumer.

In this context, the use of LLM models that lack regulation can cause instances of hallucinations. Hallucination in an AI model, as defined by IBM, is as follows:

"AI hallucination is a phenomenon where, in an LLM, often a generative AI chatbot or computer vision tool, perceives patterns or objects that are non-existent or

imperceptible to human observers, creating outputs that are nonsensical or altogether inaccurate." [22]

The lack of transparency within these applications can have negative implications, ranging from misleading results to fabricated outputs. Applications using AI should be held to a high standard in terms of regulations and transparency, allowing the consumer to make their own informed choices based on the outputs. Overall, clear disclosure of AI usage and its role in analysis is therefore essential for these applications.

2.3.2 Evaluation of Existing Ingredients Analysis Tools

By analysing existing applications for ingredient analysis, several common challenges were identified that contribute to the lack of reliability, usability, and suitability for those who have dermatological concerns. The biggest issue identified was the absence of evidence and citations; most applications did not offer verified sources to back their analysis outputs. The lack of referencing in this manner hampers user trust and the independent validation of claims made by the system.

The reliability of optical character recognition (OCR) was another important issue. Ingredient scanning frequently returned incomplete ingredient lists, misidentified products, or prompted users to select alternative items that did not correspond to the scanned product. Such errors are detrimental to the efficiency of automatic analytics and result in more errors. Monetisation had also restricted accessibility: many apps needed upfront payment or offered limited trial periods.

This limited valid assessment and lessened reach for users who didn't want or could not subscribe to services. There was also rarely any user personalisation, with many of these tools not asking for specific information about skin conditions, allergies, or past negative reactions, leading to generic outputs which neglected to cater for specific needs.

Other complications were found in inconsistent outputs, where some ingredients like salicylic acid were presented as both beneficial and detrimental without explanation or evidence. This inconsistency was aggravated by using outdated versions of AI models, where chat responses appeared more accurate than the primary analysis outputs, signalling dependence on less efficient systems. In one of the apps, citations were mentioned, but mostly on individual ingredients, rather than applying them to the overall suitability of the product, indicating a failure to use citations effectively at the product level.

2.4 Natural Language Processing with Computer Vision

-Introduction

This section explores the technical methods used to support accurate and reliable ingredient analysis within AI-driven skincare applications. It examines how computer vision, natural language processing, and output validation techniques can be combined to mitigate limitations commonly found in existing analysis tools. Beginning with Optical Character Recognition-based ingredient extraction, the chapter outlines how visual data from product labels can be converted into machine-readable text. Lastly, methods for the accuracy of data are evaluated, such as Retrieval-Augmented Generation, human-in-the-loop validation and prompt engineering, that are intended to enhance the transparency, reliability, and safety of AI-based ingredient analysis systems.

2.4.1 OCR-Based Ingredient Extraction Using Computer Vision

Optical Character Recognition (OCR) is a component of computer vision that enables systems to interpret data for natural language processing. It turns images with text into digital text that computers can read and use in different systems.

Generally, consumers take an image with a mobile phone or digital camera and feed it into an OCR-based system. The images go through noise reduction, contrast enhancement, and sharpening to enhance character visualisation prior to any extraction. This depends on image quality and resolution; the higher the resolution, the more accurate the system will be. The OCR system also detects segments of the image containing potential text and decodes the potential text into text blocks, lines, words, and ultimately individual characters.

Many computer vision models used for OCR are trained to recognise a wide range of fonts, sizes, and even handwriting, which helps ensure their reliability. Character recognition reconstructs the extracted symbols into machine-readable digital text.

The OCR engine also corrects errors, such as misread letters, and improves the text data to enhance accuracy. The final digital text is often formatted as a JSON object to enable its use in other applications. [13]

This process allows us to leverage an already tested system to address small skincare labels, which are already difficult to read to the human eye. Once the information is extracted and reassembled, it can be used across many applications, ranging from ingredient explanations to the detection of potentially harmful components hidden in the fine print.

Additionally, by using pre-trained computer vision models, output reliability and speed are optimised, and latency issues are not a significant concern. In

multimodal systems, pre-trained vision components perform OCR efficiently, while language models are applied at later stages for interpretation and analysis. Combining this with user-entered data (e.g., skin conditions, allergens) provides an effective and efficient way to analyse ingredients in skincare products while remaining convenient.

2.4.2 NLP-Based Ingredient Interpretation and Prompt Engineering

Natural Language Processing (NLP) enables computational systems to understand, interpret, and generate human language, facilitating effective human–computer interaction. In applied AI systems, NLP is used within LLM's where users can ask natural language queries and receive an NLP response.

Prompt engineering is a technique used in NLP-based language models that enables developers to selectively control the formulation of AI outputs by constraining factors such as structure, terminology, reasoning scope, and user context.

In analysis-based OCR systems, such as the approach discussed previously, prompt engineering plays a key role in interpreting extracted ingredient data. By utilising ingredient extraction, this approach enables the definition of structured outputs, including product ratings and overview systems, and supports the standardisation of hazardous ingredient naming in a manner that is understandable to the general consumer, thereby constraining language model behaviour at inference time.[8]

This technique helps reduce instances of model hallucinations by limiting unsupported information generation. Prompt design can also assist with NLP pipelines: using consistent terminology adaptation within these NLP pipelines to ensure ingredients referred to under varying names are identified as their official INCI classifications, avoiding misleading or unethical ingredient representation. Maximising prompt engineering also results in the use of large LLM's without requiring additional model training to guide them to generate more accurate, relevant, and context-aware outputs in analysis-driven applications.[9]

The optimisation of prompt engineering can allow such an LLM to be employed effectively without performing additional training, so that they can produce more

precise, relevant, and contextually appropriate outputs in analysis-driven applications.

2.4.3 Ensuring Factual Accuracy in AI Outputs

As outlined earlier, unregulated use of AI outputs can undermine the credibility of the information these applications return. In section 2.2: Limitations of Current Ingredient Analysis Tools, it was discussed that many analytics-based apps rarely provide evidence of their information sources or rely on simple ingredient explanations rather than the methods by which they operate. This implies the possibility of reliance on the pre-trained data from the LLMs they are using. A study conducted earlier this year by Jacob Schoeffer outlined issues with adherence to LLM recommendations, finding an average of 40-60% across three testing rounds, and returning information as factual without checking the facts themselves. [23]

By comparison, implementing a method that regulates output based on facts rather than trained data should use the retrieve-augment-generate (RAG) approach. ‘Retrieve’ fetches information from external sources, usually from stores that implemented OCR. Augmentation is the integration of this knowledge into the model's internal workings. And finally, generation produces the text contextually and accurately alongside the LLM’s response.

This allows LLMs to draw on articles, books, and other forms of literature to shape their outputs. This transforms how information output is managed within LLMs, preventing issues such as hallucination while providing factual evidence backed by research papers. Unfortunately, this can also lead to biased results, as developers may not use factual articles but rather those that align with their personal beliefs.

Conventional LLMs, such as BERT and GPT-3, are essentially static and depend mainly on their pre-trained data. However, as they are smaller in size, they can compute faster than some larger models, which makes them suitable for tasks where speed is more important than accuracy. By leveraging RAG for some systems, they can efficiently generate results from a trusted, credible source. [12][10]

Additionally, you can incorporate a method called ’Human in the loop’ review, which, if triggered. Call for an external check of data output via actions such as the report button. This allows human reviewers to cross-check AI outputs, evaluate the reliability of sources, and ensure that the information provided is factual rather than inferred. This can be considered post-generation validation as an additional security measure to ensure the data output is correct.

Ultimately, combining Retrieval-Augmented Generation with a human-in-the-loop validation system presents a practical solution to the limitations of unregulated LLM outputs. By grounding responses in verifiable sources and implementing a secondary verification level, these methods enhance transparency, accuracy, and user safety in AI-based ingredient analysis applications.

2.5 Conclusion

Chronic skin conditions are common and often need long-term management, but choosing suitable products is harder than it should be. Ingredient lists are difficult to interpret, marketing terms like “safe” or “for all skin types” are vague, and products rarely warn when an ingredient may be unsuitable for specific conditions. While some ingredients relieve symptoms, they can still trigger irritation or tolerance with repeated use. Regulatory guidance (e.g., SCCS negative lists) supports safety controls, but ethical transparency and naming practices are still unclear from a consumer point of view.

Testing existing ingredient analysis apps showed that current tools do not reliably solve these issues. Many systems provided outputs with little or no evidence, meaning users cannot verify claims. OCR performance was inconsistent, often returning incomplete ingredients or the wrong product match. Several apps also lacked personalisation (no allergies/conditions asked), had paywalls or short trials that limited access, and sometimes produced contradictory ingredient explanations without justification. Overall, the gap is not just regulation; it is clarity, transparency, and trustworthy interpretation. This supports the need for better systems that combine accurate extraction with verifiable, user-specific analysis.

3. Requirements and Background

3.1 Introduction

This section explores applications with similar functionality for scanning and evaluating beauty products. While few offer the same features, the most relevant options were selected. Three applications are analysed, ordered from least to most downloaded on the App Store and Google Play, with consideration of factors affecting their popularity. Each is evaluated based on functionality, UI, ease of use, and the presence of credible scientific citations.

3.2 Similar Applications

When researching similar applications, several issues became clear. Many were locked behind paywalls or offered only brief trials, while free versions showed nearly identical designs and outputs, suggesting shared AI models. Their results were often inconsistent and lacked clear reasoning, indicating reliance on pre-trained data without proper validation. This raises concerns that competitors prioritise revenue over effectively supporting users with skin conditions. This highlights the need for a solution that supports users with skin conditions more effectively.

3.2.1 Application A

The application’s design aligned with the platform's nature, featuring light styling and minimal animations. However, a few issues were identified. Despite being advertised for the Irish market, it included pricing in US dollars, links to US wholesalers, and references to US-based clinical centres.

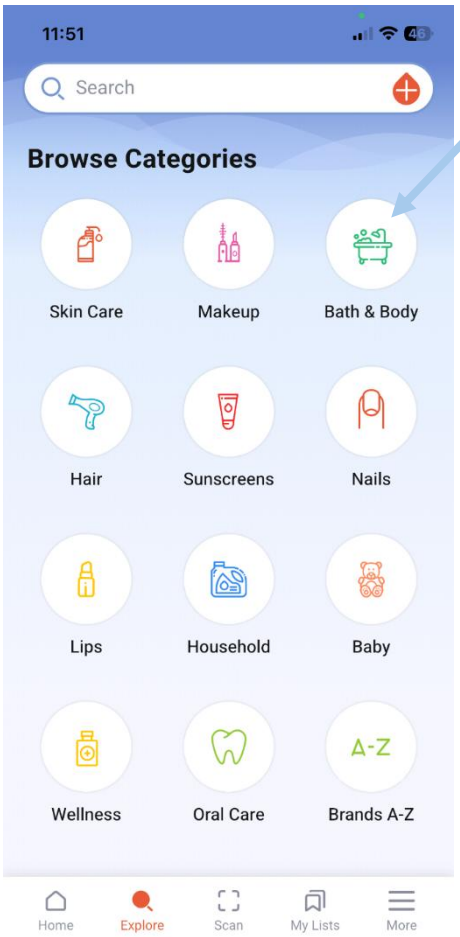


Figure 1. Shows the Explore Section of App A.

- Provides various options for product analysis, such as nails.
- Clean design.
- Follows Nielsen's 10 Usability Heuristics (principles 1–8).

(see Appendix 2)

- US Currency Icons

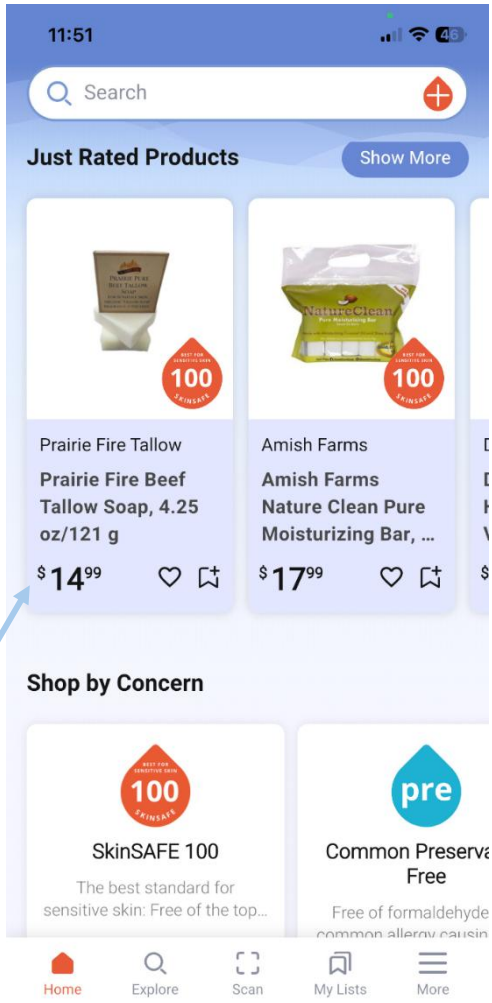


Figure 2. Home Section of App A.

Figure 5. Product Scan of App A.

Figure 4. Product Scan of App B.

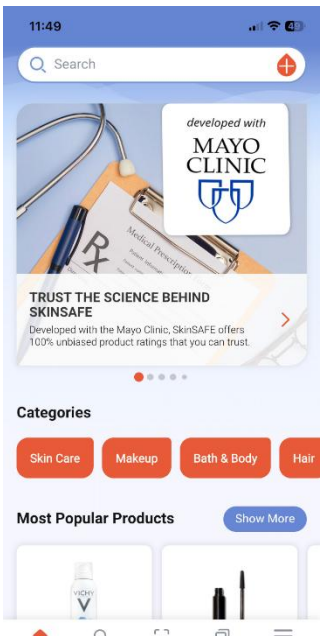


Figure 3. Landing Page promoting US Medical Centre.

The application relied on barcode detection to scan products; however, most UK and Irish products tested did not support this feature. The capture mode was automatic, with no option for manual input, and it remained unclear whether barcodes were being successfully detected. As a result, the analysis and evaluation could not be effectively assessed.

3.2.2 Application B

The second application used a simpler design approach, following a dark mode layout with visually appealing and well-designed buttons.

However, the design was oversimplified. Users could only select a single skin type, failing to account for combination skin or specific dermatological conditions.

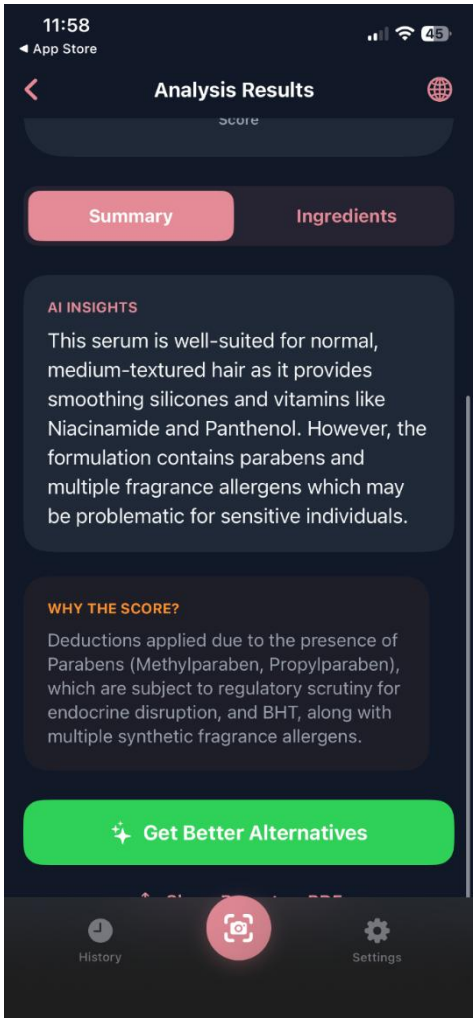
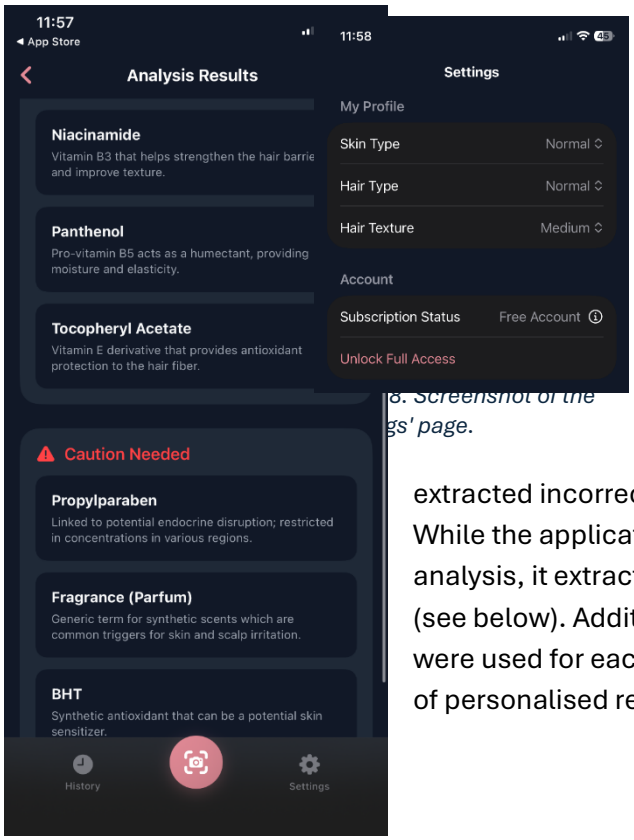


Figure 7. Result 'Summary' page of App B.

Figure 6. History page of App B.



8. Screenshot of the gs' page.

While the application provided product analysis, it extracted incorrect ingredient data (see below). While the application provided product analysis, it extracted incorrect ingredient data (see below). Additionally, only three ingredients were used for each pro and con, despite claims of personalised results.

- Mentions 'Propylparaben', a known allergen, causing redness, burning and itching.
- Is not present in the official ingredient list for this beauty product.
- Only shows three ingredients per pro and con.
- Ingredients available on Appendix 4

Figure 9. Screenshot of Analysis Results from App B.

Testing different user settings produced identical outputs, indicating limited personalisation and a lack of user-specific analysis.

The application also lacked a traditional landing page, instead relying on navigation through a lower control bar or within-page interactions.

3.2.3 Application C

The third and final application was the most promising, featuring a sleek and elegant design with a range of user customisation options. It also provided citations for each analysis, detailed ingredient explanations, a clear scoring method, and highlighted key evaluation points. Additionally, an AI agent was included to further discuss results.

Figure 10. Scoring page Application C.

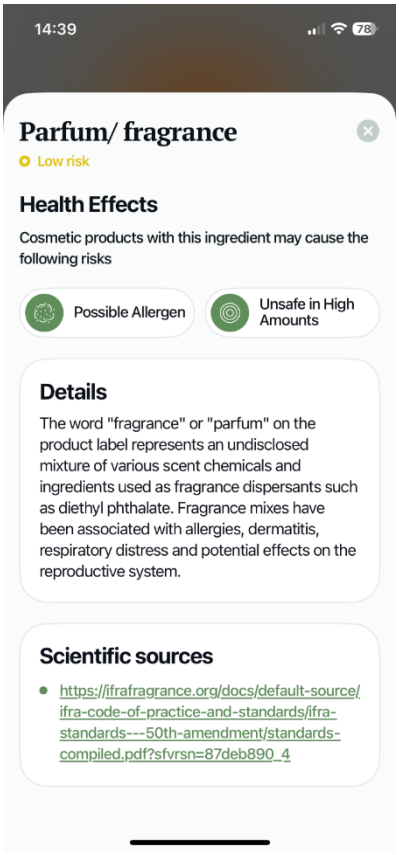


Figure 11. Result page Application C.

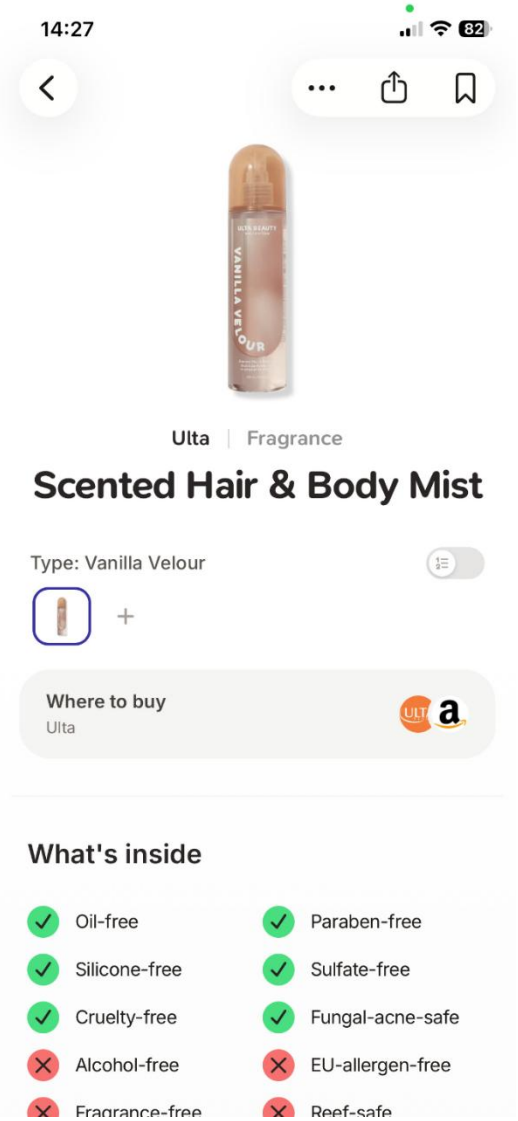


Figure 12. Result page depicting the analysed product.

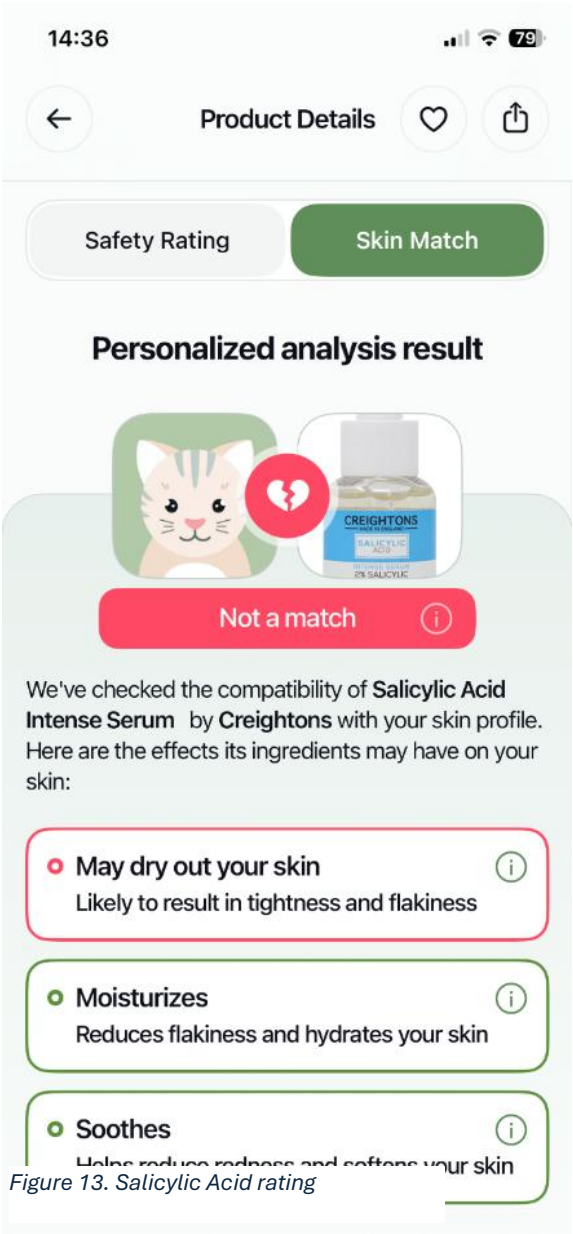


Figure 13. Salicylic Acid rating

However, several limitations were identified:

- Image analysis appeared to rely on assumptions about product shape and name, matched against an internal database
- When products were not found, incorrect items were returned
- No option was provided to confirm or reject the detected product
- As a result, evaluations were sometimes based on the wrong item (e.g. Bershka Vanilla body spray)

Issues were also found with result consistency:

- Analysis stated that *Salicylic Acid* was not compatible with the user's skin
- The AI agent later contradicted this, stating the product was suitable
- This highlights a lack of alignment between structured results and AI-generated responses

3.3 Surveys

For the survey phase, two main surveys were developed, both targeting individuals with dermatological or trichological conditions. Survey 1 aimed to assess the need for the proposed product evaluation application, while Survey 2 focused on identifying user requirements for such a system. Key findings are discussed below.

3.3.1 Survey One: Skincare Product Decision & Experiences Survey

The premise of this survey was to collect necessary information about individuals with dermatological conditions, focusing on their personal experiences and beliefs.

Primarily, attempting to gather data on these individuals, including their shopping experience and challenges in identifying non-triggering products. Below, we'll discuss three key findings and how they can help create an application that may resolve some of these issues.

- **Finding One:** Number of Users suffering from Skin Conditions

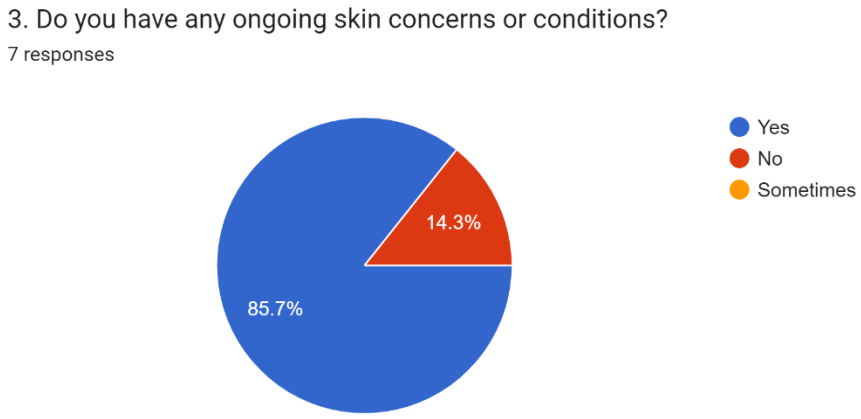


Figure 14. Survey Question 3 from Survey 1

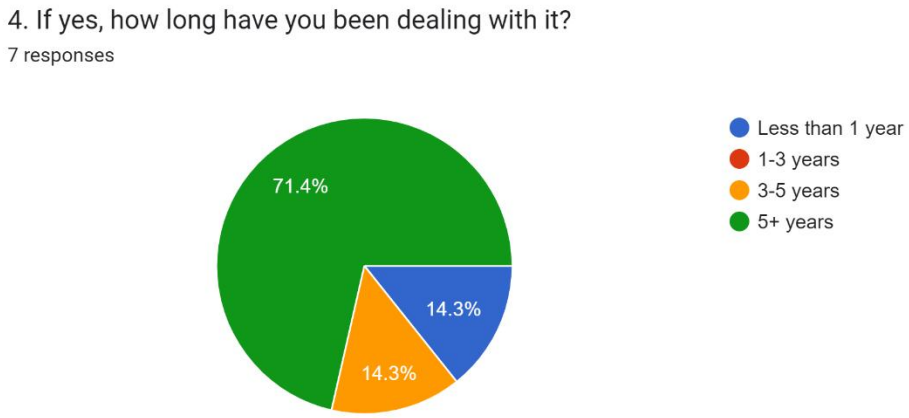


Figure 15. Survey Question 4 from Survey 1.

According to the European Union, 20–30% of the population in Europe lives with a dermatological condition, rising to 43–50% within a given year. However, our findings differed, with around 85% of participants reporting a current condition, many lasting over five years. Although these conditions may fluctuate, they can cause ongoing stress due to their recurring nature.

- **Finding Two:** Product Choice when shopping with skin conditions.

Figure 16. Survey Question 14, Survey 1.

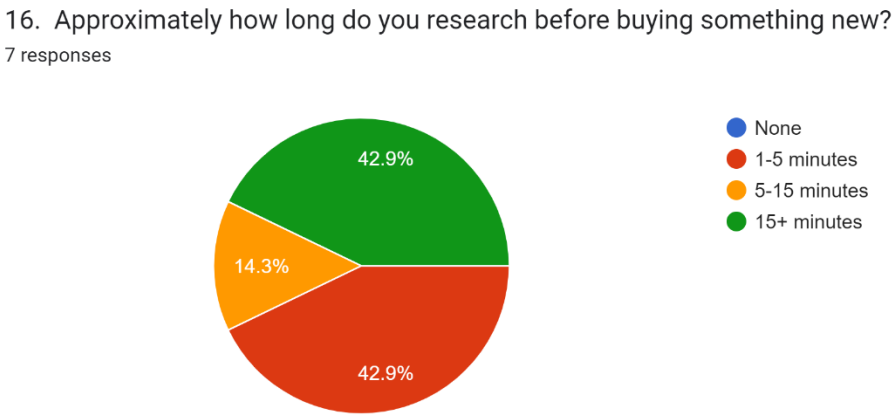


Figure 17. Survey Question 16, Survey 1.

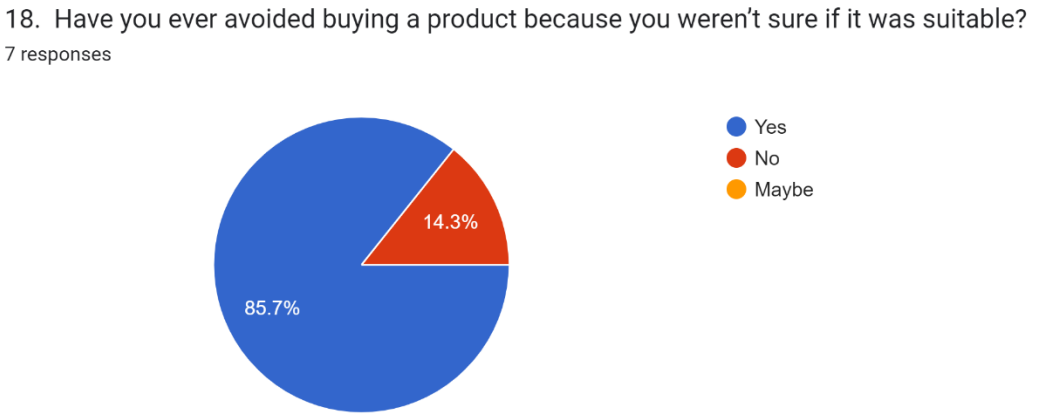


Figure 18. Survey Question 18, Survey 1.

The next step in the survey examined product suitability, with most responses assumed to reflect users with skin conditions. Many participants reported spending significant time researching products, often over 15 minutes, and up to 85.7% avoided purchases if a product seemed unsuitable. This highlights the need for an intelligent system that analyses ingredients and provides personalised evaluations to support informed decisions.

- **Finding Three**: Testers' Comments

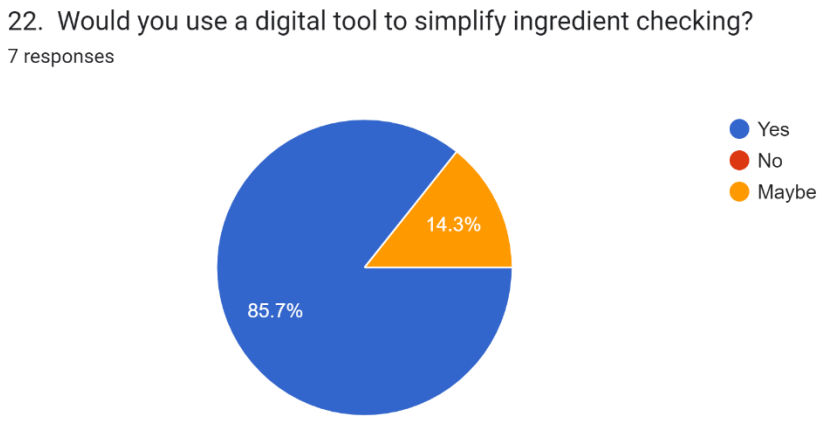


Figure 19. Survey Question 22, Survey 1.

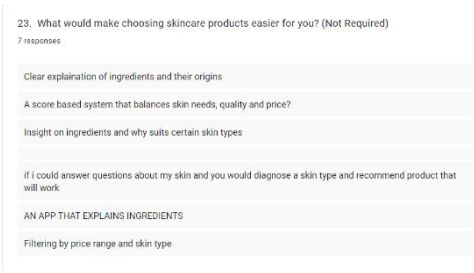


Figure 20. Survey Question 23, Survey 1.

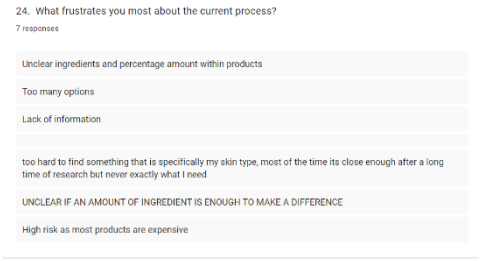


Figure 21. Survey Question 24, Survey 1.

The survey results show strong demand for a tool to simplify ingredient checking, with 85.7% of respondents saying they would use it. Many users also showed a lack of understanding of skincare ingredients and wanted clear explanations and suitability guidance.

There was a strong preference for personalised features, such as skin analysis and tailored recommendations. Users also expressed frustration with current options, citing unclear information, too many choices, and the financial risk of trial-and-error purchasing.

Overall, these findings highlight the need for a user-friendly system that provides clear, reliable, and personalised support for decision-making.

(Full list of survey responses available in Appendix [7])

3.3.2 Survey Two: Product Suitability App – Feature Expectations Survey

The primary focus of this survey was to better understand users' expectations around applications like the one proposed by this report.

Below, we'll discuss two key findings and their implications.

- **Finding One:** Personalisation and wanted features.

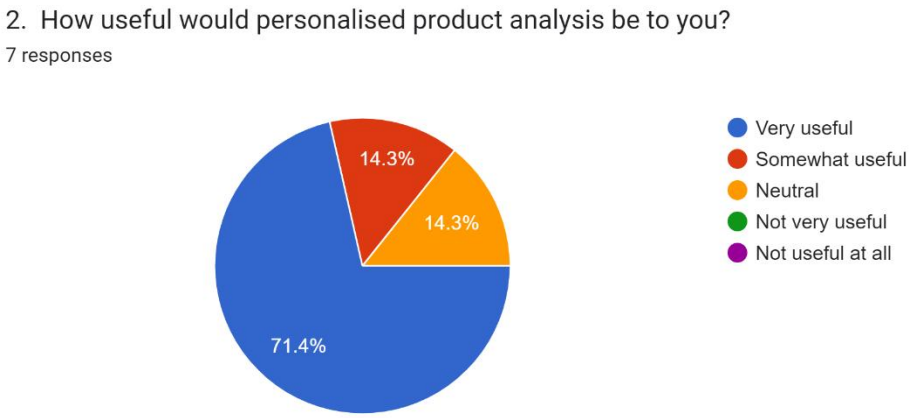


Figure 22. Survey Question 2, Survey 2.

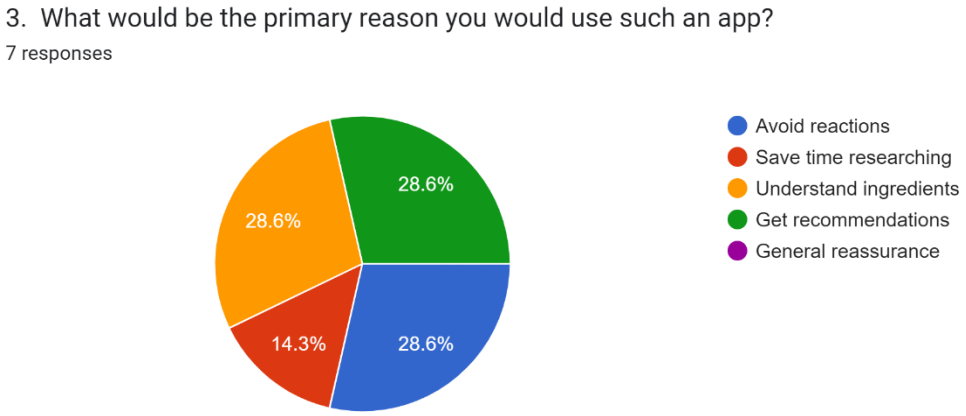


Figure 23. Survey Question 3, Survey 2.

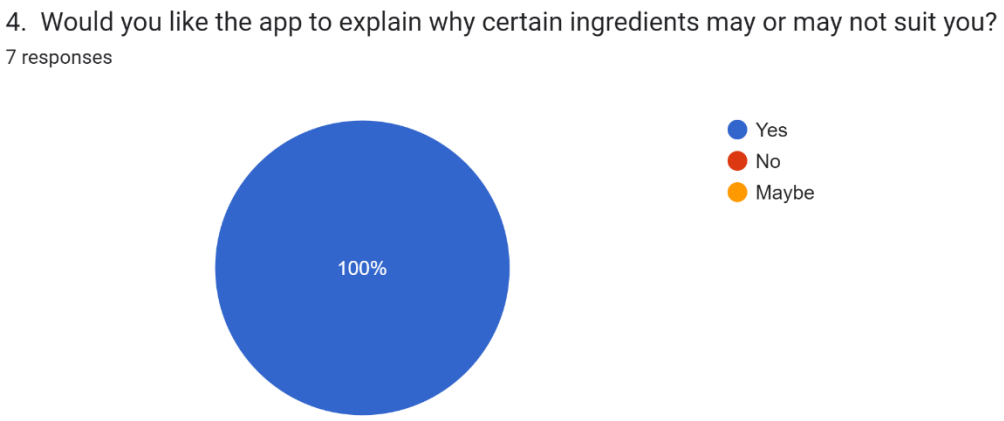


Figure 24. Survey Question 4, Survey 2.

The survey results support the value of the proposed application, with 71.4% of users finding personalised product analysis very useful and no negative responses recorded. Users were motivated by avoiding adverse reactions, understanding ingredients, and receiving recommendations, highlighting the importance of both safety and education.

Additionally, 100% of respondents wanted ingredient explanations, reinforcing the need for transparency. Overall, this shows strong demand for an application that scans products and provides clear, personalised evaluations.

- **Finding Two:** User Engagement & Purchase Influence

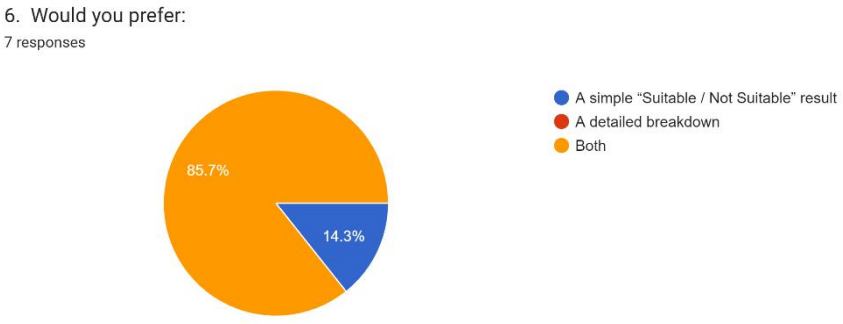


Figure 25. Survey Question 6, Survey 2.

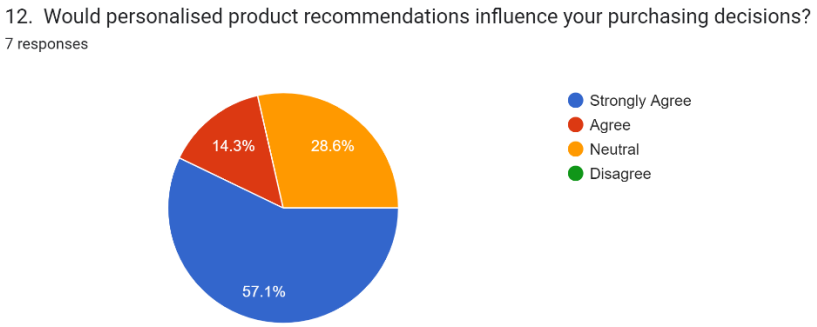


Figure 26. Survey Question 12, Survey 2.

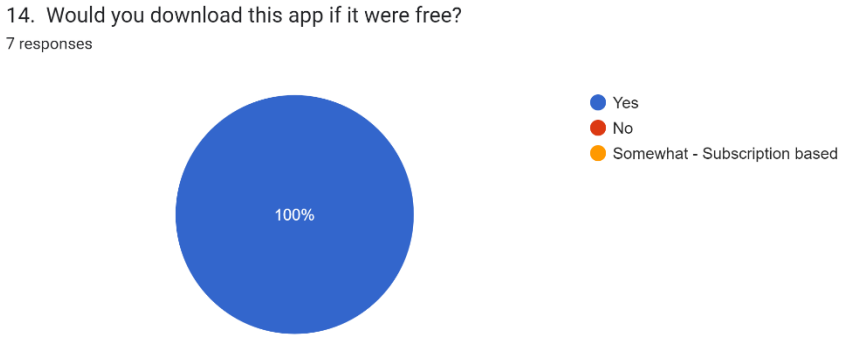


Figure 27. Survey Question 14, Survey 2.

These results further reinforce strong user acceptance of the proposed application. All respondents were willing to input their skin conditions for personalised results, showing minimal resistance where clear value is provided.

Most users (57.1%) strongly agreed that personalised recommendations would influence their purchasing decisions, with the remainder also agreeing. Additionally, 100% said they would download the app if it were free, indicating strong interest and high potential for adoption.

(Full list of survey responses available in the Appendix [8])

3.4 Personas

Below are three samples of the intended demographic for this application. These are presented as personas, fictional representations of the application's intended users. Focusing on a wide variety of individuals showcases the application's duality and how it presents itself to users.

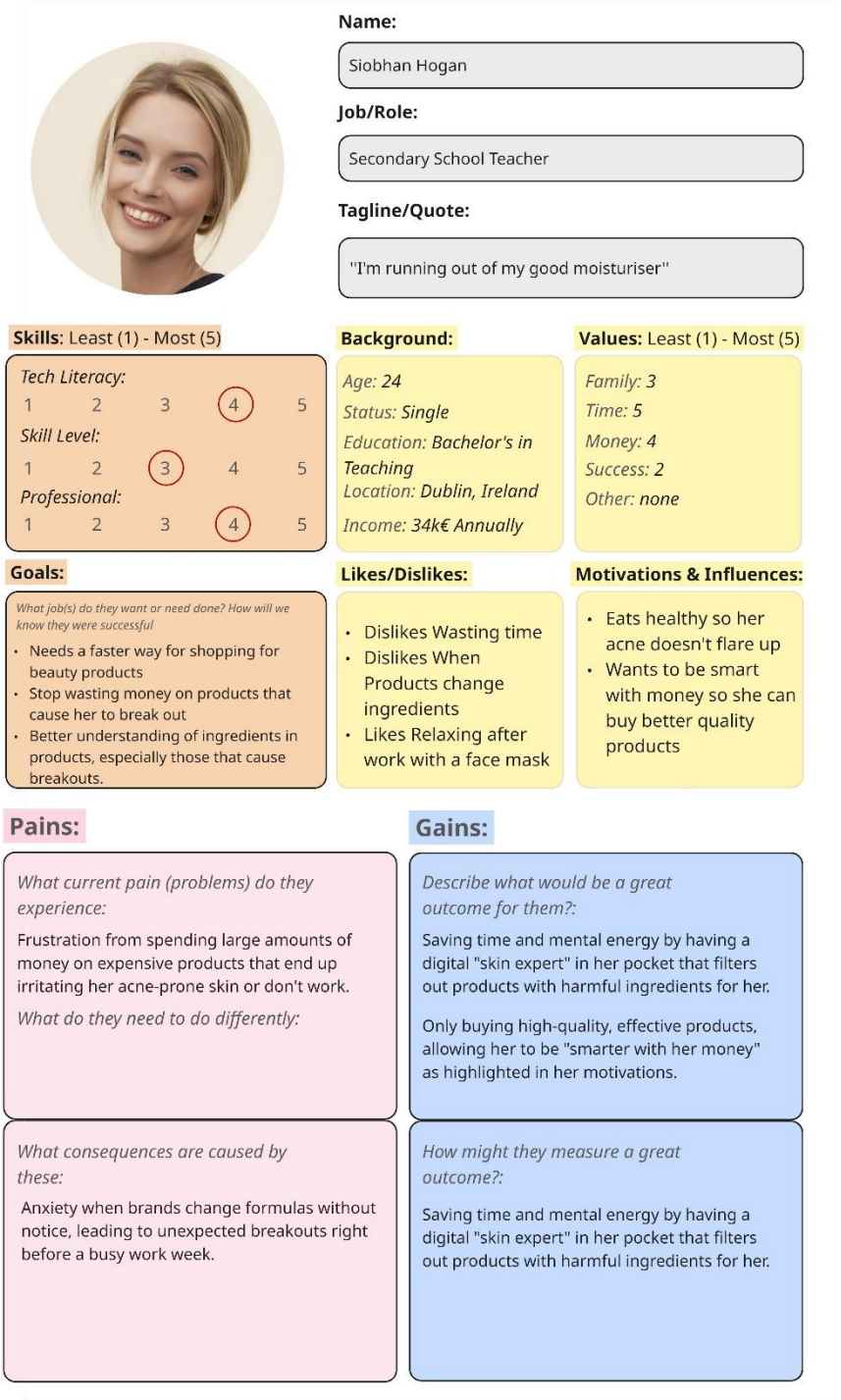


Figure 28. Persona 1 'Siobhan'.

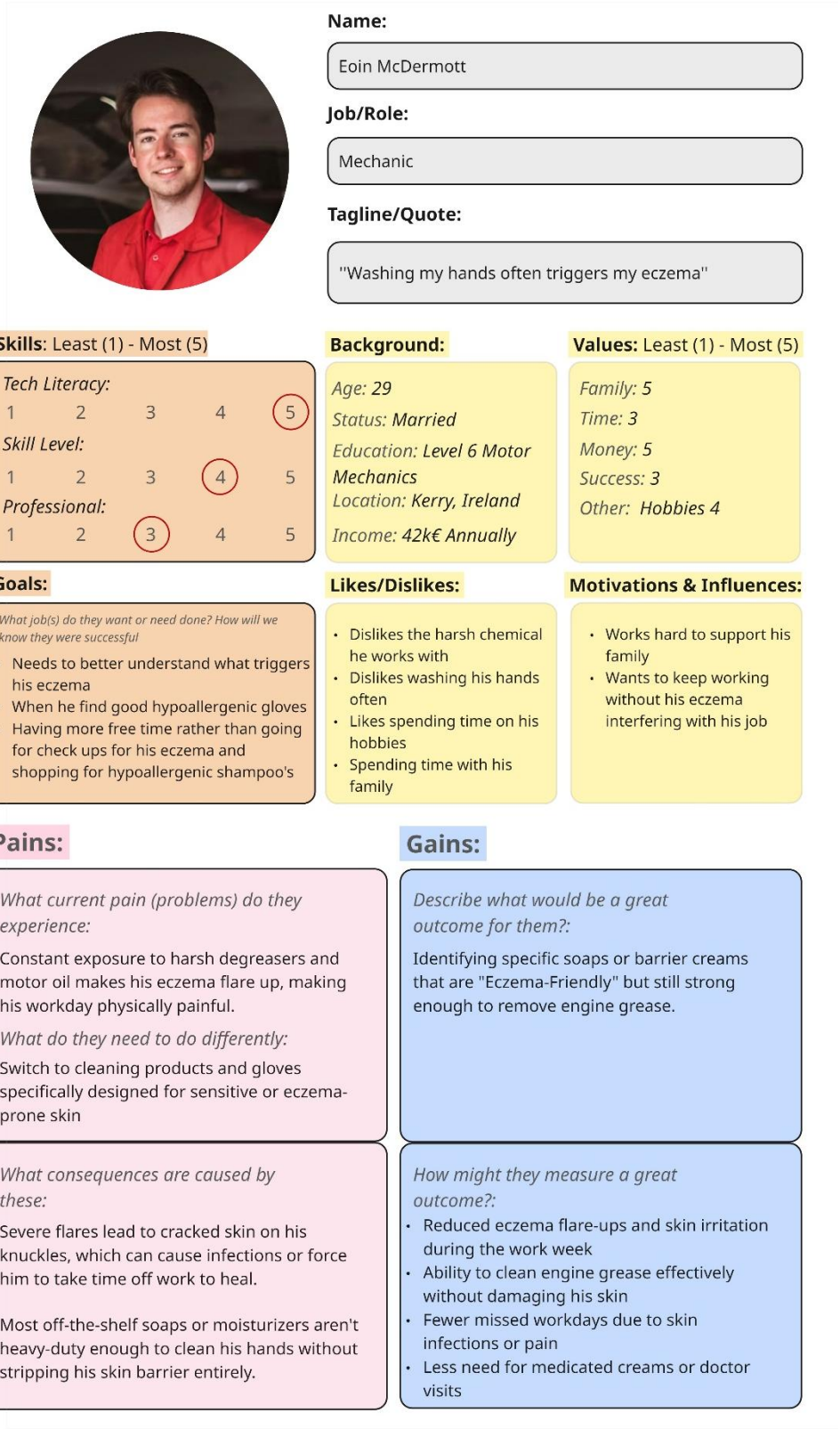


Figure 29. Persona 2, 'Eoin'.

Project: Date:

Name:

Niamh Jot

Job/Role:

Electrical Engineer

Tagline/Quote:

"If the documentation is wrong, the hardware is wrong."

Skills: Least (1) - Most (5)

Tech Literacy:
1 2 (3) 4 (5)

Skill Level:
1 2 3 (4) (5)

Professional:
1 2 3 4 (5)

Background:

Age: 34
Status: Single
Education: Master in Electrical Engineering
Location: Dublin, Ireland
Income: 72K€ Annually

Values:

Family: 3
Time: 5
Money: 4
Success: 5
Other: Innovation 5

Goals:

What job(s) do they want or need done? How will we know they were successful

- *Needs to find a way to test prototypes without waiting weeks for parts.*
- *Wants to spend more time designing and less time debugging faulty sensors.*

Likes/Dislikes:

- Likes Being organised
- Likes a good cup of coffee
- Likes having a quick night routine
- Dislikes heavy makeup products

Motivations & Influences:

- Wants to build things that actually work in the real world, not just on paper.
- Motivated by solving puzzles that others have given up on.

Pains:

What current pain (problems) do they experience:

- Spent 20 minutes reading tiny fonts on the back of a shampoo bottle only to find out it contains a known irritant anyway.

What do they need to do differently:

- Stop guessing based on brand reputation and start using data-driven scanning for every new purchase.

What consequences are caused by these:

- Frustration and "decision fatigue" while shopping, leading to just buying the same old (potentially bad) products out of habit.

Gains:

Describe what would be a great outcome for them?:

- A "Red Light/Green Light" system that flags her specific "avoid list" (like sulphates or phthalates) the moment she scans.

How might they measure a great outcome?:

- Spending less time in the aisle and more time feeling confident in what she's bringing home.

Figure 30. Persona 3, 'Niamh'

3.5 Functional, Non-Functional and System Requirements

This section outlines the requirements for the proposed system, defining the features, behaviours, and technical conditions necessary for the application to function effectively. Establishing clear requirements is an essential step in the system design process, ensuring that both user needs and technical constraints are considered throughout development. The requirements are divided into three main categories: functional requirements, which describe the core features and operations of the system; non-functional requirements, which define quality attributes such as performance, security, and usability; and system requirements, which specify the technical environment and technologies required to support the application.

(View tables on the next page)

3.5.1 Functional Requirements

Functional Requirement:	Priority
Users need to be able to make an account	**HIGH**
Users need to be able to log in	**HIGH**
Users need to be able to log out	**HIGH**
Users need to be able to update their details	**HIGH**
Users should be able to add preferences, allergies, and their conditions	**HIGH**
Users should be able to capture an image	**HIGH**
Users should be able to view scan results	**HIGH**
User should be able to view history of previous results	**HIGH**
Users should be able to make additional profiles	**MEDIUM**
Users should be able to choose whether the scan is for the main profile, the secondary profile, or both.	**MEDIUM**
Users should be able to edit previous scan results and reevaluate based on additional needs.	**MEDIUM**
The user should be able to leave feedback on the product via a Boolean question.	**MEDIUM**
Users should be able to leave feedback on the AI response	**LOW**
Users should be able to leave personalised feedback on each product.	**LOW**
User should be able to log in/Signup with Google, Apple and GitHub.	**LOW**

3.5.2 Non-Functional Requirements

Requirement:	Type:
The system should process an image and return the results within 5-10 seconds	**Performance**
The app should run on both iOS and Android.	**Performance**
The system should accept images up to 10MB in size.	**Performance**
When the user takes a picture, the app should display a guide or hit box to help capture the product from the best angle, which includes the bottle's appearance and text details.	**Usability**
The system should detect product labels and text using OCR.	**Usability**
The app should have a clean UI/UX	**Usability**
The app design should follow Nielsen Norman's 10 Design Heuristics	**Usability**
The system should allow the user to retake the image if analysis fails	**Usability**
The system should display a clear error message if the product cannot be recognised.	**Usability**
The User passwords should be hashed	**Security**
The User authentication should be done through Express	**Security**
The system shall comply with data protection laws	**Security**
Back-end should support multiple Users	**Scalability**
The backend should allow integration with external AI APIs.	**Scalability**
The system should maintain at least 99% uptime	**Reliability**

The code should be clean and commented so that future improvements can be added.	**Reliability**
The system should correctly identify the product from an image with at least 85–90% accuracy.	**Accuracy**

3.5.3 System Requirements

Requirement:	**Priority:**
The System should authenticate users using secure JWT tokens.	**HIGH**
The system should store and retrieve user and friend profiles from a PostgreSQL database.	**HIGH**
The system should identify the product brand and name from the uploaded image using an AI model.	**HIGH**
The system should generate a customised AI prompt based on the user's preferences, allergies, condition, and friends' sub-profiles.	**HIGH**
The system should allow users to capture or upload an image of a product label.	**HIGH**
The AI module should evaluate the ingredient list and return the results in a JSON format.	**HIGH**
The system should manage all data in the PostgreSQL database, including user profiles, friend profiles and scan history.	**HIGH**
User should be able to do all CRUD functionalities on their Scan History such as renaming the product name.	**HIGH**
All communication between the front-end and back-end should use HTTP encryption.	**HIGH**
The system should handle errors gracefully.	**MEDIUM**
The system should extract the product name and brand with confidence scoring.	**MEDIUM**

The system should allow the user to manually correct the detected product name if recognition fails.	**MEDIUM**
The system shall format the AI response into structured output (e.g., “Safe,” “Unsafe,” “Caution”).	**MEDIUM**
The system shall log key activities for debugging and monitoring performance.	**LOW**
The system should cache previously retrieved product ingredient data to improve performance	**LOW**
The system should allow the user to manually enter or edit ingredients if the product cannot be recognised.	**LOW**

4. System Architecture

4.1 Introduction

This section covers the key design decisions for both the client and server. On the backend, we emphasise entity relationships, the design patterns employed with our technologies, database design strategies, and the integration of prompt engineering. These choices directly relate to the client side, where we also utilise similar design patterns but focus more on UI design, including styling, layout choices, prototypes, and wireframes.

4.2 Application Design

4.2.1 Process Design (ERD)

The entity-relationship diagram is a type of flowchart that illustrates how different entities are related. It shows entities (tables) in the database and represents the relationships (connections) using Crow's Foot notation. Crow's Foot notations include one-to-one, one-to-many, and many-to-many relationships, represented by a three-pronged line.

In many-to-many relationships, we cannot store a foreign key ID within the original entity because this relationship permits multiple connections between the two entities. Instead, a third table is created to hold the IDs from both entities, along with its own primary ID.

Our ERD contains 8 tables and 3 pivot tables, with 3 many-to-many and 4 one-to-many relationships.

Below are the preliminary ERDs for the system.

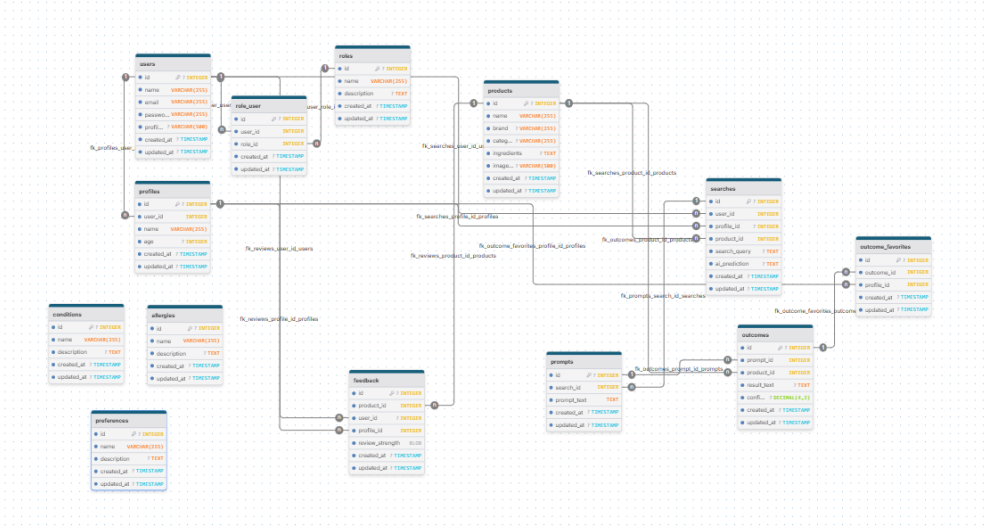


Figure 31. First ERD draft.

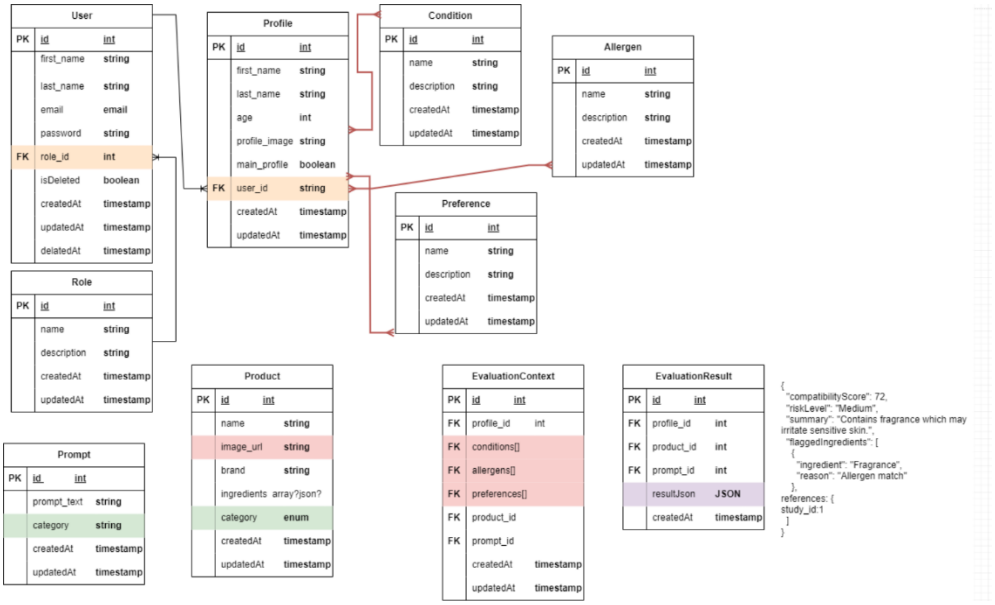


Figure 32. Second ERD draft.

This ERD is the final one.

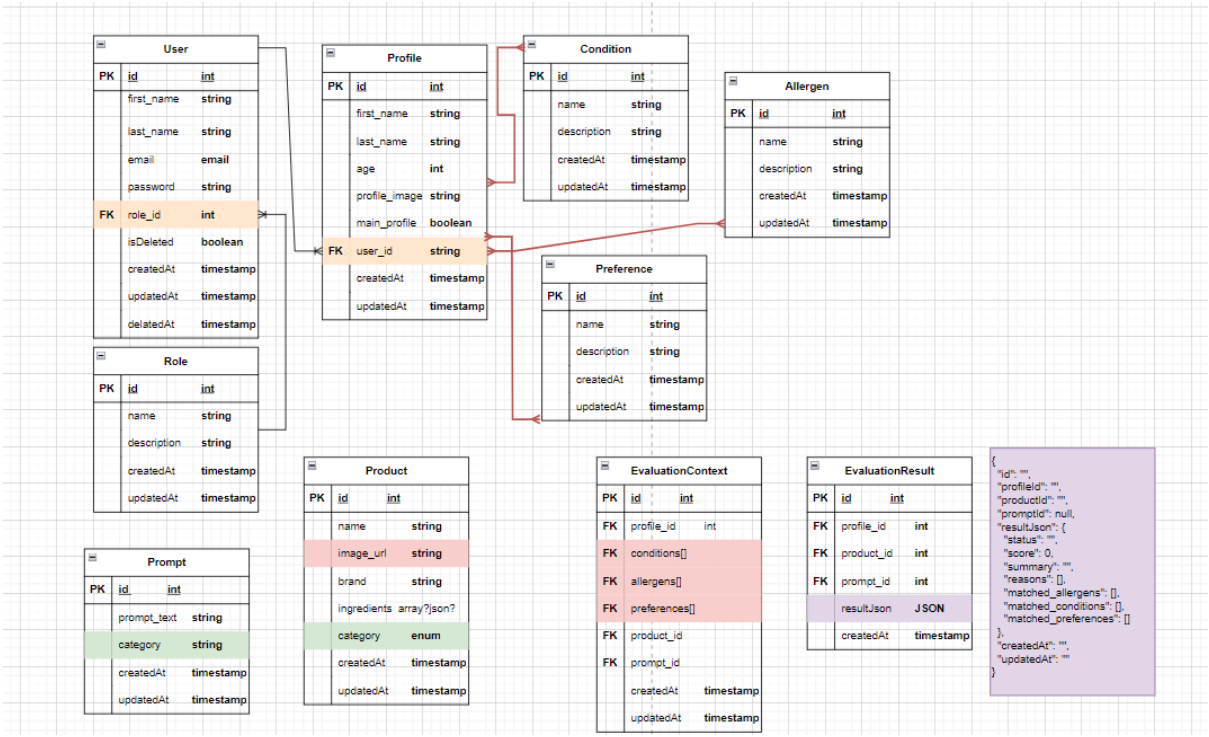


Figure 33. Final ERD.

4.2.2 Design Patterns (Layered Architecture)

TypeScript was chosen as the primary language for the server. It is a superset of JavaScript that introduces static typing through a gradual type system, improving code structure and maintainability.

The key distinction between JavaScript and TypeScript lies in type checking. JavaScript is dynamically typed, with types evaluated at runtime, whereas TypeScript performs compile-time type checking. This allows errors to be identified earlier in development, reducing the likelihood of runtime failures.

In JavaScript, the request flow is relatively simple, enabling rapid development but increasing the risk of runtime errors. A typical flow involves a user making an HTTP request, middleware processing and validating data, routing, directing the request, and controllers delegating logic to services that interact with the database. The server then returns a response with an appropriate status code.

TypeScript introduces additional abstraction layers, resulting in a more structured but slightly larger architecture. This layered approach improves reliability, scalability, and maintainability. Using Prisma ORM, the application follows a layered architecture where each component has a defined responsibility: the schema defines the database structure, DTOs define the shape of transferred data, validators enforce input constraints, repositories handle database operations, services contain business logic, controllers manage requests and responses, and routes define API endpoints.

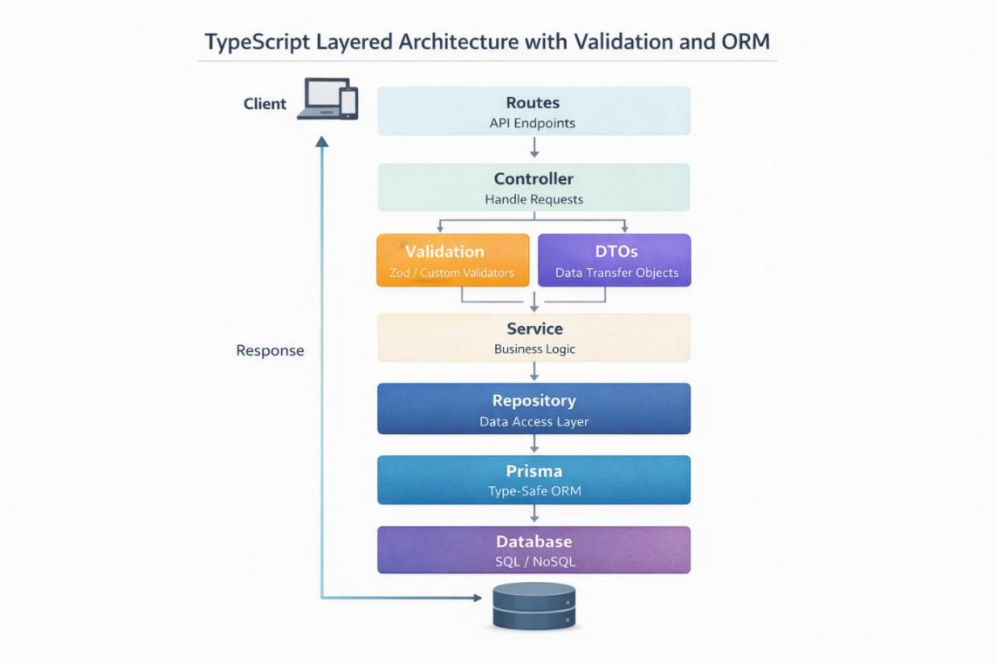


Figure 34. TypeScript Layered Architecture Diagram.

4.2.3 Database Design (JSON Structure)

As mentioned earlier, the current database consists of 11 tables, three of which are pivot tables. Certain tables contain attributes stored in JSON-based data structures. As the application is centred around AI, this approach enables more flexible handling of dynamic and semi-structured data, improving how information is interpreted and processed within the system.

Focusing on the JSON aspect, we store the results data based on the user's initial submission information. In the 'Evaluation Context' entity (available below), it takes the user's profile ID, product ID, and the matching prompt ID.

User

Attribute Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
first_name	string	100	none	none	first name of user
last_name	string	100	none	none	last name of user
email	string	100	none	none	email of user
password	string	none	none	none	password for user's account
role_id	string	none	foreign key	foreign key	foreign key for user role entity
isDeleted	Boolean	true/false	none	none	attribute for soft deleted user
createdAt	string ISO	none	none	none	timestamp of when the user was created
updatedAt	string ISO	none	none	none	timestamp of when the user was updated
deletedAt	string ISO	none	none	none	timestamp of when the user was soft deleted

Figure 36. Database Table 'User'.

Role

Attribute Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
name	string	100	none	none	name of role
description	string	100	none	none	brief description of role
createdAt	string ISO	none	none	none	timestamp of when the user was created
updatedAt	string ISO	none	none	none	timestamp of when the user was updated
deletedAt	string ISO	none	none	none	timestamp of when the user was soft deleted

Figure 35. Database Table 'Role'

Profile

Attribute Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
first_name	string	100	none	none	first name of users profile
last_name	string	100	none	none	last name of users profile
age	integer	100	none	none	age of the user's profile
profile_image	string	none	none	none	profile image for users profile
main_profile	Boolean	true/false	none	none	used to indicate if it's the user main profile
user_id	string	none	foreign key	none	used to attach profile to a user
createdAt	string ISO	none	none	none	timestamp of when the profile was created
updatedAt	string ISO	none	none	none	timestamp of when the profile was updated

Figure 37. Database Table 'Profile'

Product

Attribute Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
name	string	100	none	none	name of the product
image_url	string	none	none	none	image of the product
brand	string	100	none	none	product brand name
ingredients	json	null	none	none	ingredients of product
category	enum	{showed below}	none	none	used to indicate if it's the user main profile
user_id	string	none	foreign key	none	used to attach profile to a user
createdAt	string ISO	none	none	none	timestamp of when the product was created
updatedAt	string ISO	none	none	none	timestamp of when the product was updated

Category Enum

Shampoo
Deodorant & Antiperspirant
Cleanser
Scrub
Conditioner
Body Wash
Moisturiser
Serum
other

Figure 38. Database Table 'Product'

Condition

Attribute Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
name	string	100	none	none	name of condition
description	string	100	none	none	brief description of condition
createdAt	string ISO	none	none	none	timestamp of when the condition was created
updatedAt	string ISO	none	none	none	timestamp of when the condition was updated

Pivot Table

condition_profile

Attribute Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for entity
condition_id	string	none	foreign key	foreign key	foreign key linking to a condition
profile_id	string	none	foreign key	foreign key	foreign key linking to a profile
createdAt	string ISO	none	none	none	timestamp of when the entity was created
updatedAt	string ISO	none	none	none	timestamp of when the entity was updated

Figure 39. Database Table 'Condition' with Pivot table

Preference

Attribute_Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
name	string	100	none	none	name of Preference
description	string	100	none	none	brief description of Preference
createdAt	string ISO	none	none	none	timestamp of when the Preference was created
updatedAt	string ISO	none	none	none	timestamp of when the Preference was updated

preference_profile

Attribute_Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for entity
preference_id	string	none	foreign key	foreign key	foreign key linking to a preference
profile_id	string	none	foreign key	foreign key	foreign key linking to a profile
createdAt	string ISO	none	none	none	timestamp of when the entity was created
updatedAt	string ISO	none	none	none	timestamp of when the entity was updated

Figure 40. Database Table 'Preference' with Pivot Table.

Allergen

Attribute_Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
name	string	100	none	none	name of Allergen
description	string	100	none	none	brief description of Allergen
createdAt	string ISO	none	none	none	timestamp of when the Allergen was created
updatedAt	string ISO	none	none	none	timestamp of when the Allergen was updated

allergen_profile

Attribute_Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for entity
allergen_id	string	none	foreign key	foreign key	foreign key linking to a allergen
profile_id	string	none	foreign key	foreign key	foreign key linking to a profile
createdAt	string ISO	none	none	none	timestamp of when the entity was created
updatedAt	string ISO	none	none	none	timestamp of when the entity was updated

Figure 41. Database Table 'Allergen' with Pivot Table

Evaluation Context

Attribute_Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
profile_id	string	none	foreign key	foreign key	foreign key to profile
product_id	string	none	foreign key	foreign key	foreign key to product
prompt_id	string	none	foreign key	foreign key	foreign key to prompt
resultJSON	JSON	none	none	none	result stored in json
createdAt	string ISO	none	none	none	timestamp of when the entity was created
updatedAt	string ISO	none	none	none	timestamp of when the entity was updated
deletedAt	string ISO	none	none	none	timestamp of when the entity was soft deleted

Figure 42. Database Table 'Evaluation Context'.

Prompt

Attribute Name	Data Type	Field Length	Constraints	Relation	Description
id	string UUID	none	primary key	primary key	primary key for the entity
prompt_text	string	100	none	none	prompt text based on product category
category	enum	{showed below}	none	none	used to indicate if it's the user main profile
createdAt	string ISO	none	none	none	timestamp of when the user was created
updatedAt	string ISO	none	none	none	timestamp of when the user was updated

Category Enum

Shampoo
Deodorant & Antiperspirant
Cleanser
Scrub
Conditioner
Body Wash
Moisturiser
Serum
Other

Figure 43. Database Table 'Prompt' with category ENUM.

4.2.4 Prompt Engineering

Prompt Engineering is a crucial component of this project, as LLMs are used to process user data and generate evaluation results. It enables control over LLM outputs, ensuring accuracy and consistency, particularly in computer vision tasks where ingredient lists are extracted from user-captured product images.

In this workflow, Gemini first performs surface-level analysis of an image, identifying visual features such as packaging, colour, text, and branding. It then predicts the product and retrieves relevant data, including ingredients. However, without structured prompts, this process can be ambiguous, as the model may rely on assumptions or incomplete data. Prompt engineering addresses this by introducing constraints, such as specifying the product region (e.g. EU-based), ensuring more relevant and accurate outputs.

Another key challenge is output formatting. LLM responses can vary between plain text and JSON-like structures, which can disrupt data processing in larger systems. To solve this, prompts are designed to enforce a consistent JSON schema, specifying required variables and response structure. This ensures reliable integration from request to result.

A central feature of this project is the use of a prompt entity, allowing administrators to define and manage prompts for specific product categories, such as shampoos or facial cleansers. This improves evaluation accuracy, as different product types require different criteria. A universal prompt could lead to incorrect classifications, for example, identifying a shampoo as safe for facial use despite differences in skin sensitivity and pH.

Although prompt engineering requires initial refinement, it is essential for achieving accurate, structured, and context-aware AI outputs, making it a key element of the system’s design.

4.2.5 Retrieve Augment Generate (RAG)

One of our application's main premises is to provide safe, controlled AI responses. And one of the best ways is to use the Retrieve-Augment-Generate (RAG) approach. RAG is an AI approach in which a model first searches for relevant external information (such as documents or databases) and then uses it to generate a more accurate response. It helps reduce hallucinations by grounding the AI's answers in real, retrieved data rather than relying solely on its training.

For our application, this is crucial as it allows the AI to support its claims with credible, verifiable sources. This was done through research on various medical papers, mainly discussing ingredients analysis, factors influencing skin repair and additional support for dermatological conditions. Unfortunately, on the free tier of our application, we are limited to using Gemini 2.5 Flash, which performs slowly when handling large numbers of PDFs, so we restricted the dataset to 12 papers (list of papers is available in Appendix [15]).

RAG works by breaking text into smaller chunks and storing them as vectors. In this project, Pinecone is used as the primary vector database to store and retrieve these embeddings.

Overall, the implementation of the RAG approach significantly improved the reliability and relevance of AI-generated responses within the application. By grounding outputs in verified external sources, the system was able to provide more trustworthy and context-aware information to users. However, limitations in processing capacity and dataset size highlight potential areas for future improvement, particularly in expanding the knowledge base and optimising performance. Despite these constraints, the use of RAG provided a strong foundation for delivering safe and controlled AI interactions.

4.2.6 Technologies

This project is built using a range of technologies across both server-side and client-side development. Each layer of the system has different responsibilities, requiring appropriate tools to support functionality, performance, and user interaction. The server-side focuses on handling data processing, storage, and application logic, while the client-side is responsible for delivering an interactive and responsive mobile UX. The selected technologies reflect these differing requirements and contribute to the system's overall structure and efficiency. A summary of the key technologies used is outlined in the table below.

-Server

Prisma	An ORM is used to interact with the database in a structured, safe way.
Zod	A schema validation library used to ensure data is correct and safe.
Express	Used for securely hashing passwords for authentication.
Docker	Used to containerise the application for consistent development and deployment.
JWT	Used for handling user authentication and secure session management.
AWS S3	Cloud storage used for storing and retrieving files such as images.
Node Express TS	Server end framework that handles server logic, APIs, and backend functionality.
Jest.js	A testing framework used to test server-side code.
PostgreSQL	A relational database used to store and manage application data.
Gemini API	Used to analyse images of products and generate insights based on user preferences, such as dietary conditions and allergens.
Pinecone	Used to store vectors from tokenisation of research papers for RAG

-Client

React Native	Framework used to build the mobile application.
@react-native libraries	Used for camera’s, gradients, and local storage
Expo	A toolchain that simplifies React Native development and access to native features.

Expo Camera	
Express	Used for securely hashing passwords for authentication.
Zod	Used on the client-side for validating user input.
Bcrypt	Used for securely hashing passwords for authentication.
Gluestack UI	A UI component library for building consistent interfaces.
Expo Camera	Enables camera functionality within the mobile app.
Axios	Http Client for all incoming and outgoing requests.
Moti	Provides Tailwind-like animation
Client for Admin	
React Vite	Used to build the admin panel, offering a fast development environment with efficient hot module replacement and optimised builds.
Tailwind	A utility-first CSS framework used to rapidly style the admin interface with a consistent and responsive design.
React Router	Handles navigation within the admin panel, enabling routing between pages such as dashboards, lists, and edit views.
React DOM	Manages navigation within the admin panel, allowing routing between different pages such as dashboards and edit views.
Framer Motion	Provides animation capabilities within the admin panel, enhancing user experience through smooth transitions and interactions.
Lucide React	A lightweight icon library used to provide consistent and scalable icons across the admin interface.

4.3 User Interface

In this section, we'll discuss the inspiration behind the mobile application's overall design. Using Nielsen's 10 usability heuristics, principles of digital interface design are based on the problems users face when interacting with digital systems. Issues such as status visibility, user control and freedom, error prevention, and consistency are among the principles we will discuss in detail.

4.3.1 Style Guides and Design Choices

The application was designed to be simple, elegant, and easy to use. Its design draws on competitor platforms and independent research to address common usability issues.
A light pastel colour scheme and clear typography were selected to create a calm and approachable UX. The design is guided by established principles, including Nielsen's 10 usability heuristics such as consistency, visibility, and user control.

These principles were applied across the mobile application and the supporting admin panel, ensuring a consistent, intuitive experience for end users and administrators.
The following section outlines how these styles and guidelines were applied, including decisions around colour, typography, and layout.

-**Colour Styles**

Initially, we used a colour scheme generator to create light colours that complement her, as we needed main colours such as reds, greens, and other primary buttons for user control and status visibility.

Figure 44. Colour Schema Draft One.

Figure 45. Colour Scheme Draft Two.

Figure 46. Final Colour Scheme

-Design Elements

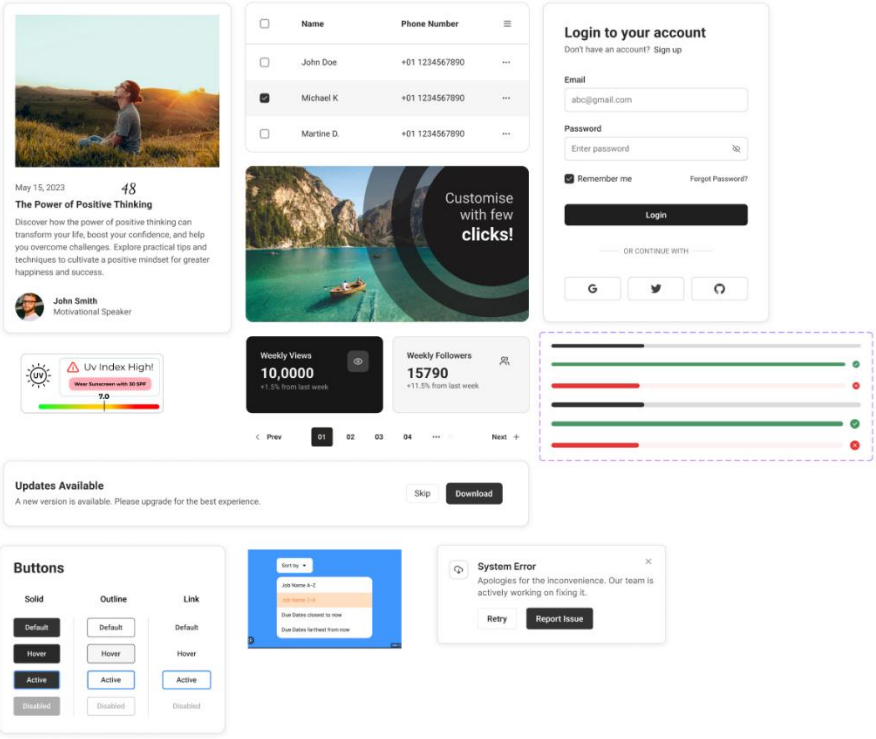


Figure 48. GlueStack Elements

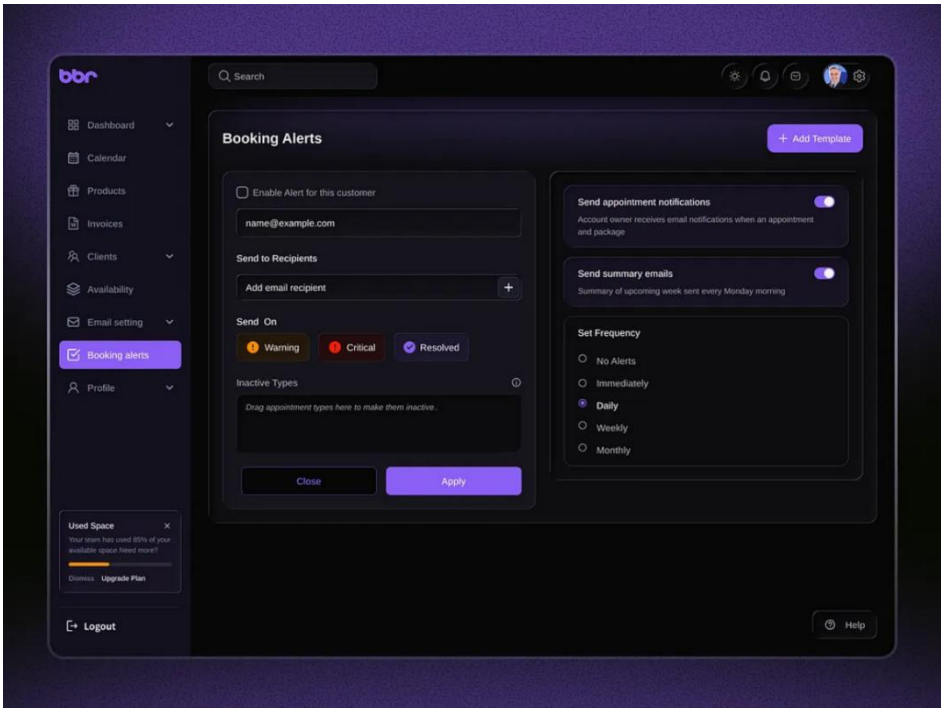


Figure 49. Admin Panel Example

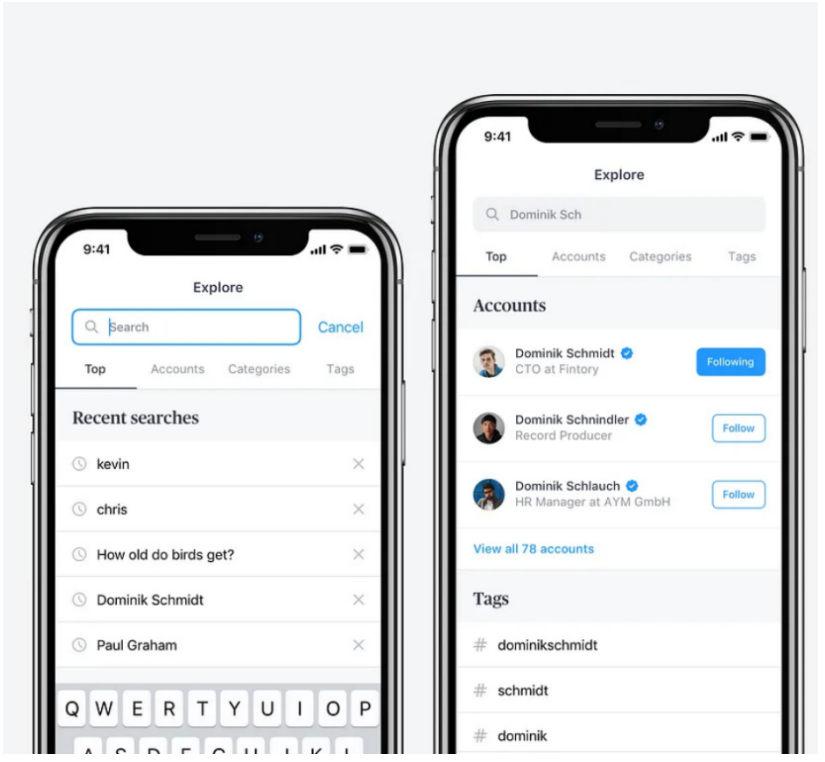


Figure 50. Search Results

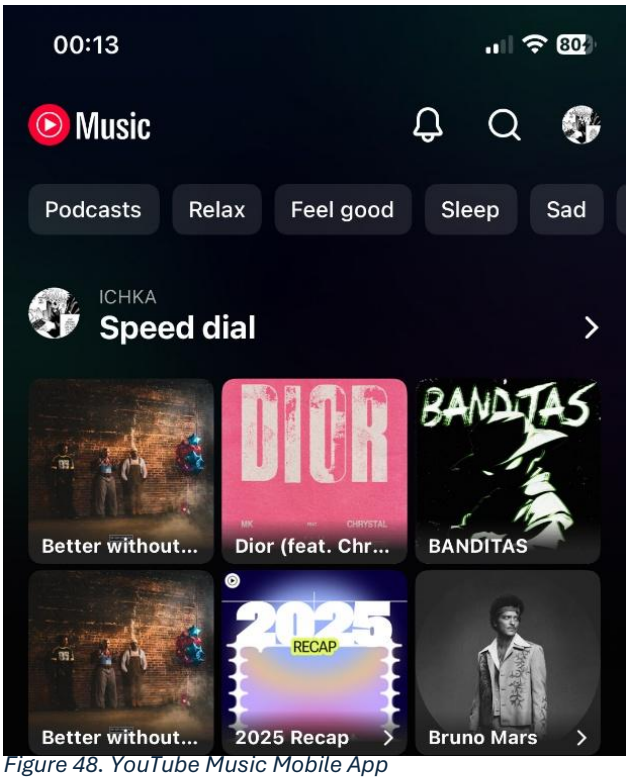


Figure 48. YouTube Music Mobile App

More can be viewed on the Miro Board Appendix [5] and the Figma board Appendix [6]

4.3.2 Flow Diagrams

To better understand the system's behaviour, a flow diagram was developed to map out the application's processes. It visually represents the interactions among components and the flow of data throughout the system. This allows for clearer analysis of how each feature operates within the overall architecture.

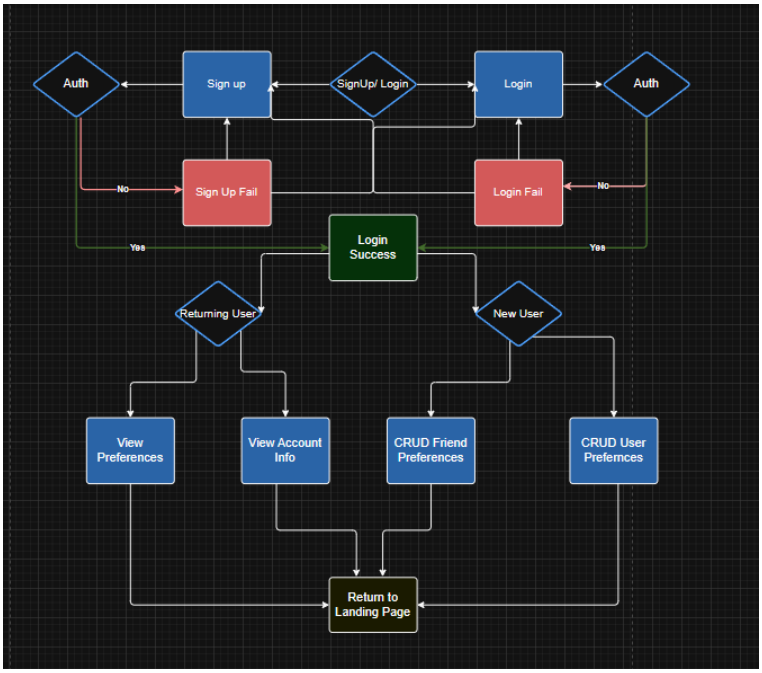


Figure 51. Auth Flow

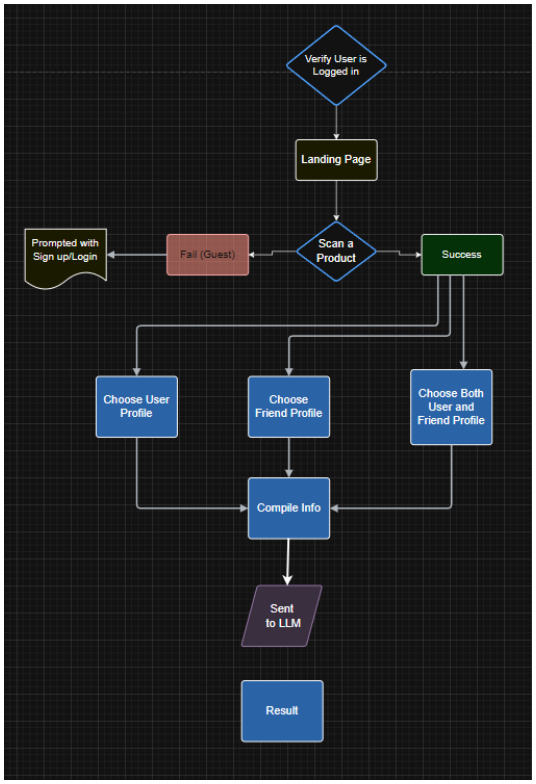


Figure 52. Evaluation Flow

Figure 53. Ingredient Extraction Flow and SerpAPI image retrieval

4.3.3 Paper Prototypes

Paper prototyping was used during the early stages of design to explore and visualise the structure and functionality of the application before implementation. This approach allowed for rapid iteration of ideas, enabling key interface elements, user flows, and interactions to be planned and refined at a low cost. By sketching initial layouts and workflows, potential usability issues could be identified early, supporting a more user-centred design process. The following section presents the paper prototypes developed for the core features of the application.

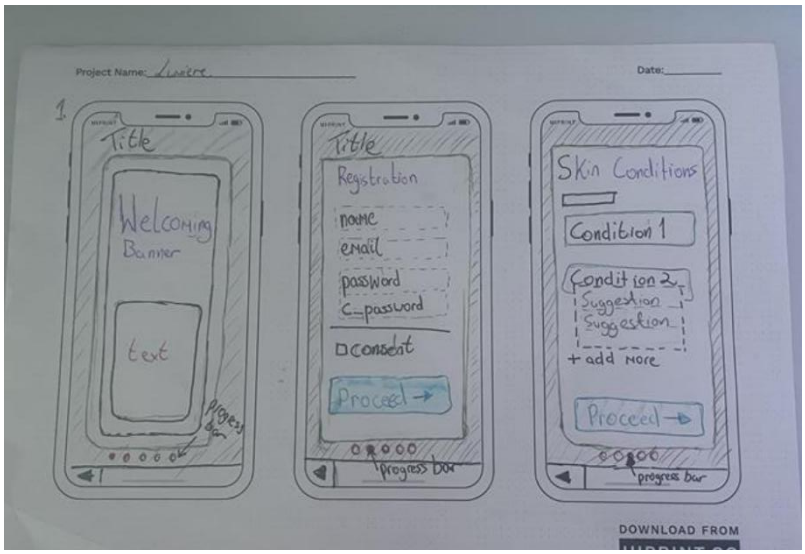


Figure 54. Low fidelity paper prototype of Welcome and register screens

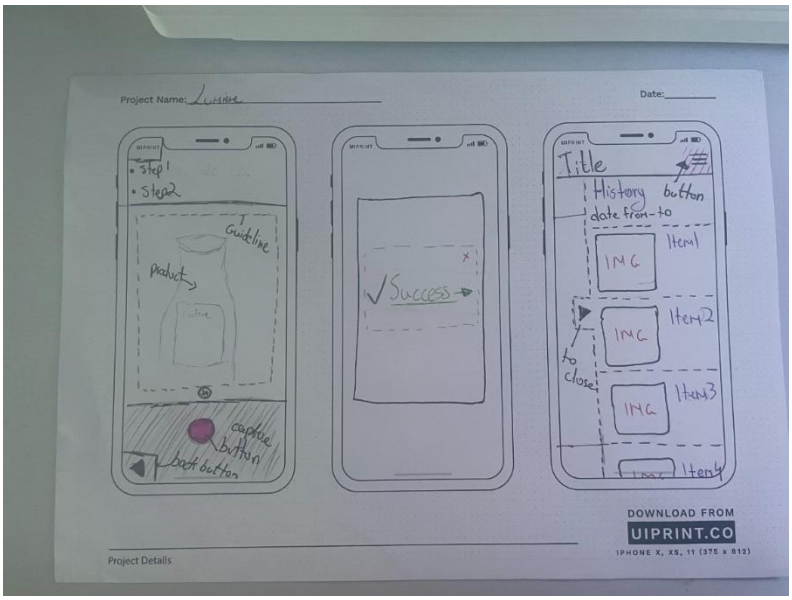


Figure 55. Low fidelity paper prototype of scanning processing

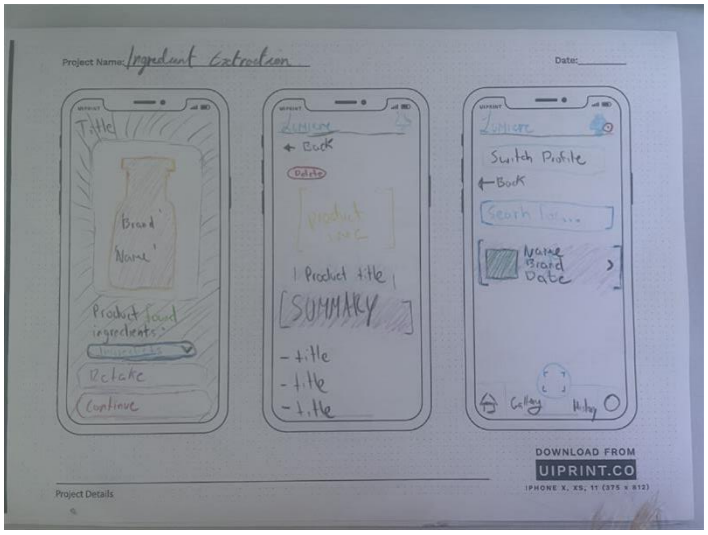


Figure 56. High fidelity of Scanning and result, alongside history.

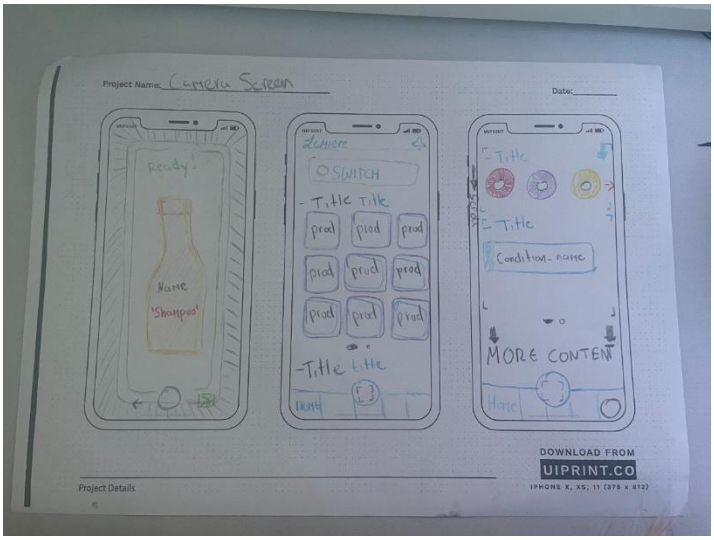


Figure 57. High fidelity of scanning and landing screens

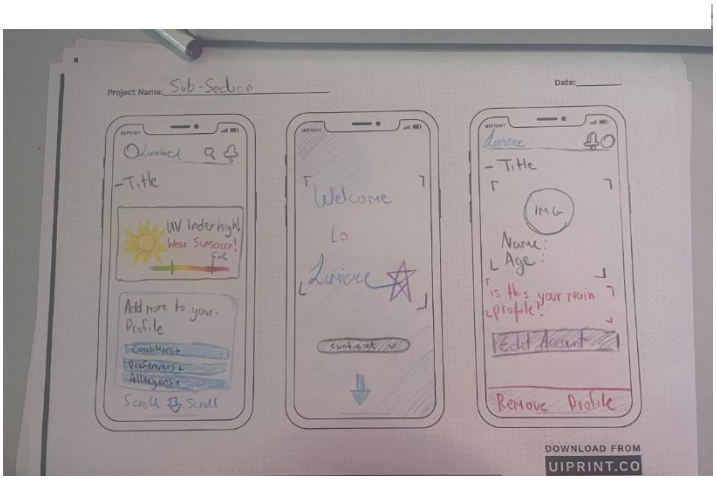


Figure 58. High fidelity of landing, welcome and profile setting screens.

4.3.4 Wireframes

Due to the large number of wireframes developed throughout the design process, only a selection of low-, mid-, and high-fidelity examples are presented in this section. These were chosen to illustrate the progression of the interface from initial concepts to a fully refined design.

Low-fidelity wireframes focus on basic layout and structure, allowing for rapid exploration of ideas. Mid-fidelity wireframes introduce greater detail, including clearer content placement and interaction elements. High-fidelity wireframes represent the finalised design, showcasing visual styling, refined UI components, and overall UX. All wireframes are available on the Figma board, Appendix [6].

-Low Fidelity

Low-fidelity wireframes were used to explore initial layout and structure, focusing on core functionality and user flow without detailed design elements.

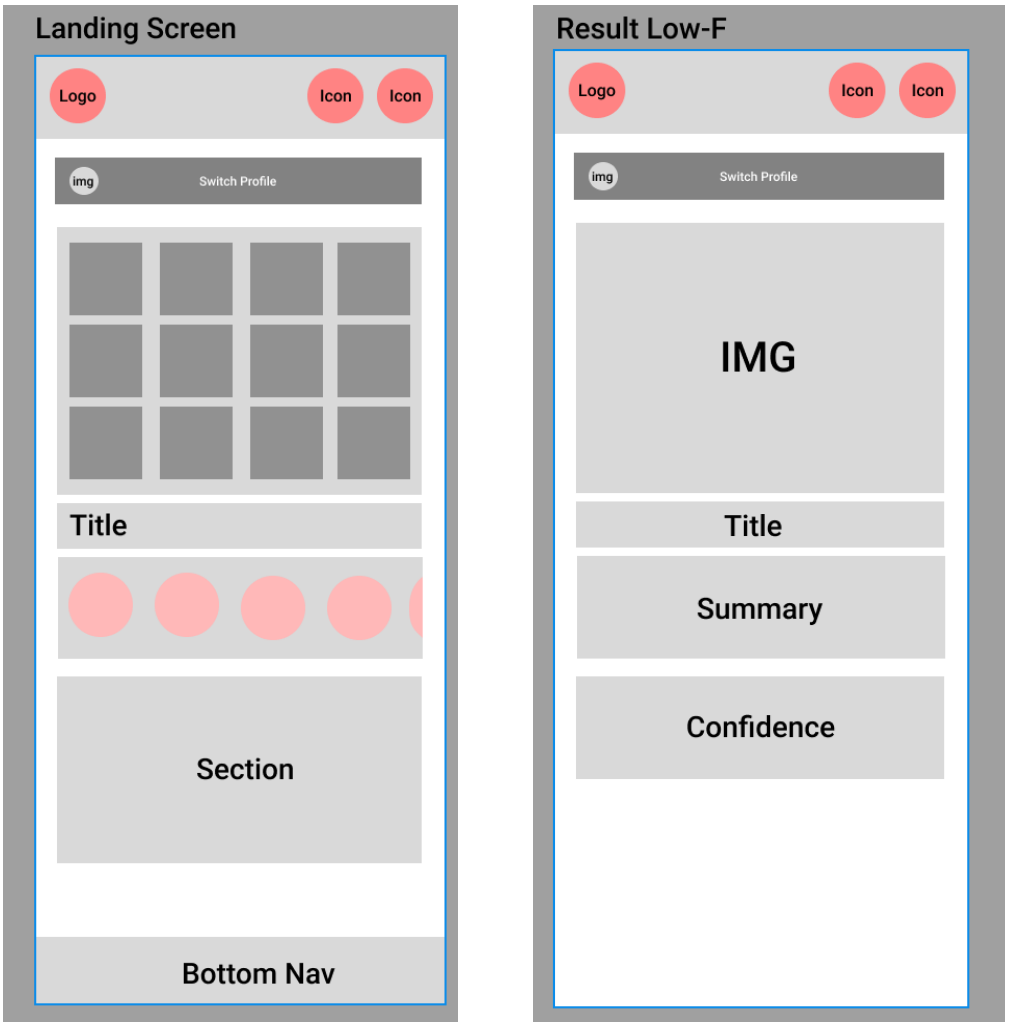


Figure 59. Landing Screen Low Fidelity

- **Mid Fidelity**

Mid-fidelity wireframes introduced more detail, including clearer content placement and basic interaction elements, helping to refine the overall interface.

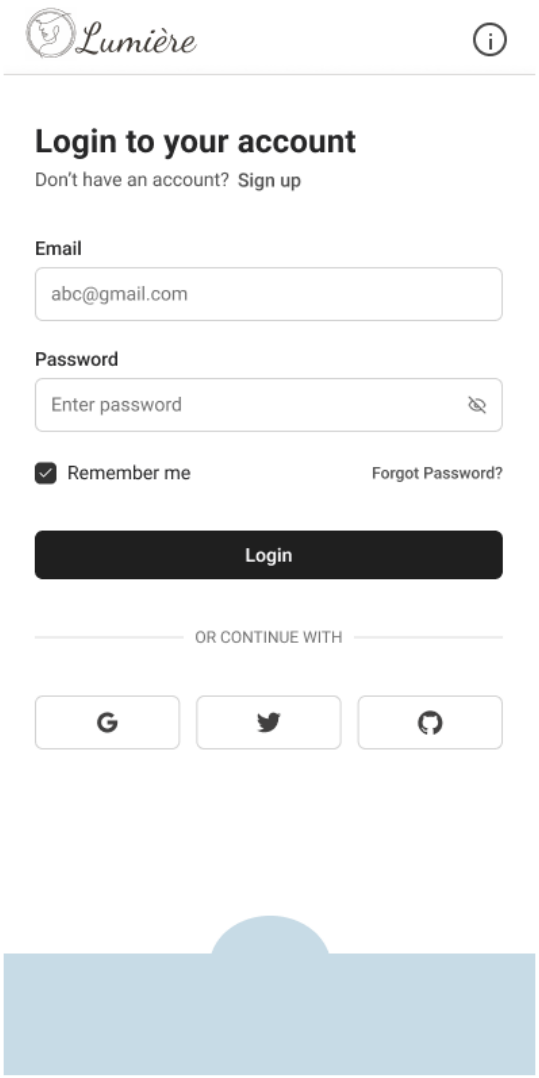


Figure 61. Medium Fidelity Login Screen

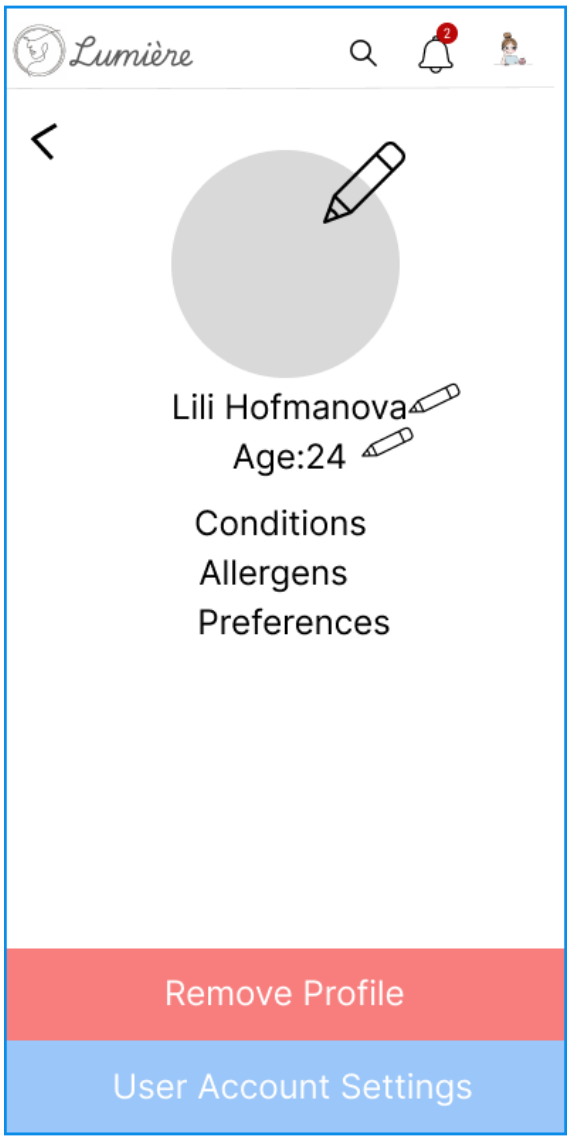


Figure 62. Medium Fidelity Edit Profile Screen

- **High Fidelity**
  High-fidelity wireframes represent the finalised design, showcasing visual styling, refined components, and the intended UX.

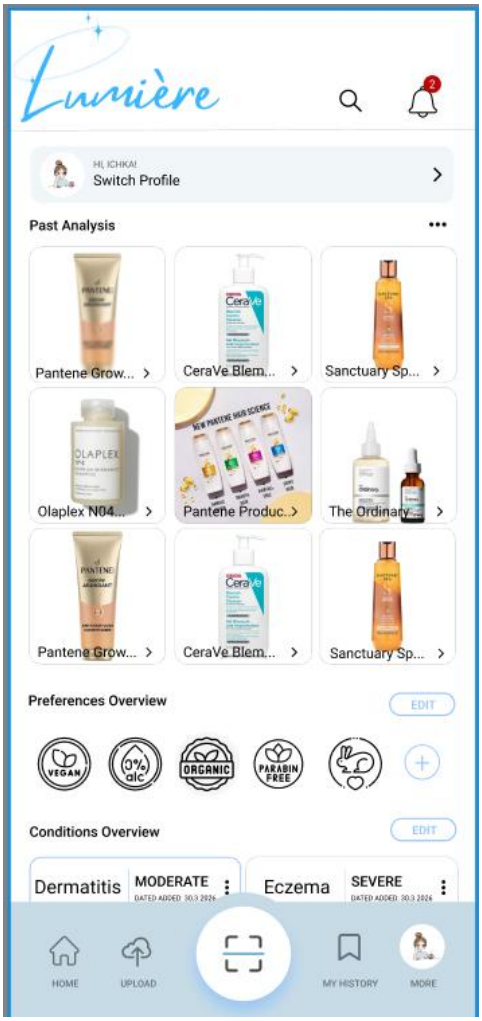


Figure 63. High Fidelity Landing Screen

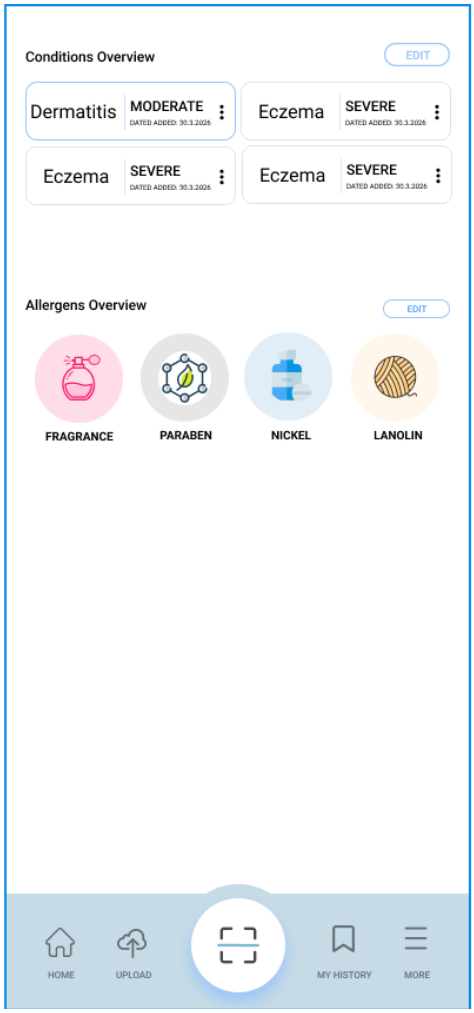


Figure 64. High Fidelity continued Landing screen.

Figure 65. High Fidelity 'All Allergens'

5. Implementation

5.1 Introduction

This section outlines the project's implementation phase, including the development environment, database setup, third-party APIs, and chosen programming languages, along with the reasoning behind these decisions. It also examines code implementation, key development challenges, and solutions, with a focus on mobile development and its differences from web platforms.

Finally, it covers testing and validation methods, version control practices, performance considerations, limitations, and potential improvements for future iterations.

5.2 Development Environment

- **Visual Studio Code** used as the primary IDE, with extensions such as **ESLint** for catching errors
- **Node.js** used as the runtime environment
- **npm** used for package management, such as S3, Google-Generative AI, and many more.
- **Expo** used to build and run the mobile application.
- **Expo Go** is used for testing on a physical device, mainly tested on IOS.
- **GitHub Copilot** used to assist with code generation, debugging and explanation of needed components and suggestions on how the project should be developed.
- **Postman** is used for testing API requests and integration testing.
- **Gemini 2.5 Flash** was used for ingredient analysis, ingredient extraction and the main model used for RAG.

5.2.1 Database Setup

- PostgreSQL database is used to store and manage user data (e.g. account information, skin conditions, and product analysis results)
- Prisma ORM is implemented for type-safe database queries and schema modelling
- The database is hosted within a Docker container to ensure consistent development and simplified environment configuration
- The backend is hosted on Render, enabling reliable deployment and scalability
- The mobile application communicates with the backend through API requests
- No persistent data is stored locally on the user's device

5.2.2 Languages Used

This project uses TypeScript, JSON, and PostgreSQL to support development and data management.

- TypeScript: Used across frontend and backend for type safety, easier debugging, and maintainability.
- JSON: Handles data formatting for API requests/responses and analysis results.
- PostgreSQL + Prisma: PostgreSQL stores structured data, while Prisma enables type-safe and efficient database access.

5.2.3 Frameworks and Libraries

The frameworks and libraries used in this project were essential for building a scalable and maintainable application, improving development speed, performance, and core functionality.

The backend was developed using a TypeScript-based Node.js and Express environment with Prisma ORM for database access, while the frontend was built with React Native in TypeScript using Expo.

Below is a list of the main runtime libraries used, along with a brief explanation of each.

Server Run-Time Libraries	
aws-sdk/client-s3	Used for storing files such as images, primarily in profile and analysis
google/generative-ai	Used for analysis entities and ingredients fetching based on user image input.
prisma/adapter-pg	Used to connect Prisma ORM with a PostgreSQL database for efficient data handling.
prisma/client	Used as an ORM to interact with the database using type-safe queries and models.
bcrypt	Used to securely hash user passwords for authentication and data protection.
Cors	Used to enable secure cross-origin requests between the frontend and backend.
Dotenv	Used to manage environment variables such as API keys and configuration settings securely.

Zod	Used for validating and enforcing data schemas to ensure input correctness.
multer	Used to handle file uploads, particularly for processing images sent by users.
jsonwebtoken	Used to implement secure user authentication through token-based authentication.

Client Run-Time Libraries	
expo	Used to make building and running the app easier across devices.
react	Used to build the app's interface using reusable components
axios	Used to send requests between the app and the backend.
@react-native-async-storage/async-storage	Used to store small bits of data locally, like user session info.
expo-image-picker	Used to let users upload or take photos for product scanning.
zod	Used to validate data and make sure inputs are correct.
gluestack-ui/themed	A component library used to build consistent and styled UI elements in the mobile application.
Admin Client Run-Time	
react-router-dom	Used to handle navigation between different admin pages such as dashboards, lists, and edit views.
framer-motion	Provides smooth animations and transitions within the admin interface to improve user experience.
lucide-react	Used to display consistent and scalable icons across the admin panel interface.
tailwindcss	A utility-first CSS framework used to design responsive and consistent layouts in the admin panel.

5.3 Challenges & Solutions

The following section outlines the challenges encountered during the development of this application, presented in a structured format detailing each error, the proposed solution, and the final fix implemented. While AI agents were utilised during development for debugging and code generation, their use was primarily limited to resolving client-side issues. In some cases, errors manifested differently than expected, and AI was used to assist in both identifying and explaining these inconsistencies. Additionally, we will only discuss major errors, specifically those that prevented the application from functioning, rather than all issues that occurred. A collection of issues that occurred can be viewed in the appendix [4].

5.3.1 Server

Issue 1: Client–Server Connection Failure

- **Error:**
  The client failed to connect to the server despite correct implementation, with no clear code-related errors.
- **Proposed Fix**:
  Assumed the local server was not properly initialised or running.
- **Actual Fix:**
  Consulted Copilot about the issue, which stemmed from the network connection. The testing device was on a different network from the server.

```
const api = axios.create({
  baseURL: API_URL,
  timeout: 60000, // Render free instances can take longer on cold start
  headers: {
    'Content-Type': 'application/json',
  },
});
const envApiUrl = process.env.EXPO_PUBLIC_API_URL?.trim();
const API_URL = normalizeAndroidLocalhost(
  ensureApiBasePath(envApiUrl || buildDefaultApiUrl()),
);
```

Issue 2: API Rate Limits During Chunking

- **Error:**
  API rate limits were reached during data chunking, causing delays and interruptions in processing.
- **Proposed Fix:**
  Switching to a different LLM to handle tokenisation and chunking.
- **Actual Fix:**
  The issue persisted across models due to similar limitations. A practical solution was implemented by limiting the system to 12 RAG research papers to stay within rate constraints.

Issue 3: Slow API Response Times

- **Error:**
  The selected model produced slow response times, affecting usability. This appeared after utilising RAG.
- **Proposed Fix:**
  Attempted optimisations in request handling and prompt structure.
- **Actual Fix:**
  Switched to a different model that was 70% faster than the previously used one and cheaper per token/request usage.

Challenge 4: Difficulty Sourcing Suitable APIs

- **Error:**
  Available APIs for seeding data either lacked sufficient detail or returned excessive irrelevant data. Most needed to be scraped or returned null on search results.
- **Proposed Fix:**
  Testing multiple public APIs to find a suitable balance.
- **Actual Fix:**
  Implemented CRUD for each analysis entity and have admin roles control those objects.

Issue 5: Deployment Failure (Prisma)

- **Error:**
  Deployment failed, likely due to Prisma configuration issues; the build command didn't run.
- **Proposed Fix:**
  Reviewed environment variables and database configuration.

- **Actual Fix:**
  Adjustments to the Prisma schema and environment setup resolved the issue, allowing successful deployment.

Issue 6: Re-evaluate

- **Error:**
  When the re-evaluation ran, it appeared not to pull any ingredients or profile current analysis data but did re-evaluate the analysis.
- **Proposed Fix:**
  Remake the re-evaluate function instead of using patch.
- **Actual Fix:**
  The function was working correctly, but didn't show the ingredients or updated profile information; a separate re-evaluate function was created.

```
async reevaluateEvaluationContext(id: string):
Promise<EvaluationContextResponseDto> {
    const existing = (await prismaRuntime.evaluationContext.findUnique({
      where: { id },
      select: { id: true, profileId: true, productId: true, promptId: true },
    })) as { id: string; profileId: string; productId: string; promptId?: string | null } | null;

    if (!existing) {
      throw new HttpError(NOT_FOUND, `Evaluation context with id '${id}' not found`);
    }

    const { resultJson } = await this.buildEvaluationResult({
      profileId: existing.profileId,
      productId: existing.productId,
      promptId: existing.promptId ?? undefined,
    });

    const updated = await prismaRuntime.evaluationContext.update({
      where: { id },
      data: {
        resultJson: resultJson as Prisma.InputJsonValue,
      },
    });

    return this.toResponseDto(updated);
  }
```

5.3.2 Client

Issue 7: State Not Updating (isComplete)

- **Error:**
  The isComplete state did not turn true after profile completion.
- **Proposed Fix:**
  Suspected backend issue
- **Actual Fix:**

  The issue was caused by relying on outdated or incomplete client-side state immediately after authentication. The solution involved asynchronously fetching the latest profile data using getMyProfile() and deriving the isComplete value directly from the returned data. This ensured that navigation decisions were based on up-to-date backend state.

```
const completeLoginFlow = async (response: AuthResponse) => {
  let shouldGoToAnalyse = false;
  let profileIdForEdit: string | undefined =
    response.user.profile_id ?? undefined;

  if (profileIdForEdit) {
    try {
      const profiles = await profileService.getMyProfile();
      const activeProfile =
        profiles.find((item) => item.main_profile) ?? profiles[0];
      shouldGoToAnalyse = Boolean(activeProfile?.isComplete);
      profileIdForEdit = activeProfile?.id ?? profileIdForEdit;
    } catch {
      shouldGoToAnalyse = false;
    }
  }

  if (shouldGoToAnalyse) {
    navigation.navigate("LandingScreen");
    return;
  }

  navigation.navigate("ProfileScreen", {
    firstName: response.user.first_name,
    lastName: response.user.last_name,
    email: response.user.email,
    profileId: profileIdForEdit,
  });
};
```

Issue 8: INCI Decoder Failure

- **Error:**
  The ingredient decoder did not return accurate or usable results.
- **Proposed Fix:**
  Debugging the existing implementation and prompts.
- **Actual Fix:**
  Switched to an alternative model (Gemini), which improved decoding accuracy.

Issue 9: Authentication Limitations (Expo URI)

- **Error:**
  Additional authentication providers, such as Google OAuth, could not be fully implemented due to limitations with Expo redirect URI handling.
- **Proposed Fix:**
  Attempted to integrate external authentication methods, including Google authentication, using standard OAuth flows.
- **Actual Fix:**
  Due to Expo URI restrictions, full support for certain providers was not achievable. As a result, authentication was limited to Expo-compatible methods, with partial or adapted support for Google authentication.

Issue 10: Mobile UI Design Constraints

- **Error:**
  Designing for a small mobile interface created layout and usability challenges. Native Wind was incompatible with GlueStack UI.
- **Proposed Fix:**
  Iterative UI adjustments and testing different layouts. Use different styling library.
- **Actual Fix:**
  A simplified and responsive design approach was adopted to improve usability across screen sizes.

Issue 11: Computer Vision Library Limitations (EAS / iOS Build)

- **Error:**
  Libraries such as React Native Vision Camera and ML Kit could not be successfully implemented within the project.
- **Proposed Fix:**
  Attempted to configure and run these libraries using Expo Application Services (EAS) and test on iOS devices.
- **Actual Fix:**
  The issue stemmed from platform and tooling limitations. iOS testing and builds required macOS, and full EAS builds required an Apple Developer account (€99/year). Due to these constraints, these libraries were not feasible to implement, and alternative approaches were used instead.

  The full proposed code is available on the branch [client/adding_mlKit].

Challenge 12: Node Modules Errors

- **Error:**
  'Property 'queueMicroTask' does not exist, one initial bundling of the mobile app via Expo Go.
- **Proposed Fix:**
  Node version and project libraries are incompatible with what was installed on the machine. Reinstall Node Version Manager and update dependencies.
- **Actual Fix:**
  Dropbox enforces a 258-character limit on file path lengths, which caused issues when the project was stored on the desktop. Moving the project to the C drive resolved this error. Additionally, the React Native Reanimated library required by the Moti animation library was found to be corrupted and incompatible with the locally installed version. To address this, a custom package.json, Babel configuration, and app.json files were created specifically for that machine.

6. Testing

6.1 Introduction

The purpose of this section is to test the implemented features and ensure they work as intended. Below, we'll discuss functional testing (API calls and HTTP requests), integration and unit testing (done via testing packages and a rest client), and feature testing, with a focus on user usability testing, which gives us insight into how the user interacts with the application and handles its different features and structure.

6.2 Functional Testing

Most of the testing was conducted using the selected REST client. Functional testing focused on validating the database structure (ERD) and the workflow sequence defined in the Miro board, with tests organised by permission level (JSON importable for REST client available in the project repository titled 'major_project.postman_collection'). This included admin, moderator, and standard user roles, each evaluated against their intended CRUD operations (full permission list available in Appendix [10]). This phase also facilitated the identification and resolution of bugs (example provided in Appendix [4]).

Testing focused on the following areas:

- Core features such as API call, ingredient extraction and product analysis
- Correct prompt usage per analysis
- Object detection is functioning as intended
- Authentication and authorisation flows (login, registration, password reset)
- Role-based access control to ensure correct permissions for admin, moderator, and user
- Data validation and error handling for API inputs and responses
- Database integrity and correct CRUD operations across all entities
- Image upload and processing performance for object detection
- Consistency of analysis results across repeated scans of the same product
- Response time and performance under normal usage conditions

Additionally, integration and unit testing were conducted using the Jest and SuperTest libraries, with a full list of tests available in the appendix [17].

Unit testing was organised by entity, focusing on validating individual components and their expected behaviours. In contrast, integration testing was conducted on a selected set of test files, as several entities shared similar CRUD operations and functionalities. This allowed for efficient coverage without unnecessary duplication of tests.

6.3 Performance and Algorithm (AI Analysis)

For this application, we rely heavily on LLMs for data processing, ingredient extraction, product evaluation, and RAG implementation. Given this significant dependence on LLMs, it is essential to carefully control the responses generated for each user. This is particularly important as the application deals with users' skin and cosmetic products, where inaccurate or inappropriate recommendations could lead to adverse reactions and potential harm.

Below we will display our choices in LLM's for the overall evaluation denoted by model, and testing variabilities. This includes areas such as:

- AI response time and system performance
- Accuracy and relevance of AI outputs
- Limitations or inconsistencies observed

Variable	Gemini 2.5 Flash	Gemini 2.5 Flash Lite	OpenAI GPT-4.1 Nano
Speed	Fast	Very fast	Fast
Reliability	High	Medium	High
Hallucination Risk	Medium	Medium	Medium–Low
Reasoning Quality	Good	Basic	Good
Best For	Balanced tasks, RAG	Simple, high-volume tasks	Safer user-facing outputs
Cost per Request (approx)	~€0.05–€0.10	~€0.03	~€0.03–€0.06
Consistency of Output	Good	Moderate	Good

As shown above, while Gemini 2.5 Flash was selected as the primary model, it was predominantly used throughout the application and specifically for tokenisation within the RAG pipeline.

Unfortunately, once RAG was implemented, the response speed decreased significantly. This was due not only to the increased number of tokens the model was required to process, including retrieved context, but also to the additional retrieval step involved in querying the vector database. Together, these factors introduced additional latency before a response could be generated for the user. While OpenAI models offered strong performance and reliability for this type of application, they were not adopted as the primary solution due to higher cost per request and overall API usage costs.

As a result, for the final application, LLM Gemini 2.5 flash Lite was chosen, it is fast, reliable and cheap to run across the application. Since RAG vectors are provided to the LLM, the risk of hallucination is inherently reduced. Furthermore, as the system is designed to rely primarily on retrieved context, the likelihood of hallucinated outputs is significantly minimised as well.

6.4 Feature Testing

Feature-level evaluation was conducted in a practical setting following client integration, with a focus on real-world performance rather than isolated unit testing. Due to the nature of AI-driven systems, evaluation prioritised response time, consistency, and reliability during regular usage. The main features assessed were object detection, ingredient extraction, RAG-based retrieval, product analysis, official image retrieval, and prompt engineering.

Object detection was evaluated based on the time taken to identify products from user-submitted images, with generally fast performance and minor delays for more complex inputs. Ingredient extraction was assessed in terms of speed and accuracy, producing consistently accurate results with minimal omissions. RAG-based retrieval was evaluated on its ability to return relevant contextual data, improving the grounding and reliability of responses. Product analysis was assessed based on response time and output relevance; although slightly slower due to RAG processing, responses were more contextually accurate and informative. Prompt engineering was also evaluated, ensuring that structured prompts consistently guided the model to produce relevant, safe, and well-formatted outputs. Additionally, official image retrieval was triggered following ingredient extraction, where extracted ingredients were used to perform a request via SerpAPI, returning relevant visual representations in most cases.

Tests were conducted using Gemini 2.5 Flash Lite.

Feature	What Was Measured	Outcome
Object Detection	Detection time from the image	Fast, minor delays on complex images
Ingredient Extraction	Speed and accuracy	High accuracy, generally quick but can take longer if the image is out of focus
RAG Retrieval	Relevance of retrieved context	Improved grounding and response reliability
Product Analysis	Response time and relevance	Slight delay due to RAG, but informative
Prompt Engineering	Output consistency and structure	Improved relevance and reduced hallucination
Image Retrieval (SerpAPI)	Retrieval time and relevance	Fast, generally accurate image results

6.5 Usability Testing

Usability testing was conducted to evaluate the application's performance alongside ease of use. The main goal was to evaluate the overall UX, focusing on ease of use, clarity of the interface, and how efficiently users could complete tasks.

Both in-person and online testing methods were used to guarantee a variety of user interactions and feedback. This approach enabled testing across different settings, providing a better understanding of how users engage with the system and of overall performance. Participants were asked to finish a set of predefined tasks that reflect typical user behaviour, such as navigating the interface and using key features.

Testing followed key usability principles such as ease of navigation, efficiency and user satisfaction. Observations focused on how easily users could understand the system without assistance, how quickly they completed tasks, and whether any confusion or errors occurred during use.

In addition to observation, users completed a short survey after testing. This provided extra insight into their personal experience, including their thoughts on usability, design clarity, and overall satisfaction.

The results of this usability testing guided further analysis and are examined in detail in the next User Testing section, where specific findings and survey results are presented.

6.6 User Testing

6.6.1 Testing Overview

Overall, five participants took part in the user testing process, each completing one or more of the five predefined tasks designed to reflect core system functionalities (available in the appendix [11]). These tasks covered a range of typical user interactions, including account creation, profile setup, product evaluation (incorporating ingredient extraction), re-evaluation of previously analysed products based on updated profile information, and general navigation of the application. A summary of participants, tasks, and key observations is presented below.

Date:	**Tester:**	**Testing Task:**	**Notes:**
18/04/2026	Matty (Online)	Task 1	Noted that the application doesn't work well on Android
19/04/2026	Edva (Online)	Task 3 & 5	Liked the design, but for a non-technical person, it was overwhelming
19/04/2026	Isobel (Online)	Task 2 & 5	Issues in notification banners
21/04/2026	Angelina	Task 1 & 4	Real-time validation was incorrect, and returned valid, but errored in the backend
25/04/2026	George	Task 3 & 4	Noted some lag in ingredient extraction, but not overall negative

For online testing, participants were required to clone the client-side repository onto their local machine and run the application using Expo on their mobile device. Testers were supplied with download scripts. Testing sessions were conducted via Microsoft Teams, where participants shared their screens as they completed the assigned tasks. Any transcripts from these sessions are provided in Appendix [13].

6.6.2 Notable Findings

The notable findings section is organised based on the tasks in which each issue was identified. We will also highlight what the tester discovered, the issue/error and how it was dealt with.

Task 1 – Register and Create Profile

Tester(s): Matty, Angelina

- **Issue Identified:** On Android devices, the system navigation buttons overlapped with the application's bottom navigation, leading to accidental mis-clicks.
- **Impact:** Users were able to complete registration and profile creation, but with reduced accuracy and increased frustration during navigation.
- **Resolution**: The safe area within the application was adjusted to prevent overlap with system UI elements.
- **Issue Identified:** Notification pop-ups required manual dismissal, interrupting the registration and onboarding flow.
- **Impact:** This disrupted the user experience and reduced interaction fluidity during initial use.
- **Resolution**: Notification behaviour was improved to support automatic dismissal and more intuitive interaction.
- **Issue Identified**: The system indicated that an email address was valid during registration; however, the backend returned an error due to the email already existing in a soft-deleted state.
- **Impact**: This created confusion for users, as the frontend validation contradicted backend behaviour, leading to failed registration attempts.
- **Resolution:** Email validation logic was updated to account for soft-deleted records, ensuring consistency between frontend validation and backend constraints.

Task 2 – Analysing Process

Tester(s): Isobel

- **Issue Identified**: Errors during login were not clearly communicated, as feedback relied on Expo logs rather than visible in-app messaging.
- **Impact:** Users experienced confusion during authentication, as failures were not clearly explained.
- **Resolution:** A dedicated notification banner was implemented to display authentication errors clearly to users.
- **Issue Identified:** The product analysis process was slightly slow during execution.
- **Impact:** This reduced efficiency but did not prevent successful task completion.

- **Resolution:** Performance optimisation was identified as an area for future improvement.

Task 3 – Analysing and Viewing History

Tester(s): Edva, George

- **Issue Identified**: Profile or analysis data occasionally failed to load correctly after navigation, likely due to token handling issues.
- **Impact:** This prevented users from reliably accessing previous scans in the history section.
- **Resolution**: Token handling was corrected to ensure consistent data retrieval and display.

Task 4 – Re-Evaluation of History

Tester(s): George, Angelina

- **Issue Identified:** During re-evaluation, the system failed to return ingredient data and citations after profile updates.
- **Impact:** This reduced user trust in the system's accuracy and limited the usefulness of the re-analysis feature.
- **Resolution**: The issue was traced to the PATCH method, and a new re-evaluation function was implemented to ensure correct data retrieval.

Task 5 – Adding Profile and Analysing

Tester(s): Isobel, Edva

- **Issue Identified**: The system failed to correctly pass or retain previous analysis context when using multiple profiles during re-analysis.
- **Impact:** This led to inconsistent or unreliable results when switching between profiles, reducing trust in multi-profile functionality.
- **Resolution**: The re-evaluation logic was updated to handle profile-specific data and ensure accurate results per profile.
- **Issue Identified:** The tester found the search option within the navigation bar to be misleading, as it suggested the ability to search for external products (like a web search), rather than being limited to in-app functionality.

- **Impact:** This caused confusion and mismatched user expectations, leading users to attempt actions that were not supported by the system.
- **Resolution:** The search feature was removed.

6.6.3 Evaluation of Post Testing Survey

Following the completion of usability testing, participants were asked to complete a structured survey to evaluate their overall experience with the application. The survey focused on key areas including usability, interface design, clarity of results, system trust, and performance. The purpose of this evaluation was to gather quantitative and qualitative feedback to assess how effectively the application meets user needs and to identify any areas requiring further improvement.

Survey responses showed a high level of user satisfaction with the application. Users rated the app as easy to use, giving it an average score of 4.8 out of 5. This suggests it has strong usability and an intuitive design. The layout and clarity of the interface received the same high rating of 4.8 out of 5, indicating that users find the system visually understandable and well-structured.

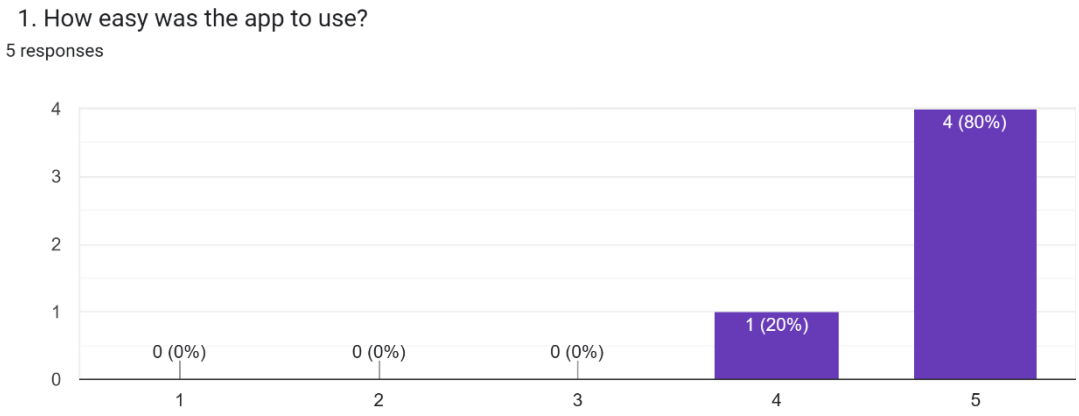


Figure 66. Question 1 from 'User Testing Survey'

Users also viewed the analysis results as clear and easy to interpret, scoring 4.6 out of 5. Overall, trust in the system was relatively high, with a score of 4.6 out of 5. This is especially important since the application relies on AI-generated outputs. It indicates that using the RAG approach positively affected its perceived reliability.

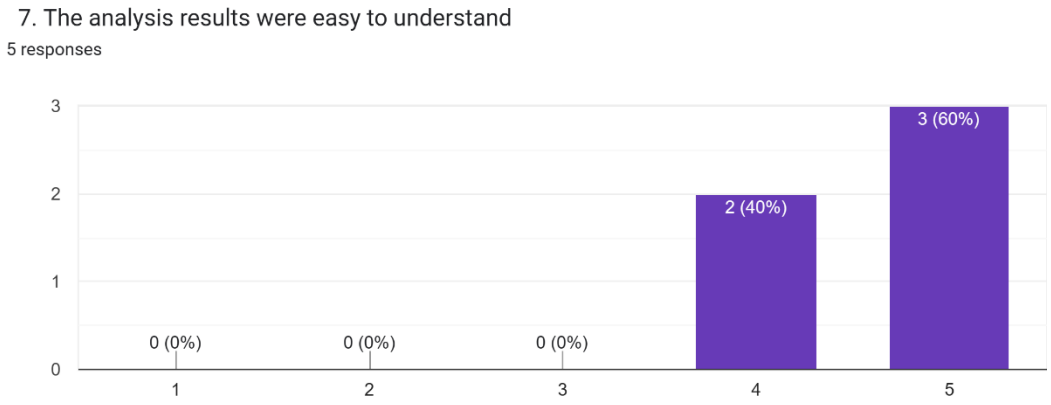


Figure 67. Question 7 from 'User Testing Survey'

Users reported a high level of confidence when using the application without assistance, with most responses rating this aspect at **4–5 out of 5**. This indicates that users felt comfortable navigating and interacting with the system independently, reinforcing the effectiveness of the interface design and overall usability.

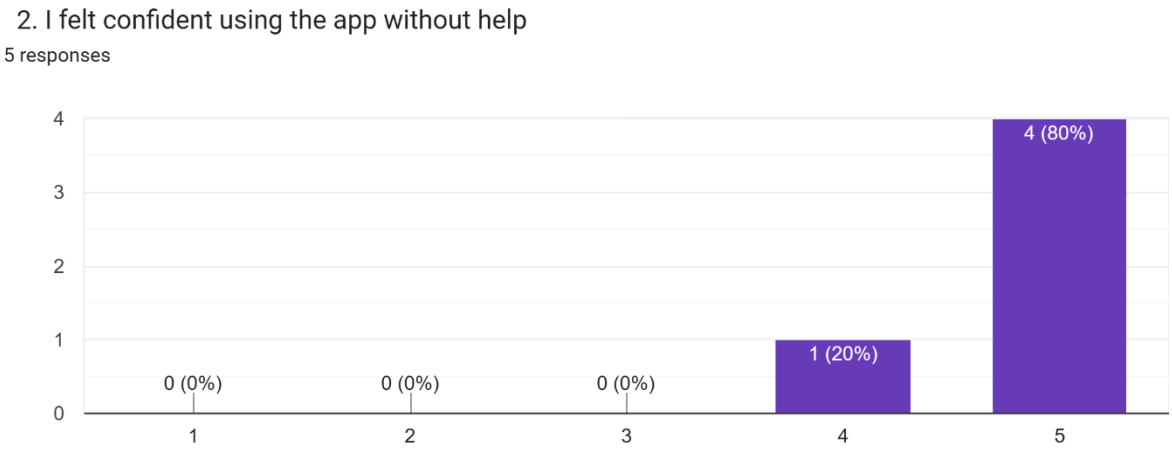


Figure 68. Question 2 from 'User Testing Survey'.

This strong confidence suggests that the application is intuitive and does not require significant external guidance, which is a key indicator of successful user-centred design.

Performance received a moderate rating of 4.4 out of 5, with some users noting slight delays during specific processes, especially product evaluation. Additionally, some users faced errors during testing, like issues mentioned in the notable findings

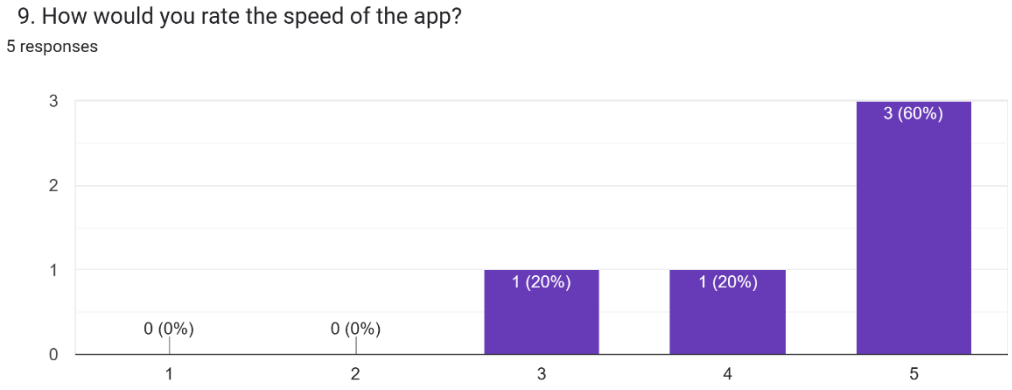


Figure 67. Question 9 from 'User Testing Survey'

section.

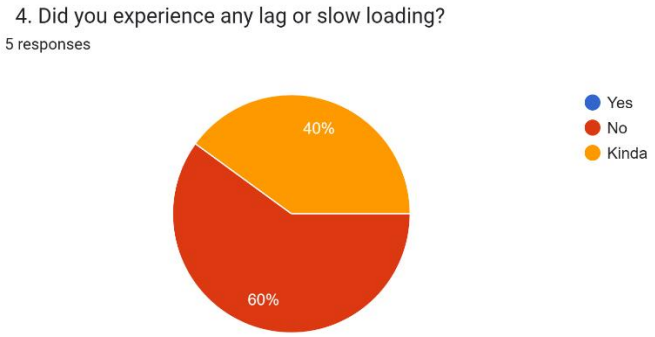


Figure 68. Question 4 from 'User Testing Survey'.

Despite these challenges, all users agreed that the application would be useful in real-life situations, and most were willing to use it again. Overall, the survey results confirm that the application is functional and well-designed, while highlighting key areas for improvement in user guidance, clarity of feedback, and performance enhancement. The full survey results are provided in Appendix [9].

7. Project Management

7.1 Introduction

An Agile/Scrum methodology was adopted for project management. Daily tasks were set on workdays to define short-term objectives within each sprint, allowing for continuous progress and iteration. Based on feedback from the project supervisor, sprint structures were adjusted to better align with time constraints and skill development goals. All progress and changes were documented using GitHub, supporting effective version control and task tracking. Finally, the system architecture and design were documented in Figma and Miro.

7.2 Schedule

The project was structured into six two-week sprints. Initially, a backend-first approach was adopted, but this proved inefficient given time constraints, leading to a more iterative development process. Some delays occurred due to feature complexity and external API integrations (e.g., GitHub and Google), particularly within Expo Go. Full sprint details can be found in the appendix [1].

Sprint 1: 02/02/26 – 16/02/26, focused on building the backend foundation, including user authentication, role-based access, database setup, TypeScript conversion, and initial testing. All objectives were met.

Sprint 2: 16/02/26 – 02/03/26, focused on enhancing authentication with S3 integration, establishing the TypeScript environment and API integrations, developing an AI-driven ingredient/product evaluation system, and validating functionality through testing of password recovery, data storage, and evaluation workflows. All objectives were met.

Sprint 3: 02/03/26 – 16/03/26, focused on integrating the frontend with the backend, developing core UI screens with authentication, implementing profile and analysis-related entities, and ensuring proper data relationships and navigation through testing. All objectives were met.

Sprint 4: 16/03/26 – 07/04/26, focused on implementing core data entities, integrating AI-driven ingredient extraction and evaluation using Gemini with RAG support, refining authentication methods, and validating the full evaluation workflow through comprehensive testing. All objectives were met.

Sprint 5: 07/03/26 – 21/04/26, focused on finalising the frontend, deploying the backend and integrating it with the app, implementing object detection, preparing and conducting user testing, and validating the complete end-to-end system through performance and functionality testing. Object detection could not be met; more on that in the previous chapter, Challenges & Solutions.

Sprint 6: 21/04/26 – 31/04/26, focused on completing user testing and documentation, refining the application based on feedback, implementing additional testing, and

performing final validation of performance, functionality, and overall system stability. All objectives were met.

7.3 Project Phases

7.3.1 Proposal

The main goal of the proposal phase was to set expectations for what this project could be, focusing on areas such as time management, realistic feature goals, system architecture, design (e.g., wireframes and paper prototypes), and the overall project aim. From here, we set goals and organised chronological steps used to complete this project. This research plan is proposed based on the overall need and purpose of this application. And finally, we discussed the supervision plan with the project supervisor and the second reader, organising meetings and weekly goals in the application implementation.

7.3.2 Literature Review and Technical Background

The primary basis for the application was to address a well-researched issue: unreliable beauty product claims and the lack of support for individuals with chronic or ongoing dermatological or trichological conditions. We focused on researching existing regulatory bodies and legislation, analysing the laws concerning ingredient control and their shortcomings. Another area of research was into the ethical sourcing of ingredients (such as Palm Oil) and the evaluation of the advantages and limitations of each ingredient. Finally, we discussed how to overcome these limitations and proposed ideas for future improvements.

7.3.3 Design

The design phase was divided into server-side and client-side components. The server design focused on system architecture, entity–relationship diagrams, database structure, and the chosen technologies, with particular emphasis on Prompt Engineering and the RAG approach in relation to the project goals.

The client-side design addressed the visual and UX aspects, including style guides, typography, colour schemes, and user workflows. This was supported by wireframes, paper prototypes, and workflow diagrams to illustrate the proposed application.

7.3.4 Implementation

The implementation phase details the development of the application across both server and client components, including deployment and debugging. The primary objective was to deliver features in line with the planned sprints over the 12 weeks, with a strong focus on the selected frameworks and libraries.

Throughout this phase, an iterative approach was adopted, allowing features to be continuously refined based on testing and identified issues. Errors in code and design were documented, alongside proposed solutions and their outcomes. Attention was given to integrating external APIs, handling AI-generated responses, and ensuring consistent data flow between system components.

This phase also involved adapting to technical challenges, especially in the mobile development environment, which required adjustments to implementation strategies to ensure overall system functionality.

7.3.5 Testing

Testing was conducted in two phases: integration and unit testing, using the Jest and Super Test libraries. Integration tests typically require multiple test cases per entity. Therefore, entities with similar behaviour were tested against a single representative object to avoid redundancy and prevent unnecessary delays in the development process. Additional testing was performed using a REST client, with documentation provided in Miro (see Appendix [5]), in which we tested each HTTP endpoint, sectioned by user role.

The primary testing, however, was conducted through user usability testing. The associated tasks, consent forms, and testing scripts are provided in their respective appendices (see Appendices [11-14]). Participants also completed surveys, the results of which informed the design process and led to minor design improvements. These results can be found in Appendix [9].

7.4 Management Tools

7.4.1 GitHub

A mono-repo strategy was employed to manage client, RAG, and server code in a single repository. Although this can simplify development and promote consistency, its advantages are more apparent in web applications with integrated deployment. For this mobile-focused project, which deploys through app stores, the benefits were less significant, but the approach still facilitated an organised development process.

As a result, the primary deployment directory root is *the server,* and the primary branch is *main*.

-Branching

Each section of the application had its dedicated branch denoted by the class the branch fell under (client or server).

Each branch was named after the database entity being developed, with additional branches created to fix features, design issues, and debug. This can be viewed below:

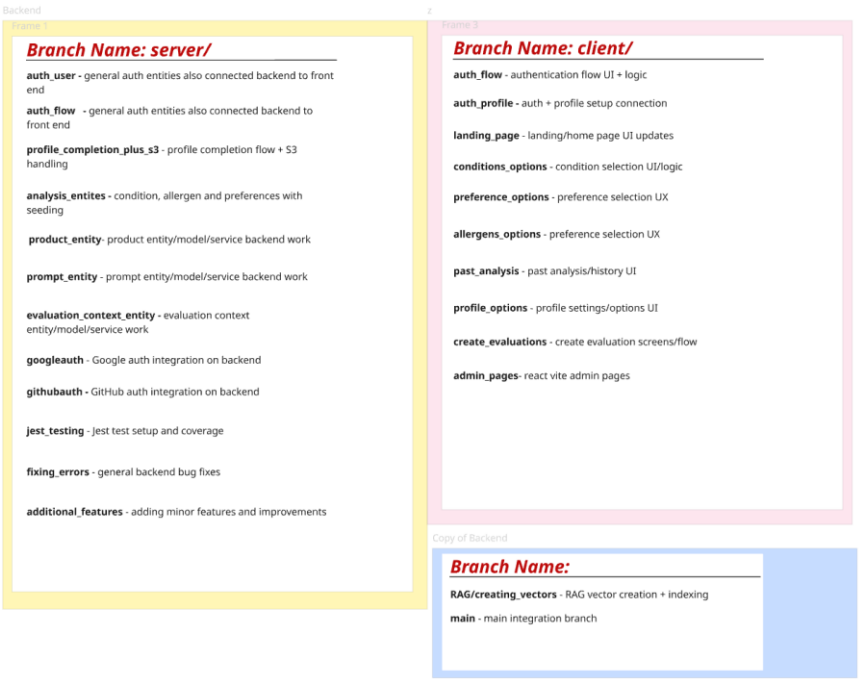


Figure 69. Screenshot of GitHub Branching

-Commits

Each commit includes meaningful comments based on the stage of implementation, including any errors encountered, how they were resolved, and possible explanations for the bug. View samples below.

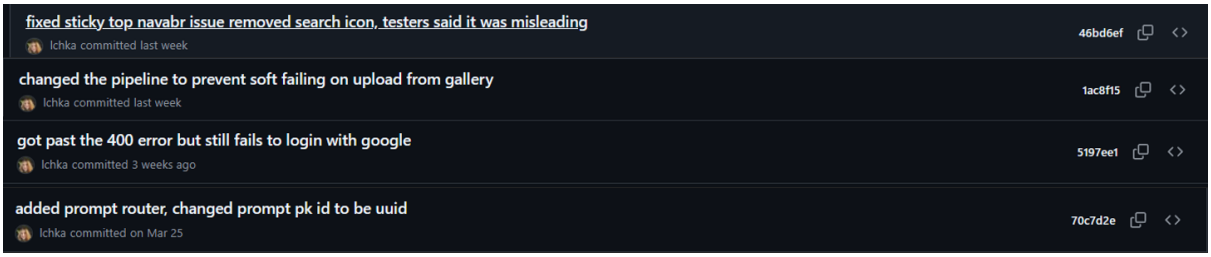


Figure 70. Screenshot depicting sample commit messages.

7.4.2 Miro

Miro was used throughout the project to support collaborative planning and design, including brainstorming features and ideas, developing client-side design concepts and style guides, mapping system architecture, outlining planned features, and organising the GitHub branch workflow. It also helped structure approaches to server-side unit and integration testing.

The Miro link can be viewed in Appendix [5].

7.4.3 Figma

Figma was used to design the UI of the mobile application, including the creation of both low-fidelity and high-fidelity wireframes for various screens. It was utilised to define layouts, components, and the overall visual structure, as well as to establish consistent style elements such as colour schemes, typography, and spacing. Reusable components were developed to ensure design consistency across the application, and user flows were prototyped to visualise interactions. Designs were iteratively refined based on feedback and usability considerations, and Figma served as a reference throughout frontend development to support accurate implementation.

The Figma link can be viewed in Appendix [6].

8. Conclusion & Future Improvements

In conclusion, the development and evaluation of this application demonstrate its effectiveness as a user-friendly and reliable AI-driven solution. By implementing a Retrieval-Augmented Generation (RAG) approach, the system produces more accurate and interpretable outputs, supported by verified external sources. This is particularly important in the context of ingredient analysis and dermatological relevance, where credibility and transparency are essential.

User testing and survey evaluations further confirm the success of the system. Participants could complete key tasks easily, and feedback indicated high satisfaction with usability, interface clarity, and overall design. Users also felt confident navigating the application on their own, implying that the system is intuitive and accessible to its target audience.

However, the evaluation process pointed out several areas that need improvement. Minor usability issues, like navigation inconsistencies and unclear feedback in some situations, were found during testing. Users also noted performance limitations, especially during product evaluation. Although these issues did not greatly affect functionality, they offer important guidance for future development.

Building on these insights, several enhancements could further improve the application. Adding a profile transfer feature would let users share profile data with others, making it more accessible and flexible. Additionally, a recommendation system based on product ingredients and user preferences could enhance the application by suggesting suitable products based on previous scans, allowing users to receive personalised recommendations aligned with their past interactions.

Including a user feedback option within the AI analysis, such as a true/false or helpful/not helpful choice, would let users validate responses. This feedback could refine future outputs, especially when products with similar ingredients do not always yield consistent results.

Finally, as noted in the survey responses, usability could improve by adding a product search feature. This would allow users to search for products directly in the application instead of relying only on scanning or uploading images. Overall, these improvements would enhance both the functionality and user experience of the system.

References

1. Ajayi, S. A., Olaniyi, O. O., Oladoyinbo, T. O., Ajayi, N. D., & Olaniyi, F. G. (2024). *Sustainable sourcing of organic skincare ingredients: A critical analysis of ethical concerns and environmental implications*. **Asian Journal of Advanced Research and Reports, 18(1), 65–91.** **Ajayi, S.** A., Olaniyi, O. O., Oladoyinbo, T. O., Ajayi, N. D., & Olaniyi, F. G. (2024). Sustainable sourcing of organic skincare ingredients: A Critical analysis of ethical concerns and environmental implications. *Asian Journal of Advanced Research and Reports*, *18*(1), 65–91. **https://doi.org/10.9734/ajarr/2024/v18i1598**
2. Lionetti, N., & Rigano, L. (2018). Labeling of cosmetic products. *Cosmetics*, *5*(1), 22. https://doi.org/10.3390/cosmetics5010022
3. Vendruscolo, C. W., Bagatin, E., & Leonardi, G. R. (2025). The science behind the label: Evaluating claims in Dermatologist-Recommended cosmetics. *Dermatological Reviews*, *6*(4). https://doi.org/10.1002/der2.70045[2/der2.700452/der2.70045](https://doi.org/10.1002/der2.700452/der2.70045)
4. Torres, A., Rego, L., Martins, M. S., Ferreira, M. S., Cruz, M. T., Sousa, E., Almeida, I. F., Torres, A., Rego, L., Martins, M. S., Ferreira, M. S., Cruz, M. T., Sousa, E., & Almeida, I. F. (2023). How to promote skin repair? In-Depth look at pharmaceutical and cosmetic strategies. *Pharmaceuticals*, *16*(4), 573. https://doi.org/10.3390/ph16040573
5. Chinecherem, U. M., Uju, E., & Geraldine, N. (2023). PALM OIL PRODUCTION FOR THE FOOD AND COSMETICS INDUSTRY IN AFRICA: ETHICS AND SUSTAINABILITY IMPLICATIONS. In *IntechOpen eBooks*. https://doi.org/10.5772/intechopen.1001107
6. Slanzi, D. (2022, July 19). *Sustainability and transparency in the cosmetic industry: the clean beauty movement and consumers' consciousness*. https://unitesi.unive.it/handle/20.500.14247/11811
7. Klaschka, U. (2016b). Natural personal care products—analysis of ingredient lists and legal situation. *Environmental Sciences Europe*, *28*(1), 8. https://doi.org/10.1186/s12302-016-0076-7
8. Shi, E., Yu, S., Hu, H., Dai, H., Yang, Q., Wu, J., Pan, Y., Sun, L., & Liu, T. (2025). Prompt Engineering for Healthcare: Methodologies and applications. *Meta-Radiology*, 100190. https://doi.org/10.1016/j.metrad.2025.100190
9. Chen, L., Weng, H., Pardeshi, M. S., Chen, C., Sheu, R., & Pai, K. (2025). Evaluation of prompt engineering on the performance of a large language model in document information extraction. *Electronics*, *14*(11), 2145. https://doi.org/10.3390/electronics14112145
10. Gomez, A. P., Panarotto, M., & Isaksson, O. (2024). Evaluation of Different Large Language Model Agent Frameworks for Design Engineering Tasks. *The Design Society*, 693–702. https://doi.org/10.35199/norddesign2024.74
11. Mohammad, R. (2024). Advancing generative AI with RAG: Enhancing relevance, creativity, and reliability in language models. *IJCET*, IJCET_15_04_027.

12. *Unlocking Data with Generative AI and RAG*. (n.d.). Google Books. https://books.google.ie/books?hl=en&lr=&id=NukhEQAAQBAJ&oi=fnd&pg=PP1&dq=json+data+with+rag&ots=BbrNpstjpr&sig=-5t7pfV7KEV9IqE-BzBj88arhLM&redir_esc=y#v=onepage&q=json%20data%20with%20rag&f=false
13. Sayallar, Ç., & Sayar, A. (2025). Harmful ingredient detection from cosmetic products using optical character recognition and bi-LSTM model. *Signal Image and Video Processing*, *19*(4). https://doi.org/10.1007/s11760-025-03923-0
14. Richard, M., Paul, C., Nijsten, T., Gisondi, P., Salavastru, C., Taieb, C., Trakatelli, M., Puig, L., & Stratigos, A. (2022). Prevalence of most common skin diseases in Europe: a population-based study. *Journal of the European Academy of Dermatology and Venereology*, *36*(7), 1088–1096. https://doi.org/10.1111/jdv.18050
15. *chronic, adj. & n. meanings, etymology and more | Oxford English Dictionary*. (n.d.). https://www.oed.com/dictionary/chronic_adj
16. Tabassum, N., & Hamdani, M. (2014). Plants used to treat skin diseases. *Pharmacognosy Reviews/Bioinformatics Trends/Pharmacognosy Review*, *8*(15), 52. https://doi.org/10.4103/0973-7847.125531
17. Tuckman, A. (2017). The potential psychological impact of skin conditions. *Dermatology and Therapy*, *7*(S1), 53–57. https://doi.org/10.1007/s13555-016-0169-7
18. Ferreira, M., Matos, A., Couras, A., Marto, J., & Ribeiro, H. (2022). Overview of Cosmetic Regulatory Frameworks around the World. *Cosmetics*, *9*(4), 72. https://doi.org/10.3390/cosmetics9040072
19. Bialas, I., Zelent-Kraciuk, S., & Jurowski, K. (2023). The Skin Sensitisation of Cosmetic Ingredients: Review of Actual Regulatory Status. *MDPI*. https://www.mdpi.com/2305-6304/11/4/392
20. Scientific Committee on Consumer Safety, S. (n.d.). *Annex II*. European Commission. https://ec.europa.eu/growth/tools-databases/cosing/reference/annexes/list/II
21. European Union, E. (2009). REGULATION (EC) No 1223/2009 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 30 November 2009 on Cosmetic Products. *Official Journal of the European Union*.
22. Ibm. (2025, November 17). AI Hallucinations. *IBM*. https://www.ibm.com/think/topics/ai-hallucinations
23. Schöffer, J., Jakubik, J., Vössing, M., Kühl, N., & Satzger, G. (2025). AI reliance and decision quality: fundamentals, interdependence, and the effects of interventions. *Journal of Artificial Intelligence Research*, *82*. https://doi.org/10.1613/jair.1.15873

Appendix

1. Sprints

Sprint: Sprint 1	
Start Date: 02/02/26	End Date: 16/02/26

Goals | *Discuss goals required to complete in this sprint*

Week 1: By the end of week 1 we should have a basic database with user authentication set in, including user roles, user seeding, and protected routes.

- Implement PostgreSQL, create a mock database
- Begin Node Instance, install relevant packages,
- Create user models,
- Create user role middleware and separate admin from ordinary user roles, set policies for each role
- Create a router within index.js, set up protected routes
- Create a seeder and seed three users for each role.
- Create a user/auth Controller- register, login and logout
- Complete User Authentication + User CRUD
- Test User Authentication within the REST client

Week 2: By the end of week 2 we should have finalised the user authentication, with the relevant roles attached, with relevant policies and permissions. Began working on client side with testing different LLM keys for result evaluation. Convert backend code to TypeScript to match front end.

- Create Profile model alongside its controller
- Convert to typescript
- Add user permission and policies for each role
- Began Client side, test Gemini and OpenAI keys

Testing

- Test for database entry – is the user appearing in the database?
- Test register, does it auto-log in the user?
- Test logout – does the token get removed?
- Can a user only update their own user details?
- Can the admin (only admin) CRUD the Preferences
- Is backend working correctly after typescript conversion

Sprint: Sprint 2	
Start Date: 16/02/26	End Date: 02/03/26

Goals | *Discuss goals required to complete in this sprint*

Week 1: By the end of week 1, we should have user authentication fully complete, including s3 profile image storage, with a token the user must enter into the REST client. Furthermore, we should have set up the TypeScript Environment with API keys for Gemini and . Finally, we should set up an ingredient pulling system using InciDecoder.

- Create S3 Bucket
- Finalise Draft ERD
- Work on Mobile Wireframes
- Organise Thesis Headings and re-read the research chapter
- Work on Functional, Non-Functional and System Requirements.
- Using the client-side pull ingredients from the product based on image analysis with the chosen LLM Model.

Week 2: By the end of week 2, we should have a mock evaluation system, taking in the user's hard-coded skin condition, hardcoded product, ingredient pulling for said product and a final evaluation that** is stored in the database **attached to the user id.

- Take the ingredient list pulled from the product.
- Collect user skin condition information [hard-coded].
- Preprocess ingredients and skin condition data for consistency.
- Design prompts to ask the AI if the product is suitable for the user.
- Use prompt engineering to guide Gemini's analysis (e.g., highlight irritants, allergens).
- Generate suitability analysis and explanations from Gemini.
- Store or return the analysis for frontend display or further processing

Testing

- Test for user reset password, does the token expire correctly?
- Test for user reset password, does the token match and allow the user to reset the password.
- Test for user reset password, does the reset password button correctly trigger the event to send a token to the user's email?
- Using ingredients, can we prompt the user for a skin condition and, using Gemini, evaluate the product?
- Are RAG vectors storing correctly?

Sprint: Sprint 3

Start Date: 02/03/26	End Date: 16/03/26

- Does the evaluation get stored in the database using the user ID.

Goals | *Discuss goals required to complete in this sprint*

Week 1: By the end of week 1, we should have completed the remaining goals from the last sprint. Additionally, we should have connected the backend to the frontend, created a landing screen for the front end, with a basic registration/login screen, and replaced hardcoded user data in the backend for analysis.

- Connect backend frontend(locally)
- Began frontend design and began landing page
- Create basic register and login pages (with auth)
  Replace User data for analysis entities with user inputted data

Week 2: By the end of week 2 we should have implemented profile features in the frontend (connecting profile to registration) and tested via expo. In the backend we began coding conditions, preferences, and allergens. Connecting them with the respective ERD set relationships and improve navigation in the front end. Finally attaching conditions, preferences, allergens to the profile create and test in backend.

- Implement profile features in the frontend
- Implement conditions entity
- Implement allergen entity,
- Implement preference entity
- Connect analysis entities to the profile and test store.
- Connect all analysis entities correctly via Prisma based on ERD relationships.

Testing

- Test CRUD Preference
- Test CRUD Allergen
- Test CRUD Condition
- Test Register -> Profile (Frontend)
- Test Landing Screen is appearing on user phone and does not allow to be viewed by unauthenticated users. (guest)
- Test Navigation Routes

Sprint: Sprint 4	
Start Date: 16/03/26	End Date: 07/04/26

Goals | *Discuss goals required to complete in this sprint*

Week 1: By the end of week 1, we should implement seed preferences, conditions and allergens with real data. Implement Prompt entity and seed prompts. Implement the Product entity and attach ingredient extraction via the Gemini flash model, and remove the Inci decoder as it is unreliable. Began Evaluation Context entity. Implement RAG and attach to Gemini analysis. Chunk relevant papers.

- Seed allergen
- Seed Condition,
- Seed Preference,
- Implement Prompt Entity,
- Implement the Product entity and attach ingredient extraction
- Remove Inci Decoder
- Began Evaluation Context entity
- Implement RAG and store on Pinecone.

Week 2: By the end of week 2, we need to complete the evaluation context entity and the seed prompt entity, based on the Enum of categories via the product and prompt entity. Attach relevant permissions and policies based on user role for all new entities. Test evaluation context, prompt output, and finalise the registration->profile/login process for the frontend. Implement RAG and attach to Gemini analysis. Attach Google, Apple, GitHub auth logins.

- Complete evaluation context entity.
- Attach the Gemini model to the evaluation context entity.
- Create Prompt objectives based on categories.
- Attach permissions and polices, test per user role
- Generate suitability analysis and explanations from Gemini.
- Store or return the analysis for frontend display or further processing
- Create API keys for each auth method and implement in front + backend.

Testing

- Test CRUD product entity.
- Test CRUD prompt entity
- Test Prompt seeded objects.

- Test Image upload/ingredient extraction based on Gemini flash model
- Test CRUD evaluation context entity.
- Can we implement RAG and use the stored vectors and attach does to the evaluation using the stored vectors?
- Does the evaluation get stored in the database using the user ID.
- Test login with additional auth methods via expo.

Sprint: Sprint 5	
Start Date: 07/04/26	End Date: 21/04/26

Goals | *Discuss goals required to complete in this sprint*

Week 1: By the end of week 1, we should finalise screens for: welcome, landing, analyse result, image capture, history, register, login and additional minor screens. Set up object detection via package (tbc) for camera screen. Host the backend on the chosen hosting platform and reattach the front end to the hosted backend. Complete navigation and add minor design aesthetics to screens. Began paperwork and necessary testing tasks for user testing.

- Complete all screens
- Host on Render and attach frontend to hosted backend
- Complete navigation and add minor design aesthetics.
- Began paperwork necessary for user testing, such as consent forms, testing tasks and scripts.
- Implement object detection via detection package and test on multiple different objects.

Week 2: By the end of week 2, we should have the front end finalised and ready for testing, and all testers and tasks allocated and set up. Full test of backend to front end and add additional loading screens, and test performance back-to-back. Re-test and validate the response from: object detection, ingredient extraction, analysis and re-analysis. Add any additional features, such as feedback on the analysis response or profile transfer, if there is time.

- Testing paperwork is completed and testing tasks are allocated to testers.
- Back-to-back testing from new hosted backend
- Add additional loading screens.
- Test performance back-to-back
- Re test performance for object detection.
- Re-test performance for analysis and re-analysis
- Re-test performance for ingredient

Testing

- Test all screens + navigation, make sure everything works properly and nothing breaks

- Test frontend connection to hosted backend (login/register + pulling data)
- Test object detection + image capture on different objects
- Test full flow from image → ingredients → analysis → result
- Test analysis vs re-analysis, are results consistent?
- Test saving and pulling evaluation history per user
- Test performance with back-to-back requests (API + frontend)
- Test error handling + loading screens (failed requests, slow responses, bad input)Can we implement RAG and use the stored vectors and attach does to the evaluation using the stored vectors?
- Does the evaluation get stored in the database using the user ID.

Sprint: Sprint 6	
Start Date: 21/04/26	End Date: 31/04/26

Goals | *Discuss goals required to complete in this sprint*

Week 1: By the end of week 1, we create the post-testing survey, complete user testing, and cite all findings in the main report. Add any additional fixes to the frontend from user testing. Complete reflection, introduction and future work in the report. Finally, add front-end testing with Jest & SuperTest.

- finalise all screens
- Fix any minor issues based on user testing's
- Cite user findings
- Complete reflection, introduction and future work chapters.
- Add any frontend testing as needed with Jest and SuperTest

Week 2: By the end of week 2, we should re-check all performances and finalise the sprint schedule subchapter in the report. Additionally, we should re-run all testing on the backend and frontend.

- Re-check performance of ingredient extraction
- Re-check analysis and re-analysis
- Re-run integration tests front and backend
- Re-run unit tests for the front and backend.

Testing

- Test full app after user testing changes (all screens + navigation still working)
- Test frontend + backend connection after fixes (no broken endpoints)
- Test ingredient extraction, analysis and re-analysis after updates
- Test performance again (back-to-back requests, response times)
- Test integration between frontend and backend (full user flow

- Test unit + integration tests (frontend + backend using Jest & SuperTest
- Test for any new bugs introduced from fixes or changes
- Test overall usability improvements based on user feedback

2. Jacob Nielson's 10 Usability Heuristics

This diagram was adapted from Aufait UX heuristic evaluation article

3. List of Product Ingredients

Official Ingredients: Aqua, Stearyl Alcohol, Behentrimonium Chloride, Cety! Alcohol, Bis-Aminopropyl Dimethicone, Partum, Isopropyl Alcohol, Benzyl Alcohol, Citric Acid, Sodium Benzoate, Tocopheryl Acetate, Disodium EDTA, Polysorbate 20, Panthenol, Panthenyl Ethyl Ether, Niacinamide, Histidine, Hexyl Cinnamal, Hydroxycitronellal, limonene

4. Issues in Implementation table

Errors occurred: Rows highlighted in yellow indicate where AI-assisted code generation was used, while rows highlighted in green represent sections where AI was utilised for explanatory support.

#	Error / Issue	Category	Stage	Cause	Resolution
1	Backend and frontend could not be connected when testing via Expo Go	Server	Early Development	The mobile device and backend server were not on the same Wi-Fi network, preventing communication	Checked terminal output and ensured both devices were connected to the same network
2	User profile was not linked to the correct user during registration	Server	Early Development	The authenticated user token was not included, so the system could not retrieve the user ID	Implemented a GetCurrentUser method to correctly associate profiles with the user ID
3	AWS S3 access issues prevented users from retrieving product resources	Server	Mid Development	IAM policies were incorrectly configured, blocking access even when read permissions appeared enabled	Reviewed and corrected IAM roles and bucket policies
4	Stored product images could not be retrieved from S3	Implementation	Mid Development	A mismatch between how images were stored and retrieved caused access failure	Corrected inconsistencies in image handling logic
5	Ingredient extraction returned incorrect or null values using INCI Decoder	Implementation	Mid–Late Development	INCI Decoder produced inconsistent outputs and soft failures when parsing AI-generated ingredient text	Replaced INCI Decoder with Gemini for more reliable extraction
6	External APIs for conditions and allergens were unusable	Implementation	Late Development	APIs lacked documentation, had incomplete data, were expensive, or required scraping	Replaced with internally seeded dataset and implemented admin CRUD functionality
7	RAG implementation significantly increased response time	Implementation	Late Development	Additional steps such as embedding retrieval and prompt augmentation increased latency	Accepted as a trade-off for improved contextual accuracy
8	Third-party authentication (Apple, Google, GitHub) could not be fully implemented	Server	Late Development	Apple required a paid developer account, while Google and GitHub failed due to incorrect redirect URIs in Expo Go	Retained implementation but left features non-functional without EAS/production setup

#	Error / Issue	Category	Stage	Cause	Resolution
9	ML Kit real-time object detection could not be implemented	Implementation	Late Development	Required EAS build services which were unavailable within project constraints	Feature was not implemented and alternative scanning approach was used
10	EAS build could not be used due to lack of macOS environment	Implementation	Late Development	iOS builds required macOS which was not available	Limited iOS testing and avoided EAS-based features
11	EAS build restricted due to Apple Developer account cost	Implementation	Late Development	Apple Developer Program required €99/year subscription which was outside project budget	Avoided use of EAS deployment features requiring Apple credentials
12	Gemini API rate limiting restricted RAG dataset size	Implementation	Late Development	API tier limits caused failures when processing more than ~12 research papers and broke Pinecone uploads	Limited dataset to 12 papers based on testing within API constraints
13	Frontend state (isComplete) was not updating correctly	Client	Late Development	Conflicting conditional logic created a false positive where the frontend showed true but backend remained false, with no clear logging	Simplified conditional logic after analysing issue with AI (Copilot)
14	Gemini model latency required switching models	Implementation	Late Development	Performance limitations in Gemini Flash 2.5 caused slow response times	Switched to Gemini Flash 2.5 Lite following performance improvements
15	Gluestack documentation inconsistencies slowed UI development	Client	Late Development	Component usage varied significantly across versions with limited documentation for the version used	Generated an AI-assisted cheat sheet to standardise usage
16	Styling conflict between NativeWind and Gluestack	Client	Late Development	NativeWind was incompatible with the Gluestack styling system	Removed NativeWind and used inline/component-based styling
17	Prisma client failed to initialise during deployment	Server	Late Development	Deployment environment did not correctly run Prisma generation steps	Updated build command to match local development setup
18	Incorrect citation links in AI-generated JSON output	Implementation	Late Development	Model-generated citation structure but lacked consistent mapping to actual source links	Created asset mapping RAG papers to hosted links for correct citation output
19	Evaluation context prompt was being overwritten	Implementation	Late Development	Prompt entity structure caused main evaluation prompt to be overridden, leading to inconsistent responses	Refactored relationships to allow modular prompt composition

#	Error / Issue	Category	Stage	Cause	Resolution
20	Integration testing failed due to strict typing with Supertest	Server	Late Development	Strong TypeScript typing conflicted with Supertest request handling and middleware (e.g., next functions)	Reduced test scope and created representative integration test files
21	Repeated 503 errors from external API	Client	Late Development	Requests failed due to high frequency and lack of retry logic, causing immediate failure	Refactored logic to allow retry attempts instead of failing immediately
22	React Native navigation inconsistencies	Client	Late Development	State and caching issues caused incorrect screens/components to render after reloads	Used more direct component linking instead of relying fully on navigation state
23	queueMicrotask error occurred only on local device	Implementation	Late Development	Likely caused by Node/NVM inconsistencies or dependency issues, though exact cause was unclear	Implemented custom microtask handler and imported into core server files
24	Difficulty designing mobile UI due to limited experience	Client	Throughout Development	Greater experience in web development made adapting to mobile UI paradigms challenging	Improved UI iteratively using testing, experimentation, and component libraries

5. Miro Board

Miro Link: https://miro.com/app/board/uXjVJBYsjNM=/?share_link_id=159790122455

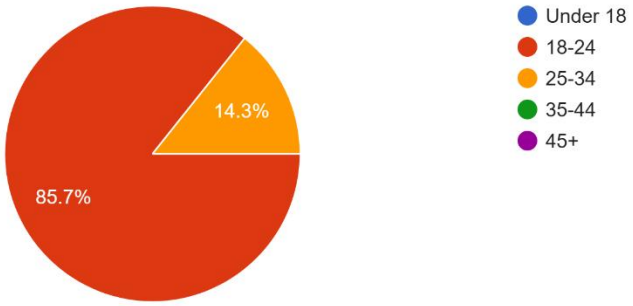
6. Figma Board

Figma Link: https://www.figma.com/design/ivB38UFTVyuOW0WCgEVWUz/MajorProject?node-id=0-1&m=dev&t=39j2XidqhPF4Vyfu-1

7. Survey: Skincare Product Decision Survey

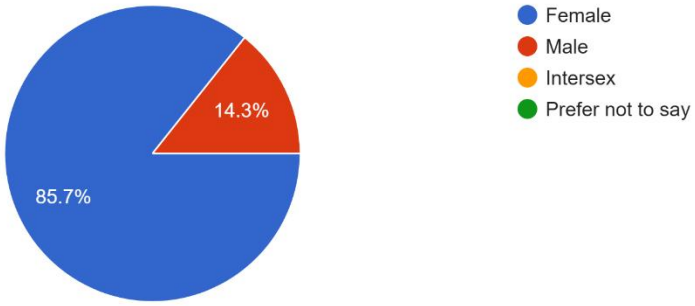
1. What age group are you in?

7 responses



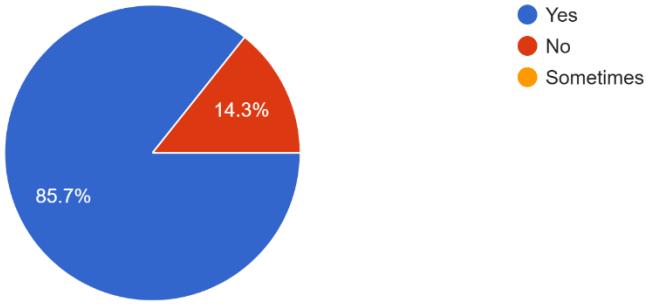
2. What is your biological sex?

7 responses



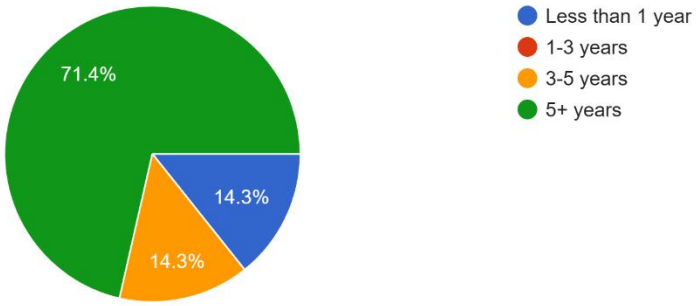
3. Do you have any ongoing skin concerns or conditions?

7 responses



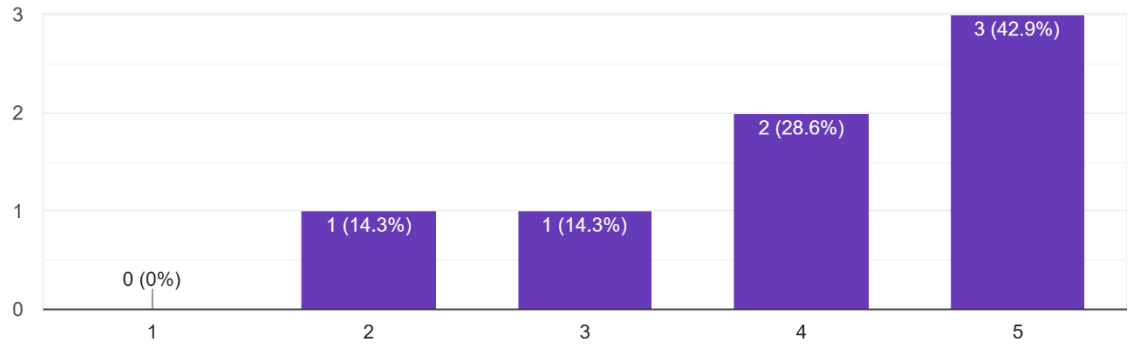
4. If yes, how long have you been dealing with it?

7 responses



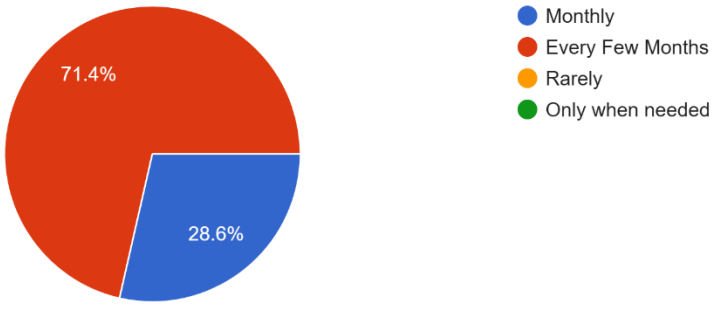
5. How important is skincare/cosmetic safety to you? (1–5 scale)

7 responses



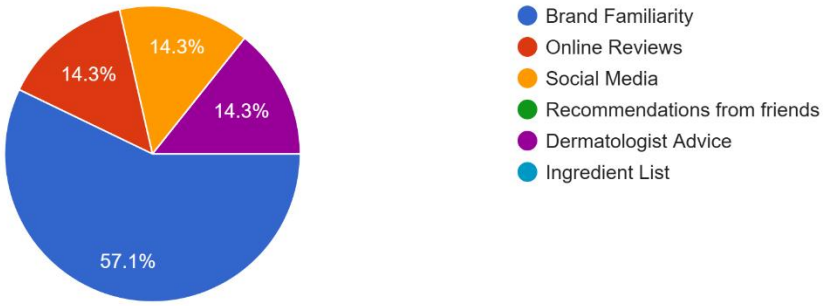
6. How often do you purchase skincare or cosmetic products?

7 responses



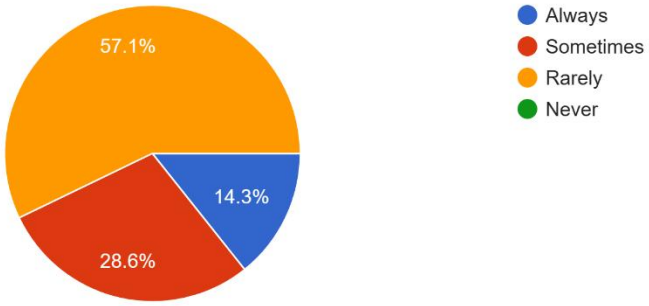
7. When buying a new product, what do you usually rely on?

7 responses



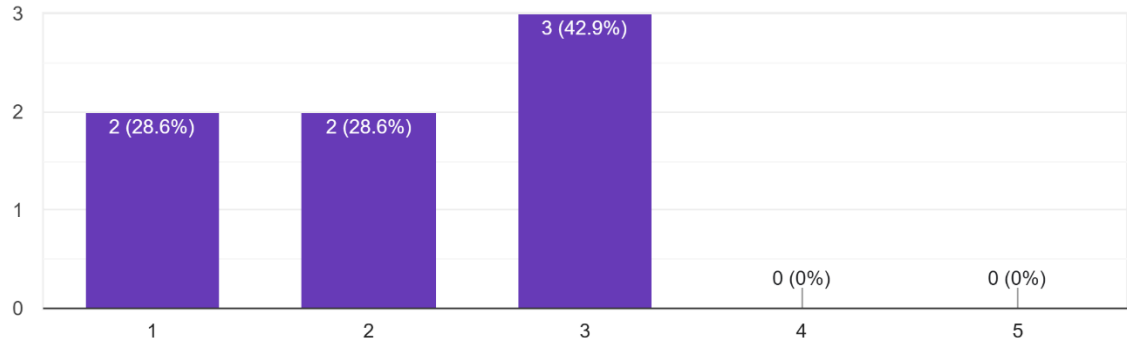
8. Do you read ingredient labels before buying?

7 responses



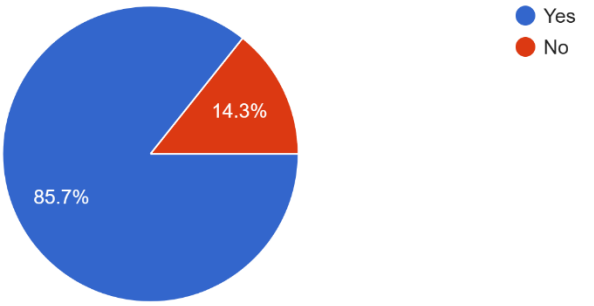
9. How confident do you feel understanding ingredient lists? (1–5 scale)

7 responses



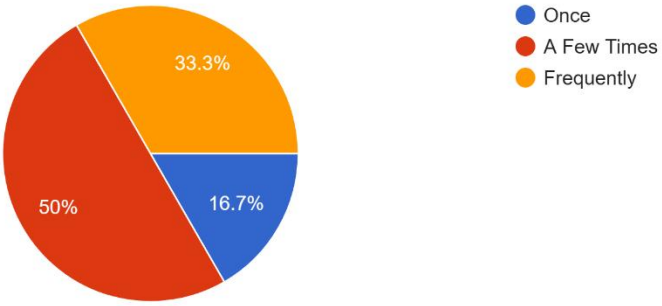
10. Have you ever purchased a product that negatively affected your skin?

7 responses



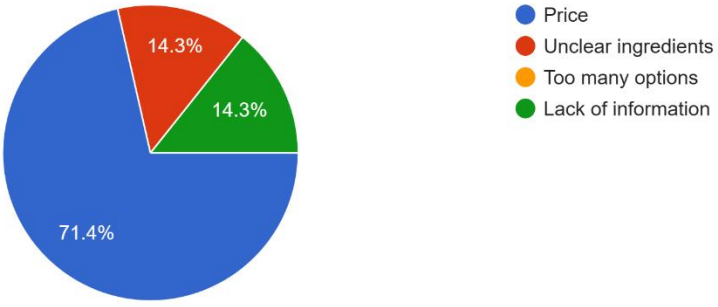
11. If yes, how often has this happened?

6 responses



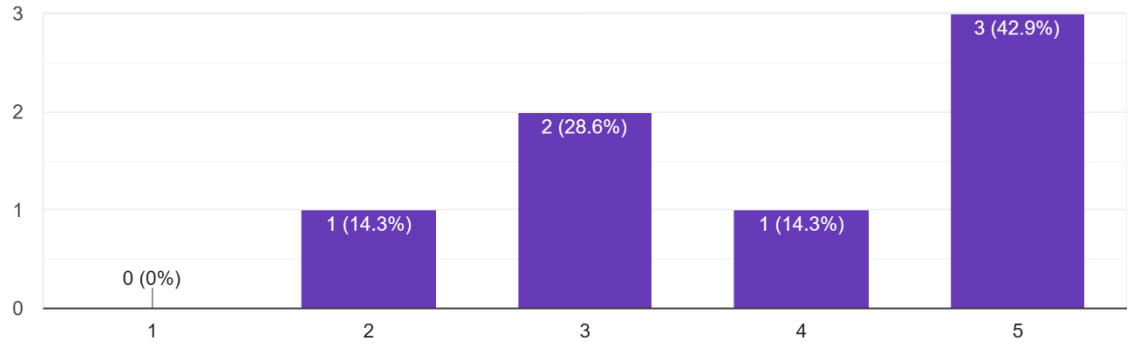
12. What usually causes hesitation when buying a product?

7 responses



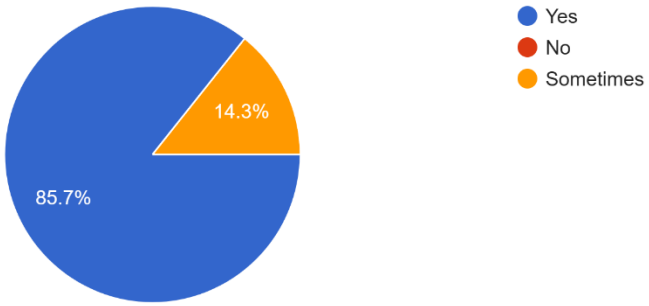
13. On a scale of 1–5, how stressful is choosing new skincare products?

7 responses



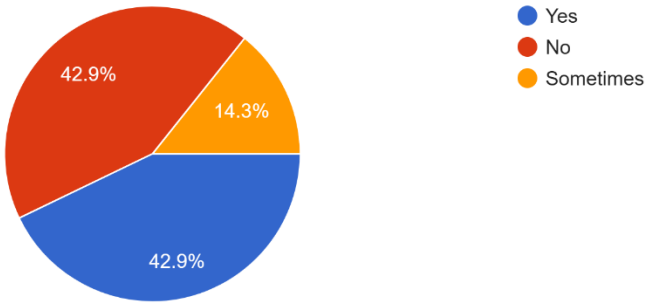
14. Do you feel there is too much conflicting information online about skincare?

7 responses



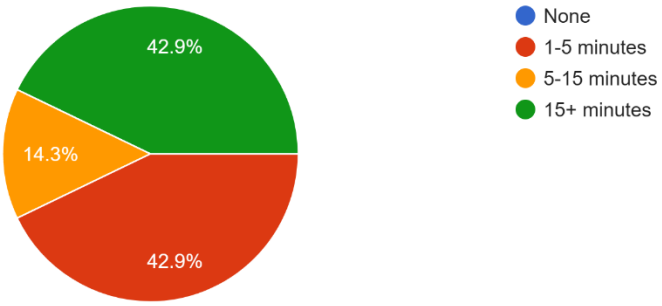
15. Have you ever spent extra time researching ingredients before purchasing?

7 responses



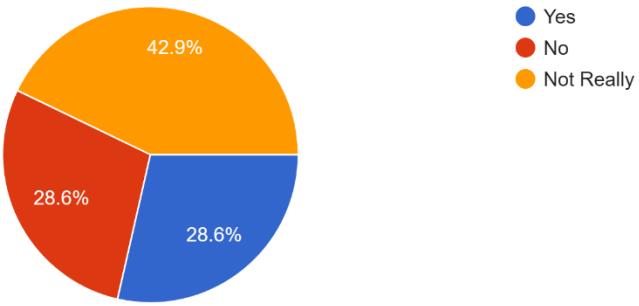
16. Approximately how long do you research before buying something new?

7 responses



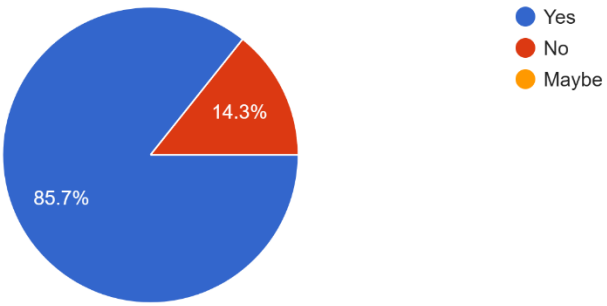
17. Do you feel current tools (Google, TikTok, blogs, etc.) give you personalised advice in forms of advertisements?

7 responses



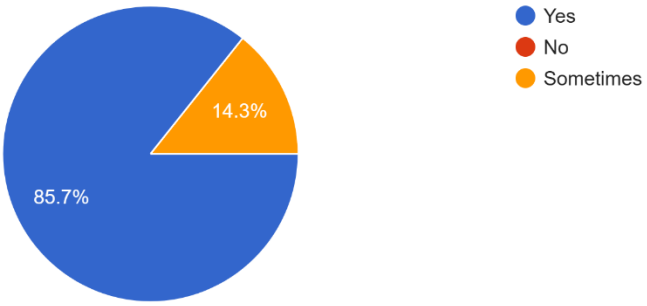
18. Have you ever avoided buying a product because you weren't sure if it was suitable?

7 responses



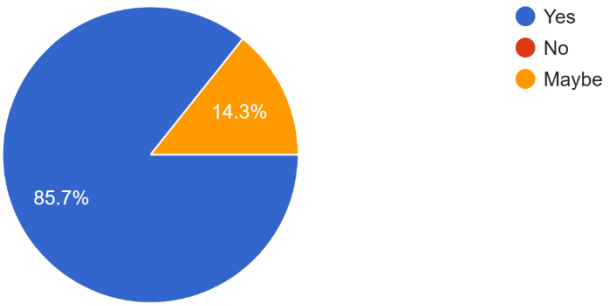
19. Would you say product selection feels overwhelming at times?

7 responses



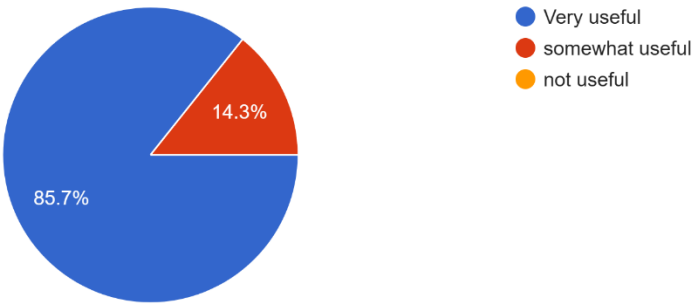
20. Would you find it helpful to have a quicker way to check if a product suits your specific skin needs?

7 responses



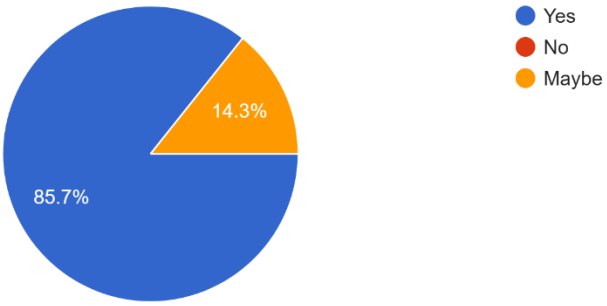
21. Would personalised guidance based on your own skin profile be useful?

7 responses



22. Would you use a digital tool to simplify ingredient checking?

7 responses



23. What would make choosing skincare products easier for you? (Not Required)

7 responses

Clear explaination of ingredients and their origins

A score based system that balances skin needs, quality and price?

Insight on ingredients and why suits certain skin types

if i could answer questions about my skin and you would diagnose a skin type and recommend product that will work

AN APP THAT EXPLAINS INGREDIENTS

Filtering by price range and skin type

24. What frustrates you most about the current process?

7 responses

Unclear ingredients and percentage amount within products

Too many options

Lack of information

too hard to find something that is specifically my skin type, most of the time its close enough after a long time of research but never exactly what I need

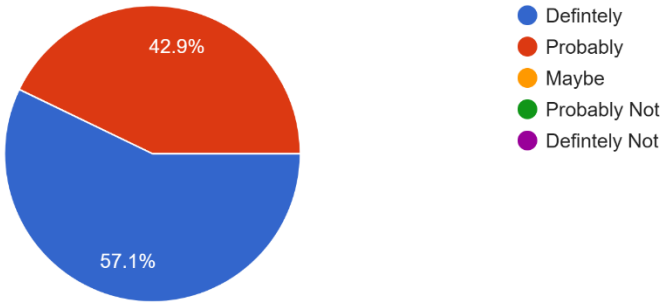
UNCLEAR IF AN AMOUNT OF INGREDIENT IS ENOUGH TO MAKE A DIFFERENCE

High risk as most products are expensive

8. Survey: Product Suitability App – Feature Expectations

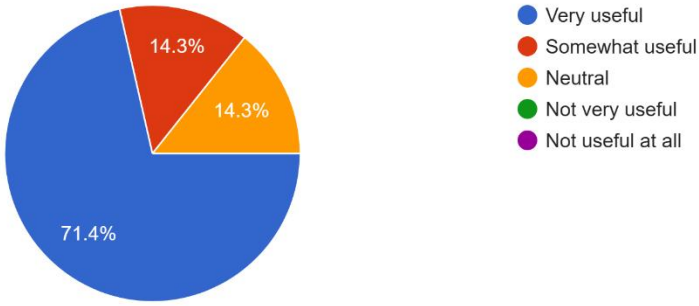
1. Would you use an app that helps determine whether a skincare product suits your specific skin needs?

7 responses



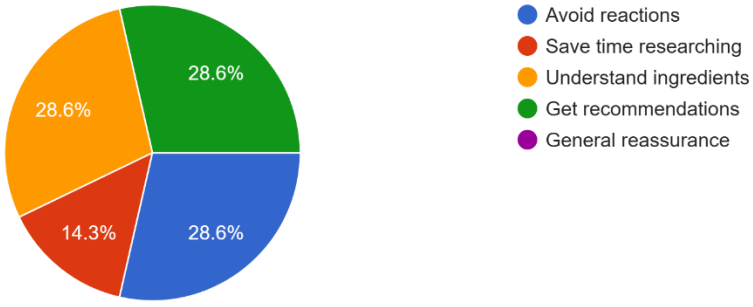
2. How useful would personalised product analysis be to you?

7 responses



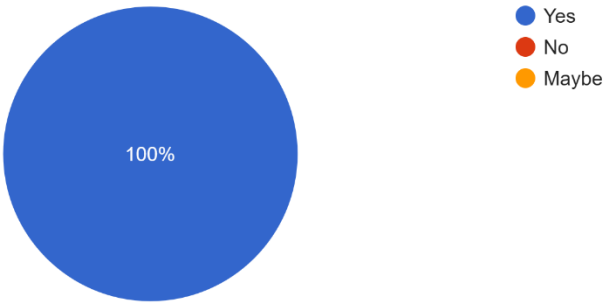
3. What would be the primary reason you would use such an app?

7 responses



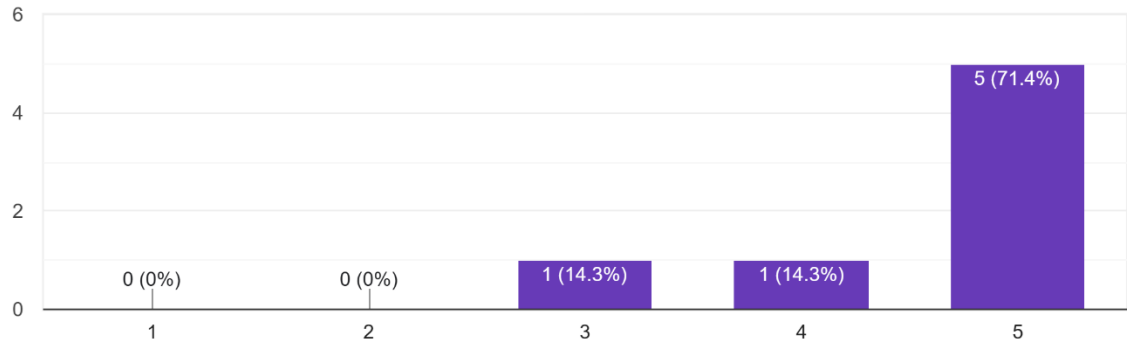
4. Would you like the app to explain why certain ingredients may or may not suit you?

7 responses



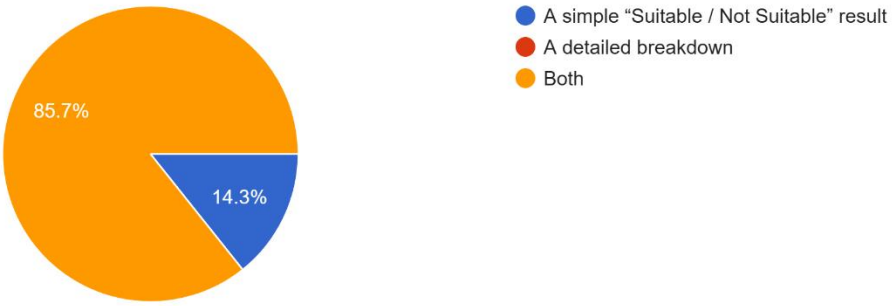
5. How important is ingredient transparency to you? (1–5 scale)

7 responses



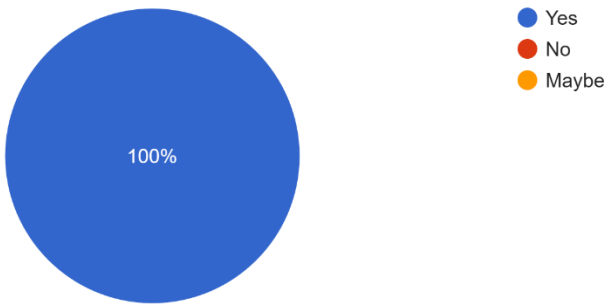
6. Would you prefer:

7 responses



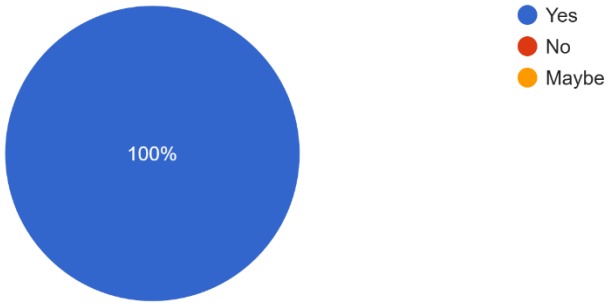
7. Would you be willing to input your skin condition(s) to receive personalised results?

7 responses



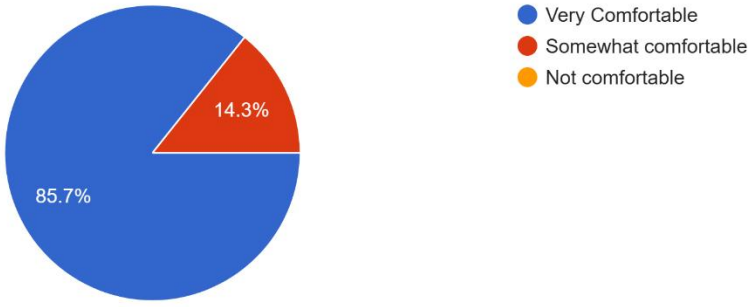
8. Would you input known allergens to improve accuracy?

7 responses



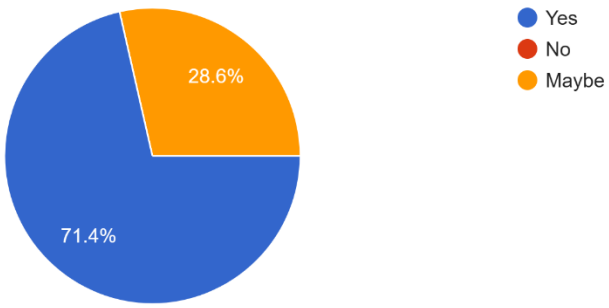
9. How comfortable are you with an app storing your skin profile for better recommendations?

7 responses



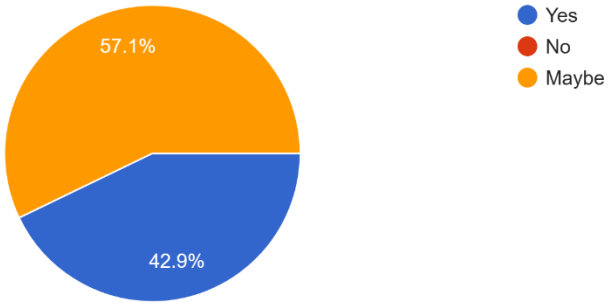
10. Would you like the app to learn from products you liked or disliked?

7 responses



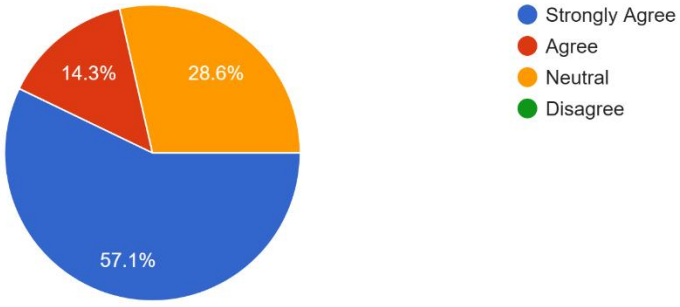
11. Would a simple thumbs up / thumbs down feedback system be sufficient?

7 responses



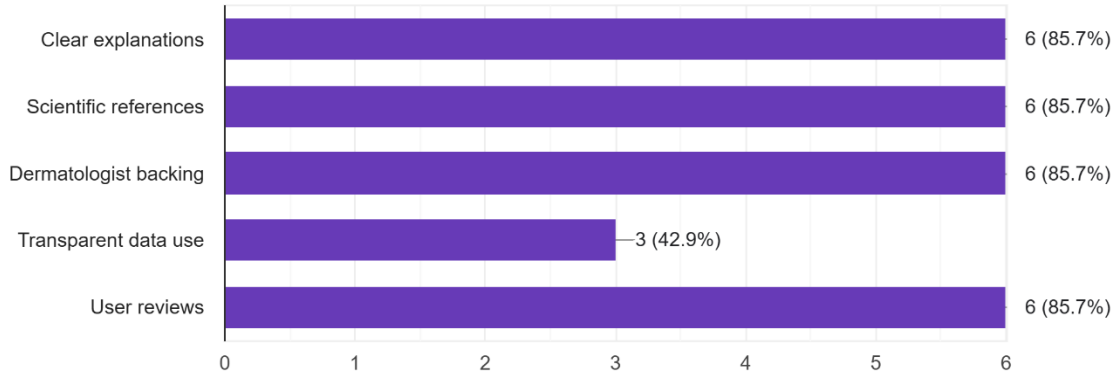
12. Would personalised product recommendations influence your purchasing decisions?

7 responses



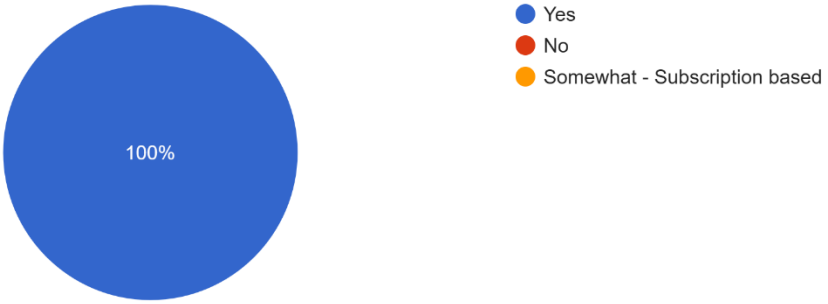
13. What would make you trust such an app?

7 responses



14. Would you download this app if it were free?

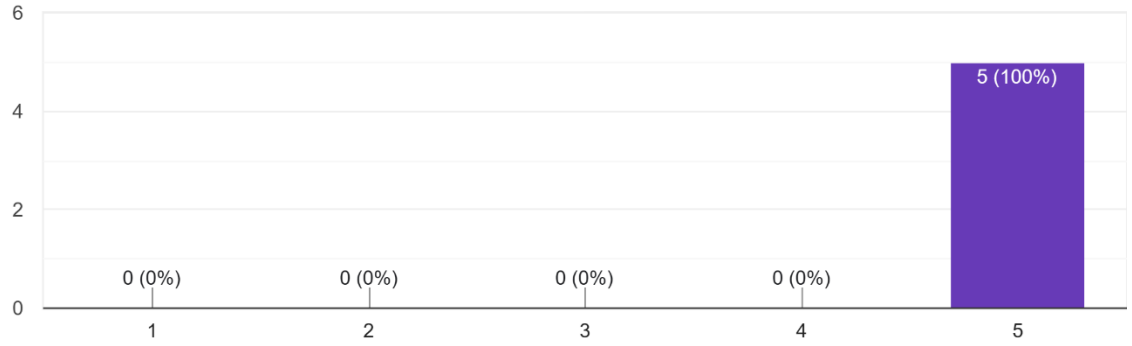
7 responses



9. Survey: User Testing Survey

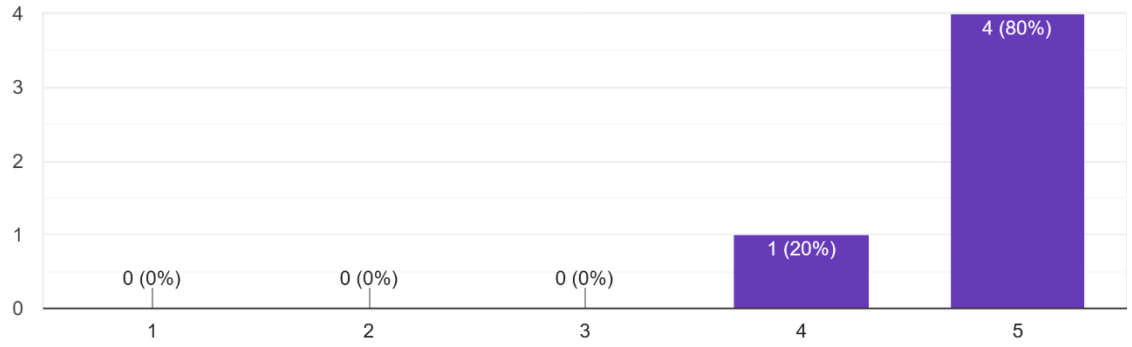
1. How easy was the app to use?

5 responses



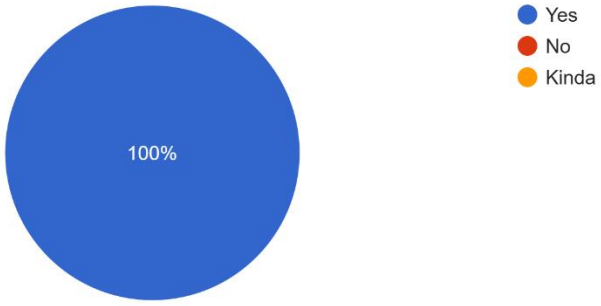
2. I felt confident using the app without help

5 responses



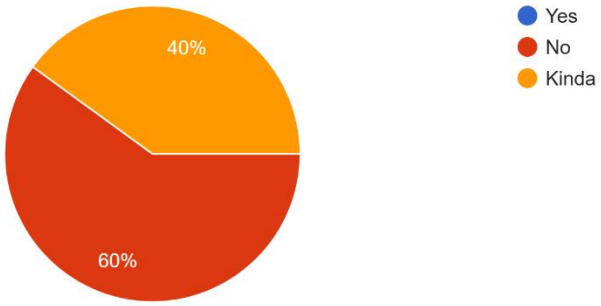
3. Was the app easy to navigate?

5 responses



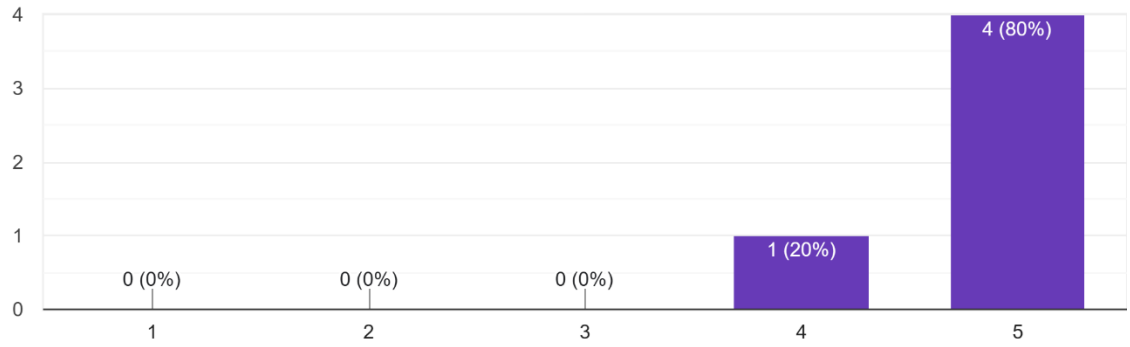
4. Did you experience any lag or slow loading?

5 responses



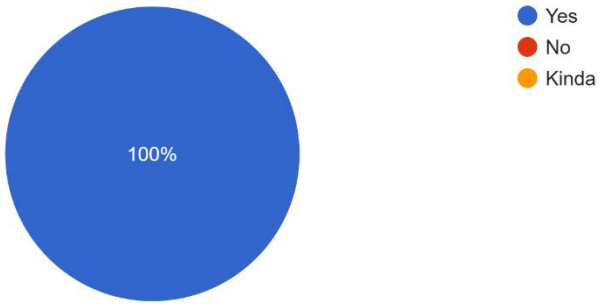
5. The layout and design were clear and understandable

5 responses



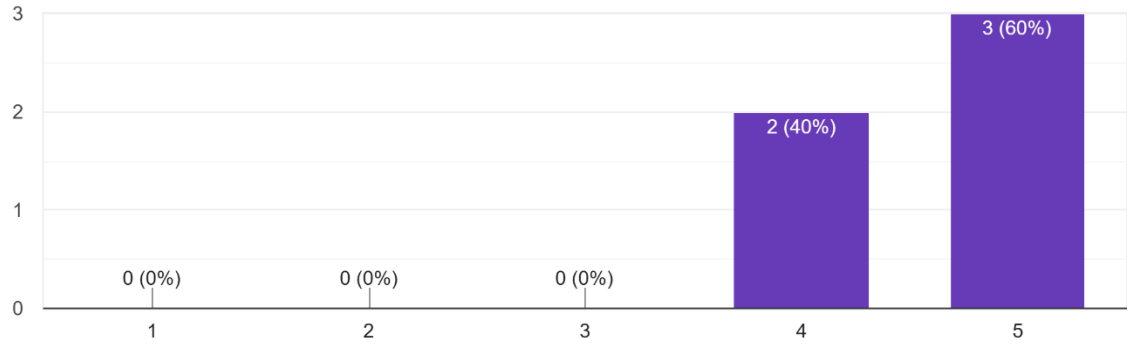
6. Was it easy to scan or upload a product?

5 responses



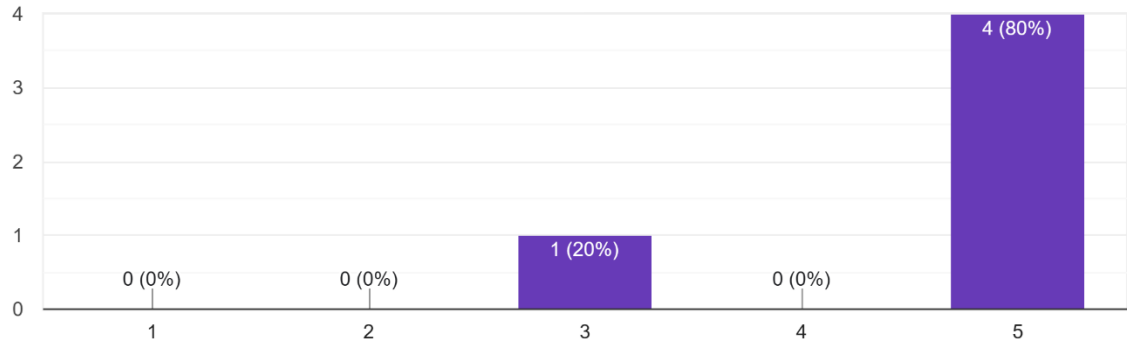
7. The analysis results were easy to understand

5 responses



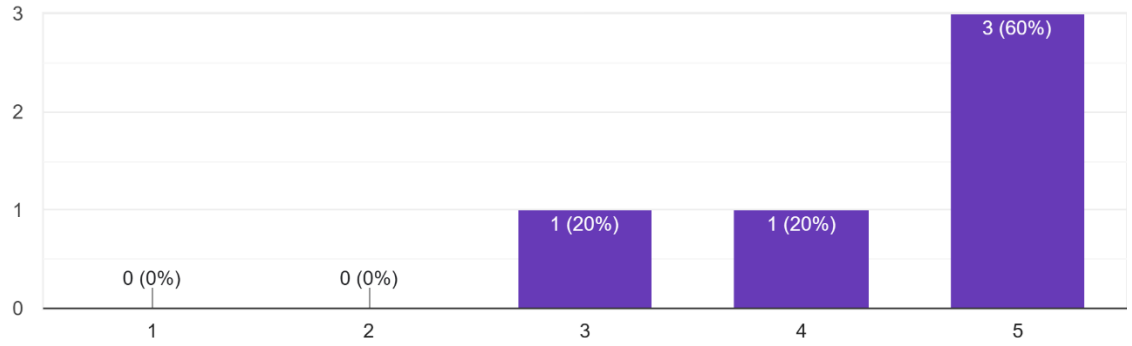
8. I trust the results provided by the app

5 responses



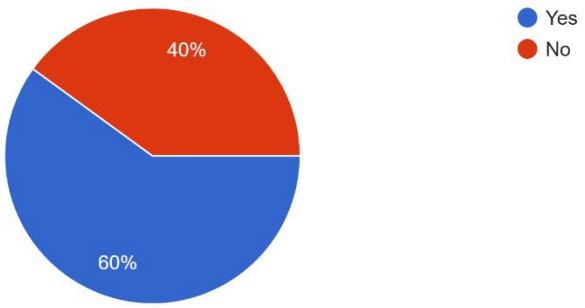
9. How would you rate the speed of the app?

5 responses



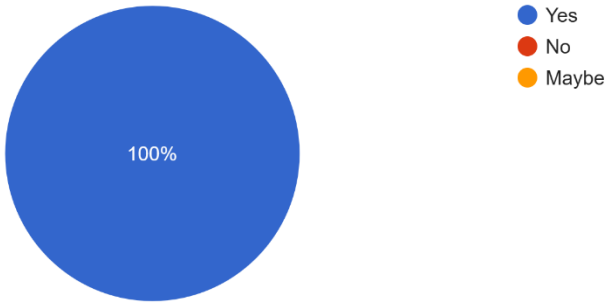
10. Did you encounter any errors?

5 responses



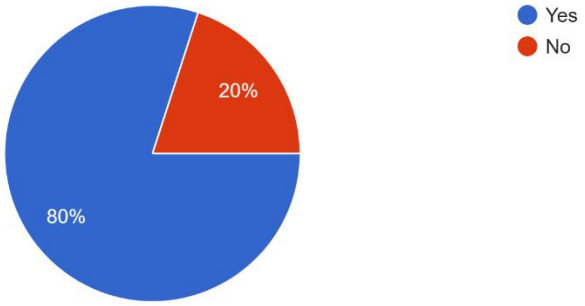
11. Do you think this app would be useful in real life?

5 responses



12. Would you use this app again?

5 responses



13. What did you like most about the app? Optional

5 responses

Clear and easy to understand, there were no questions left unanswered when viewing results of the uploads

startup loading

simple ui, nice design

Super easy to understand and navigate, I felt the flow was very natural. Styling was very visually pleasing. I thought the product analysis was so impressive! I liked the product safety badge and score, and the fact that the information included citations.

preference saving

14. What did you like least about the app? Optional

3 responses

the buttons at the bottom of an android device made some in-app buttons harder to click due to overlay design

Encountered an error upon register but was easily fixed.

inconsistent cosmetics design

15. What improvements would you suggest? Optional

4 responses

If possible to implement a search feature for items without image requirement

less sections

None, very cool app.

use a color paltte for icons

10. Permissions Table

Feature	Permission	Admin	Moderator	User
Users	user:view-all	yes	yes	no
Users	user:view-own	no	no	yes
Users	user:create	yes	no	no
Users	user:update-any	yes	yes	no
Users	user:update-own	no	yes	yes
Users	user:delete-any	yes	yes	no
Users	user:delete-own	no	no	yes
Preferences	preference:view	yes	yes	yes
Preferences	preference:create	yes	no	no
Preferences	preference:update	yes	yes	no
Preferences	preference:delete	yes	no	no
Allergens	allergen:view	yes	yes	yes
Allergens	allergen:create	yes	no	no
Allergens	allergen:update	yes	yes	no
Allergens	allergen:delete	yes	no	no
Conditions	condition:view	yes	yes	yes
Conditions	condition:create	yes	no	no
Conditions	condition:update	yes	yes	no
Conditions	condition:delete	yes	no	no
Profiles	profile:view-all	yes	yes	no
Profiles	profile:view-own	no	no	yes
Profiles	profile:view-own-all	yes	yes	yes
Profiles	profile:create	yes	no	yes
Profiles	profile:update	yes	yes	yes
Profiles	profile:delete	yes	no	yes
Prompts	prompt:view	yes	yes	no
Prompts	prompt:create	yes	no	no
Prompts	prompt:update	yes	no	no
Prompts	prompt:delete	yes	no	no
Products	product:view	yes	yes	yes
Products	product:create	yes	yes	yes
Products	product:update	yes	yes	yes
Products	product:delete	yes	no	no
Evaluation Context	evaluation-context:view	yes	yes	yes
Evaluation Context	evaluation-context:create	yes	yes	yes
Evaluation Context	evaluation-context:update	yes	yes	yes
Evaluation Context	evaluation-context:delete	yes	no	yes
Admin	admin:panel-access	yes	no	no
Admin	role:manage	yes	no	no

Admin	system:settings	**yes**	no	no

11. User Testing Tasks

Task 1: Register and Create Profile

Research Question:	What issues arise from the registration and profile creation?
Scenario:	Imagine you heard about this application from a friend and decided to sign up to become a user.
Task:	Start by registering as a user in the application (please do not use your personal information—sample details will be provided). Once registered, complete your profile using the provided information. Then navigate through the app as you normally would, exploring the available features and content. As you use the app, please verbally mention any issues, confusion, or thoughts you have. When you are finished, navigate to your profile and log out of the application.
Inputs:	**Registration Information:** Email: john@gmail.com First Name: John Last Name: Doe Password: Password123! **Profile Information:** First Name: John Last Name: Doe Age: 32 Condition: Choose Random Preference: Choose Random Allergen: Choose Random Profile: Choose Avatar

Thank you for your participation. We value all your input.

Task 2: Analysing Process

Research Question:	What issues arise when users attempt to analyse an image?
Scenario:	Imagine you were recommended this application from a friend or colleague, and you decided to capture a product.
Task:	Start by logging out of the account if you are currently logged in. Once logged out, please log in using the details provided below (do not use your own personal information). After logging in, navigate through the app as you normally would. Locate the scan or upload feature and capture or upload a product image using the provided example. Allow the app to process the image and generate the ingredient extraction and analysis. Review the results and take note of any warnings or recommendations shown. As you complete this task, please verbally mention any issues, confusion, or thoughts you have. Once finished, log out of the application.
Inputs:	**Login Information:** Email: StephanieJoane@example.com Password: MyPassword321!

Thank you for your participation. We value all your input.

Task 3: Analysing and Viewing History

Research Question:	Are there any issues with viewing history after an analysis.
Scenario:	You've just picked up a new skincare product and want to check if it's safe to use. Use the app as a new user to scan or upload the product image and view the analysis results. After reviewing the result, go to the history section and find your previous scan. Mention anything that is confusing or unclear.
Task:	Confirm you are logged out. From there, log in using the details provided below (do not use your own personal information). Once logged in, navigate through the app and locate the scan feature. Capture or upload a product image and allow the app to process and display the analysis results. Review the outcome shown on the screen. From there, navigate to the history section and locate the scan you just completed. Please verbally state if you come across any issue(s). Once satisfied, proceed to log out.
Inputs:	**Registration Information:** **Login Information:** Email: StephanieJoane@example.com Password: MyPassword321!

Thank you for your participation. We value all your input.

Task 4: Re-Evaluation of History

Research Question:	What issues arise if the user would like to re-evaluate a previous analysis?
Scenario:	You have updated your current allergen list and would like to re-analyse previously 'safe' products to determine whether they remain 'safe' to use.
Task:	Confirm you are logged into the application. Navigate to your profile and update your allergens using the provided information. Once updated, go to the history section, locate the most recently analysed product, and find the re-analyse **banner**. Re-analyse the product and review the updated results. Please verbally state if you come across any issue(s) or notice any differences in the analysis. Once satisfied, proceed to log out.
Inputs:	**Login Information:** Email: StephanieJoane@example.com Password: MyPassword321! **Allergen Update Information:** Allergen: Remove Random and Choose Random

Thank you for your participation. We value all your input.

Task 5: Adding Profile and Analysing

Research Question:	What issues arise with creating additional Profiles
Scenario:	You have been using the application for some time and now want to create an additional profile for a friend, so you can scan and analyse products on their behalf safely.
Task:	Confirm you are logged into the application. Navigate to your profile and create a new profile for your friend using the provided information. Once created, switch to the new profile and use the scan feature to capture or upload a product image. Select both profiles for scanning. Allow the app to analyse the product and display the results. Using the switch profile view results per profile. Please verbally state if you come across any issue(s). Once satisfied, proceed to log out.
Inputs:	**Login Information:** Email: StephanieJoane@example.com Password: MyPassword321! **Profile Information:** First Name: Amy Last Name: Doe Age: (Leave Blank) Condition: Choose Random Preference: Choose Random Allergen: Choose Random Profile: Choose Avatar

Thank you for your participation. We value all your input

12. All Consent Forms

-Matty

Project Title: *Lumière – Cosmetic Ingredient Analysis App*

Principal Investigators: Laura 'Lili' Hofmanova

You are invited to participate in a user test of *Lumière*, a mobile application designed to identify potentially harmful cosmetic ingredients for individuals with specific skin conditions. *Lumière* promotes transparent analysis of cosmetic ingredients and their effects on individuals. The purpose of this test is to gather feedback on the app's usability, features, and overall user experience. Your participation will involve using the application and sharing your thoughts throughout the process, which will be recorded on video. This form outlines the terms and conditions of your involvement in this testing phase.

Participant Information:

- **Participant Name: Matthew Dent**

- **Date of Birth** *(DD:MM:YYYY)*:**16/08/2003 Age: 22**

- **Nationality: Irish**

- **Gender** (mark 'X'):
  - **Female**
  - **X Male**
  - **Non-Binary**
  - **Prefer Not to Say**

- **Contact Information** *(optional)*: **N00220082@iadt.ie**

Consent to Participate:

I, Matthew Dent voluntarily agree to participate in the user test for the beauty app stated above. I understand that my participation is entirely voluntary, and I have the right to withdraw at any time without penalty.

Purpose and Procedure:

I understand that the purpose of this user test is to evaluate the usability and effectiveness of the **Lumière mobile application**.

The procedure will involve:

• Using the app under observation

• Providing feedback on the app's features, design, and functionality

• Allowing the session to be recorded on video for analysis purposes

Confidentiality and Data Usage:

I understand that any information collected during this user test, including video recordings, will be kept confidential. My personal information will not be shared with any third parties and will only be used for the purpose of this user test.

Rights of Participants:

I understand that I have the right to:

• Refuse to answer any questions or perform any tasks that I am uncomfortable with

• Withdraw from the user test at any time without penalty

• Request that any personal data collected during the test be deleted

Use of Recorded Material:

I understand that video recordings of the user test sessions may be used for analysis, research, or promotional purposes related to the sustainable app. However, my identity will be kept confidential, and my personal information will not be disclosed without my explicit consent.

Contact Information:

If I have any questions or concerns about this user test or my participation in it, I can contact at *N00222003@iadt.ie*.

By signing below, I acknowledge that I have read and understood the terms and conditions outlined in this consent form, and I agree to participate in the user test under these conditions.

Participant's Signature: MDent

Date *(DD:MM:YYYY)*:18/04/2026

CONTINUE ON THE NEXT PAGE FOR RECORDING CONSENT.

Recording Consent Form

Thank you for participating in our usability research!
We'll be recording your session so it can be reviewed to better understand your experience and improve the app.
Please read the statement below and sign where indicated.

Your insights help us improve the experience for individuals managing skin conditions and making informed cosmetic choices.

I understand that my usability test session will be recorded. I grant *Lumière* research team permission to use this recording for internal use only, for the purpose of improving the designs being tested.

Please indicate your consent by ticking the appropriate options below:

☑ Video Recording

Signature: MDent

Print your name: MATTHEW DENT

Date *(DD:MM:YYYY)*: **18/04/2026**

Project Title: *Lumière – Cosmetic Ingredient Analysis App*

Principal Investigators: Laura 'Lili' Hofmanova

You are invited to participate in a user test of *Lumière*, a mobile application designed to identify potentially harmful cosmetic ingredients for individuals with specific skin conditions. *Lumière* promotes transparent analysis of cosmetic ingredients and their effects on individuals. The purpose of this test is to gather feedback on the app's usability, features, and overall user experience. Your participation will involve using the application and sharing your thoughts throughout the process, which will be recorded on video. This form outlines the terms and conditions of your involvement in this testing phase.

Participant Information:

- **Participant Name:** Isabel Ripoll
- **Date of Birth** *(DD:MM:YYYY)*: 02 / 02 / 1977 **Age:** 49
- **Nationality:** Colombian
- **Gender** (mark 'X'):
  - ✓ **Female**
  - ○ **Male**
  - ○ **Non-Binary**
  - ○ **Prefer Not to Say**
- **Contact Information** *(optional)*:_____

Consent to Participate:

I, [Participant's Full Name], voluntarily agree to participate in the user test for the beauty app stated above. I understand that my participation is entirely voluntary, and I have the right to withdraw at any time without penalty.

Purpose and Procedure:

I understand that the purpose of this user test is to evaluate the usability and effectiveness of the **Lumière mobile application**.

The procedure will involve:

- **Using the app under observation**
- **Providing feedback on the app's features, design, and functionality**
- **Allowing the session to be recorded on video for analysis purposes**

Confidentiality and Data Usage:

I understand that any information collected during this user test, including video recordings, will be kept confidential. My personal information will not be shared with any third parties and will only be used for the purpose of this user test.

Rights of Participants:

I understand that I have the right to:

- **Refuse to answer any questions or perform any tasks that I am uncomfortable with**
- **Withdraw from the user test at any time without penalty**
- **Request that any personal data collected during the test be deleted**

Use of Recorded Material:

I understand that video recordings of the user test sessions may be used for analysis, research, or promotional purposes related to the sustainable app. However, my identity will be kept confidential, and my personal information will not be disclosed without my explicit consent.

Contact Information:

If I have any questions or concerns about this user test or my participation in it, I can contact at *N00222003@iadt.ie*.

By signing below, I acknowledge that I have read and understood the terms and conditions outlined in this consent form, and I agree to participate in the user test under these conditions.

Participant's Signature: Isabel Pipo

Date *(DD:MM:YYYY)*: 02 / 02 / 1977

CONTINUE ON THE NEXT PAGE FOR RECORDING CONSENT.

Recording Consent Form

Thank you for participating in our usability research!
We'll be recording your session so it can be reviewed to better understand your experience and improve the app.
Please read the statement below and sign where indicated.

Your insights help us improve the experience for individuals managing skin conditions and making informed cosmetic choices.

I understand that my usability test session will be recorded. I grant *Lumière* research team permission to use this recording for internal use only, for the purpose of improving the designs being tested.

Please indicate your consent by ticking the appropriate options below:

- ✓ Voice Recording
- ○ Video Recording

Signature: Isobel Ripoll

Print your name: ISOBELL RIPOLL

Date *(DD:MM:YYYY)*: 02/02/1977

Project Title: *Lumière – Cosmetic Ingredient Analysis App*

Principal Investigators: Laura 'Lili' Hofmanova

You are invited to participate in a user test of *Lumière*, a mobile application designed to identify potentially harmful cosmetic ingredients for individuals with specific skin conditions. *Lumière* promotes transparent analysis of cosmetic ingredients and their effects on individuals. The purpose of this test is to gather feedback on the app's usability, features, and overall user experience. Your participation will involve using the application and sharing your thoughts throughout the process, which will be recorded on video. This form outlines the terms and conditions of your involvement in this testing phase.

Participant Information:

- **Participant Name:** EDVARDAS LIAUKUS
- **Date of Birth** *(DD:MM:YYYY)*: 18/12/2001 **Age:** 24
- **Nationality:** Lithuanian
- **Gender** (mark 'X'):
  - ○ **Female**
  - ✓ **Male**
  - ○ **Non-Binary**
  - ○ **Prefer Not to Say**
- **Contact Information** *(optional)*:_____

Consent to Participate:

I, [EDVARDAS Name], voluntarily agree to participate in the user test for the beauty app stated above. I understand that my participation is entirely voluntary, and I have the right to withdraw at any time without penalty.

Purpose and Procedure:

I understand that the purpose of this user test is to evaluate the usability and effectiveness of the **Lumière mobile application**.

The procedure will involve:

- **Using the app under observation**
- **Providing feedback on the app's features, design, and functionality**
- **Allowing the session to be recorded on video for analysis purposes**

Confidentiality and Data Usage:

I understand that any information collected during this user test, including video recordings, will be kept confidential. My personal information will not be shared with any third parties and will only be used for the purpose of this user test.

Rights of Participants:

I understand that I have the right to:

- **Refuse to answer any questions or perform any tasks that I am uncomfortable with**
- **Withdraw from the user test at any time without penalty**
- **Request that any personal data collected during the test be deleted**

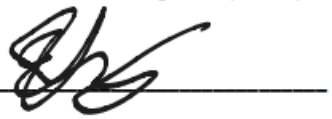
Use of Recorded Material:

I understand that video recordings of the user test sessions may be used for analysis, research, or promotional purposes related to the sustainable app. However, my identity will be kept confidential, and my personal information will not be disclosed without my explicit consent.

Contact Information:

If I have any questions or concerns about this user test or my participation in it, I can contact at *N00222003@iadt.ie*.

By signing below, I acknowledge that I have read and understood the terms and conditions outlined in this consent form, and I agree to participate in the user test under these conditions.

Participant's Signature: 

Date *(DD:MM:YYYY)*: 19/04/2026

CONTINUE ON THE NEXT PAGE FOR RECORDING CONSENT.

Recording Consent Form

Thank you for participating in our usability research!
We'll be recording your session so it can be reviewed to better understand your experience and improve the app.
Please read the statement below and sign where indicated.

Your insights help us improve the experience for individuals managing skin conditions and making informed cosmetic choices.

I understand that my usability test session will be recorded. I grant *Lumière* research team permission to use this recording for internal use only, for the purpose of improving the designs being tested.

Please indicate your consent by ticking the appropriate options below:

- ◉ Voice Recording
- ○ Video Recording

Signature: 

Print your name: EOVAROAS

Date *(DD:MM:YYYY)*: 21/04/2026

Project Title: *Lumière – Cosmetic Ingredient Analysis App*

Principal Investigators: Laura 'Lili' Hofmanova

You are invited to participate in a user test of *Lumière*, a mobile application designed to identify potentially harmful cosmetic ingredients for individuals with specific skin conditions. *Lumière* promotes transparent analysis of cosmetic ingredients and their effects on individuals. The purpose of this test is to gather feedback on the app's usability, features, and overall user experience. Your participation will involve using the application and sharing your thoughts throughout the process, which will be recorded on video. This form outlines the terms and conditions of your involvement in this testing phase.

Participant Information:

• **Participant Name:** Angelina Morris

• **Date of Birth** *(DD:MM:YYYY)*: 02 / 11 / 2003 **Age:** 22

• **Nationality:** Irish

• **Gender** (mark 'X'):

- [x] **Female**
- [ ] **Male**
- [ ] **Non-Binary**
- [ ] **Prefer Not to Say**

• **Contact Information** *(optional)*:_____

Consent to Participate:

I, [Angelina Morris], voluntarily agree to participate in the user test for the beauty app stated above. I understand that my participation is entirely voluntary, and I have the right to withdraw at any time without penalty.

Purpose and Procedure:

I understand that the purpose of this user test is to evaluate the usability and effectiveness of the **Lumière mobile application**.

The procedure will involve:

- **Using the app under observation**
- **Providing feedback on the app's features, design, and functionality**
- **Allowing the session to be recorded on video for analysis purposes**

Confidentiality and Data Usage:

I understand that any information collected during this user test, including video recordings, will be kept confidential. My personal information will not be shared with any third parties and will only be used for the purpose of this user test.

Rights of Participants:

I understand that I have the right to:

- **Refuse to answer any questions or perform any tasks that I am uncomfortable with**
- **Withdraw from the user test at any time without penalty**
- **Request that any personal data collected during the test be deleted**

Use of Recorded Material:

I understand that video recordings of the user test sessions may be used for analysis, research, or promotional purposes related to the sustainable app. However, my identity will be kept confidential, and my personal information will not be disclosed without my explicit consent.

Contact Information:

If I have any questions or concerns about this user test or my participation in it, I can contact at *N00222003@iadt.ie*.

By signing below, I acknowledge that I have read and understood the terms and conditions outlined in this consent form, and I agree to participate in the user test under these conditions.

Participant's Signature: Angelina Morris

Date *(DD:MM:YYYY)*: 21 / 04 / 2026

CONTINUE ON THE NEXT PAGE FOR RECORDING CONSENT.

Recording Consent Form

Thank you for participating in our usability research!
We'll be recording your session so it can be reviewed to better understand your experience and improve the app.
Please read the statement below and sign where indicated.

Your insights help us improve the experience for individuals managing skin conditions and making informed cosmetic choices.

I understand that my usability test session will be recorded. I grant *Lumière* research team permission to use this recording for internal use only, for the purpose of improving the designs being tested.

Please indicate your consent by ticking the appropriate options below:

- [x] Voice Recording
- [ ] Video Recording

Signature: Angelina Morris

Print your name: ANGELINA MORRIS

Date *(DD:MM:YYYY)*: 21/04/2026

Project Title: *Lumière – Cosmetic Ingredient Analysis App*

Principal Investigators: Laura 'Lili' Hofmanova

You are invited to participate in a user test of *Lumière*, a mobile application designed to identify potentially harmful cosmetic ingredients for individuals with specific skin conditions. *Lumière* promotes transparent analysis of cosmetic ingredients and their effects on individuals. The purpose of this test is to gather feedback on the app's usability, features, and overall user experience. Your participation will involve using the application and sharing your thoughts throughout the process, which will be recorded on video. This form outlines the terms and conditions of your involvement in this testing phase.

Participant Information:

- **Participant Name:** George Diffley
- **Date of Birth** *(DD:MM:YYYY)*: 22/08/2005 **Age:** 20
- **Nationality:** IRISH
- **Gender** (mark 'X'):
  - ○ **Female**
  - X **Male**
  - ○ **Non-Binary**
  - ○ **Prefer Not to Say**
- **Contact Information** *(optional)*: georgediffley05@gmail.com

Consent to Participate:

I, GEORGE DIFFLEY, voluntarily agree to participate in the user test for the beauty app stated above. I understand that my participation is entirely voluntary, and I have the right to withdraw at any time without penalty.

Purpose and Procedure:

I understand that the purpose of this user test is to evaluate the usability and effectiveness of the **Lumière mobile application**.

The procedure will involve:

- **Using the app under observation**
- **Providing feedback on the app's features, design, and functionality**
- **Allowing the session to be recorded on video for analysis purposes**

Confidentiality and Data Usage:

I understand that any information collected during this user test, including video recordings, will be kept confidential. My personal information will not be shared with any third parties and will only be used for the purpose of this user test.

Rights of Participants:

I understand that I have the right to:

- **Refuse to answer any questions or perform any tasks that I am uncomfortable with**
- **Withdraw from the user test at any time without penalty**
- **Request that any personal data collected during the test be deleted**

Use of Recorded Material:

I understand that video recordings of the user test sessions may be used for analysis, research, or promotional purposes related to the sustainable app. However, my identity will be kept confidential, and my personal information will not be disclosed without my explicit consent.

Contact Information:

If I have any questions or concerns about this user test or my participation in it, I can contact at *N00222003@iadt.ie*.

By signing below, I acknowledge that I have read and understood the terms and conditions outlined in this consent form, and I agree to participate in the user test under these conditions.

Participant's Signature: George Duffy

Date *(DD:MM:YYYY)*: 25 / 04 / 2026

CONTINUE ON THE NEXT PAGE FOR RECORDING CONSENT.

Recording Consent Form

Thank you for participating in our usability research!
We'll be recording your session so it can be reviewed to better understand your experience and improve the app.
Please read the statement below and sign where indicated.

Your insights help us improve the experience for individuals managing skin conditions and making informed cosmetic choices.

I understand that my usability test session will be recorded. I grant *Lumière* research team permission to use this recording for internal use only, for the purpose of improving the designs being tested.

Please indicate your consent by ticking the appropriate options below:

- ◉ Voice Recording
- ○ Video Recording

Signature: George Diffley

Print your name: GEORGE DIFFLEY

Date *(DD:MM:YYYY)*: 25/ 04/ 2026

13. Transcripts (Isobel, Edvardas)

Isobel 0:06
Ohh, Jesus Christ!
The.
Prod second, I will turn my mic off.
Okay, task three, analysing and viewing history. Oh, that's already okay.
Logging in.
Look in.
L.
Okay, once logged in, navigate through the app and locate the scan feature, which I think is just this big button. Camera access needed. Yes, grant permission, allow.
Okay, capture, upload a product dimension, allow the app to process, display the analysis result, read the outcome shown on the screen. Okay, here is the product I want to test.
Okay, and product found, confirm the product for continuing. Yes, this, is it? Yes, it is.
Oh, it was, yes.
Yes.
Yeah.
Avoid? Okay, why am I avoiding it?
Okay.
preservatives, skin sensitizers, and can cause allergic contact dermatitis.
Cause that per skin reactions in individual sensitive skin, but it's not like that is cruel if you're vegan.
They matched profile settings, and...
Okay.
Okay, one now.
Navigate to the history section and locate the scan you just completed.
The big button for history.
And this is the scam.
Okay, I can see all of the information again.
Can go back, that's nice.
Please replace it if you come across any issues. Okay, no issues found.
And I should log out.
Okay.
Okay, task for re-evaluation of history.
Confirm you are logged into the application. Okay, I'm not, so I will log in.
Login.
Okay, nice. But now...

Navigate to your profile and update your allergens using the provided information.
Profile.
Other gents.
Whichever the tester would like.
Okay, let's see. I'm allergic to...
Let's say...
This one?
This one, maybe.
Let's see.
Updated, go to the history section and locate previously analysed product marked as caution.
Or make like.
History section.
All wait.
Yeah.
History.
This is a void.
OK, reanalyze the product and review the updated results.
Here, we evaluate.
OK, now it's still a void.
Ten out of 100.
By this result, because the nice.
Yep, so no issues found. Everything is nice and...
Easy to navigate.

Laura Hofmanova (Student) stopped transcription

Edvardas Liakus 0:24
Shake the ****.
I already have an account, so I am going to log in.
Ohh.
Must have misspelt it.
OK.
So, I want to create a profile for my friend.
Amy Toyle.
Okay, cheque her condition.
It's acne.
And she does, I think, have a L.
She does have a preference for vegan products.
And then we're gonna save a picture, and then...

We go with the save.
Continue.
And then we switch into her.
And then we're gonna look for a scan.
So gonna scan.
Let's scan this.
Just, ohh, I need it.
That was quite quick.
Let's continue and select both of our profiles.
And see that analyzation.
Okay, just gonna read through. It does say caution for the product, so...
Prod it does look legit, but it comes to analysis that...
not as safe as you would think so.
That's interesting.
OK, that's.
Useful information.
So, I come to log out.
Oh.
I'm just gonna look at this.
Settings and I log out.

Laura Hofmanova (Student) stopped transcription

14. Download Script

Lumiere Client Application

This is the mobile client for Lumiere, developed using Expo (React Native).
The application enables users to scan cosmetic products, analyse ingredients, and manage personalised skincare insights.

1. Getting Started

Follow the steps below to run the application locally using Visual Studio Code and a mobile device.

2. Prerequisites

Ensure the following are installed:

- **Node.js (version 18 or higher recommended)**
- **npm (version 9 or higher)**
- **Visual Studio Code**
- **Expo Go on your mobile device**

To install Expo Go on your phone:

- **Open the App Store (iPhone) or Google Play Store (Android)**
- **Search for "Expo Go"**
- **Download and install the application**

Note:
You do not need to create an account to use Expo Go. If prompted to sign in, you may skip this step.

3. Installation

Open Visual Studio Code and use the terminal to clone the repository and install dependencies:

- **git clone https://github.com/lchka/lumiere.git**
- **cd lumiere/client**
- **npm install**

4. Environment Configuration

Uncomment the a .env file inside the /client directory.

This ensures the application connects to the hosted backend.

5. Running the Application

In the VS Code terminal, start the development server:

npx expo start

A QR code will appear in the terminal or browser.

6. Running on Your Phone

1. **Open the Expo Go application on your phone**
2. **Select the option to scan a QR code**
3. **Scan the QR code shown in your terminal or browser**
4. **The application will open automatically on your device**

Ensure your phone and computer are connected to the same WiFi network.

7. API Configuration

The application connects to a backend server.

The backend is hosted on Render:
https://majorproject-1toi.onrender.com/

Important notes:

- **The hosted server may go to sleep after a period of inactivity (approximately 15 minutes)**
- **Before testing, open the link above in a browser to reactivate the server**
- **Initial requests may take longer while the server is waking up**

If running locally, update the API URL in:

/client/src/config/api.ts

Example:

export const API_URL = "http://YOUR_IP:3000";

Additional notes:

- **Do not use localhost, as it will not work on a mobile device**
- **Use your computer's local IP address (for example, http://192.168.1.10:3000)**
- **Ensure the backend server is running before testing**

8. Troubleshooting

Application does not load on phone

- **Ensure your phone and computer are connected to the same WiFi network**
- **Restart the Expo server:**

npx expo start -c

Dependency issues

rm -rf node_modules
npm install

Camera or permissions issues

- **Ensure camera permissions are enabled on your device**
- **Restart Expo Go if permissions were recently granted**

15. RAG Papers Used

- A daily regimen of a ceramide-dominant moisturising cream and cleanser restores the skin permeability barrier in adultswith moderate ecz https://onlinelibrary.wiley.com/doi/epdf/10.1111/dth.14970
- A review on the role of moisturizers for atopic dermatitis- https://synapse.koreamed.org/upload/synapsedata/pdfdata/0253apa/apa-6-120.pdf
- Effects of tocotrienol on aging skin A systematic review - https://www.frontiersin.org/journals/pharmacology/articles/10.3389/fphar.2022.1006198/full
- Epidermal Barrier Dysfunction in Atopic Dermatitis - https://revista.spdv.com.pt/index.php/spdv/article/view/1405
- Lipid-based formulations in cosmeceuticals and biopharmaceuticals- https://link.springer.com/content/pdf/10.1186/s41702-020-00062-9.pdf
- protective effects of cosemetic ingredients - https://link.springer.com/content/pdf/10.1007/s11033-021-06918-5.pdf
- The Role of Moisturizers in Addressing Various Kinds of Dermatitis: A Review - https://www.clinmedres.org/content/15/3-4/75?utm_campaign=DSL_clinique-smart-night-custom-repair-feuchtigkeitscreme-bewertungen
- Role of Topical Emollients and Moisturizers in the Treatment of Dry Skin Barrier Disorders - https://link.springer.com/article/10.2165/00128071-200304110-00005
- Safety Assessment of Palm-Derived Ingredients as Used in Cosmetics - https://journals.sagepub.com/doi/full/10.1177/10915818241237797
- Study of the protective effects of cosmetic ingredients on the skin barrier, based on the expression of barrier-related genes and cytok - https://link.springer.com/article/10.1007/s11033-021-06918-5
- The Effect of Ceramide-Containing- https://cdn.mdedge.com/files/s3fs-public/Document/September-2017/081010087.pdf
- The Enigma of Bioactivity and Toxicity of Botanical Oils for Skin Care - https://www.frontiersin.org/journals/pharmacology/articles/10.3389/fphar.2020.00785/full

16. Links to Hosted Server and Hosted Admin Panel

Note: The server is hosted on Render's free tier and may experience occasional downtime due to inactivity.

Server: https://majorproject-1toi.onrender.com

Admin Panel: https://majorproject-1-73tu.onrender.com

Admin Login Details:

Email: admin@example.com

Password: Admin123!

17. Jest and SuperTest Tests

Integration

Module	Method	Endpoint	Status	Expected Result
Allergen	POST	/api/allergens	401	Unauthenticated
Allergen	POST	/api/allergens	403	No permission
Allergen	POST	/api/allergens	400	Invalid body
Allergen	POST	/api/allergens	201	Creates allergen
Allergen	GET	/api/allergens	200	Returns all
Allergen	GET	/api/allergens/:id	200	Returns allergen
Allergen	GET	/api/allergens/:id	404	Not found
Allergen	PATCH	/api/allergens/:id	200	Updates
Allergen	PATCH	/api/allergens/:id	404	Not found
Allergen	DELETE	/api/allergens/:id	200	Deletes
Allergen	DELETE	/api/allergens/:id	404	Not found
Allergen	GET	/api/allergens/profile/:profileId	200	Returns profile allergens
Allergen	GET	/api/allergens/profile/:profileId	404	Not found
Auth	POST	/api/auth/register	201	Registers user
Auth	POST	/api/auth/login	200	Logs in
Auth	POST	/api/auth/google	200	Google login
Auth	POST	/api/auth/logout	200	Logout
Auth	GET	/api/auth/me	200	Returns user
Auth	GET	/api/auth/me	401	Unauthenticated
Prompt	POST	/api/prompts	201	Creates prompt
Prompt	GET	/api/prompts	200	Returns all
Prompt	GET	/api/prompts/:id	200	Returns prompt

Module	Method	Endpoint	Status	Expected Result
Prompt	GET	/api/prompts/:id	404	Not found
Prompt	PATCH	/api/prompts/:id	200	Updates
Prompt	DELETE	/api/prompts/:id	200	Deletes
User	GET	/api/users/me	200	Returns current user
User	GET	/api/users	200	Returns all
User	GET	/api/users/:id	200	Returns user
User	GET	/api/users/:id	404	Not found
User	PATCH	/api/users/:id	200	Updates
User	DELETE	/api/users/:id	200	Soft deletes
User	DELETE	/api/users/force/:id	200	Permanently deletes
User	PATCH	/api/users/restore/:id	200	Restores the user

- Unit

Module	Test Cases / Assertions
Allergen	Create allergen successfully · Get all · Get by ID · Update · Delete
Allergen	Get allergens for profile · Validate input · Reject invalid update
Allergen	RBAC: Allow admin · Forbid normal user
Auth	Register successfully (token) · Fail if role not found
Auth	Login successfully (token) · Fail with invalid credentials
Auth	Verify token · Throw error for invalid token · Logout success
Auth	Validate register input · Reject invalid login input
Condition	Create successfully · Get all · Get by ID · Update · Delete · Profile fetch
Condition	Validate input · Reject invalid update · RBAC: Admin vs User
Evaluation Context	Create · Evaluate product · Admin: Get all

Module	Test Cases / Assertions
Evaluation Context	Get by ID · Profile · Product · User
Evaluation Context	Update · Re-evaluate · Delete · Validate & Reject inputs
Evaluation Context	RBAC: User / Mod / Admin permissions
Gemini / AI	Extract product from image · Fallback to ingredients-only · Web fallback
Gemini / AI	Error handling (missing file) · Evaluate & Normalize result
Preference	Create · Get all · Get by ID · Update · Delete · Profile fetch
Preference	Validate input · Reject invalid update · RBAC: Admin vs User
Product	Create · Get all (Admin) · Get by ID (Ownership)
Product	Update & Delete (with ownership) · Validate input · Empty update
Product	RBAC: User / Mod / Admin · Forbid unauthorized delete
Profile	Create · Get all · Get by ID · Get by User ID · Current Profile
Profile	Update · Delete · Validate input · Empty update
Profile	RBAC: User / Mod / Admin · Forbid restricted actions
Prompt	Create · Get all · Get by ID · Update · Delete
Prompt	Validate input · Empty update · RBAC: Admin only
User	Get current · Get all · Get by ID · Update by ID · Update own
User	Soft delete · Force delete · Restore user
User	RBAC: Admin or Self · Deny other modifications · Validate input
Weather / UV	Get UV index (valid coords) · Error handling (invalid coords)
Weather / UV	Handle service failure via next()