Design Project

Department of Computer Science

Design Report

Authors:

Ivan Mitev s2690543

Fabio Allous s2345080

Rares Moldovan s2711656

Alex Toma s2836033

Danil Vorotilov s2485788

Supervisor:

João Rebelo Moreira

Research Group:

Semantics, Cybersecurity & Services (SCS)

October 30, 2024

UNIVERSITEIT TWENTE.

Abstract

This report is part of the "M11 Design Project" course at the University of Twente. A Digital Product Passport (DPP) application was implemented on the request of the Semantics, Cybersecurity and Services (SCS) department of the Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS). In the scope of this project a DPP is described in the following manner *"The main idea of the DPP is to enable the implementation of a Digital Twin of any product, tracking all associated information of the product, from the idea to the consumer and the end of life."* and our goals were to (1) make a literature review of existing implementations, (2) implement a prototype and (3) validate the prototype through a use case. This implementation is designed to be a consumer product which focuses heavily on the storage and security of varying data. The system is viable in the exploration of ease of use, database architecture and implementation and it also serves as an argument for the importance of unified data formats throughout the EU. At the end of the report, extending the system with further capabilities is discussed. This possibility for extension was also kept in mind during the design process as in its essence a DPP is meant to be implemented on a very large scale.

Chapter 1

INTRODUCTION

The adoption of the European Green Deal, aiming to reach carbon neutrality by the year 2050, has led the European Union to propose more and increasingly ambitious green projects and regulations. One such regulation is Ecodesign for Sustainable Product Regulation (ESPR) (Protokol, 2024) and is focused on improving the circularity and energy and environmental sustainability metrics of products within the European economy. In order to meet its ambitious energy efficiency goals, the EU needs to optimize the use of materials and energy throughout both the manufacturing process and life cycle of consumer products.

The Digital Product Passport is a solution designed to tackle this challenge by ensuring information about manufactured goods is readily available to individuals and businesses alike. Calls for proposals began under the Digital Europe Programme in 2023. The objectives included: *"To enable sharing of key product related information that are essential for products' sustainability and circularity"*,*"help consumers in making sustainable choices"*, *"allow authorities to verify compliance with legal obligations"*, and to encourage *"circular value retention and optimisation"* (HaDEA, 2023). The expected outcomes, on the other hand, were that a DPP would be deployed and validated in at least two value chains, generating recommendations for deployment in other value chains. It was also expected that the project would explore the needs of *"standardization and specifications to ensure interoperability, security and acceptance by all the stakeholders"* (HaDEA, 2023).

Despite the novelty of the DPP as a concept, numerous implementations have already been developed and independent research has been done on the topic. These implementations mainly target the sectors of electronics, batteries and textiles and have focused either on use in the production and distribution process or after the product is sold to a customer. Two of the available passports stand out, CIRPASS and Circularise, and they will be explored further in the Domain chapter.

Currently, consumers struggle with obtaining full and accurate information about products they own. Researching unadvertised qualities such as origin of materials, possibly hazardous composition, or checking if an item has health and safety or energy consumption certificates

entails endless scrolling on the internet. On top of that, manufacturers don't always provide this information on their websites and customers are required to get their information from unreliable third party websites.
Businesses also face a serious, although intrinsically different, information access problem. As oftentimes multiple companies are involved in a product's manufacturing, distribution, servicing and recycling processes, these companies have to share product information between themselves. However, no standardized data models have been universally adopted. This leads to a waste of resources gathering the characteristics of a product and possibly an even bigger waste if the information gathered is incorrect.

Through the implementation of a Digital Product Passport which ensures authentic, readily-available and secure data, customers will be able to make easier, more educated decisions about which products to buy. Additionally, businesses will be able to optimize their processes, leading to a more efficient and profitable processes for them and a more environmentally friendly product lifecycle. Both of these improvements will lead to progress towards the European Commission's goals for the DPP project.

The rest of the report is structured in the following way. The second chapter of this report will be an overview of the domain of this project. The third chapter will be a literature review of various research examining DPPs. The fourth chapter will be a review of already existing solutions. The fifth chapter will be concerned with the functional and nonfunctional requirements of the project. The sixth chapter will justify and explain the global and architectural choices made by the team. The seventh chapter will explain and justify implementation choices such as used technologies and the design of the database. The eighth chapter focuses on the tests conducted in order to ensure the quality of the system. The ninth chapter explores how the system can be built upon in the future. The ten and final chapter provides the team's evaluation on the development process. After that, a bibliography and Appendices are provided.

Chapter 2

DOMAIN ANALYSIS

This chapter will consist of explaining and analyzing the DPP domain. This step is necessary to understand the core concepts, terminology, processes and requirements within the DPP domain. By understanding these elements we make sure that the development and design of our system aligns with the industry standards and regulations. This section will explore the DPP domain, focusing on its purpose, key stakeholders, essential data elements, compliance requirements, and the challenges and opportunities involved in implementing DPPs effectively

2.1 Overview of the DPP domain

2.1.1 Purpose and scope

DPPs are digital records of products containing their lifecycle information such as their origin, the materials used, recycling options and more. They are important for industries like electronics, textiles, and automobiles due to their purpose of supporting sustainability goals and achieving the requirements of regulations that focus on product transparency and traceability.

2.1.2 Core Concepts:

DPPs give information about a product's lifecycle by tracking it from raw materials to disposal, which affirms transparency at every stage. In terms of traceability and data interoperability, DPPs ensure data flows smoothly across the supply chain providing stakeholders with accurate available information. Products also offer unique identifiers like QR codes or barcodes, linking them to their digital passports, easing the process of data retrieval for all stakeholders.

2.2 Key Stakeholders and Data Elements

Stakeholders can be classified into three categories.
Manufacturers and suppliers. They provide data on production methods, material sources and environmental impact.
Consumers. They access information on the product like sustainability, repair options and recycling instructions.

Regulatory bodies. They ensure compliance with environmental and privacy standards.

The data elements found in a DPP are usually data on material composition, manufacturing processes, recycling details and repair instructions. This information is made available in order to make consumers aware of the origins of the product and helps them make informed decisions and support goals for sustainability.

2.3 Processes and Compliance Requirements

2.3.1 Data Collection and Sharing

DPPs collect data at multiple points in the product life cycle. It starts at material extraction, continuing through manufacturing, distribution, use and disposal. Unique data is available at each stage, making a comprehensive record of the product's journey. Controlling who accesses the information and data sovereignty principles ensure that only the correct people see the information, maintaining the privacy and security of the data. Data sharing is maintained by interoperability standards, which allow various systems and organizations to work seamlessly with DPPs. These standards make sure stakeholders can access reliable information.

2.3.2 Compliance with Regulations

The DPP needs to align with current international regulations such as the Ecodesign for Sustainable Products Regulation (ESPR) and General Data Protection Regulation (GDPR). Under ESPR, DPPs help to improve product transparency by providing lifecycle information such as material composition, environmental effects, and end-of-life options, therefore promoting a circular economy. DPPs use defined formats like GS1, and ISO to provide interoperability and uniform data sharing across regions and platforms.

GDPR compliance requires DPPs to collect and share a minimal amount of data by gathering only essential information and by using strong security measures like data encryption to protect any data being linked. This strategy ensures that DPPs meet both sustainability and data protection standards, promoting transparency and accountability throughout a product's lifecycle.

2.4 Challenges and Opportunities

2.4.1 Challenges

In terms of data complexity, managing and standardizing data across complex supply chains can be challenging. Interoperability to make sure that the DPPs are compatible across any system and have seamless data exchange across any region requires adherence to standards like ISO. Additionally, making sure that the data is private and secure is also critical. DPPs need good privacy measures, such as encryption and access control, to ensure that only authorized parties can view sensitive information and that data integrity is maintained throughout the product life cycle.

2.4.2 Opportunities

Successful implementations of DPPs on a smaller scale have already proven to be of significant economic value. A product lifecycle which is easier to track leads to a more efficient manufacturing process, lower material expenditures and overall higher profit margins for companies(Circularise, 2023). This, implemented on the scale of the EU can lead to lower prices for electronics, batteries and textiles across the whole union. Moreover, DPPs can aid with other, not so easily quantifiable goals such as persuading Europeans into more sustainable consumerism habits. When it is commonplace knowledge exactly how much electricity a product uses or how many trees have been felled to manufacture it, societal pressure leads to people steering away from buying it. DPPs create a wide range to influence business and society, both immediately and in the future, in subtle and in obvious ways.

Chapter 3

RESEARCH INTO DPP LITERATURE

This chapter will be concerned with previous research into DPP technology. It will specifically focus on the relevant information needed for a successful passport, methods of data collection and storage. It will also examine the functional and non-functional requirements established by previous researchers. This research is crucial when making design and architectural choices for this project

3.1 A Reference Architecture for Digital Product Passports at Batch Level to Support Manufacturing Supply Chains

The first paper(Wiesner et al., 2024) focuses on the analysis of potential requirements of a DPP in the context of a large European manufacturer's supply chain. It establishes a set of information requirements and proposes a reference architecture with a proof of concept design. Although this paper examines DPPs from a different perspective, it still leads to key takeaways which are relevant in designing and developing a user-focused DPP solution.

3.1.1 Data Insights

This paper recognizes that during a product's lifecycle, different stakeholders are concerned with different levels of data granularity and presents it as a gap for DPP development. While a manufacturer might be concerned with how many tonnes of metal or thousands of computer chips are required to produce a batch of a specific phone model, consumers are interested in the weight of a single unit of that phone or its computer chip's model and performance. On the other hand, there is information, such as carbon emissions, which is viable to both parties.

Another gap the paper explores is *"Missing Product Data"*(Wiesner et al., 2024). As a manufacturer gathers data from in-house processes as well as third parties, such as material providers, this can lead to missing information about the product. The paper suggests "interfaces for data exchange"(Wiesner et al., 2024) such as blockchain or Data Spaces when this problem arises from a third party. This concern can be translated to consumer-targeted DPPs as they require information from multiple parties such as the manufacturer, distributors and companies which perform maintenance and repairs.

3.1.2 Architectural Insights

In its section on reference architecture, the paper proposes a set of design principles, some of which are also viable in developing a consumer-focused DPP solution. The architecture suggests using standardized, REST-based APIs to send data to and from the database in a JSON format. It also calls for the use of standardized data models and the use of a microservice architecture, separating the app's functionalities in loosely coupled services. On top of that, it advocates for a role-based access control policy to ensure certain functionalities are protected and an event-driven architecture to aid with data management. All of these suggestions are also useful and widely employed in the development of mobile applications which have to handle vast amounts of data.

3.2 Thesis from graduating University of Twente students

This section explores two thesis, presented by graduating University of Twente students, Luis Hamad-Chetan and Malina Wiesner. Wiesner is also a co-author of the paper discussed in the previous section and in fact the two papers share a lot of key points and conclusions. However, it is still crucial to analyze her master thesis as it is far more detailed and gives deeper insight into potential information requirements. Hamad-Chetan and Wiesner's papers are broadly concerned with the capabilities and requirements needed to successfully implement DPP solutions and although Hamad-Chetan's paper explores Environmental Management Systems in more detail, its analysis of DPPs is relevant.

3.2.1 Information requirements

Both papers recognize the need to extend the data requirements of DPPs beyond the *"identified legal background of data disclosures for suppliers and manufacturers"* (Hamad-Chetan, 2024). This data is separated into seven information categories: "Product identification", "Company information", "Functional and technical specifications", "Material and composition information", "Product design and service", "Usage and repair history", "Environmental indicators and certification" (Wiesner, 2023). These categories respect the requirements of the EU's Ecodesign for Sustainable Products Regulation (ESPR) regulation and also transcend beyond the domain of supply chain management and manufacturing. Each category will be further explored in the section on information requirements for this specific project

3.2.2 Non-functional and operational requirements

Hamad-Chetan's paper proposes a number of functional and operational requirements. These include a unique identifier, which links the individual passport to the product it represents and also a range of broader, more abstract requirements. They are security and accessibility, which mostly relate to the way data is stored and managed, as well as flexibility and interoperability, which is concerned with the ability of the DPP to adapt to an ever-changing set of regulations and stakeholders interested in the system.

3.3 International Data Spaces (IDS)

3.3.1 What is IDS

International Data Spaces, or IDS, is an environment "where organizations can share data while retaining full control over its use" (Steinbuß et al., 2022). IDS can be seen as a kind of digital highway that enables secure data exchange between companies. It was created by the International Data Spaces Association, or IDSA, in 2017. They are the governing body responsible for creating these regulations. One of the key benefits of IDS is control. The data owners have complete authority over their data. They can decide who can access it and under what conditions. One of the key features of IDS is that it doesn't require companies to store data centrally. There's no central storage, which means the data stays with the owner, maintaining privacy and security.

3.3.2 Why use IDS in DPP

Because a DPP requires collecting and distributing varying data across different countries with varying legislation and companies with different data formats, any relevant solution requires a functional Data Space environment in order to be widely adopted. An IDS provides unmatched capabilities for data authenticity, availability and most importantly, standardization and sovereignty. Any EU-wide initiative on standardizing product information will be doomed to fail if the safe handling, collection and distribution of this information is not guaranteed. Moreover, European and sometimes even local legislation needs to be respected. An IDS provides a solution to all of these problems(Steinbuß et al., 2022). In order to funnel data in and out of an IDS, each company uses what is called a "data connector". Data connectors "extract data, sometimes periodically, from one or more upstream data sources and land that data in another database" (Incorta, 2022). Although outside of the scope of this project, the ability to easily connect to a data connector is extremely important for a DPP.

Chapter 4

RESEARCH INTO DPP IMPLEMENTATIONS

In this chapter, two currently available DPP implementations are explored, Circularise and CIRPASS. For each implementation, its technologies and use case are explored as well as its limitations. After that, the DPPs will be assessed and their relevance to this project will be discussed. This research is crucial as it forms the basis of many design and architectural decisions made during this project.

4.1 Circularize

Circularise is a Dutch company, established in 2016, with the goal of establishing *"end-to-end supply chain traceability"*(Circularise, 2023). They aim to achieve this through their DPP solutions in the automotive, batteries, construction, electronics, metals and plastics industries. Circularise's DPPs are mainly offered as a B2B solution for companies looking to optimize their production and recycling processes and avoid potential fines for using prohibited materials. However, they also offer a consumer-targeted DPP to their clients. Their product offers Life Cycle Assessment(LCA) and Product Carbon Footprint calculations through the use of reliable primary data which leads to accurate sustainability claims for their clients. Additionally, they use blockchain technology to store their data and ensure data availability, authenticity and security.

Circularise's DPP implementation provides data very similar to the types discussed in the previous section. In their own words, they include data relating to *"Product Identification, Materials, Product Design, Technical Specifications, Product Lifecycle, Product Maintenance, User Manuals, Warranty Information, Reuse and Recycling Information, Energy Recovery, Waste, Hazardous Waste and Environmental Information"*(Daphne & Stretton, 2023). The company also. Another thing they focus on in their consumer-facing DPP is its public accessibility through the scanning of barcodes and authenticity, which as mentioned above is guaranteed through blockchain. Since many companies which buy their consumer-facing DPP are also clients for their other passport solutions, Circularise puts heavy emphasis on its selective data-sharing practice which enables their clients to deeply customize the data they share with consumers.

4.2 CIRPASS

CIRPASS was a project, funded by the European Commission (EC), which aimed to "*lay the ground for the gradual piloting and deployment of a standards-based DPP"*(CIRPASS, 2024) and ran between 2022 and 2024. After that, CIRPASS2 was started with the goal of demonstrating functioning DPPs through pilot deployments in the electronics and textile industries, among others. One of the many deliverables of CIRPASS was a framework for designing functional and compliant product passports(Archer et al., 2024). The project arrived at these results by conducting a detailed assessment on 32 European-level DPP initiatives, so they can be taken as a form of general consensus on how to design a DPP according to current industry and European standards.

One of the design practices CIRPASS recommends is using a decentralized approach to store data since it maximizes the robustness, resilience and security of the data. By letting the creator hold and manage the data, the potential lack of security or misuse is avoided. A disadvantage of this approach is that it requires an authority on data formatting, storage and distribution to ensure each creator or distributor manages their data properly. Another practice recommended by CIRPASS is that the DPP systems should be Product Centric, focusing on the individual product. The reason for this is that in the circular economy model, the product and its lifecycle are at the center of interest. The only way to minimize the material and financial waste throughout the product lifecycle is by having complete and reliable information about it. Finally, in their framework CIRPASS suggests that a DPP should provide an *"interoperable playing field"(Archer et al., 2024)*. It views a product passport as carrying all B2B, B2C and B2G information in a single data space, although being accessible through different interfaces. Therefore, interfaces covering those three use cases should be easily implementable and compatible with each other, hopefully through the use of standardized technology. These design practices have been used as a foundation for numerous DPP implementations, including those of CIRPASS 2.

Chapter 5

REQUIREMENT ANALYSIS

This section will delve into the proposed system. This includes an explanation of the requirements, use case diagrams, user stories, mock-up designs, technologies used, testing and documentation.

5.1 Requirements Elicitation

The first step towards a clearly defined set of functional and non-functional requirements is elicitation. Throughout the initial meetings with the project's supervisor, João Moreira, the context and use cases of this specific DPP implementation were explained to the group. João supplied a very useful set of research papers and already available solutions, which were discussed in the two previous chapters. On top of that, he provided the team with the device for which the DPP will be created, a Google Nest Protect 2nd generation smoke and carbon monoxide alarm. Using all of these resources, use case diagrams and user stories were created, which were then used as the basis for a set of functional and non-functional requirements

5.2 Use Case Diagram

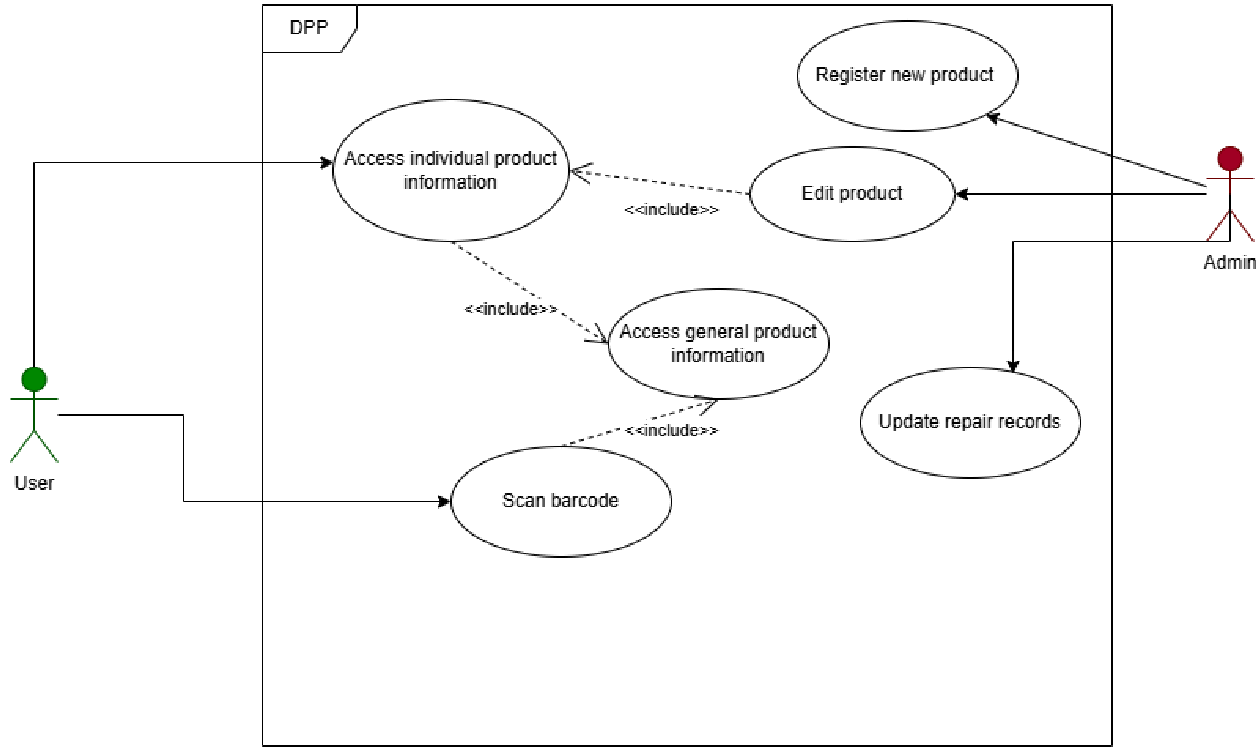


Figure 5.2

Throughout the course of the requirements elicitation phase, the use case diagram in Figure 5.2 was defined. The target product is a consumer-facing DPP, so most of the system is focused on the needs and wants of a universal type of user. However, a special type of user with administrative privileges is still required in order to manage and create data about the products on an individual and general level. Therefore, according to the separation of concern design principle, it was decided that the two types of users would exist separately and independently from each other. The only function they would share is accessing product information as this is also needed for auditing.

All other diagrams created during this phase and used in the analysis of requirements can be found in Appendix A.

5.3 Stakeholder Requirements

This section presents a list of user stories which are then rephrased as functional requirements.

5.3.1 User

1. As a user I want to access product information about products I own.

A user has to be able to access information about products they own, such as manufacturer, carbon emissions, environmental and health certificates, production materials, etc.

2. As a user I want to be able to register a new account with credentials of my choice.
   A user has to be able to create a personal account with a password and email they desire. Using this approach instead of having premade accounts raises users' trust in the system and makes it more familiar and easier to use.

3. As a user I want to be able to sign in to my own personal account.
   A user has to have their own personal account, where they can view the inventory of the products they own. This account needs to be secured via a password and inaccessible to any malicious actors who don't know the password.

4. As a user I want to be able to sign out of my account.
   A user has to be able to sign out of their account. They require this feature for security reason in case a malicious actor gains access to their device

5. As a user I want to access the repair history of products I own.
   A user has to be able to access all of the registered repairs for any product they have claimed ownership of. Those repairs are especially common in electronics or other products with a long lifespan and possibility to be resold.

6. As a user I want to be able to access the last active location of electronic devices.
   A user has to be able to trace the last known location of any device they own if it supports such tracking capabilities. Those devices are generally electronics and their last location is recorded through a history of bluetooth pairings or through their last connection to a mobile or wifi network. Any additional method of collecting this data can be seen as too invasive

7. As a user I want to be able to know when my electronic device was last turned on.
   A user has to be able to track the last known position of any device they own which supports such tracking capabilities. This information is gathered in the same way as the information in the last story.

8. As a user I want to access general information about products I am interested in buying.
   A user has to be able to examine the available passport information about a product they are interested in buying, either in a physical store or online. When a user wants to do this, they scan the barcode of the product, which contains a unique ID, and the information is presented to them.

9. As a user I want to view the individual product passport of a product I own.
   A user has to be able to view an individual product's DPP. This lets the user track information which is specific to this single unit of the product model, such as date of manufacture and repair history. This action is also viable for reselling products with long life-cycles such as electronics.

5.3.2 Admin

1. As an admin I want to be able to view information about all products in the database.
   Admins need to be able to view all products in the database to track repairs, locations and for general data management purposes.

2. As an admin I want to be able to update repair records for specific products.
   Admins need to be able to issue repairs. As repairs on products need to be tracked and monitored, users with non-administrative privileges shouldn't be able to access this feature for data authenticity reasons

3. As an admin I want to be able to edit information about products.
   Admins need to be able to edit the information about products. This can include general product information such as new certifications or updates to the user manual or individual information.

4. As an admin I want to be able to create new products.
   Admins need to be able to create new products in the database. When a company produces a new batch of any given product or rolls out a new model, they will ask the admin to add them to the database.

5. As an admin I want all of my special privileges to be inaccessible to normal users
   Admins have responsibilities such as data maintenance and addition of new data which are crucial to the business of the manufacturers, distributors and repair shops they represent. Therefore, all functions specific to the admin need to be accessible only via a secure password or a special administrator account.

5.4 System Requirements

The requirements for the system were set according to the MoSCoW scale. The MoSCoW scale includes four categories, features the system must have, those the system should have, those the system could have and those the system won't have

5.4.1 Must have

1. The system must have a mechanism to register new accounts.
   The mechanism used is AWS Cognito which provides quick, easy and most importantly secure registration through an Amazon Web Services back-end. After a user inputs their email and desired password, they receive a code through their email. After they input the code, the registration is complete and they are given the opportunity to sign in.

2. The system must have a mechanism to login with an existing account.
   The mechanism used is AWS Cognito which provides a login service through accounts created in the registration procedure. A user can either choose to register or login when they input their credentials. If they already have an account, they choose login and are redirected to the home screen. This should require no additional steps other than inputting the credential and clicking a single button

3. The system must display a table of unit-specific details about any individual product owned by the user.
   After a user accesses their product inventory, they must be able to click on a specific product and after that must be redirected to an individual product screen. This screen must display further details about the product. This feature is accessible within five button clicks from the home screen.

4. The system must have a logout feature.
   The mechanism used is AWS Cognito. When a user decides they have finished using the application, they have the ability to end their session. In order to access their account, they need to input their credentials again. This feature is accessible within one button click from the home screen.

5. The system must have a mechanism for adding new products.
   An admin must be able to add new products to the database. As manufacturers are always coming up with new products, this feature is crucial for meeting the use case requirements of consumer-facing DPP. This feature is accessible within two button clicks from the home screen.

6. The system must separate features available to all users and features available only to admins.
An admin's privileges must be protected through a password. Normal users shouldn't be able to update the records of the database as ensuring data authenticity is one of the goals of the DPP system.

7. The system must display general product information about the model of a product.
One of the user's use cases for the system is accessing general information about products they want to buy. Therefore, a user must be able to access this information even about products they don't own. This should be accessible within one button click from the home screen.

5.4.2 Should have

1. The system should have a mechanism which allows for the editing of already existing products.
In some cases new information is received about products which are already in the system. This can include certifications or altered energy efficiency ratings. An admin can change this information through their account. As this is a rare occurrence, the time/effort it takes to edit records is not a valid measurement, so this requirement is judged on a pass or fail basis.

2. The system should display details about any repairs made to an individual product.
Each individual unit of a product has its own lifecycle. Two phones of the same model can go through a different number of repairs after being sold. Therefore, a user is concerned with the repair history of their individual product. This functionality is not critical to the DPP system but it aids in the achievement of numerous stakeholder goals. It should be accessible within three button clicks from the home screen.

3. The system should have a mechanism for adding repair records through the application
When a product is being repaired, the repair shop is required to publish a new update to the product's repair record. While this can be done directly through the back-end database, it should be the responsibility of the administrator and they should be able to do it from the application. This feature should be available within two button clicks of the home screen.

4. The system should have a barcode-scanning function which returns general product information.

This requirement accompanies number 8 from the "must" section and makes its use easier. For the sake of user convenience, general product information should be accessed without having to manually input any general product identifiers. This should be done with a barcode as advised and implemented by already existing systems. This should be accessible within one button click of the home screen.

5. The system should have a mechanism for users to retrieve information about an individual product.
   Sometimes, even without owning a product, the user wants to access individual information about this product via serial number. This requirement accompanies requirement 3 from the must-have section. This should be accessible within one button click of the home screen.

6. The system should have capabilities to store a record of the last known traces of a product.
   Some products have tracing capabilities such as last known location or the last time they were connected to a network. Information about their traceability should be stored and presented in the individual product information section. This should be accessible within five button clicks of the home screen.

5.4.3 Could have

1. The system could display a complete history of past locations and times of products being turned on.
   Some users may own very expensive devices which are constantly being moved, such as heavily specified electronics or drones. They would benefit from being able to know the historical whereabouts of their equipment. This would also be useful to business owners who want to prevent misuse of company property. However, this is not crucial to the project as it is concerned with only a single Google Nest device.

2. The system could require a two factor authentication when registering new accounts.
   Some users doubt the security and trustworthiness of newly developed applications. Therefore, to ensure stricter security when registering and to build trust with first-time users, a two factor authentication system could be implemented where the user receives an email with a verification code. To ensure a pleasant user experience this code will be sent within thirty seconds of pressing the associated button.

5.4.4 Won't have

1. The system won't have separate accounts for repair shops, distributors and manufacturers.
   In a fully fleshed out DPP, there would be separate accounts for repair shops, distributors and manufacturers. However, as this implementation is focused on the consumer-side of DPPs, all of these accounts have been combined into the admin account.

Chapter 6

GLOBAL AND ARCHITECTURAL DESIGN

In this chapter the global and architectural design choices of the system are introduced, explained in detail and justified. A high-level overview of the whole system is presented and both the preliminary and final view and functionality of the pages and the back-end is discussed.

6.1 Global Design Choices

The development of a Digital Product Passport is heavily dependent on the specific use-case of the passport. This specific system is consumer-first and is heavily focused on a positive user experience and easy learnability. All of the design choices described below are made with this in mind. Moreover, the scope of the project, which is creating a DPP for a single item (a Google Nest unit) from the IoT Cyberlab, was also considered. Therefore, some design choices were made to make the implementation easier for the case of having only one product. However, possible ways to extend this to a more general solution are discussed.

6.2 Mock-up designs

After the initial meetings with the supervisor, in which the team collected research resources and possible requirements, the team developed the mock-up design above. This was done using Figma and the mock-up was presented to João. Through the mock-up, the team discovered that more pages were necessary for the final product. The final design of the pages remained similar with any changes being made after consultation with the supervisor, based on advice received during one of multiple peer-review sessions or because of technical difficulties with the implementation. The mock-up designs can be found in Appendix B.

The pages defined in the mock-up were:

- a home screen - presenting the choice to either sign-in or register
- a registration/sign-in screen - used to sign-in or register
- a product inventory - a screen which would be shared between an admin and a user. However, while an admin would be able to view all products, a user would be able to view only their own products

- an individual product screen - presenting key information, logically divided into sections, which can be extended by clicking the corresponding section
- A screen to add repairs - only accessible to admins

6.3 Pages

This section provides a detailed overview of the application's screens.

6.3.1 Intro, Register, Login and Verification Code Screens

The intro screen is the first screen of the application. It presents 2 key options for user interaction: "Sign Up" and "Log in". To get started, users must create an account. This is done by pressing the "Sign Up" button. They need to use an email address and create a password. The password needs to be 8 characters long, must contain at least 1 capital letter and a special character. In addition, users will receive a 6-digit code which needs to be added to the verification screen. Upon successful creation of an account or login, users are redirected to the home screen.

6.3.2 Home and Product Screens

The application serves 2 types of users: normal user and admin. Once logged in into the application, the user is able to scan the barcode of a desired product on the home screen. The "Scan a barcode" button triggers the built-in camera application of the user's device. Once scanned, the user is forwarded to the product screen. The product screen consists of all general information about the scanned product(See Chapter 7.3.1).Moreover, the user can also see the last scanned product. The home screen has a "Logout" button which allows the user to log out of the application.

6.3.3 Admin, Register a product and Repair Screens

Compared to the normal user, admins have the authority to manage the product-related activities such as product creation and modification, and repair management. Admins can create and update individual products, specifying the corresponding fields of the product(See Chapter 7.3.2 for all the attributes linked to the individual product). Moreover, admins can manage the repairs of each of the products(See chapter 7.3.3 for all the attributes linked to the repairs of a product).

6.3.4 IndividualProductScreen or Display Product Screen

This screen is accessed by individual users after they enter a serial number matching a product in their possession. Upon clicking the button below the TextField in the home screen, which is indicated by the "Display product" text, the user is then taken to a page where they can view the details of their product, listing details such as purchase date, last known date at which the product was known to be spotted, the last known location where it was spotted, as well the serial number and global trade identification number of the product the user owns.

6.4 Architectural Design Choices

The application was designed according to the Model-View-ViewModel system, or MVVM, which is the main architectural design system used in the field of mobile applications. In this system, the Model deals with the data and business logic, the View and the ViewModel represent the presentation logic, or what the user is able to see. Additionally, the ViewModel represents the connection between the Model and the View, enabling the transfer of data in between the two for retrieval or insertion into a database along with the display of such data via the View to the user. For those unfamiliar with the Android or IoS ecosystem, it would appear that we are in fact using the MVC, or Model-View-Controller model, but we are not. The core difference will be explained below, along with a description of the files in the system that are a part of each component, namely View, Models, and ViewModels.

The most significant difference between the MVC model and the MVVM model, which was used, is the fact that in the MVC model there is tight data-binding between the View and the ViewModel. Moreover, unlike the case of the MVC model where the controller has to update the view manually, in the context of MVVM the view is not constrained solely by its data binding to the viewmodel but can update itself. On top of that, the MVVM model offers enhanced flexibility in terms of its data binding potential, namely the fact that one viewmodel can bind to several views, or one view can be integrated with several viewmodels. Furthermore, the MVVM design pattern also offers much clearer separation of concerns. All of this has also been discussed in recent research, in a comparison between the performance of the two design patterns(Wilson et al., 2022).

Moving on to the structure of the application itself, the View layer is made up of the 10 screens described in the previous section, along with helper functions, namely custom composable elements created from base composable elements, such as buttons, text fields, rows or columns. These have the purpose of displaying the application's UI and enabling the user to interact with the interface.

Next up, the ViewModel layer consists of 2 classes, namely the DataViewMod.kt class and the AuthenticationViewMod.kt class. Each of these have separate roles, and they do not interact with one another. The AuthenticationViewMod is responsible for the navigation between pages within our app, and, as the name would suggest, it is also responsible for ensuring a smooth authentication for the users. The DataViewMod is responsible for the retrieval of data requested by the user via the View when an event is triggered, or the insertion of data from the View to the Firestore database, upon the same conditions being met, with the help of the FirebaseService.kt file. The AuthenticationViewMod.kt file contains functions responsible for calling the Amplify Service.kt file, which will be discussed in the upcoming paragraph

Finally, the Model layer is composed of 4 different files, namely the Models.kt, the UIStates.kt file, the FirestoreService.kt file and lastly the AmplifyService.kt file. The Models.kt file stores the Models used to manipulate the database objects and their retrieval along with their insertion, comprising a total of 3 data classes, each modeling an instance of the IndividualProduct table, the Product table and the Repair tables in our Firestore database.

On the other hand, the UIStates file has 6 data classes, 3 of which have the purpose of filling the data requested by the user in the View upon their retrieval via the Model, or acting as an intermediate step before enabling the creation of a full object for either its insertion or retrieval from the database. The other 3 allow for the remembrance of states within TextFields in the login, authentication and register screens, and communicating This is implemented via the remembering of states in each of the screens that make use of the Firestore database.

The Firestore Service.kt file is responsible for extracting and inserting data from the backend, upon an event triggered by the user, via a callback.It has a total 4 functions,   2 of which are suspend functions. The suspend keyword in this scenario is identical to the Java runnable keyword, marking the usage of parallel threads in our program. This is as a result of having the need to obtain the data required by the user in real time via GET requests. However, since the communication conducted with Firebase Firestore is asynchronous, the need to have parallel threads become essential, otherwise without these, we would be blocking our main UI thread when requesting data either when scanning a barcode or when inputting a serial number. The other 2 functions located within the FirestoreService.kt file are the functions responsible for the CREATE, or POST requests. These functions are responsible for the insertion of objects into our database.

The final part of the Model layer is represented by the AmplifyService.kt file which is responsible for sending authentication challenges to the Cognito user pool in order to either login, register/sign up, verify the code that was sent on the email to the user, log out of the application(from the home page). All the functions mentioned here, within the AmplifyService.kt file, is Before all these functions however, is the configure amplify function, which has the purpose of the initial AWS amplify initiation. All of the technologies discussed in this subsection will be delved into more deeply in chapter 7.

Chapter 7

IMPLEMENTATION CHOICES

7.1 Technologies used

At the start of the project, the question of which technologies would best suit the project's needs was asked. Considering the different values of the stakeholder groups, it was decided that what was needed was a secure implementation which focused on data availability and authenticity. At the same time, the passport needed to be widely available and usable by people of varying demographics and backgrounds. Also, the solution needed to be scalable and it needed to remain relatively cheap when scaled. Naturally, the team arrived at the conclusion that they needed to make a mobile phone application using cloud services in order to meet those requirements.

7.1.1 Kotlin and Android

After it was decided that the DPP needs to be in the form of a mobile application, the team had to settle on an operating system. The two most popular mobile operating systems are Android and iOS. However, Android far surpasses iOS in the EU, with 64.5% against Apple's 35% of the market share(Sherif, 2024). Therefore, it was decided that an Android application would be most suitable for a system which is to be used in the EU.

While many programming languages can be used for Android development, Kotlin and Java are the two most popular (Successive Digital, 2022). In recent years, Kotlin has surpassed Java because of its many advantages. Two of them, concise code and code safety, are the most important for this project. First of all, apps developed on Kotlin generally require less lines of code for the same functionalities and are proven to be more readable for developers (Successive Digital, 2022). Due to the time constraint of around 10 weeks, this was a persuasive argument for the use of Kotlin. Secondly, code written in Kotlin is safer for developers as it avoids the NullPointerException crash, which is the main reason for app crashes in Java (Successive Digital, 2022). On top of that, the Kotlin compiler performs more checks than the Java compiler before running which allows apps to crash and helps developers root out bugs. For these reasons, the team settled on the Kotlin programming language.

7.1.2 AWS Amplify

AWS Amplify is a toolkit used to quickly set-up and develop applications using the Amazon Web Services ecosystem. It is especially suited for mobile and web applications. It acts as a kind of a “glue” service which lets developers host their applications in the AWS cloud and quickly attach functionality from other AWS services. Amplify’s big advantage is that it allows for apps to be set up quickly and is therefore preferred by developers who are making prototype or MVP solutions (Be a Better Dev, 2021). Another selling point is the ability to have multiple environments at the same time. Similarly to Git and other version control software, Amplify supports simultaneous branch-like environments on which different developers can test different functionalities and store different versions of the project. However, this can also prove to be a problem as just like in version control tools, in Amplify there can also be a type of “merge conflict”. This leads to having to redo a portion of the work or redeploy some of the AWS resources when joining everything in a single environment. Nevertheless, the ability to quickly deploy a cloud-based application was a big enough advantage to sell Amplify to the project group.

7.1.3 AWS Cognito

One of the services Amplify provides access to is AWS Cognito. Cognito is Amazon’s user authentication solution and it provides secure sign-up and sign-in capabilities. Cognito’s main advantage is that the user authentication is handled in a fast and secure manner and many of the developer’s responsibilities about user security are moved to Amazon. It also provides two-factor authentication when registering which is handled through an automated email system. On top of that, this technology has a quick set-up and easy integration into mobile applications (DiStasio, 2024). With so many advantages, Cognito was an easy choice for user authentication.

7.1.4 Firebase Firestore

Firebase is Google’s answer to Amazon’s AWS. For numerous reasons which are going to be described in section 7.2 the team decided to not use DynamoDB, the database solution provided by Amazon. For reasons explained in more detail in section 7.3, it was decided that a NoSQL serverless solution was required. Firebase Firestore is one such serverless, NoSQL solution which benefits from robust security protocols(BlueWhaleApps, 2021), which is crucial for the use-case of this project. On top of that, Firestore is designed to be easily scalable so it offers the possibility of extending the functionality of this project while keeping the original database and building up from it.

7.2 Technologies NOT used

Explaining why certain, at first glance logical, technological choices were not made is just as important as justifying chosen technologies. The team encountered a few technologies which were either too challenging to implement or had a better alternative, in the context of this project.

7.2.1 Blockchain

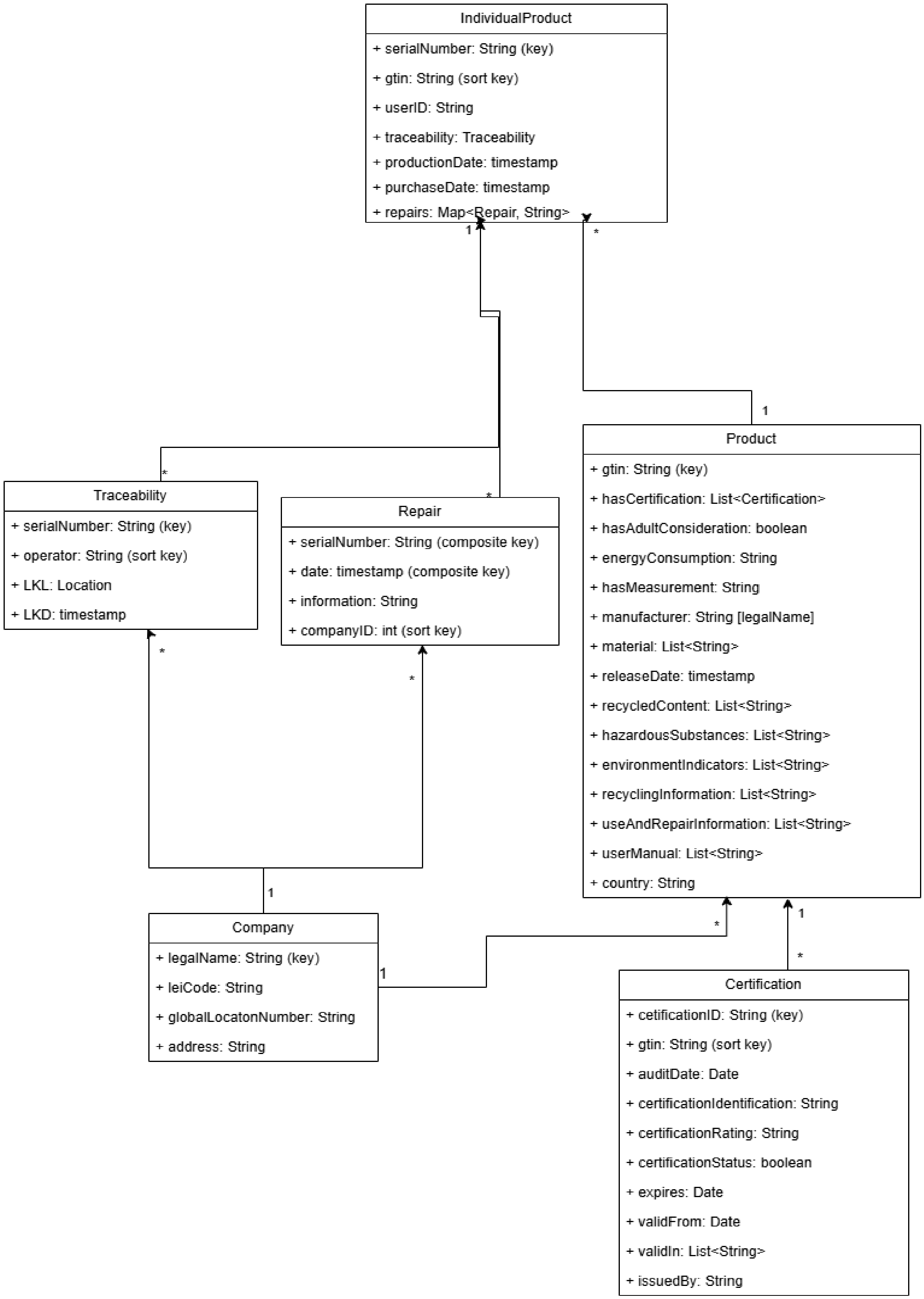
A common solution to any problem involving data authenticity and security is using blockchain technology because data written on the blockchain is immutable. In fact, blockchain was discussed in most of the literature reviewed for this project, specifically because of those two reasons(Archer et al., 2024). However, blockchain presents a huge problem when it comes to cost. A blockchain requires users to pay a fee when they retrieve or write data (Soomaney, 2024). This scales terribly as the more data you want to store, the more you have to pay. A DPP requires constant reads and writes per user, accessing their owned products or examining products they don't own. On top of that, it also requires reads and writes every time a manufacturer wants to register a new batch of freshly produced units or a repair shop wants to publish repairs. This constant stream of operations on the database proves too expensive to justify the benefits that come with using a blockchain.

Another common argument for using blockchain is that they guarantee trust in a distributed system where no one entity can have control over the database. In the case of DPPs, this means that each company can upload data without worrying that sensitive information such as units produced, units sold, etc. will be sold to or used by malicious third parties. However, many DPP researchers (Wiesner, 2023) (Archer et al., 2024) would agree that International Data Spaces (IDS) are a more suitable alternative. IDSs are "a trusted environment where organizations can share data while retaining full control over its use"(International Data Spaces, 2024). In its Reference Architecture Model, the International Data Spaces Association provides a set of strategic requirements for a Data Space, including "trust, security and data sovereignty, and standardized interoperability"(Steinbuß et al., 2022). Therefore, a properly maintained and designed Data Space will solve the same problems as the blockchain but also many more such as the problem of DPPs needing interoperable data formats.

7.2.2 DynamoDB

DynamoDB is Amazon's noSQL serverless database and an alternative to Firebase Firestore. It satisfies all of the requirements for a database for this project and on top of that is part of the AWS ecosystem which suggests it might have an easier integration with Amplify. However, this proved to not be the case. While the team relentlessly tried to implement DynamoDB, the steps of the implementation constantly led to implementing new AWS services. To access DynamoDB from the application a RestAPI was needed. This was created and added using AWS APIGateway. However, API Gateway required AWS Lambda functions to connect each CRUDE operation to the database. While some progress on connecting DynamoDB was made, like implementing a working GET function within the application which retrieves information from the database, it was eventually abandoned in favor of the simpler to implement Firestore.

7.3 Database Schema



The database diagram represents the structure of the Firebase Firestore database implemented in the app's backend. The design of the database draws on the literature and implementations discussed in chapters 3 and 4. On top of that, it is designed to comply with naming and general design principles advocated by Schema.org. Schema.org is a widely adopted collection of database schemas started by Google, Microsoft, Yahoo and Yandex. This is done with the goal of helping manufacturers comply with a single data format which is crucial if the DPP is to be widely adopted. Moreover, having a standardized data model from the start will make the adoption of an International Data Space easier.

The above-shown database diagram can be used as a blueprint either for an SQL or NoSQL diagram. The team has determined that a DPP requires the NoSQL version, which allows for unstructured and semi-structured data. All data requirements for DPPs mentioned in this project are just suggestions and no consensus on the required information and data formats has been reached. This means that companies can largely choose what data and which data format they use when sending information about their products. The lack of widespread International Data Spaces only intensifies these problems as companies can either send their data through an IDS or directly to the administrators of the DPP. Moreover, this can quickly change as new projects and legislation on DPPs are being approved at a rapid rate. To account for this dynamic environment, we have decided to use a NoSQL database but have still developed an universal DB design as we expect that an SQL solution could be required in the future.

Each database table is explained below and an example of what it represents in the real world is given. Also, each relationship between tables is explained only when the two tables have already been discussed.

7.3.1 Product

This table stores general product information which is shared between all units of a specific product's model. An example of a Product is a Google Nest v2. In this case, the table would store all of the information which is the same between all of the Google Nest units across the globe, no matter who owns them. This information includes a unique identification number, the GTIN, with which manufacturers register and differentiate products and no two products can have the same GTIN number. Users encounter the GTIN in their everyday lives when scanning barcodes, as the GTIN is the number below or above the lines of a barcode. Other information stored here includes the manufacturer, the materials used in production, user manuals that come with the product, information about energy and environment indicators, countries of production and assembly, certificates issued to the product, information about recycling and others.

7.3.2 Individual Product

This table stores information on the level of an individual unit of a product. This information is unique for any single unit, produced by a manufacturer. An example of this is the Google Nest unit used in this project. Although it shares a lot of traits with all other units produced by Google, some of those traits are unique to it. A trait which is unique and different between all units is the serial number. Other unique information includes the user, the production date, the purchase date, the location and time the product was last turned on and what repairs have been done to the product. An IndividualProduct has a many to one relationship with Product, meaning that many units of the same product can share a GTIN, just like all Google Nests have the same GTIN.

7.3.3 Repair

This table stores a record of individual repairs made on an individual product. An example of this would be a phone which goes in for maintenance, at a third-party or official repair shop, for a cracked screen. A repair is identified by the serial number of the product being repaired and the timestamp of the repair, as it is impossible to have two separate repairs being performed on a product at the same time. This table also includes additional information about the repair and the ID of the company conducting it. It has a many to one relationship with IndividualProduct as many repairs can be done on a single product but each repair concerns only one product at a time.

7.3.4 Traceability

This table represents the location history of each individual product. An example of this would be a Google Nest unit which was connected to multiple different smart home systems. Its owner would be able to access a history of the device's connections. The table stores the serial number of the product, the name of the operator (in case the device is connected to a mobile or wi-fi network) and the location and timestamp of the connection. Each connection appears as an individual item in the table. This table has a many-to-one relationship with IndividualProduct as a product has a history of connections but a single instance of the connection can't be shared between two separate devices.

7.3.5 Company

This table represents the record of companies which either manufacture, distribute or repair products, or are network providers for the connections stored in the traceability table. An example of a Company would be Google which manufactures the Google Nest but also MediaMarkt which is a distributor and Ziggo which is a network provider, providing the network to which the device is connected. Each company has a unique legal name. In the rare cases names match a salt can be added to the name for uniqueness. Also, some companies have a lei code and a global location number of their facility. They also have an address. Company has a one-to-many relationship with Traceability, Repair and Product. A company can provide many connections, repairs or products but each of those can be created only by a single company.

7.3.4 Certification

This table represents the record of certificates issued to products. An example of this is the RoHS(Restriction of Hazardous Substances) certificate issued to the Google Nest used in this project. A certificate has a unique identifier, an audit date, validity, an expiration date, rating, description (identification), issuer and a list of countries in which it is valid. A Certification has a many-to-one relationship with Product as a single Product can have many certificates but a certificate is always product-specific.

Chapter 8

TESTING

8.1 Test Plan

Our test plan is to combine log testing to test specific functionalities within the android project and the AWS backend. After testing each component individually, we can proceed to system testing. We will not be conducting system tests to test functionalities which were previously failed within the unit testing stage or not unit tested at all. This is because we are aiming to test as many parts of the project individually which are involved in the system being tested before conducting the system test. This removes doubt about each functionality of the project not working as its intended, and only tests the integration between functionalities.

8.1.1 Unit Testing

The main tool used for unit testing in our environment, as well debugging, was represented by the LayoutInspector tool, which can be found in the Android Studio Device Manager tab. The LayoutInspector has the purpose of analyzing at which the composable elements within our applications recompose, or put simply, reinitialize. Thus, the LayoutInspector helped us to pinpoint specific performance issues tied to the frontend of our application, so as to avoid having slow loading times as well as distorted colors. In addition to this, the LayoutInspector was used when determining the relative offset and position of visual elements in our application.

8.1.2 System Testing

System testing was utilized to make sure that at each step of integration of cloud services, the expected result was achieved. System testing was conducted by using the android "android.util.Log" library. On successful results, a success message was logged. On unsuccessful results, if it was a network error the status code was printed or if it was an error thrown by the android compiler, the stack dump was logged and read to further find the root cause for the error. All of the system's components are integrated together as intended. The system tests have covered the integration of AWS Amplify authentication service into the android project as well as the testing of Firebase Services.

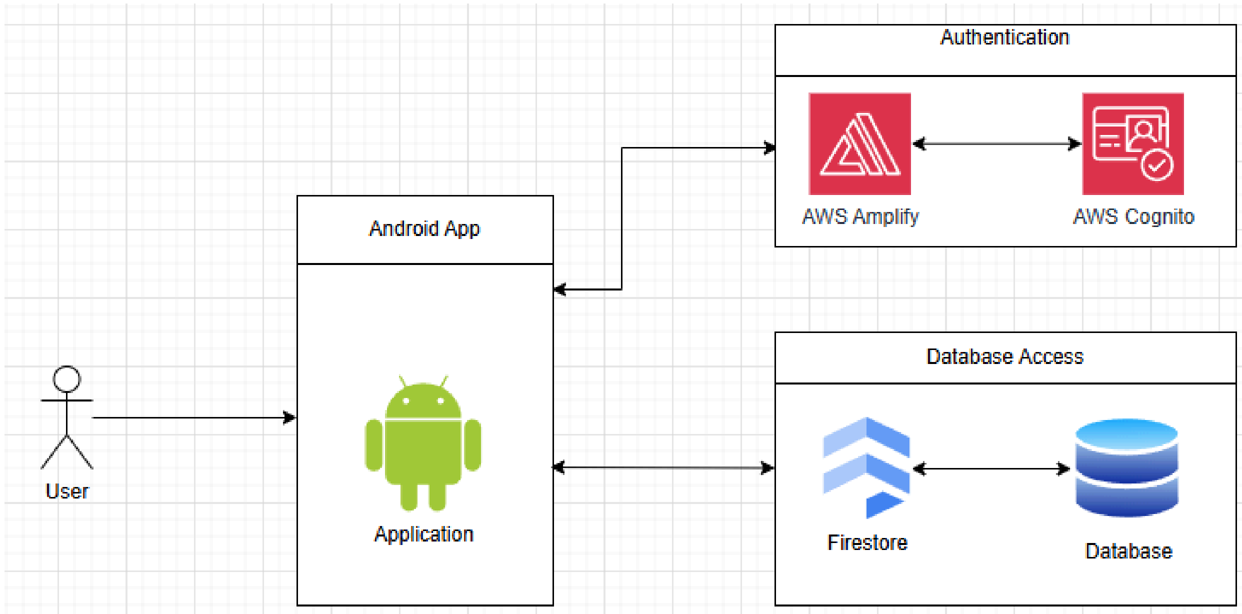


Figure 8.1: Application Backend

The final system test was as such: Go through the authentication process, then inside the home page (Appendix B.7) we had a button which we would click, which would then execute a custom function. This proved to be a slow method, and in hind-sight, it would have been a good idea to invest some time into making automated system tests

8.1.3 Usability Testing

Through usability testing, our team was able to gain and act upon valuable information for bettering our implementation of the visual interface and usability of the Digital Product Passport. This was done by asking people to use our application and give us their feedback. Each of the peer-feedback review sessions, where we had about 20 potential users who reviewed our applications represented a great opportunity in this sense, given the decently sized pool of potential users, and each time we gained valuable constructive criticism from the user pools to which we showcased our visual interface and functionalities.

8.1.4 Functionality Testing

This section shows the functionalities of our application which we should test. Each functionality is marked with a priority. The three priorities are: High (H), Medium (M) and Low (L). The high priority level indicated that the correctness of the functionality is essential to the application. If there are problems with a functionality which has “High” testing priority, all cases where the functionality can throw an exception must be tested. Additionally, the logic of the functionality must be tested as well. This is because if the functionality with “High” priority fails, other functionalities will fail as well, because they are interdependent. The functionalities with “Medium” priority level are generally not related to the user but to the maintenance of the application. For example, database manipulation through an admin account. What differentiates “Medium” functionality from “High” is that it is not an essential functionality for the Application to run, however it could have unwanted side effects for the user if not properly tested The “Low” priority functionalities have very little impact on the user and are mostly functionalities related to the visual and interactive part of the application.

Type of Functionality	**Functionality to test**	**Priority Level**
Authentication	Register a new account	H
	Log into an existing account	H
	Log out	H
	Accept terms and conditions	L
Database	Retrieve product information	H
	Retrieve general product information	H
	Retrieve information of owned products	M
	Posting a new product	M
	Updating repair records	M
Camera	Scanning barcode	H

Table 8.1: Functionality priority table

8.2 Testing results

This section covers the results of the tests mentioned in subsection 8.1. Results for Unit tests were emitted as we did not utilize unit tests. Functionality testing is also emitted as it is not a result based test, rather a table which gives each functionality a ranking of its priority for the core functionality of the project.

8.2.1 System Testing

After Conducting the system tests, we were able to successfully verify the functionality correctness of the microservices used. AWS Amplify and AWS Cognito, which were used for authentication, were successfully integrated into the project. This was seen when the user pool panel in AWS Cognito contained the email of the user which we registered through the app. Additionally, the android logging library "android.util.Log" was used to display error messages in the Logcat tab inside android studio. This has helped us find out the results of a call to AWS amplify or Firebase from our android app.

1. Test: Try signing up with a new email and password.
   Expected result: Receive an email with a verification code
   Actual Result: Passed

2. Test: Signing up with an already existing email and password.
   Expected result: Registration is unsuccessful, the password-email combination in the userpool remains unchanged
   Actual Result: Passed

3. Test: Inputting a verification code and registering a new account.
   Expected result: A new account is created in the Cognito user pool and the user is redirected to the next screen.
   Actual Result: Passed

4. Test: Logging in with an already existing account.
   Expected result: The log-in is successful and the user is redirected to the home screen.
   Actual Result: Passed

5. Test: Logging in with an non-existing account.
   Expected result: The log-in is unsuccessful and the user doesn't have access to the home screen.
   Actual Result: Passed

6. Test: Scanning the barcode of a product that is already in the database (the Google Nest)
   Expected result: The general product information about the device is displayed.
   Actual result: Passed

7. Test: Searching the serial number of a device, already in the database (the Google Nest).
   Expected result: Retrieving the individual and general product information.
   Actual result: Passed

8. Test: Registering a new device, which is not already in the database.
   Expected result: A new device appears in the IndividualProduct table with the corresponding product information.
   Actual result: Passed

9. Test: Updating a device, which is already in the database, by reregistering it with the same identifiers but inputting different information in the other fields.
   Expected result: The device is updated with the new product information. No duplicates should appear.
   Actual result: Passed

8.2.2 Usability Testing

Here are the following features of the app which we have tested.

1. Register an account for the app
2. Log into the app
3. Scan the barcode with the camera
4. Register a product for repairs
5. Register a new product

After telling the participants to do the actions above, we have received valuable feedback which we mention below.

1. The app doesn't indicate which field is which
   Solution: add labels on the fields to let the user know which information should be put into which field.
2. The app doesn't indicate why I couldn't register
   Solution: Add a textfield which reports a human readable error and an action which should be taken to fix that error.
3. I do not want my password to be shown as I am typing it
   Solution: add a protection mechanism to change every character typed into a "•".
4. I want to be able to go from the login page to the registration page and vice versa instead of going back to the welcome screen to access them.
   Solution: Add links from login page to register page and vice versa with the labels "Don't' have an account? Register here" and "Have an account? Log in" respectively.
5. I do not want to fill out all attributes for a product for it to be registered for a repair.
   Solution: Add an "*required" on the "serialNumber" and "gtin" fields, so that the required attributes for identifying a product are explicitly shown to the user#
6. The home page has buttons which are offset from the center
   Solution: Align all the buttons to the center.

7. The initial admin page buttons were in the color red, and thus were associated with the delete feature.
   Solution: Change the color of the buttons into blue, green and magenta.

Chapter 9

FUTURE PLANNING

9.1 Support for Additional Features

There are many use cases for the DPP which we have not included in our main project as they would take too long to implement. This subsection of the report will briefly go over these use cases. Although the team members are not planning on implementing these features, we do wish to talk about the features we would develop nevertheless.

9.1.1 List of Additional Features

- **Expand database to include more information**
  Lots of vital information for a DPP solution was left out. The suggested fields which would be added are the following: general, source, footprint, ownership, maintenance, documentation and instructions (2024, September 18). was left out to simplify the database schema. These fields are essential for a DPP solution as

- **Record the product's lifecycle state**
  Our final implementation only keeps track of individual items and doesn't include information about which part of the production process the product is currently in. This feature would extend the database to add attributes which would track the product lifecycle state. Additionally there would be an ability for users to change the state of a product within the app. This feature would rely on users to accurately track the product's lifecycle.

- **Ability to monitor history of each product's state**
  Every registered product would have its own state monitoring where the product is in the circular lifecycle phase. Additionally, each product would have a list of previous states which this product has been in. This would give valuable information on which state the product is in, and the time the product spends in each lifecycle state. This information would further help manufacturers find out where their product goes after its use.

- **Analytical dashboard for producers**
  The producers of a product would need their own dashboard to view statistics of their products. This dashboard would allow querying of digital product passports of the products the manufacturer is responsible for. This would give the manufacturer valuable information about the product such as: how many products are in which state of the product life cycle, purchases, where the product goes after end of use and other vital information for analyzing digital product passport data.

- **Future compliance with International Data Spaces (IDS) standards**
  In the future, we would want our software to be compliant with IDS standards. This would give us the ability to be both a data supplier and data client of an IDS and generally engage in data transactions with a Data Space. A key step towards achieving this would be the implementation of a data connector framework (Giussani & Steinbuss, 2024). This will lead to easier data provision by manufacturers and access by consumers, giving us the ability to scale-up the project.

Chapter 10

EVALUATION

10.1 Responsibilities

The assigned responsibilities listed below:

- **Ivan:** Requirement Analysis, Presentation Slides, Poster Design, Report, Database Design, DynamoDB(failed)
- **Rares:** Firebase Backend, Requirement Analysis, Presentation Slides, Poster Design, Report, Kotlin Frontend, Authentication(primary implementation)
- **Danil:** Authentication(initial setup & parallel implementation), Report, DynamoDB(failed)
- **Fabio:** Front end Design, Presentation Slides, Poster Design, Report
- **Alex:** Firebase Backend, Front end Design, Figma Prototypes, Presentation Slides, Poster Design, Report, Kotlin Frontend

10.2 Requirement Gathering Evaluation

A thorough research into current DPP implementation was conducted. From this research we were able to evaluate which design is used by the currently leading DPP firms. Additionally, a thesis from a University of Twente student was analyzed to derive relevant requirements for the project.

10.3 Planning Evaluation

As two of the members were online and the other three present physically, the team opted for a more lenient schedule. Firstly, a team group chat was created where everyone could collaborate and present their ideas. Secondly, weekly meetings with the supervisor were scheduled and the Google Nest Protect, the device of concern in this project, was given to the team. As the project progressed and the functional and non-functional requirements were elicited, a mock-up design was proposed to the supervisor. After mutual agreement on the mock-up, a database design was proposed and agreed upon.

As the project moved forward, meetings with the supervisor were held less frequently as the main focus went towards development of the system. However, textual communication (through Microsoft Teams) was maintained during the whole duration of the project. As the project moved towards its final stages and deliverables had to be submitted, the team raised the work-speed and team calls were held almost daily.
Because of the nature of working both online and on-site there was some lapse in communication. This led to some work having to be redone and other work being of lesser quality. However, all deadlines were respected and it is the team's sincere belief that every deliverable is of sufficient quality.

10.4 Implementation Evaluation

The project had its ups and downs during the implementation phase. We faced some hurdles, but we also managed to create the final product. The AWS backend team has underestimated the complexity and time required to understand the AWS infrastructure and how to integrate AWS microservices into the project. This was especially seen with the integration of DynamoDB into the project, where the failure to do so caused many setbacks, which had to be fixed before the presentation. The solution to these setbacks was to use a Firebase backend. Additionally, AWS didn't have the right APIs to retrieve relevant information about an authenticated user, such as which group the user belonged to, which was needed to know which user was an admin and which wasn't. Therefore, the authentication was later changed to Firebase as well to accommodate this requirement.

10.5 Development Cycle

We have split the project into 4 phases: requirement solicitation, design, implementation, testing. Each phase provides valuable insight into the next. These are similar to the software development cycle as can be seen below.

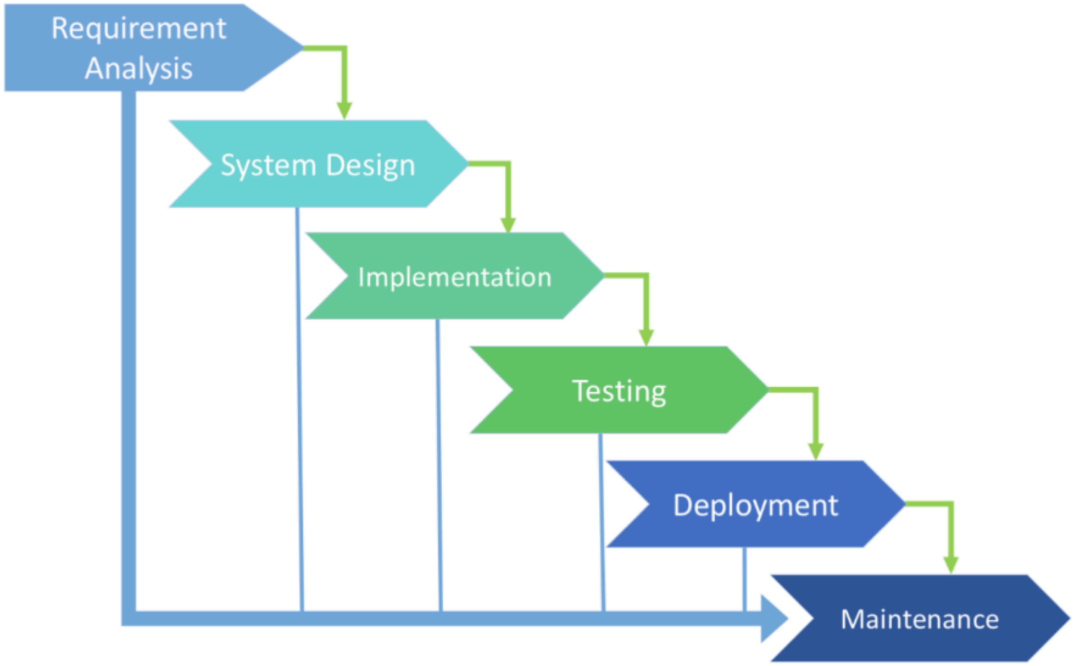


Figure 10.5 SDLC Waterfall Model

The major differences between the software development cycle and our development is that in our project we skip the deployment stage and maintenance stage, as the project does not require these stages. We instead mention how we would maintain the project, and skip on the deployment stage completely

10.6 Conclusion

The Digital Product Passport emerges as a promising solution to bridge the information gap between manufacturers, consumers and regulators. It is a vital technology for driving the transition towards a more sustainable and circular economy. By providing critical product information in an accessible way, the DPP empowers consumers to make informed choices. As the EU strives to achieve its ambitious climate and environmental goals, the DPP plays a pivotal role in accelerating the adoption of sustainable practices by the consumer. Our project explored current DPP solutions as well as the literature on the value that DPP brings as a concept.

The lack of experience with AWS microservices has slowed down the backend development for this project, however the team thankfully recovered by using a different cloud provider, Firebase. In hindsight, a lot more time should have been spent figuring out the AWS microservices or the switch to Firebase should have been sooner to implement our design sooner. Although there were difficulties with setting up the live demo for the presentation, the final product submitted had all the planned features.

Bibliography:

Digital product passport (DPP): The Complete Guide. Protokol. (2024, September 18). https://www.protokol.com/insights/digital-product-passport-complete-guide/

European Commission. (2023, May 2). *Digital product passport*. European Health and Digital Executive Agency (HaDEA). https://hadea.ec.europa.eu/calls-proposals/digital-product-passport_en

Wiesner, M., Moreira, J., Guizzardi, R., & Scholz, P. (2024). A reference architecture for digital product passports at batch level to support manufacturing supply chains. *Lecture Notes in Business Information Processing*, 307–323. https://doi.org/10.1007/978-3-031-59465-6_19

HAMAD-CHETAN, L. (2024). *Environmental Management Systems: a platform towards a Digital Product Passport reality* (thesis).

Wiesner, M. (2023b, December 14). *From component to product digital capabilities for the digital product passport in the production sector*. University of Twente. https://purl.utwente.nl/essays/97870

Product traceability platform for Supply Chain Compliance. Circularise. (2023). https://www.circularise.com/

Daphne, T., & Stretton, C. (2023, March 17). What data goes into a digital product passport? *Circularise*. October 31, 2024, https://www.circularise.com/blogs/data-in-a-dpp

About CIRPASS. CIRPASS. (2024). https://cirpassproject.eu/about-cirpass/#

Archer, P., Olsson, S., & Joussen, A. (2024). *DPP System Architecture*. https://doi.org/10.5281/zenodo.12206138

Schema.org. (2024, September 17). https://schema.org/

Sherif, A. (2024, March 7). *Mobile OS Market Share Europe 2010-2023*. Statista. https://www.statista.com/statistics/639928/market-share-mobile-operating-systems-eu/#:~:text=As%20of%202023%2C%20Android%20was,percent%20of%20European%20mobile%20phones

7 reasons why you should use Kotlin for Android App Development. Successive Digital. (2022, October 18). https://successive.tech/blog/kotlin-for-android-app-development/#:~:text=The%20smaller%20code%20volume%20means,and%20complex%20app%20development%20projects.

Daniel. (2021, October 29). *What is AWS amplify? pros and cons? - be a better Dev*. Be a Better Dev. https://beabetterdev.com/2021/09/22/what-is-aws-amplify/

DiStasio, A. (2024, May 16). *Pros and cons of using Amazon Cognito for user authentication: Gavant Blog*. Gavant Software. https://www.gavant.com/library/pros-and-cons-of-using-amazon-cognito-for-user-authentication

Pathik. (2021, March 4). *7 reasons to choose google cloud firestore as your database solution*. BlueWhaleApps. https://bluewhaleapps.com/blog/7-reasons-to-choose-google-cloud-firestore-as-your-database-solution

Soomaney, A. (2024, October 15). *Understanding blockchain data storage, costs, methods and Sustainability*. Medium. https://medium.com/@abhinav_cryptotax/understanding-blockchain-data-storage-costs-methods-and-sustainability-620610218a9f

Home. International Data Spaces. (2024, October 24). https://internationaldataspaces.org/#:~:text=The%20International%20Data%20Spaces%20Association,full%20control%20over%20its%20use.

Steinbuß, S., Otto, B., Teuscher, A., & Bader, S. (2022). *IDS-RAM 4*. IDS Knowledge Base. https://docs.internationaldataspaces.org/ids-knowledgebase/ids-ram-4/front-matter/frontmatter

International Data Spaces. (2024, May). Data Connector Report. https://internationaldataspaces.org/wp-content/uploads/dlm_uploads/IDSA-Data-Connector-Report-No-15-May-2024.pdf

What are data connectors?. Incorta. (2022, September 16). https://www.incorta.com/glossary/what-is-data-connector#:~:text=Data%20connectors%20are%20software%20components,data%20science%20applications%20and%20frameworks

Giussani, G., & Steinbuss, S. (2024). Data Connector Report. *International Data Spaces Association*. https://doi.org/https://doi.org/10.5281/zenodo.13838396

Datarob. (2020). *Understanding the software development life cycle*. https://datarob.com/essentials-software-development-life-cycle/

Wilson, A., Wedyan, F., & Omari, S. (2022). An empirical evaluation and comparison of the impact of MVVM and MVC GUI driven application architectures on maintainability and Testability. *2022 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, 101–108. https://doi.org/10.1109/idsta55301.2022.9923083

Appendix A

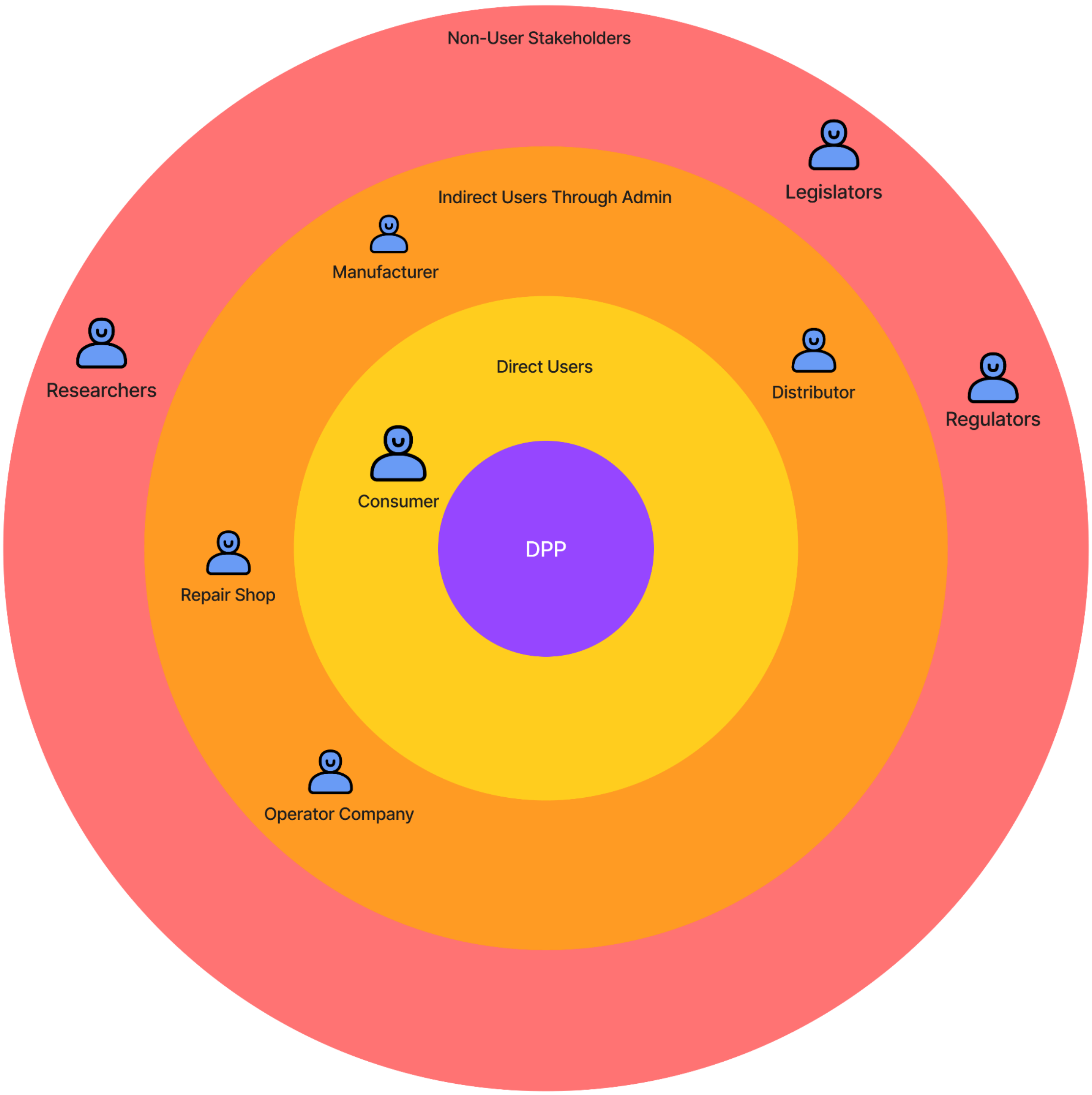


Figure A.1: Stakeholder onion model

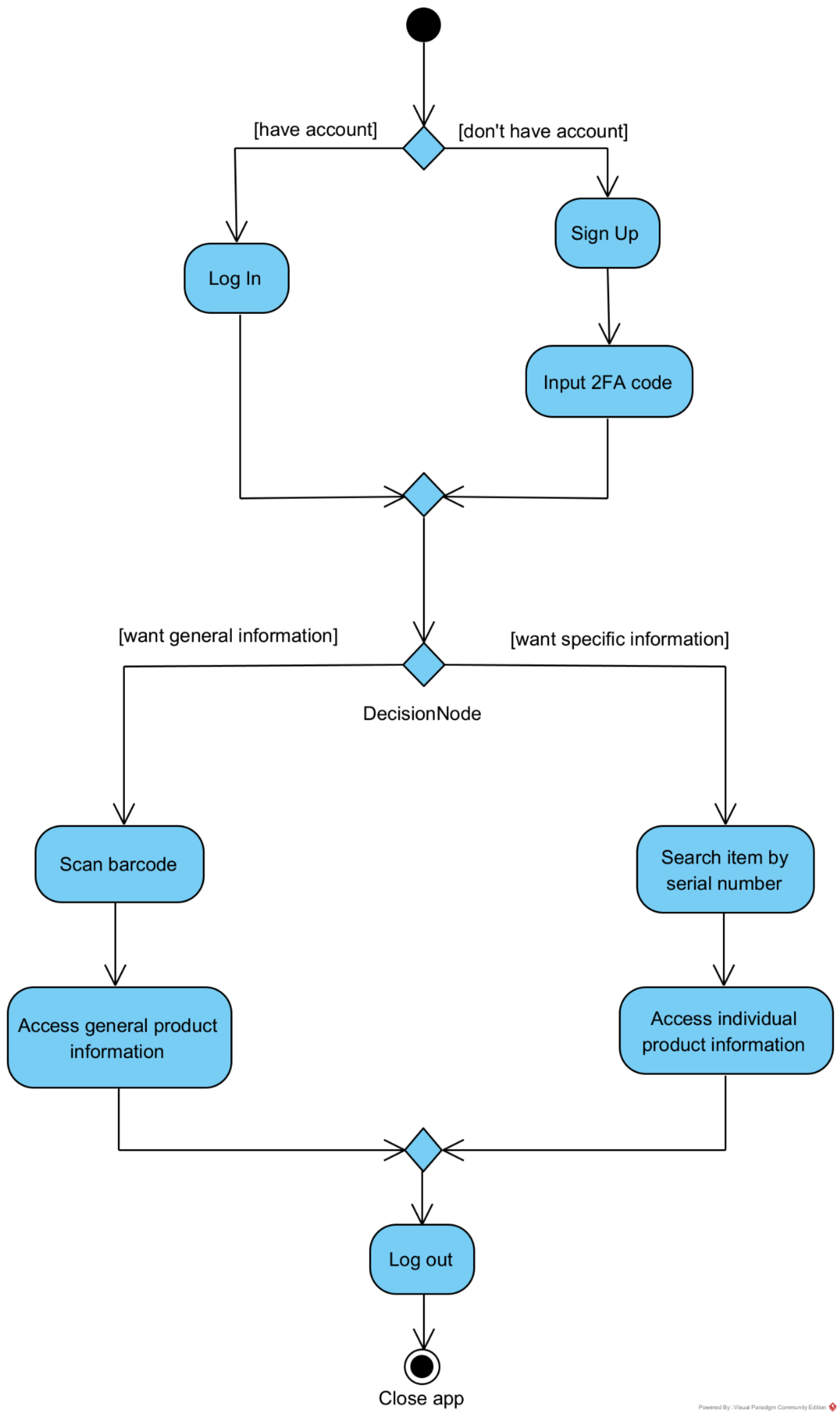
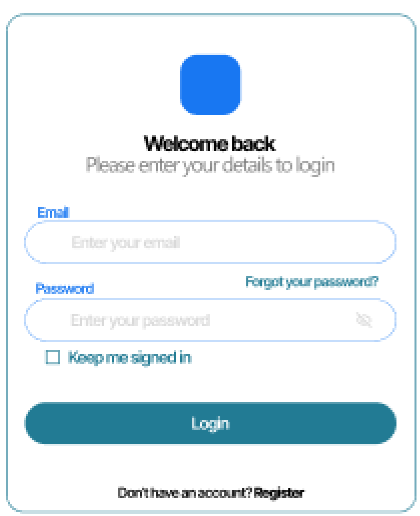


Figure A.2: User Activity Diagram

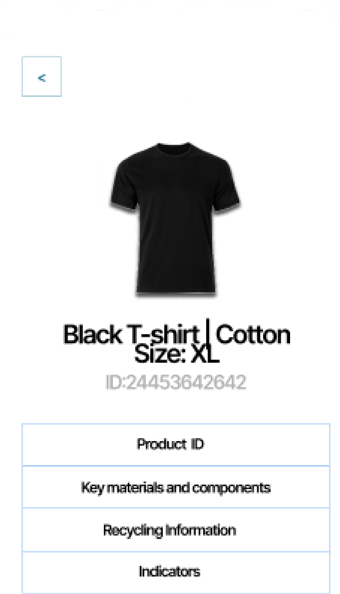
Appendix B

Gitlab Link: https://gitlab.utwente.nl/s2711656/DPP.git

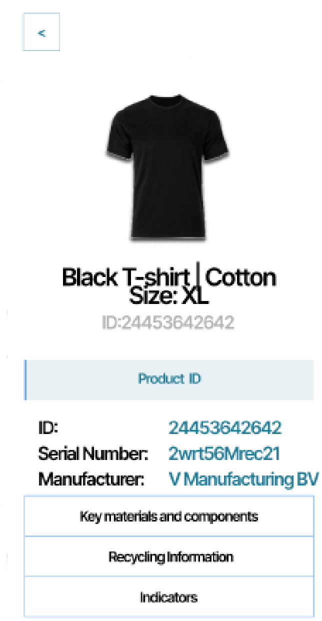
Figure B.1: Welcome Screen



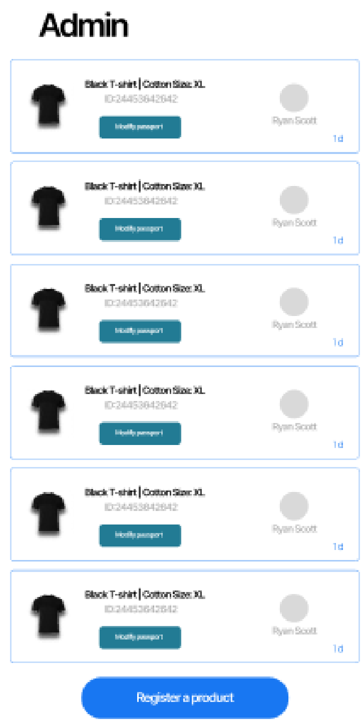
Appendix B.2: Login Screen



Appendix B.3: Product Screen

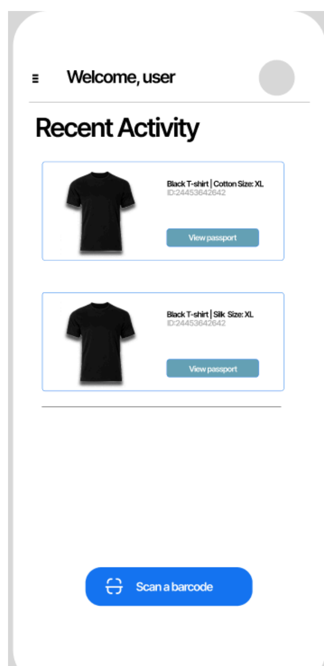


Appendix B.4: Product Screen with “Product ID” attribute expanded



Appendix B.5: Admin Screen

Appendix B.6: Repairs Screen



Appendix B.7: Home page