Bern University
of Applied Sciences

Digital Vaccination Certificate

Course of study	Bachelor of Computer Sciences
Author	Marius Schär
Advisor	Dr. Annett Laube
Co-advisor	Dr. Reto Koenig
Expert	Mathis Marugg

Version 1.0.0 of January 20, 2022

Abstract

Due to the ongoing COVID-19 pandemic, the issue of a “proof of vaccination” has been raised in multiple countries. The core properties of a proof of vaccination include non-forgeability and protection of the certificate holder’s privacy. Moreover, the verification procedure must be both easy and sound.

In this thesis I compare existing state level solutions, pointing out their strengths and weaknesses in the given context for the above mentioned properties. Then a concept, as well as a prototype implementation is given as a proof of concept for a solution that fulfills all the core properties and requirements.

Contents

1. Introduction

Due to the ongoing COVID-19 pandemic, the issue of a "proof of vaccination" has been raised in multiple countries. A proof of vaccination must not be forgable, it must protect the Certificate Holder's privacy, and it must be easy to verify.

This thesis continues the work done during Project 2 by further refining the concept and implementing a prototype of our own system, as well as comparing other solutions to this problem with my system and highlighting their respective advantages and disadvantages.

I believe that my proposed solution has several benefits over the Swiss Covid Certificate and other existing solutions with regards to privacy, availability, convenience, and soundness.

As in Project 2, I did not examine the political issues surrounding these certificates, instead focusing on the technical aspects.

Several solutions to the "proof of vaccination" problem already exist. Most notably for Switzerland is the Swiss Covid Certificate (SCC), which is an implementation of the European Union's Digital Covid Certificate (aka. Digital Green Certificate, EU DGC) The EU DGC is the closest thing to a standard for a proof of vaccination, although a lot is left to the member states in the implementation.

Worldwide there are many solutions to the problem, from completely centralized and proprietary systems [12], to those making use of existing open standards. [22]

This thesis introduces the key concepts required for a proof of vaccination system and compares existing state level solutions, specifically the Swiss Covid Certificate and a centralized system, to my own, novel solution.

The concept, and a prototype implementation as a proof of that concept, are given for the my own system: ARCTIC - the **A**utomatically **R**eadable **C**hain of **T**rust based **I**mmunization **C**ertificate.

It is **A**utomatically **R**eadable by computers, which simplifies the verification procedure for human verifiers and enables automated gates based on the certificate. It is based on a **C**hain of **T**rust, meaning that, while there is a centralized authority, it does not issue certificates directly. Thus the vaccination data is kept between the certificate holder and the person performing the vaccination. Lastly, it is an **I**mmunization **C**ertificate, in that it is based on FHIR (Fast Healthcare Interoperability Resources) Immunization records and cryptographically secured.

2. Prerequisites

This chapter introduces the prerequisites required for a proof of vaccination system by first discussing what the subject of the proof is and how information can be secured using digital signatures, then the role public key infrastructure plays and the concept of a "Chain of Trust" (CoT), how a proof of vaccination might be stored, transported and presented.

2.1. FHIR

In order to construct a proof of vaccination, the act of immunizing (e.g. by vaccination) a person must be represented in digital form. As discussed in the Project 2 that preceded this thesis, FHIR is ideally suited to this task. [26, Section 4.1.1]

FHIR, the Fast Healthcare Interoperability Standard (pronounced "fire"), is a standard by Health Level Seven International (HL7) that describes an API and data format for exchanging health data. The latest version of FHIR, 4.0.1, is the first normative version of the standard, released in 2019. [16] FHIR is organized by "resources" that represent entities, processes, and other concepts that are relevant in the medical field. The "Immunization" resource contains all the data required for a proof of vaccination. [17].

The Immunization resource could include information about the reason an Immunization was performed or not performed, who performed the immunization (down to a single healthcare worker), what quantity of the vaccine was administered, how the patient was educated, and even the reactions to the immunization that the patient experienced. While these are important data points in a medical context, they are not needed for a vaccination certificate and are thus left out (the standard declares many elements as optional).

An important component of FHIR is the coding system and value sets. To gain maximal compatibility there exist, for example, Europe wide value sets which specify how different concepts are mapped into the resources. Listing 2.1 shows an example of such a coded value (`"vaccineCode"`). It specifies which vaccine was used in the immunization represented by this resource. In this example it's the Moderna COVID-19 Vaccine. The `coding` specifies both a `system` and a `code`. The `system` in this case is the "EU Public Health - Union Register of medicinal products" which assigns an identifier (e.g. "EU/1/20/1507" for the Spikevax COVID-

19 vaccine) to every medicinal product approved by the European Union. [4] Each coded value also contains either a `"text"` or `"display"` element, which specifies a human readable text to be displayed instead of the identifier.

The EU has published a separate document containing the value sets to be used to represent vaccines, manufacturers, and diseases in the EU DGC. [7, Section 2.3]

```
"vaccineCode": {
  "coding": [{
    "system": "https://ec.europa.eu/health/documents/community-register/html/",
    "code": "EU/1/20/1507"
  }],
  "text": "Spikevax (previously COVID-19 Vaccine Moderna)"
}
```

Listing 2.1: Excerpt of a FHIR record specifying a vaccine

Example

There follows a description of each attribute in an Immunization resource that is relevant to a proof of vaccination, based on the complete example of an Immunization resource in Listing 2.2

- The `"status"` element is required by FHIR, but is usually `"completed"` in a certificate. Could also be used to indicate that an immunization was not performed, together with the “statusReason” attribute.
- The `"resourceType"` element is always `"Immunization"`, indicating the type of FHIR resource.
- The `"vaccineCode"` element indicates which vaccination was used.
- The `"patient"` element links the Immunization resource to the person subject to this immunization (i.e. the person that received the vaccine and holder of the certificate).
- The `"recorded"` element indicates when the certificate was issued.
- The `"occurenceDateTime"` element indicates when the immunization took place.
- The `"manufacturer"` element indicates who manufactures or markets the vaccine that was used.
- The `"lotNumber"` contains the vaccine production lot number used.
- The `"protocolApplied"` specifies how the immunization was performed, indicating how many doses were given (`"doseNumberPositiveInt"`) and how many are required (`"seriesDosesPositiveInt"`). This is because different patients (e.g. people who have had COVID-19) require different numbers of

doses to complete their immunization. This also allows for flexibility with e.g. booster shots. The `"targetDisease"` is always COVID-19 in this context. Multiple `"targetDisease"`s can be recorded, because some vaccines (e.g. the MMR vaccine) target multiple diseases (e.g. measles, mumps, rubella).

```
{
  "status": "completed",
  "resourceType": "Immunization",
  "vaccineCode": {
    "coding": [{
        "system": "https://ec.europa.eu/health/documents/community-register/html/",
        "code": "EU/1/20/1507"
      }],
    "text": "COVID-19 Vaccine Moderna"
  },
  "patient": {
    "type": "Patient",
    "identifier": {
      "value": "S00004156"
    }
  },
  "recorded": "2021-06-16",
  "occurrenceDateTime": "2021-06-03",
  "manufacturer": {
    "type": "Organization",
    "identifier": {
      "value": "ORG-100031184"
    },
    "display": "Moderna Biotech Spain, S.L."
  },
  "lotNumber": "3002541",
  "protocolApplied": [{
      "doseNumberPositiveInt": 2,
      "seriesDosesPositiveInt": 2,
      "targetDisease": [{
          "coding": [{
              "system": "http://snomed.info/sct",
              "code": "840539006"
            }],
          "text": "COVID-19"
        }]
    }]
}
```

Listing 2.2: Example of an Immunization resource

2.2. Identification

A proof of vaccination must be securely linked to the certificate holder in order to ensure that it cannot be copied and used by another person.

Many systems make use of a passport or national Identity Document (ID) card that certificate holders must present alongside the certificate itself to establish the link between the proof of vaccination and the person. [11] The passport or

ID are well suited to this, because forgery or use of a document not applying to person carrying it are illegal, thus the penalty for simply taking e.g. a sibling's ID and certificate is quite severe (up to three years in prison and a fine). [27]

There are different ways of linking a certificate to an ID. Many systems choose to include the full name (i.e. first and last name) and the date of birth on the certificate. During verification, the information on the certificate must then be compared to the presented ID. Another possibility is the "document number". Each ID is assigned a unique document number, which can be included in the certificate instead of the full name and date of birth.

2.2.1. Machine Readability

The Swiss passport and ID card are both examples of Machine Readable Travel Documents (MRTD) as specified in the International Civil Aviation Organization's (ICAO) document 9303. [19] The ICAO is, among other things, a standards body creating standards relating to air travel. It is important to note that the ICAO does not itself have any regulatory authority. It makes recommendations, which member states may then choose to enforce. [18]

The ICAO document 9303 stipulates that all MRTDs must have a Machine Readable Zone (MRZ). An MRZ is designed to be easily read using machine vision, thus a font (OCR-B), position, size, and it's contents are well defined. [19, part 3, section 4] The exact format of an MRZ depends on the size of travel document (i.e. passport booklets are TD3, whereas ID cards are TD1). [19, parts 4, 5] Since TD1 and TD3 sized MRTDs cover the majority of Swiss citizens, the thesis will focus on those.

An MRZ contains the name, date of birth, and gender of the ID holder, as well as the document type, document number, and expiration date (usually the same information that is also contained in human-readable format elsewhere on the ID) Check digits are used to ensure that the information was read correctly. [19, part 3, section 4.9]

An example of a MRTD of size TD1 is given in Figure 2.1.

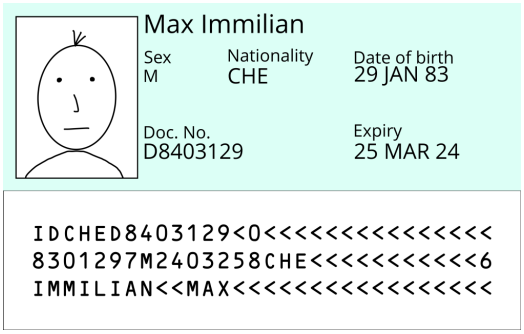


Figure 2.1.: MRTD Example (MRZ is the white area)

2.2.2. Security

Identity documents often have multiple levels of security to protect against forgery. Some of these features enable a layperson to detect a forgery, while others may require forensic analysis. ICAO document 9303 part 2 states the mandatory and optional security features of MRTDs and how to detect and verify them. [19]

These features may involve the material an MRTD is made from being difficult to obtain, manufacturing techniques, or even cryptographic signatures on e-IDs.

An inspection in visible wave-lengths may provide defense against the most basic forgery techniques, but more advanced readers usually scan an MRTD in the infra-red, visible, and ultra-violet spectrum.

2.3. Chain of Trust

Digital signatures enable verifying that a given message was written by a known sender (authenticity), and that that message wasn't tampered with (integrity). This concept is fundamental to any proof of vaccination system, as the problem boils down to: A certificate holder (*Charlie*) claims to a verifier (*Victor*) to be vaccinated by a doctor (*Bob*), by presenting a certificate (proof of vaccination). To verify that, A) *Bob* issued this certificate (authenticity) and B) *Charlie* did not alter it (integrity) a digital signature is used. *Victor* checks the validity of the certificate signature against the known public key of *Bob*. If the signature is valid, *Victor* can be reasonably sure that *Bob* created the certificate and that the proof of vaccination is valid. This essentially creates a "Chain of Trust" (CoT) from *Bob* to the certificate, i.e. iff *Victor* knows and trusts *Bob*, and the signature is valid, the certificate must be valid.

Given *Charlie's* vaccination record V_c, *Bob's* key pair K_R (consisting of the public key K_R^+ and the private key K_R^-) and an identifier for K_R, $H_R \equiv \text{hash}(K_R^+)$ the proof of vaccination C is constructed as follows:

$$\begin{aligned} S_{V_c} &= \text{sign}(V_c, K_R^-) \\ C &= H_R + S_{V_c} + V_c \end{aligned}$$

To verify the validity of the proof of vaccination C, *Victor* uses H_R to look up *Bob's* public key K_R^+ in his list of known and trusted certificate issuers, then checks the authenticity and integrity of V_c by verifying the signature:

$$\text{verify}(S_{V_c}, K_R^+)$$

In the above example, *Bob* is the ultimate trust root of the certificate for *Victor*. In that case, the CoT has a single link (i.e. *Bob's* signature of the certificate).

However, it may be desirable to extend the CoT so that *Bob* is not directly issuing proofs of vaccination. Consider the following example, in which *Bob* authorizes *Alice* to create proofs of vaccination. By making use of the CoT, it is sufficient that *Victor* knows and trusts *Bob* to verify a proof of vaccination issued by *Alice*. In addition to the previously given V_c, K_R, and H_R also consider *Alice'* key pair K_A and an "authorization signature" SA_A such that *Bob* signs *Alice'* public key K_A^+:

$$SA_A = \text{sign}(K_A^+, K_R^-)$$

A proof of vaccination C with a longer Chain of Trust can now be constructed as follows:

$$\begin{aligned} S_{V_c} &= \text{sign}(V_c, K_A^-) \\ C &= H_R + SA_A + K_A^+ + S_{V_c} + V_c \end{aligned}$$

To verify the validity of the proof of vaccination C, *Victor* uses H_R to look up *Bob's* public key K_R^+ in his list of known trust anchors, then checks the authenticity and integrity of K_A^+ by verifying the authorization signature:

$$\text{verify}(SA_A, K_R^+)$$

This establishes that *Bob* has authorized *Alice* to issue proofs of vaccination. *Victor* then uses an identical process as in the previous example to verify the authenticity and integrity of V_c:

$$\text{verify}(S_{V_c}, K_A^+)$$

Thus we have created a chain of trust with three links as illustrated in Figure 2.2, where A $\equiv H_R$, B $\equiv SA_A$, C $\equiv K_A^+$, D $\equiv S_{V_c}$, and E $\equiv V_c$.

This shows that iff *Victor* can rely on the trust anchor A, he can establish that the vaccination record E can be trusted. If however, any link in the CoT breaks (i.e. a signature is invalid), the presented proof of vaccination is no longer valid.

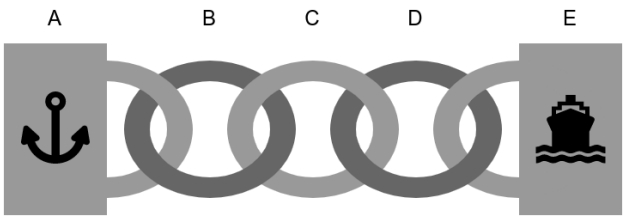


Figure 2.2.: Chain of Trust (CoT)

2.4. Encoding

2.4.1. Objects

To transport structured data it must be encoded or serialized. For the use case of a proof of vaccination encoding objects is especially interesting, as a vaccination record is a structure of nested objects (see section 2.1).

One way to serialize an object for transport is using JavaScript Object Notation (JSON) as defined in ECMA-404. [5] JSON is a subset of JavaScript, and supports two different modes of nesting: the first is objects (aka. dictionaries or key-value pairs) which map a name or key to a value (a value can be a string, number, boolean, null, object, or array) the second is arrays, which contain a list of values (as described before).

JSON has the advantage of being widely supported due to the popularity of JavaScript, but it also has some disadvantages: firstly, it does not support values of binary data, which would be useful when transporting cryptographic material. Secondly, it is fairly verbose (i.e. space in-efficient), which can be prohibitive in low-bandwidth communication.

CBOR, the Concise Binary Object Representation, as defined in RFC 8949, provides a solution to both of these downsides. [2] It is based on the JSON data model, supporting the same value types and nesting modes, but also supports binary data as a value type. Additionally, CBOR is designed to provide fairly small message sizes. [3] A JSON encoded object only contains printable characters, while a CBOR encoded object is a byte stream that also contains non-printable characters. This leads to the main downside, that an object encoded as CBOR is no longer human readable, but that can be mitigated by use of diagnostic tools such as cbor.me.

An important feature of CBOR for the purpose of cryptographically signing CBOR objects is that the encoded representation of objects (called “maps” in CBOR) has a well defined, deterministic key order. According to the standard, “The keys in every map MUST be sorted in the bytewise lexicographic order of their deterministic encodings.” ([2, Section 4.2.1]). This makes it possible and convenient to not only check signature validity against the exact bytes an object was delivered as, but also against a re-encoded version of that object with identical data contents.

The CBOR Object Signing and Encryption (COSE) extension on the CBOR standard specifies how to sign and/or encrypt objects using CBOR. It was inspired by the JSON Object Signing and Encryption (JOSE) standard, but takes advantage of the above described features of CBOR. COSE also defines a message structure to be used. [25]

2.4.2. 2D Barcodes

A 2D barcode, as opposed to a 1D barcode, encodes data in two dimensions. The most prominent example of this is the Quick Response (QR) Code, which was developed in the 1990s by Denso Wave. [29] Others include Aztec Code, Data Matrix, and PDF417. Denso Wave has patented the QR-Code, but allows it to be used free of charge as long as users follow the specification. [30]

QR Codes have many different uses, but they all rely on transmitting some piece of information to the user via a camera. They can be many different sizes (called "Versions"), which determine the amount of data that can be contained, from Version 1 (21x21 pixels) to Version 40 (177x177 pixels). [32] QR Codes make use of different levels of error correction, which allows them to be read even if they are partially damaged or obscured (e.g. torn paper, reflection on screen). The error correction level is configurable from L, M, Q, to H which allow $7\,\%$, $15\,\%$, $25\,\%$, and $30\,\%$ of data bytes respectively to be recovered. [31]

QR Codes also offer different encoding modes, such that certain limited types of information can be stored more efficiently. The following examples are at error correction level L: a QR Code can contain $2.953\,\mathrm{kB}$ of binary data, 7089 digits of numerical information, and 4296 characters in "alphanumeric" mode which are restricted to: the capital letters `A-Z`, the digits `0-9`, and 9 special characters (i.e. `$ % * + - . / : <space>`). [20, section 7.4.4]

2.4.3. Binary

Many communication channels (notably 2D barcodes and JSON), do not support direct binary data. The binary data must usually be encoded in such a way, that it only contains printable characters (or even a subset of those). The most popular such encoding is Base64 as specified in RFC 4648. It uses 64 characters of the ASCII alphabet (`[A-Za-z0-9+/]` and a special padding character `=`) to represent 6 bits each, such that 3 bytes with 8 bits ($3 * 8 = 24$ bits) map to 4 characters with 6 bits ($4 * 6 = 24$ bits). The output is padded with `=`, such that is it always a multiple of 4 characters. The standard also specifies a URL- and filesystem-safe version, called "Base64url", which uses `-` and `_` instead of `+` and `/` respectively. [24].

Recently a variation of Base64, Base45, has been specified for use with QR and Aztec codes. It essentially works the same as Base64, but uses a more restricted set of the 45 characters allowed in a QR Codes alphanumeric mode, encoding 2 bytes per 3 characters. It is still in the draft stage and shouldn't be used or referenced because it isn't stable yet. [23]

Despite that, it has been widely implemented in the different proof of vaccination systems, because it allows a more efficient encoding of data. This can be shown by multiplying the capacity of the QR Code, which differs by mode, with the efficiency of the encoding ($\frac{\text{bits in}}{\text{bits out}}$):

$$\begin{aligned}
\text{binary mode with Base64} &= 2953 \times \frac{18}{24} \approx 2214\,\mathrm{B} \\
\text{binary mode with Base45} &= 2953 \times \frac{16}{24} \approx 1968\,\mathrm{B} \\
\text{alphanumeric mode with Base45} &= 4296 \times \frac{16}{24} \approx 2864\,\mathrm{B}
\end{aligned}$$

It is obvious that Base45 is a less efficient encoding (≈ 0.67) than Base64 (≈ 0.75), but that is canceled out by the ability of a QR Code to store more ($1.45\times$) symbols in alphanumeric mode than in binary mode.

3. Requirements

This chapter describes the requirements for a proof of vaccination system, based on the task description (Appendix A) and prior work done in Project 2 [26].

To describe the requirements and use cases, several personae that were created for this purpose in Project 2 are updated and also used here.

The requirements are divided into functional (section 3.1) and non-functional (section 3.2) requirements.

Personae

An **Issuer** issues proofs of vaccination to vaccinated persons, either at the time of vaccination or at a later date through the information contained in a HIS. An Issuer is authorized by the Trust Root.
(also known as Alice)

The **Trust Root** is a central, trusted authority, such as the FOPH, FOITT, or HIN. The Trust Root signs Issuers' key pairs, in order to prove that they are trustworthy and authorized to issue proofs of vaccination. The Trust Root publishes a list of revoked key pairs in case an Issuer is found to not be trustworthy. (e.g. found issuing proofs of vaccination to non-vaccinated people)
(also known as Bob)

The **Certificate Holder/Owner** (aka. Patient) is a vaccinated person carrying a proof of vaccination. Using the proof of vaccination, they can prove a Verifier that they are vaccinated, and thus gain access to restricted areas (e.g. air travel, fitness centers, restaurants, or clubs)
(also known as Charlie)

The **Verifier** has the responsibility of verifying proofs of vaccination. To do this, the Verifier only has to know and trust the Trust Root, not every Issuer. This role may be carried out by border agents, bouncers or in areas where a passport is required, the Verifier may also be an automated system making use of MRTDs.
(also known as Victor)

3.1. Functional Requirements

F01	Data Entry
Scenario	Alice enters the data for a new certificate into the system.
Module	Issuer
Priority	Mandatory
Description	Alice enters the necessary data for a new certificate into the system. The system performs a plausibility check on the given data. The data must conform to the FHIR data schema outlined in Appendix B: data_schema.json which was developed during Project 2. [26]
	The certificate includes the following data as part of that schema: ▶ Date of vaccination ▶ Vaccine Lot number ▶ A patient identifier (an ID document number) ▶ Vaccine manufacturer ▶ How many doses Charlie received ▶ How many doses Charlie is expected to receive ▶ The disease targeted by the vaccine

Table 3.1.: Functional Requirement 01 - Data Entry

F02	Certificate Issuance
Scenario	Alice issues a certificate to Charlie
Module	Issuer
Priority	Mandatory
Description	Alice issues a certificate to Charlie. This means the system creates an ARCTIC certificate.

Table 3.2.: Functional Requirement 02 - Certificate Issuance

F03	Certificate Handout
Scenario	Alice gives the completed ARCTIC certificate to Charlie
Module	Issuer
Priority	Mandatory
Description	Alice can deliver the certificate to Charlie in one or more of the following ways: ▶ Printed on paper ▶ Digitally as a PDF, e.g., as a download link ▶ Charlie scanning a QR-Code on Alice's system.

Table 3.3.: Functional Requirement 03 - Certificate Handout

F04	proof of vaccination to a Verifier
Scenario	Charlie proves to Victor that he is vaccinated.
Module	Checker
Priority	Mandatory
Description	Charlie shows Victor his ARCTIC QR Code and the ID associated with it. Victor scans the QR Code using the verifier app and is then prompted to scan the MRZ of the presented ID. If the certificate is valid and the ID matches, the verifier app shows Victor that everything is OK. If not, the app displays a corresponding error message. Victor then checks if the photo on the ID matches the person presenting the ARCTIC. If this matches as well, Charlie is cleared to enter.

Table 3.4.: Functional Requirement 04 - proof of vaccination to a Verifier

F05	proof of vaccination to an automated system
Scenario	Charlie uses his ARCTIC certificate to access an ID restricted area, e.g. an airport.
Module	Access
Priority	Mandatory
Description	Charlie uses his ARCTIC certificate to pass an automated gate by presenting to the machine his ARCTIC QR Code as well as ID. The machine automatically checks the validity of the certificate and the ID using the MRZ and RFID Chip. If everything matches the gate opens, if not Charlie must seek a human verifier.

Table 3.5.: Functional Requirement 05 - proof of vaccination to an automated system

F06	Certificate Renewal
Scenario	Alice renews Charlies ARCTIC
Module	Issuer
Priority	Optional
Description	Because the ARCTIC certificate is bound to an ID, it must be possible to renew the certificate in case the ID expires. In order to get his certificate renewed, Charlie takes his old and new IDs, as well as old certificate to Alice. She verifies it and issues a new certificate with the same data, but a new patient identifier (bound to the new ID).

Table 3.6.: Functional Requirement 06 - Certificate Renewal

3.2. Non Functional Requirements

Q01	Non-Falsifiable
Scenario	An ARCTIC cannot be forged by an attacker.
Priority	Mandatory
Description	The data on a ARCTIC certificate must be secured with a cryptographic signature by the Issuer, which is checked by the Verifier. This means that a certificate cannot just be generated out of thin air without knowing the private key of an Issuer. If an attacker copies a certificate, they won't be able to show the corresponding ID.

Table 3.7.: Non-Functional Requirement 01 - Non-Falsifiable

Q02	No internet connection required
Scenario	Issuance and Verification are possible without an internet connection.
Priority	Mandatory
Description	A certificate can be issued by Alice without an internet connection. An internet connection may be required for the key distribution, but not for issuing certificates. An ARCTIC certificate can be verified by Victor without an internet connection. An internet connection may be required for the key distribution, but not for verifying. All data, except for the trust root public key is contained within the certificate.

Table 3.8.: Non-Functional Requirement 02 - No internet connection requried

Q03	Anonymity
Scenario	An ARCTIC certificate is pseudonymous unless combined with an ID.
Priority	Mandatory
Description	No personally identifying data of Charlie is stored in the certificate. An attacker can only access the ID document number and vaccination information if they get hold of the certificate. The ID is required to resolve back to Charlie as a person.

Table 3.9.: Non-Functional Requirement 03 - Anonymity

Q04	Privacy
Scenario	Charlies data remains between Alice and Charlie.
Priority	Mandatory
Description	The vaccination data is only included in the certificate, and never sent to a central authority.

Table 3.10.: Non-Functional Requirement 04 - Privacy

Q05	Usability
Scenario	Victor does not need to perform manual checks to verify an ARCTIC certificate.
Priority	Mandatory
Description	Victor scans the ARCTIC QR Code and an ID using the verifier app, which tells Victor if that combination is valid. The QR Code of the ARCTIC and the MRZ of the ID must be readable without much hassle.

Table 3.11.: Non-Functional Requirement 05 - Usability

4. Solution

The solution I propose to the problem of proof of vaccination is the "Automatically Readable Chain of Trust-based Immunization Certificate", ARCTIC for short. It has a less centralized approach than other systems with a focus on a convenient, yet sound verification procedure through automation. It also has good resilience against denial of service attacks and works well in low-data environments.

4.1. System Overview

In Figure 4.1 the general architecture is laid out. The red-dashed arrows represent actions which require a periodic remote-connection between participants, while the blue-solid arrows show actions which either require no connection or only a local (face to face) one. This shows that the essential actions of issuing, holding, and verifying a certificate do not require any connection to the outside world, which is an advantage for both privacy and availability.

To issue certificates, the Issuer must be authorized by the Trust Root. This establishes the beginning of a Chain of Trust (CoT). If an Issuer turns out to be untrustworthy (e.g. issuing fraudulent certificates) they are added to the Issuer Revocation List (IRL). The IRL is periodically checked by the Verifier to ensure they stay up to date. For the CoT to work, the Verifier must know and trust the Trust Root, but not each individual Issuer.

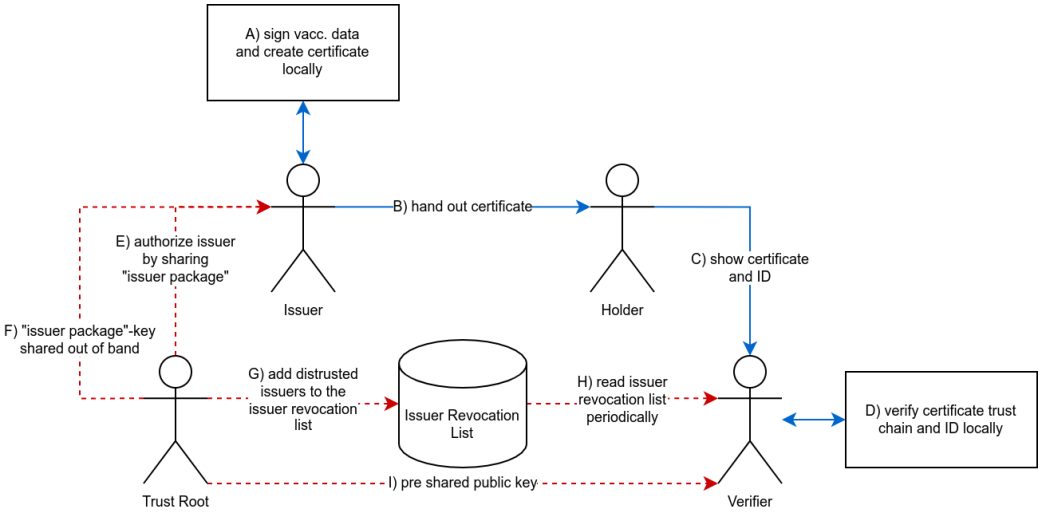


Figure 4.1.: ARCTIC System Overview

4.2. Protocol

A proof of vaccination in the ARCTIC system takes the form of a container that contains the following five sections (as shown in Listing 4.1), that together form a Chain of Trust as described in section 2.3.

- The `"data"` element contains a FHIR Immunization resource (see section 2.1) as a nested object.
- The `"dataSignature"` element is a digital signature by the Issuer that ensures the authenticity and integrity of the `"data"` element. The input to the signature is the CBOR encoded (see subsection 2.4.1) Immunization resource, signed by the Issuer's private key.
- The `"issuerKey"` element is the X.509 encoded public key of the Issuer.
- The `"authorizationSignature"` element is a digital signature by the Trust Root that authorizes the Issuer's key pair to be used for issuing. It's input is the X.509 encoded public key of the Issuer (same as the `"issuerKey"` element), signed by the Trust Root's private key.
- The `"trustRootIdentifier"` element is a Base64 encoding of the SHA-256 hash of the Trust Root's public key. This is used by the Issuer to identify which Trust Root generated this certificate.

The cryptography used for these signatures is based on Elliptic Curves (EC), specifically the Elliptic Curve Digital Signature Algorithm ECDSA, because EC cryptography provides good security at with short key lengths. [21] This is important when transmitting such cryptographic material through low-bandwidth channels i.e. 2D barcodes.

```
{
  "data": {<fhir Immunization resource as a nested object>},
  "dataSignature": "<binary data omitted>",
  "issuerKey": "<binary data omitted>",
  "authorizationSignature": "<binary data omitted>",
  "trustRootIdentifier": "<SHA-256 hash in base64 omitted>"
}
```

Listing 4.1: ARCTIC Container Format

4.2.1. Encoding

To hold and present an ARCTIC, the container is packaged into a convenient QR Code that can either be presented on a piece of paper or a device with a screen, as long as the side length of the QR Code is at least $30\,\mathrm{mm}$.

The encoding of the QR Code takes place in the "pipeline" shown in Figure 4.2:

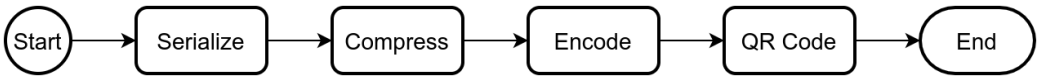


Figure 4.2.: ARCTIC to QR Code pipeline

1. The given ARCTIC container (as previously described) is first CBOR *serialized*.
2. The resulting binary stream is the *compressed*, which reduces the already small CBOR size by another $15\,\%$ to $\approx 780\,\mathrm{B}$. The exact size depends on the data in the Immunization resource and the random numbers used in ECDSA.
3. The compressed binary stream is then Base64 *encoded* for compatibility with QR Codes (see subsection 2.4.3). At the end of this stage, the ASCII string `vaccCERT` is prefixed to the encoded string, to distinguish ARCTIC QR Codes from other QR Codes and make the scanning more convenient.
4. The QR Code is generated in binary mode, using error correction level L (see subsection 2.4.2).

The Base64 encoding described in step 3 was chosen because of how well supported it is. A more efficient variant of this pipeline would use Base45 instead of Base64 and generate the QR Code in alphanumeric mode, for the reasons described in subsection 2.4.3. Another variant would put the compressed binary stream into the QR Code in binary mode without any encoding to printable characters, but this is unsupported in several popular QR Code libraries.

4.2.2. Issuer Revocation

The issuer revocation list contains an entry per line which are formatted as follows:

```
<issuer>;<signature>
```

Where the `<issuer>` is the Issuer Id, a Base64 encoded SHA-256 hash of the issuer public key (similar to the Trust Root Identifier), and the `<signature>` is the Base64 encoded signature of the Issuer Id, signed by the Trust Root. The IRL is made accessible to the Verifiers as a UTF8 encoded text file served on a web server over HTTPS.

4.2.3. Verification

To verify an ARCTIC the Verifier's App takes the following steps:

1. Scan the presented QR Code and identification document (MRTD)
2. Decode it into the ARCTIC container by reversing the encoding steps
3. Verify the authenticity and integrity of the CoT as described in section 2.3
4. Match up the patient identifier in the ARCTIC and the scanned MRTD
5. Check any additional business rules (e.g. days since vaccination, type of vaccine, etc.)

If anything unexpected occurs in steps 2, 3, 4, or 5, such as an invalid signature or the wrong ID being presented, the ARCTIC is rejected.

4.3. Processes

The following processes are visualized in Figure 4.1:

A) When a vaccination is administered, the issuer enters the information required to create a FHIR Immunization record (see section 2.1). [1] This record is then processed as described in section 4.2 to create an ARCTIC container, which is then encoded to a QR Code.

B) The QR Code created during A) is printed and handed to the patient/holder.

C) When prompted by a Verifier, the Holder must present both the QR Code from B) and associated ID. The QR Code can either be on paper or a screen, making it technology agnostic.

D) The verifier uses the verification app to scan the QR Code and ID presented by the Holder. The verification app executes the verification procedure described in subsection 4.2.3 and tells the verifier if the certificate is considered valid i.e. is the chain of trust intact, does the presented ID match the certificate, and are the business rules fulfilled.

[1] the vaccine used,the lot number, the vaccination date, the protocol used (how many doses), and the document number of the identification document presented by the patient

E) To authorize an Issuer, the Trust Root shares an "issuer package" containing the beginning of a chain of trust (Trust Root Identifier, Issuer Key Pair, and Authorization signature. See section 2.3)

F) Because the Issuer private key is contained in the issuer package, the package is encrypted. The encryption key is shared out of band with the package.

G) If an Issuer turns out to be untrustworthy, they are placed on the Issuer Revocation List (IRL), which is formatted as described in subsection 4.2.2.

H) The Verifier must periodically (i.e. daily) read the IRL mentioned in G) and reject certificates issued by those Issuers, even if they are valid otherwise.

I) The Verifier must know the Trust Root's public key. This key is pre-shared, and is not be updated while in operation.

4.3.1. Information Distribution

Table 4.1 shows how various pieces of information are distributed in the ARCTIC system. It uses the notation introduced in section 2.3 for the different components of a Chain of Trust. The novel notation used here is "ID", which represents the identity document presented by the Holder, or the Holder's identity in general.

Any values contained within the column of a participant are known solely to that participant in that and any previous steps. Values spanning multiple columns are shared in that step between the participants. Participants don't forget anything. The common knowledge shared among all participants is the identity of the Trust Root (K_R^+ and H_R).

It's clear from this visualization, that the ID and vaccination information V_c, contained in C, is never seen by the central authority i.e. the Trust Root.

ID	Step	Trust Root	Issuer	Holder	Verifier
A	Initial	K_R^-		ID	
B	Authorize Issuer	SA_A, K_A^+, K_A^-			
C	Vaccinate Patient		V_c, ID		
D	Issue Certificate		$C = H_R + SA_A + K_A^+ + S_{V_c} + V_c$		
E	Hand out Certificate		C		
F	Verify Certificate			C, ID	

Table 4.1.: Information Distribution with ARCTIC

5. Prototype

To prove that ARCTIC works as proposed in chapter 4, a prototype was built, which encompasses the necessary functionality. There are four distinct parts to the prototype: a general foundation of models and utilities, a desktop app for issuing certificates, an android app for verification, and gateway-stand-in for the automated access.

5.1. Foundation

Because a model of the ARCTIC as well as various utilities for signature generation and verification, encryption, encoding, and serialization were necessary on three different platforms Java was chosen to build the foundation of the prototype.

The model consists of the entities necessary to build a FHIR Immunization resource as described in section 2.1, including some hard-coded value sets for the various vaccines and protocols available.

For the CBOR encoding, the popular Jackson library was used with some custom serializers for the model classes and cryptographic keys. Public keys are encoded in the X.509 format, the `java.security`-implementation's default, while for signatures the raw byte streams are written to the CBOR object. Notably, CBOR's deterministic key ordering (as described in section 2.4) is not fully supported in Jackson. In the JVM version of the library, map keys are encoded in the order that they are present in the Java Class that is being encoded (although this order can be manually overridden with the `@JsonPropertyOrder` annotation), while in the Android version of the library the keys are encoded in "Length-First Map Key Ordering", such that: " If two keys have different lengths, the shorter one sorts earlier; If two keys have the same length, the one with the lower value in (bytewise) lexical order sorts earlier. " ([1, Section 3.9]) This corresponds to the old version of the CBOR specification. To avoid problems with this, the shared model in the ARCTIC prototype implementation specifies the length-first encoding manually using the `@JsonPropertyOrder` annotation.

As discussed in section 2.4 Base45 is a more efficient encoding, when compared with Base64, when it's written to QR Codes. For ease of implementation, Base64 was used in the prototype, because it is well supported in the Java standard library. A version 2 of ARCTIC should use Base45 to improve data transmission efficiency.

5.1.1. Administration

subsec:proto.admin In order to perform the duties of the Trust Root, authorizing and revoking issuers, two simple Java executables were created within the Model project. They are manually run directly from the IDE for simplicity of implementation.

The authorization program takes a file containing the Trust Root private key, as well as a name and password for the package. When run, it generates a `<name>.package.bin` file that can be shared with the Issuer. The password to the package must be shared separately.

The revocation program takes an issuer package file and the corresponding password and outputs a line of text that can be added to the revocation list. The line is manually appended to the issuer revocation list, which is a simple text file residing on a web server.

5.2. Issuance

The issuance application is a JavaFX app, meaning it can be run on most desktop operating systems. JavaFX is a framework used to build graphical desktop applications in Java. It is often considered the successor to the Swing Widget Toolkit.

The user interface is divided into three parts, which from left to right are: the issuer identity management, the data entry, and the certificate output.

The issuer identity management section allows importing and "activating" issuer packages. When an issuer package is activated, the password must be entered to unlock it. If successful, it is used to issue certificates. The active identity is shown above the list.

When issuing a certificate, the issuer enters the relevant data in the Data Entry section. They have the choice of the different vaccines and protocols that exist in the value sets. Basic data validation is performed on the entered data: no fields can be left empty, and the selected vaccination date must be in the past. Upon clicking "Issue Certificate" the QR Code appears on the right, and the form is reset for the next certificate.

The certificate is generated using the shared library, and encoded into a QR Code using the ZXing ("Zebra Crossing") library. The generated QR Code can be exported as a PNG, or the PNG can be included in a PDF to make printing easier. The PDF generation is done using the openhtmltopdf library, which takes an HTML file that's used to define the layout, and look and feel of the final PDF. An example of this PDF is included as Appendix C.

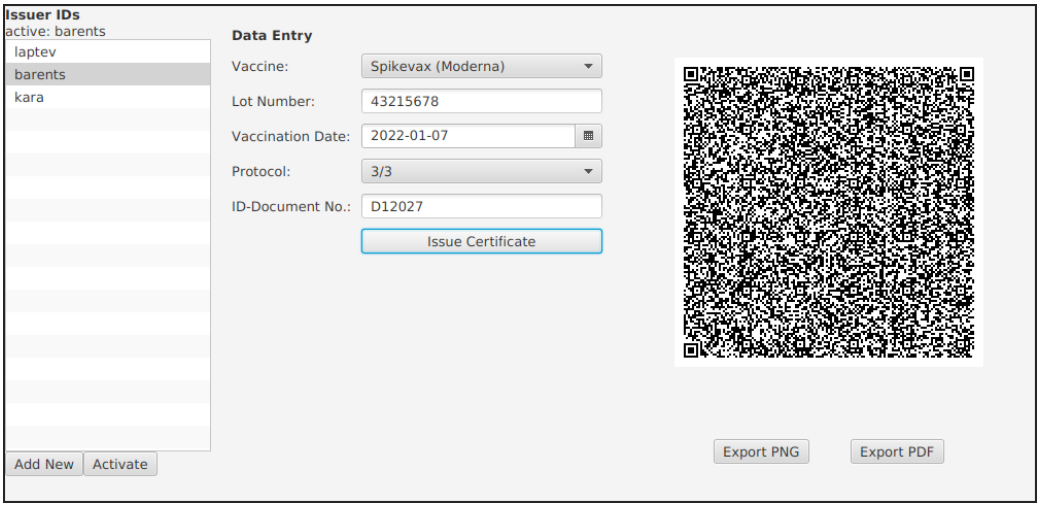


Figure 5.1.: Issuance UI

5.3. Verification

The verification app was built in Kotlin (as is standard for Android), but as Kotlin has compatibility for Java libraries that was no problem. When verifying a certificate, the Verifier first scans the QR Code (as shown in Figure 5.2). This was implemented using the ZXing Android Embedded library, which is based on the ZXing library referenced in section 5.2.

Next, the Verifier scans the MRZ of the presented MRTD (as shown in Figure 5.3). This was implemented by extracting and modifying the MRZ reader component from the polling-station-app project into it's own library that can be used outside the polling-station context.

Finally, the app verifies the trust chain using the shared library, checks that the Issuer isn't on the IRL, that the scanned ID matches the ID number in the certificate, and that the business rules are fulfilled. The result of these checks are then displayed to the Verifier. If all the above checks are OK, the certificate is considered valid (see Figure 5.4). If there is a failure condition, that is displayed to the verifier as well, e.g. if the presented ID doesn't match the certificate (see Figure 5.5), if the issuer is revoked (see Figure 5.6), or if multiple issues are present (see Figure 5.7). Using the "Scan next" button, the Verifier can verify the next certificate. A production version of the verification app would hide this information from the Verifier to reduce complexity, but for a prototype it's useful and illustrative.

The current prototype reloads the IRL on restart to make testing convenient. A production implementation would store the IRL locally and only reload once a day. The IRL signatures are checked on reload, as described in subsection 4.2.2.

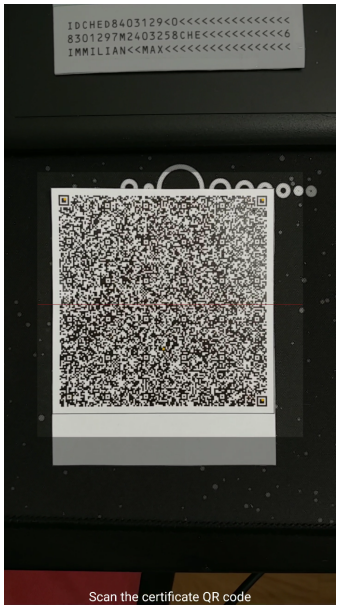


Figure 5.2.: QR Code Scanning

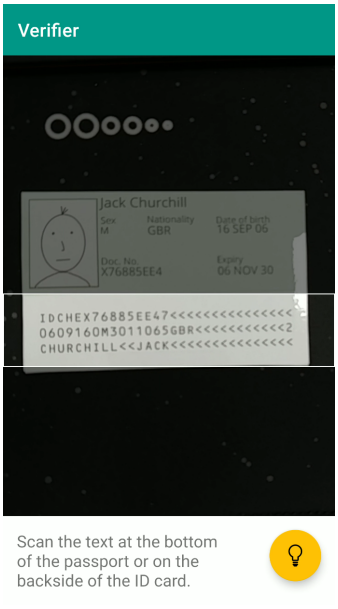


Figure 5.3.: ID Scanning

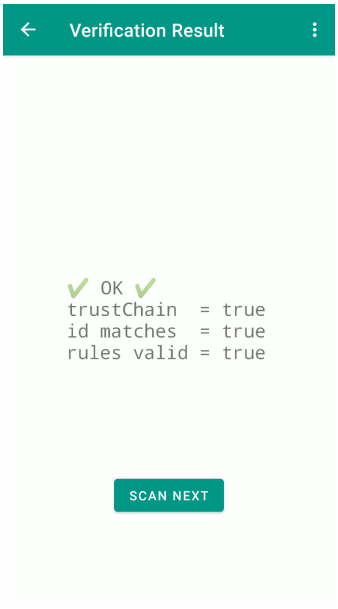


Figure 5.4.: Verification OK

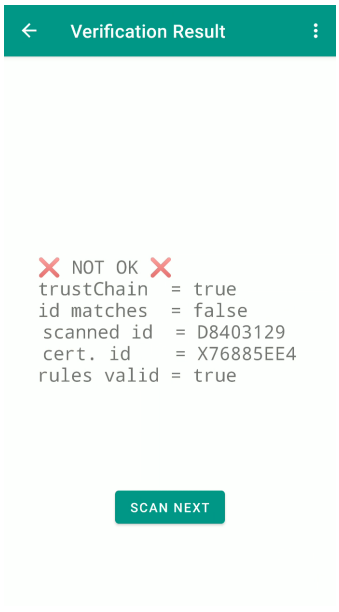


Figure 5.5.: Bad ID Match

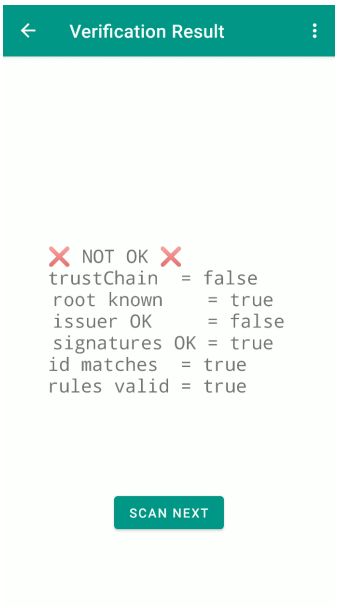


Figure 5.6.: Issuer Revoked

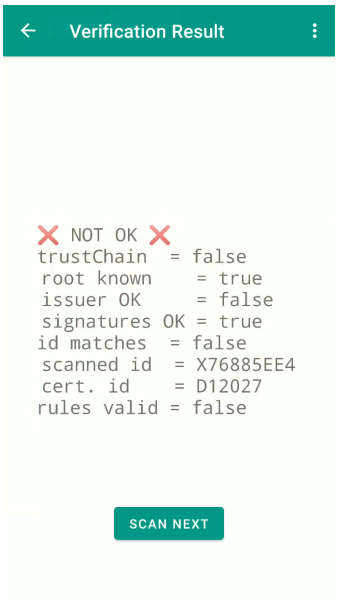


Figure 5.7.: Verification NOK

5.4. Automated Access

The gateway-stand-in for the automated access prototype has four major components: an Android device to act as a scanner, a Raspberry Pi (RPi) single board computer as a control unit, four LEDs for user feedback, and a 3d printed frame to hold it all together.

The verification app described in section 5.3 also offers a “Gateway Mode”. In that mode, the app connects to a Mosquitto MQTT Broker on the RPi to communicate the scanning results. The communication is achieved through a USB cable using SimpleRT reverse tethering, which allows the RPi to act as a router for the Android device. The RPi receives three different events from the Android device: ready to scan QR Code, ready to scan ID, and the scanning results (QR Code contents and ID number). These events are processed in a Java application, which uses the shared library to verify the scanned certificate and an LED driver script written in Python to control the LEDs using four of the RPi’s 40 GPIO pins.

Figure 5.8 shows the 3d model with three important features: 1) the feedback LEDs, 2) a cutout for the Android device camera, and 3) a position for the user to place the ID. In Figure 5.9 the wiring and placement of an example ID is shown, while Figure 5.10 and Figure 5.11 show the LEDs used for user feedback. They light up to tell the user to: first present the QR Code (A), then present the ID (B), and finally light up either green (C) if the certificate is valid, or red (D) if the certificate is invalid.

Due to an Android Device being used as a scanner, it only operates in (near) visible wavelengths of light. Actual ID scanners used at e.g. airports make use of infrared and ultraviolet wavelengths to check some of the security features of the MRTDs. More details on the security features of MRTDs are described in subsection 2.2.2.

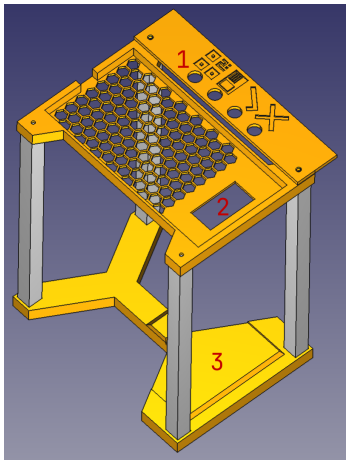


Figure 5.8.: 3D model

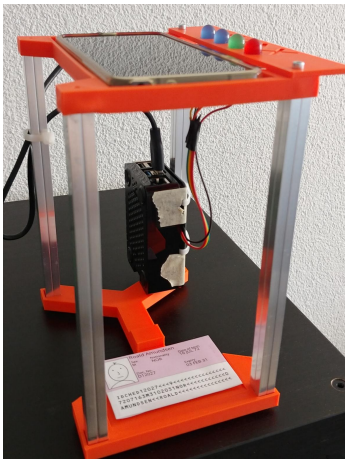


Figure 5.9.: ID placement

Figure 5.10.: LEDs and symbols

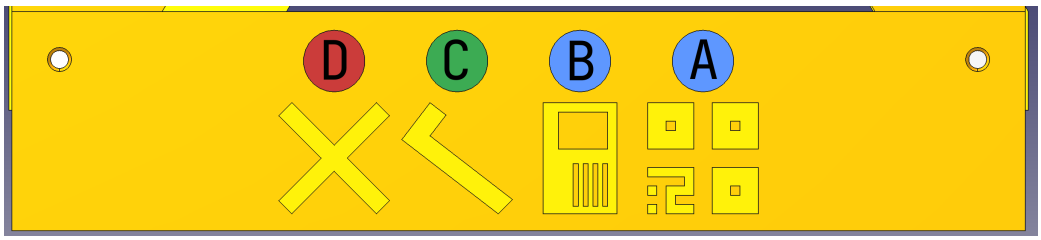


Figure 5.11.: LED Detail

6. Testing

The prototype built in chapter 5 must be tested, to ensure the planned functionality works as designed. These tests will be performed manually, using the requirements and tasks as reference. If a feature is not present or does not work as designed, this will be documented.

6.1. Test Data

To simplify testing and protect the authors identity, several fake IDs were created. This was done by creating a simple MRZ generator, which takes the required information (Listing 6.1) and outputs a properly formatted MRZ with check digits (Listing 6.2). The generated MRZs are then added to the ID template (Figure 6.1).

Various certificates were generated, some intentionally invalid and malformed, to test the Verifier and Automated Access components.

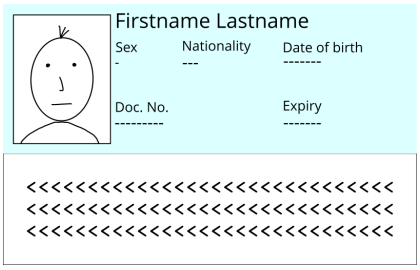


Figure 6.1.: Fake ID template

```
1 val type = "ID"
2 val issuer = "CHE"
3 val docNum = "D1337"
4 val dateOfBirth = "060916"
5 val sex = "M"
6 val expiry = "301106"
7 val nationality = "CHE"
8 val surname = "WILHELM"
9 val givenName = "TELL
```

Listing 6.1: Input into the MRZ generator

```
1 IDCHED1337<<<<9<<<<<<<<<<<<<<
2 0609160M3011065CHE<<<<<<<<<<<8
3 WILHELM<<TELL<<<<<<<<<<<<<<<<
```

Listing 6.2: Output from the MRZ generator

6.2. Test Cases

Each test case is structured the same way: each includes a scenario and description which lays out the actions to be taken. The "Expected" row specifies the desired outcome, while the "Alternatives" are slightly altered actions or results that should also be considered when testing. The last row shows the state of the test execution: success (everything as described), partial (mostly successful with minor deviations), or fail.

C01	Issue a certificate
Scenario	Alice Issues a certificate to Charlie.
Description	Alice Issues a ARCTIC to Charlie, who received his second dose of Moderna today. Charlie uses his ID card as an ID and wants the ARCTIC as a PDF.
Expected	Alice successfully issues a ARCTIC and hands it out to Charlie in the form of a PDF.
Alternatives	A) Alice mistakenly tries to issue a certificate for a future date: she is unable to do so because the system checks data plausibility. B) Charlie has recovered from COVID-19 and only requires a single dose: the certificate states that he has received one of one doses.
Success	see Figure 6.2 and Appendix C as a demonstration.

Table 6.1.: Test Case 01 - Issue a certificate

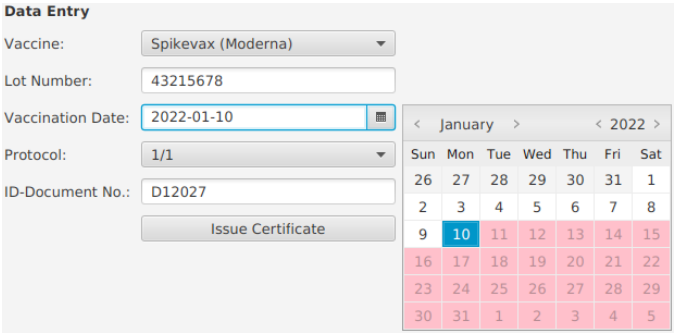


Figure 6.2.: Issuance UI - Calendar showing date restriction

C02	Verify a certificate
Scenario	Victor verifies Charlie's certificate.
Description	Charlie shows Victor his ARCTIC and ID. Victor scans both documents, the app tells him if the ARCTIC and combination of ARCTIC and ID is valid.
Expected	The app shows victor that everything is OK.
Alternatives	▶ An attacker shows a certificate that is not theirs, but copied from Charlie: the app shows Victor that the ID doesn't match. ▶ Someone shows a certificate that was signed with a key that is not signed by the Trust Root: the app shows Victor that the chain of trust is broken. ▶ Someone shows a certificate that was signed with a key that is on issuer revocation list: the app shows Victor that the certificate has an invalid Issuer. ▶ Someone shows a certificate for a vaccination that was two years ago: the app shows Victor that the business rules are not fulfilled. ▶ The ARCTIC and ID are not readable due to bad lighting.
Success	see Figures mentioned in section 5.3 as a demonstration.

Table 6.2.: Test Case 02 - Verify a certificate

C03	Authorize an Issuer
Scenario	Bob authorizes Alice as an Issuer.
Description	Bob authorizes Alice as an Issuer using the certificate authority module. Victor is able to verify a certificate that is issued by Alice.
Expected	Alice is able to successfully issue valid ARCTIC certificates.
Success	

Table 6.3.: Test Case 03 - Authorize an Issuer

C04	Revoke an Issuer
Scenario	Bob revokes an Issuer that was found issuing ARCTIC certificates fraudulently.
Description	Bob adds the malicious Issuer to the issuer revocation list.
Expected	The certificates the malicious attacker issued are no longer valid.
Alternatives	If Victor doesn't check the issuer revocation list, ARCTIC certificates by revoked Issuers are still wrongly considered valid by Victor.
Success	see Figure 6.3 with the revoked issuer and Figure 6.4 with the outdated Issuer Revocation List

Table 6.4.: Test Case 04 - Revoke an Issuer

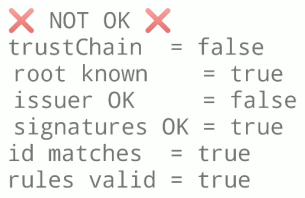


Figure 6.3.: Result of a revoked Issuer

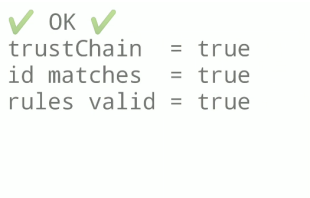


Figure 6.4.: Result with an outdated IRL

C05	Automated Access
Scenario	Charlie uses an automated gate to gain access to an area that requires an ARCTIC certificate.
Description	Charlie presents the Automated Access Module with a valid combination of ARCTIC QR Code and ID.
Expected	After checking the presented items, the Automated Access Module allows Charlie to enter if they are a valid combination.
Alternatives	▶ If the presented combination is invalid, access is denied. ▶ If the Automated Access Module has trouble reading any of the items, an error message is provided to Charlie. *(optional)*
Partial	If a technical issue with scanning either the ID or QR Code is present, there is no feedback.

Table 6.5.: Test Case 05 - Automated Access

C06	Forged ID
Scenario	An attacker uses a forged ID and someone else' ARCTIC.
Description	An attacker uses a forged ID and someone else' ARCTIC to fool a human verifier. The attacker may choose to only forge the MRZ portion of the ID.
Expected	The verifier app shows that the combination of ID and ARCTIC is valid, but the Verifier notices that the ID is fake.
Alternatives	The attacker tries this with the automated gate.
Partial	A human Verifier is able to spot the fake ID when comparing the faces due to the unusual materials used. The automated gateway does not notice anything is wrong. This would be preventable with a professional MRTD scanner, because they are equipped to check security features (see subsection 2.2.2). Another defense might be to check the digital signatures contained in the chip of the MRTD using NFC. However, not all IDs have this feature, notably the Swiss ID card.

Table 6.6.: Test Case o6 - Forged ID

7. Comparison

This chapter explores the Swiss Covid Certificate as well as a fictional centralized system and compares them to the ARCTIC. The Swiss Covid Certificate SCC was chosen because it is relevant for Switzerland, and because it is an implementation of the European Union's Digital Green Certificate (EU DGC) relevant for much of the European Union. A fictional centralized system was chosen because all of the centralized systems investigated were proprietary, and thus little information could be gained about their inner workings. The fictional centralized system discussed here will be referred to as "CENT".

7.1. Swiss Covid Certificate

The Swiss Covid Certificate is mostly an implementation of the European Union Digital Covid Certificate. This was done in order to have easy compatibility with the neighboring countries. The system documentation and implementation are available on GitHub [10].

Design

The Swiss Covid Certificate not only supports proof of vaccination, but also proof of recovery and proof of test result. This means it has some features that are not relevant for proof of vaccination (such as test result transfer codes or `3G/2G/2G+` verification) that won't be further explored here. This first part describes the design elements that are shared with the EU DGC

As the design document for the EU DGC states:

> The verifier of a certificate should be able to establish that:
>
> - The certificate has been issued by an authorized entity
> - The information presented on the certificate is authentic, valid, and has not been altered
> - The certificate can be linked to the holder of the certificate
>
> (EU eHealth Network [9, p.4])

Furthermore, it states in the main design principles and business requirements: "

- Cross-border interoperability
- Data protection [...]
- Data security and privacy by design and by default
- Inclusiveness (especially medium-neutrality)
- Modularity and scalability
- Open standards
- [...]

" (EU eHealth Network [9, p.4–6])

With these goals in mind, the solution proposed by the EU is cryptographic signing with a public key sharing infrastructure to facilitate cross-border interoperability. Privacy and data protection are ensured by using a decentralized system, where each certificate carries the data locally, not sharing health records and personal data in a centralized way. This approach also scales well: by making use of the PKI, member countries can determine their own systems for issuing certificates or assigning signing-privileges, as long as they share the signing keys with the other members.

Medium agnosticism is achieved through use of 2D barcodes (QR Codes), which can either be displayed on a screen or on augmented paper " (i.e. paper certificate with printed machine- readable parts such as barcodes, QR Codes, Machine Readable Zones) " (EU eHealth Network [9, p.5])

Certificate Contents

A EU DGC consists of a document which contains the vaccination data which is wrapped in a container to ensure integrity and authenticity. As shown in in Figure 7.1 the vaccination data is first encoded in the CBOR format. The CBOR document is then signed using COSE and compressed. Finally it's Base45 encoded and a QR Code is created from that text.

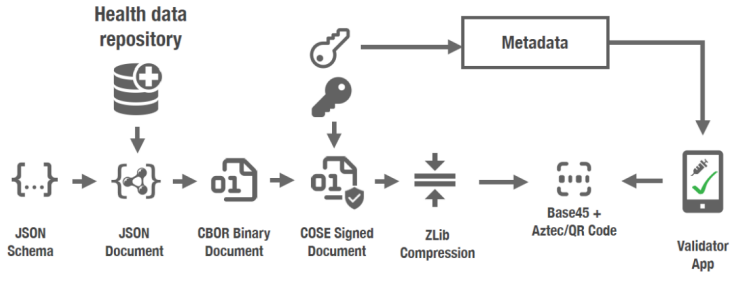


Figure 7.1.: EU DGC data format summary

The “Metadata” in the diagram alludes to the PKI, meaning validator apps know the trusted signing keys.

The vaccination document of a EU DGC certificate is best explained using the example shown in Listing 7.1 EU eHealth Network [8, `simple.json`] Essentially, this format is a re-mapping of the information contained in a FHIR Immunization record to a simpler, less nested structure.

The `nam` element encodes the given (`gn`) and family (`fn`) name of the carrier of the certificate, both as human readable text, and as it would be shown on the MRZ on an MRTD Together with the date of birth (`dob`), this identifies the carrier in combination with an official identification document.

The `v` element contains the vaccination information. This is where, instead of the `v`, a `t` or `r` element could be placed to create a certificate for testing or recovery respectively.

- `tg` - TarGet disease (always COVID-19)
- `vp` - Vaccine or Prophylaxis used
- `mp` - Medical Product approval code
- `ma` - Manufacturer of the vaccine
- `dn` - Dose Number - the number of doses administered
- `sd` - Series Doses - the number of doses required
- `dt` - DaTe of vaccination
- `co` - COuntry of vaccination
- `is` - ISsuer of the certificate
- `ci` - Certificate Identifier (UVCI)

The Unique Vaccination Certificate Identifier (UVCI) has multiple purposes. The most often used, is that individual DGCs can be revoked (e.g. if the name is misspelled or it is discovered that a certificate was fraudulently issued).

However, it could also be used to verify certificates manually, such as over the phone if a “non-augmented paper” version of a certificate is presented. In that case the system is essentially centralized, because UVCIs need to be verified with the issuing member state. To enable this, a UVCI can come in one of three flavours. Common to all three is that they begin with a version and country indicator and end with a check sum. The first flavour is the most modular, containing sections for the issuing entity, vaccine and an opaque unique id. The second only contains an opaque unique id, and the third contains a section for the issuing entity and an opaque unique string. Member states are free to decided how they encode and

generate the various sections, but the UVCI should not contain any personally identifying information and may only consist of alphanumeric characters. [6]

```
{
  "ver": "1.3.0",
  "nam": {
    "fn": "Smith-Jones",
    "fnt": "SMITH<JONES",
    "gn": "Charles Edward",
    "gnt": "CHARLES<EDWARD"
  },
  "dob": "1964-01-01",
  "v": [
    {
      "tg": "840539006",
      "vp": "1119349007",
      "mp": "EU/1/20/1507",
      "ma": "ORG-100031184",
      "dn": 1,
      "sd": 2,
      "dt": "2021-06-11",
      "co": "NL",
      "is": "Ministry of Health Welfare and Sport",
      "ci": "URN:UVCI:01:NL:DADFCC47C7334E45A906DB12FD859FB7#1"
    }
  ]
}
```

Listing 7.1: official DGC example

Swiss Specializations

What the certificate looks like and what it contains cannot be customized, as that is part of the EU DGC specification that every member country must implement if they want to take part.

The Swiss system makes uses of a centralized signing service, where certificates can either be issued through a web interface by authorized persons or through an API by HISes of the Cantons (e.g. vacme.ch).

The centralized service can issue a temporary (valid for 48 h) “Certificate Light” (CL). The CL is only valid in Switzerland, as it is incompatible with the EU DGC specification and contains only the full name, date of birth, and cryptographic signatures, but no medical data. This has two advantages: it does not expose the certificate Holder’s health information and the QR Code is smaller, thus easier to read. To obtain a CL, the Holder uploads their full certificate to the centralized service and in turn receives the CL. This can be done through the Swiss Covid Certificate Wallet app.

Figure 7.2 shows the simplified data flow in the Swiss Covid Certificate system. The red-dashed arrows represent actions which require a remote-connection between

participants, while the blue-solid arrows show actions which either require no connection or only a local (face to face) one.

A) When a vaccination is administered, the issuer sends the vaccination details to the centralized signing service. The signing service then returns the complete SCC certificate.

B) The QR Code and the vaccination details in a human readable form are printed out and handed to the patient/holder. In many cases, the holder also receives their certificate in electronic form through the vaccination appointment system (i.e. `vacme.ch`).

C) When prompted by a Verifier, the Holder must present both the QR Code and an official identity document (e.g. ID card, driver's license, SwissPass). The QR Code can either be on paper or a screen (e.g. the SCC wallet app), making it technology agnostic.

D) The Verifier uses the verification app to scan the QR Code, and if the UVCI doesn't appear on the certificate revocation list. Then, the Verifier manually checks if the presented ID corresponds to the name and date of birth displayed by the verification app.

E) The trust list (a list of public keys authorized to issue certificates) and certificate revocation list (a list of invalid EU DGC certificates) is regularly updated from the centralized EU repository.

F) The Verifier must periodically download the trust list and the certificate revocation list to make sure they're up to date.

G) If a certificate is invalid (as described previously), it's added to the certificate revocation list and no longer considered valid.

The certificate Holder wallet app featured a prominent button to refresh trust list and certificate revocation lists. It also validated the current certificate, followed by an animation featuring a green check mark. This was sometimes used by Verifiers to verify certificates. This is obviously unsound, because the Verifier trusts the phone presented by the Holder instead of scanning the certificate QR Code with a trusted device. No instances of this design flaw being taken advantage of are known, and the Issue is addressed as of January 2022. [28]

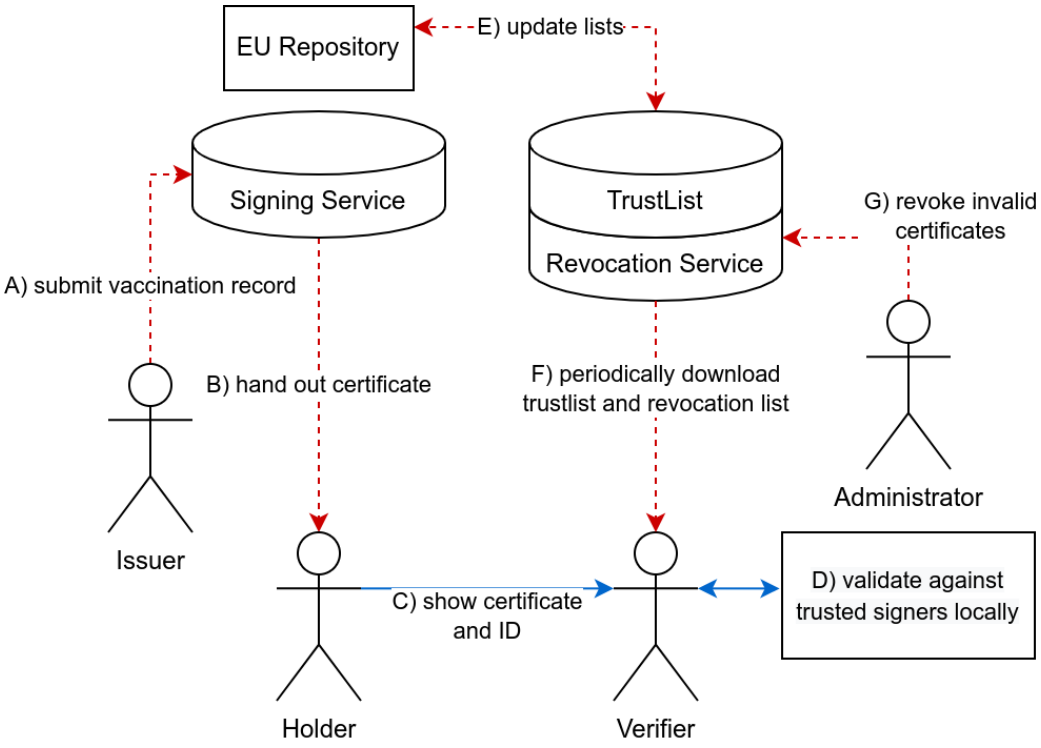


Figure 7.2.: SCC System Overview

7.1.1. Information Distribution

Table 7.1 shows how various pieces of information are distributed in the SCC system. It uses the notation introduced in section 2.3 and subsection 4.3.1 for the different components of a Chain of Trust. The common knowledge shared among all participants is the EU Trust List and the Certificate Revocation List.

It's clear from this visualization, that the ID and vaccination information V_c, contained in C, are seen by the Central Authority, effectively creating a centralized vaccination registry.

ID	Step	Central Authority	Issuer	Holder	Verifier
A	Initial			ID	
B	Vaccinate Patient		V_c, ID		
C	Enter Data	V_c, ID			
C	Issue Certificate	$C = S_{V_c} + V_c$			
D	Hand out Certificate	C			
E	Verify Certificate			C, ID	

Table 7.1.: Information Distribution with SCC

7.2. Centralized

The centralized system depicted here (CENT) is fictional and for illustrative purposes. It's a hypothetical system like it could be run by e.g. a national health service. Many proof of vaccination systems worldwide are poorly documented and proprietary, making it difficult to find a real example, and nigh impossible to figure out how it actually works. Because of this, this fictional system is an amalgamation of information snippets gathered from different systems, as well as hypothetical possibilities.

A real world example of a centralized system would be Argentina's "Carnet Unificado de Vacunación Digital" (CUVD), a digital vaccination card introduced in 2019 and adapted for use during the COVID-19 pandemic. [12] [13] It requires the use of the proprietary eGovernment app "miArgentina", and verifying user identity through a facial scan with that app. [14] [15] It was not possible to determine further technical detail or a detailed description on how verification works.

Figure 7.3 shows the data flow in this system:

A) When a vaccination is administered, that is registered in the central vaccination database.

B) When prompted, the Carrier shows the Verifier their ID

C) The Verifier scans the ID and checks with a centralized service if the Carrier is allowed to enter.

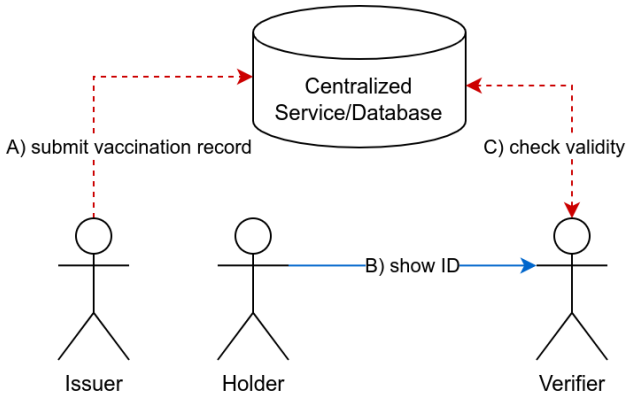


Figure 7.3.: CENT System Overview

As an alternative to checking the central database using a smartphone over the internet, it would also be possible to have a telephone service that the Verifier can call, manually read the details of the ID, and receive their answer. CENT doesn't require the Holder to carry any additional items, because their ID is sufficient.

It is possible for the Verifier to send a location along with the ID details, which would enable geographic tracking of individuals, as well as building a social graph

of people often spending time together.

7.2.1. Information Distribution

Table 7.2 shows how various pieces of information are distributed in the SCC system. It uses the notation introduced in section 2.3 and subsection 4.3.1 for the different components of a Chain of Trust.

It's clear from this visualization, that the ID and vaccination information V_c, are seen by the Central Authority, due to the centralized vaccination registry. However, it should be noted that while the Verifier can find out about the vaccination status of the Holder (due to having a valid or invalid certificate), no detailed information about the vaccination is revealed to the Verifier.

In Step D, both the Verifier and the Central Authority gain access to the ID (indicated in red), as well as any meta data associated with this verification, such as time and location.

ID	Step	Central Authority	Issuer	Holder	Verifier
A	Initial			ID	
B	Vaccinate Patient		V_c, ID		
C	Enter Data	V_c, ID			
D	Verify Certificate	ID		ID	

Table 7.2.: Information Distribution with CENT

7.3. Comparison

This section draws a comparison between the Swiss Covid Certificate (SCC) system, the fictional centralized CENT system, and my proposed ARCTIC system described previously, in terms of the data the certificates contain, the processes used, the availability and privacy of the system, as well as any special features. In section 7.4 these comparisons are summarized in a compact format.

Data

In terms of vaccination data all three systems process roughly the same information: which vaccine was used (and the associated manufacturer), when the vaccination took place, number of doses administered, number of doses scheduled, and the lot number.

They all contain some method to link the certificate to the Holder. In case of the SCC it is the full name and date of birth. The CENT links the Holder directly to the vaccination in the centralized vaccination database. The ARCTIC contains the identification document number to link an MRTD to the certificate, which can be used to identify the Holder.

Processes

The processes for Issuing are different between the three systems. The CENT makes use of a centralized vaccination database, meaning the person receiving the vaccine is automatically known to have received it. In the SCC, a centralized service is used to issue certificates, either through an API for other health systems (e.g. vacme.ch) or a web interface. The cantons authorize persons allowed to issue certificates. The ARCTIC has a different approach to the other two: by having a central Trust Root authorizing Issuers, the Issuers can then issue certificates locally without the involvement of the Trust Root.

The CENT only requires the Holder to presented an ID to the Verifier, while the SCC and ARCTIC require the Holder to additionally present a QR Code. An image or printout of the QR Code suffices, but the SCC offers a wallet app.

To verify a certificate in the SCC and ARCTIC systems, the Verifier scans the QR Code and is told if the certificate is authentic and if the business rules are met (i.e. number of doses, date). In the SCC, the Verifier must then manually compare the full name and date of birth to an ID presented by the holder, while in the ARCTIC system the Verifier scans the MRZ on the MRTD presented by the holder, and the ID link is established automatically. In the CENT system, the Verifier scans the ID of the Holder and contacts a central service with the ID information to establish the validity of the certificate.

In terms of convenience and usability, both ARCTIC and CENT ensure a correct verification through automation, even if the Verifier is lazy. The SCC requires manually checking several pieces of information on the ID, which may get tiring after performing many verifications.

Revocation in the SCC can be done on a certificate-by-certificate basis using the UVCI that's included in each certificate, whereas the ARCTIC only allows revoking entire Issuers (in case they become untrustworthy). This has the consequence that potentially many certificates would need to be re-issued when an Issuer is revoked in the ARCTIC system. To revoke a CENT certificate, the vaccination record is deactivated or deleted from the central database.

Availability

The centralized nature of the CENT and SCC mean that both rely on a connection to the central service in order to issue certificates, the ARCTIC only requires a connection for the first-time setup of an Issuer. Afterwards, certificates can be issued locally.

Both the SCC and ARCTIC rely on the Verifiers having a periodic connection to the certificate and issuer revocation lists respectively, but a connection is not needed to verify every certificate. In contract to that, the CENT requires a connection to the central service for every verification.

Privacy

In terms of health data, both the SCC and CENT make use of a centralized service to issue certificates. This means that health data are not only kept between the patient and the administering healthcare worker, but is also shared with the central authority. The ARCTIC system keeps the health data between the patient and the administering healthcare worker, because they issue certificates directly and locally.

With the SCC and ARCTIC the Verifier has access to the vaccination information at the time of verification. The exception to this is the Certificate Light included in the SCC. The CENT does not expose the vaccination information to the Verifier.

If an SCC certificate is lost or stolen, it can be traced back to a person fairly easily due to the inclusion of the full name and date of birth. Thus, this information is potentially exposed without the knowledge of the Holder. If an ARCTIC certificate is lost or stolen, the vaccination information is linked to the pseudonymous identification document number. The Issuing entity of the identity document can break this pseudonymity.

With all three systems it is possible for a malicious group of Verifiers to build

movement profiles and social graphs of the certificate Holders, by remembering their ID or other identifying features of the certificates and correlating that by location and time. This is also possible for the central authority in the CENT system, while the SCC and ARCTIC prevent this by verifying certificates locally.

Special Features

The SCC offers the Certificate Light as a privacy friendlier variant of the normal certificate. However, requirements for `2G/2G+/Testing` make the Certificate Light less relevant because the validity of a certificate in one of those modes reveals information about the contents (vaccinated/tested). Verification of light certificates is not possible when using the aforementioned modes in the official verifier app.

The ARCTIC enables automated access to restricted areas (such as air travel) through the link with a Machine Readable Travel Document (MRTD). While is is a theoretical possibility with the SCC, it was not implemented.

7.4. Summary

Table 7.3.: Comparison Summary between the three solutions

Topic	**Swiss Covid Certificate (SCC)**	**CENT**	**ARCTIC**
Data within certificate	full name and date of birth, vaccination data, issue date, and a UVCI	none (only centralized storage)	document number of an ID, vaccination data (FHIR) and issue date
Issue	centralized service using either and API or Web-Interface	centralized service based on a vaccination registry	decentralized Issuers which are authorized by a central Trust Root
Carry	certificate QR Code + an identity document with a picture (e.g. passport, ID card, driving license, residence permit)	an identity document with a picture	certificate QR Code + an MRTD.
Verify	EU trust list and certificate revocation list 1) scan certificate QR Code 2) verify signature 3) verify business rules 4) compare name + birthdate with ID 5) verify picture on ID steps 4) and 5) are manual	communication with a centralized verification service 1) scan ID 2) verify online using the service 3) verify picture on ID step 3) is manual	pre-shared trust root key and issuer revocation list 1) scan certificate QR Code 2) scan MRTD 3) verify chain of trust 4) verify MRTD matches cert. 5) verify business rules 6) verify picture on MRTD step 6) is manual
Revocation	individual certificates can be revoked using the UVCI	individual certificates can be revoked by the centralized service	Issuers, and all certificates they issued, can be revoked
Availability	Issuing requires the centralized signing service, verification happens offline. The trust list and the revoked certificates are shared online and must be downloaded periodically.	All actions require the centralized service.	Issuing and verification are done locally and offline. The revoked issuers are shared online and must be downloaded periodically.

Table 7.3: ... continued from the previous page

Topic	Swiss Covid Certificate (SCC)	CENT	ARCTIC
Movement Data	(Groups of) malicious verifiers (e.g. restaurant chains, event coordinators) can remember and share observed UVCIs or other identifying features, and from that construct a movement profile and social graph.	*see SCC* + The central service can easily construct a movement profile and social graph.	*see SCC*
Health Data	The central signing service knows every certificate that was issued, creating an implicit central vaccination registry. A certificate allows assigning the contained vaccination data to a person (name + date of birth). The Certificate Light is a workaround to prevent this.	The central services know every certificate that was issued. Vaccination data is not leaked to the Verifiers.	The certificate allows assigning the contained vaccination data to a pseudonymous document number. During verification (in combination with a MRTD) the vaccination data can be assigned to a person (name + date of birth).
Ease of Verification	Verifier must scan QR Code and manually compare 3 pieces of information (first name, last name, date of birth) between the verification app the ID	Verifier must scan the ID	Verifier must scan the QR Code and ID.
Other Risks	Improper certificate verification by trusting the Holder's wallet app.	The central authority can, through mistake or malice, arbitrarily ban individuals from entry to a certificate controlled area by deleting or deactivating their record in the centralized database.	Because no manual inspection of the MRTD is required, forgery may be easier (see subsection 2.2.2). Specifically, an attacker could substitute the MRZ of their ID with that from another person, and use that other person's certificate. This can be mitigated during the verification by a visual inspection of the ID, and through use of standard mechanisms in an automated gateway.
Special Features	Certificate Light that doesn't contain any vaccination data		automated access to restricted areas using automated MRTD scanning and integration with existing HIS with FHIR.

8. Project Management

The work for this thesis is organized entirely using GitLab; both for version control of all documents and prototypes, as well as for task management.

An iterative approach was chosen, where the project was broken down into a backlog of GitLab Issues in the beginning phase, and organized into a Kanban-style board (see Figure 8.1).

The work order and priority was planned in a Gantt chart Project Plan (see Figure 8.2). In this project plan, each item may mention in parentheses a number (e.g. "`Write test concept (#26)`"). This number links the items to the GitLab tasks, which are automatically imported into this document using a Python script (see section 8.4) to make them accessible to readers of this document. That section makes a distinction between completed and rejected tasks. Any tasks that were rejected are shown in red, and provide a "Reason" row that explains the decision.

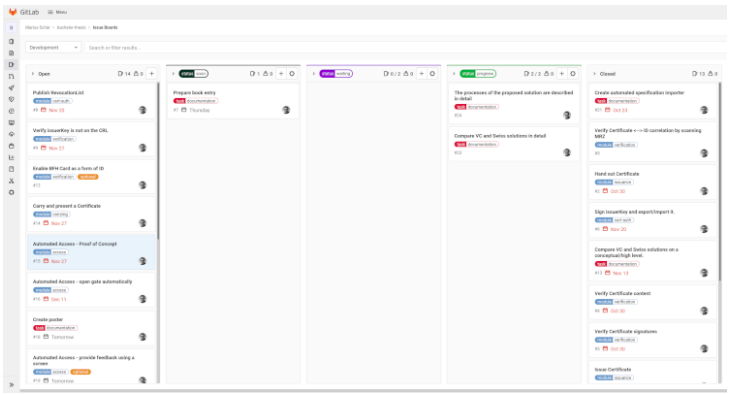


Figure 8.1.: Kanban Board

8.1. Results

While there were some delays, the overall timeline worked out, thanks to the later part of the work being planned with a lot of slack. In this section I will point out particular examples of especially good or especially bad decisions, while the general actual timeline is shown in Figure 8.3.

Report Structure

Writing the report, especially the comparison to other systems, took longer than anticipated. This was due to two factors: the structure of the report was very convoluted in the beginning, making it difficult to write clearly. In weeks 11 and 12 the report was restructured into two parts. The first part being structured like a paper on the subject of Proofs of Vaccination and the second, project management, part. This restructuring meant large parts of the report had to be re-written, stretching from week 13 to 17.

Automated Access Prototype

It was originally planned to implement the Automated Access prototype using a Raspberry Pi (RPi) and a webcam. This path was abandoned in favour of using an Android device as a scanner. This was due to poor performance, with one frame of camera feed being processed in $8\,\mathrm{s}$ to $10\,\mathrm{s}$. This means the minimum time to read the MRZ of an ID was $8\,\mathrm{s}$, twice that if it fails to read perfectly on the first try. After attempting to implement this over several days, the decision was made to use an Android device as a scanner, because this level of performance is tedious and unacceptable, even for a prototype. Additionally, the verification app had already proven that the Android device has much better performance in this task.

Shared Library

The decision to build a shared library to handle the encoding and decoding, as well as the issuance and verification of ARCTIC containers really payed off. At first this approach is slower due to the overhead of building a re-usable library, but every further usage is both faster and less error-prone. As described in section 5.1, there was some weird behaviour with cross-platform portability, but that could be resolved fairly easily once the root cause was discovered.

Centralized System

A goal of the thesis was to compare the SCC and the ARCTIC to other state's solutions. This was quite challenging, as most other systems fall into one of two categories: either similar to the SCC (using a different protocol, but still a central authority issuing certificates secured with digital signatures), or closed source and intransparent. The former is not interesting to cover, because they are so similar, the latter is impossible to cover because very little information is available, i.e. "this QR Code lets you eat at restaurants" and various discussion of the politics surrounding the certificates, but no technical information. In the end a fictional centralized system was chosen, because it was not possible to find enough information about any one system.

8.2. Project Plan

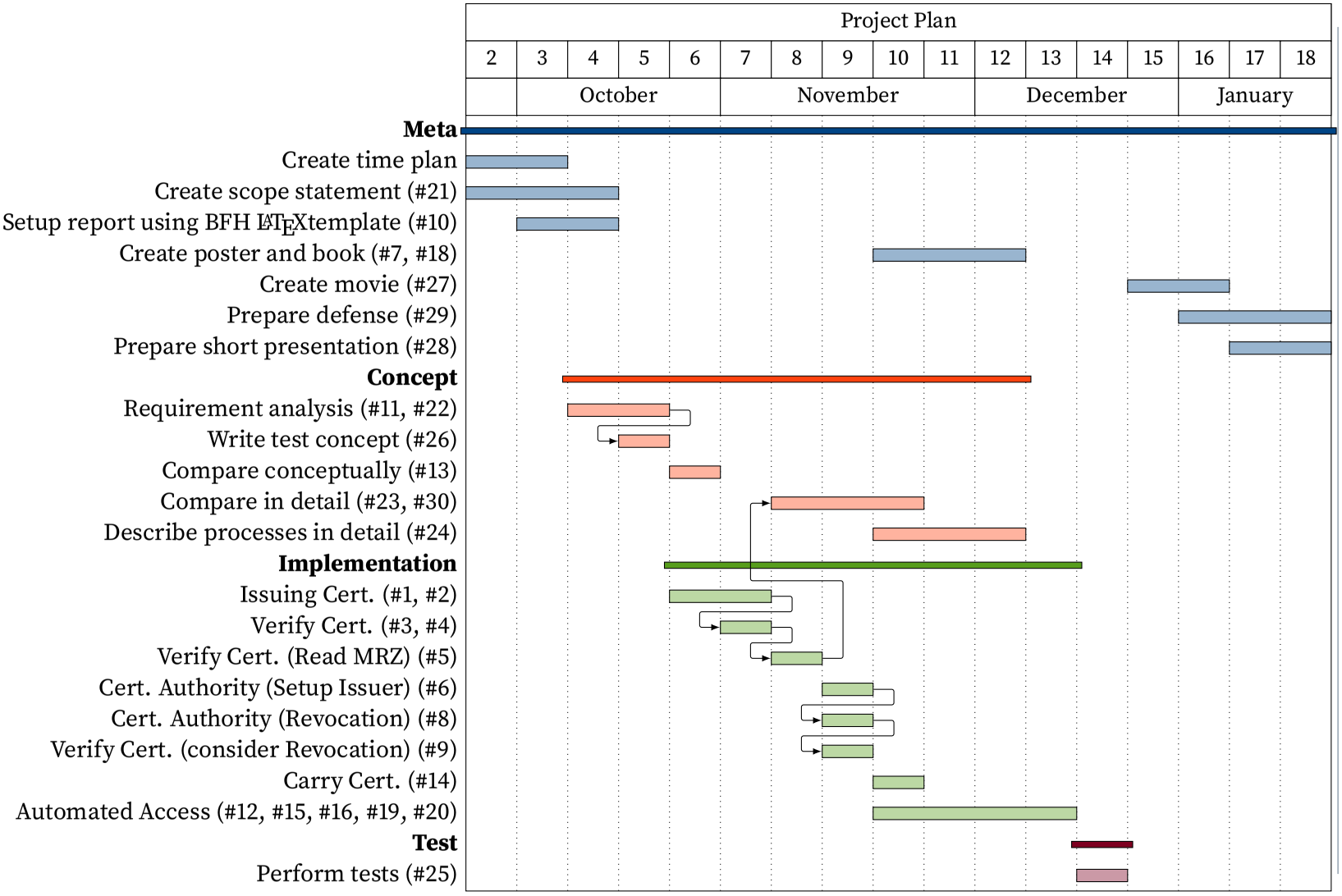


Figure 8.2.: Project Plan created at the beginning of the thesis

8.3. Project Plan (Actual)

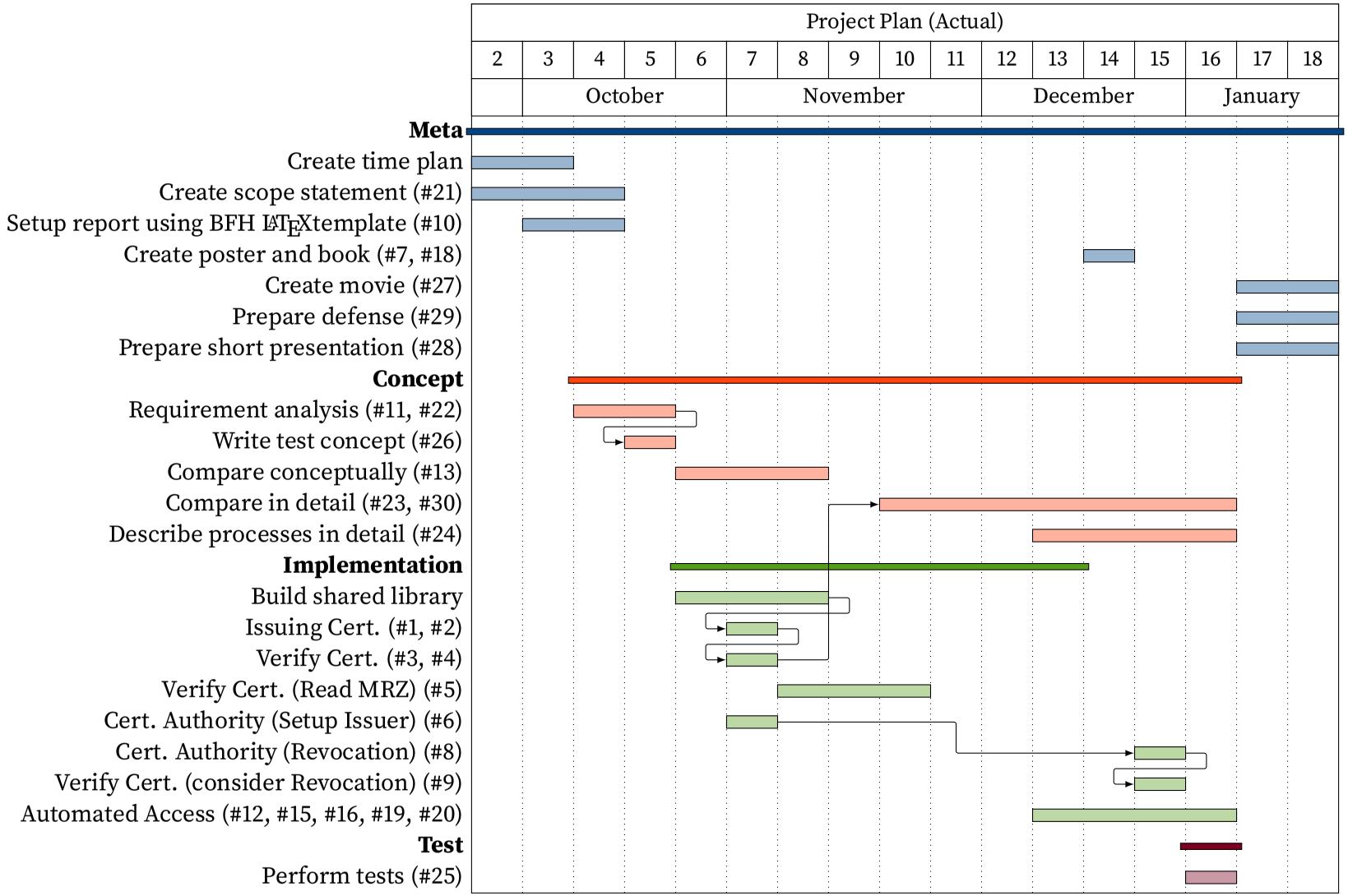


Figure 8.3.: Figure 8.2 updated to show actual timeline

8.4. Tasks

#1	Issue Certificate
Module	Issuance
Scope	Mandatory
Description	Alice must be able to issue certificates.
Criteria	☑ This functionality must be provided by a desktop app ☑ Entered data must be validated to conform to the specification ☑ Finished certificate must be displayed

Table 8.1.: Task 1 - Issue Certificate

#2	Hand out Certificate
Module	Issuance
Scope	Mandatory
Description	Alice must be able to hand out the certificate by exporting it.
Criteria	☑ A finished certificate must be exportable as a PDF ☑ A finished certificate must be exportable as a PNG

Table 8.2.: Task 2 - Hand out Certificate

#3	Verify Certificate signatures
Module	Verification
Scope	Mandatory
Description	Victor must be able to scan a certificate and be told if its signatures are valid. Being valid entails: - The trust root is known - The issuerKeySignature is valid - The dataSignature is valid
Criteria	☑ The verification must be implemented as an Android App ☑ The validation criteria must be fulfilled ☑ The verification code should be re-usable for the automated access prototype

Table 8.3.: Task 3 - Verify Certificate signatures

#4	Verify Certificate content
Module	Verification
Scope	Mandatory
Description	Victor must be able to scan a certificate and be told if its content is valid. Being valid entails: - doseNumber == seriesDoses - All the identifiers are a known value and approved (may be configurable). - vaccination date is more than 10 (configurable) days in the past
Criteria	☑ Must be functionality in the same Android App as #3 ☑ The verification code should be re-usable for the automated-access prototype ☑ The validation criteria must be fulfilled

Table 8.4.: Task 4 - Verify Certificate content

#5	Verify Certificate <–> ID correlation by scanning MRZ
Module	Verification
Scope	Mandatory
Description	Victor must be able to scan a certificate and an ID and be told if they match. Matching means, tat the patient.identifier.value matches the ID's document number.
Criteria	☑ Must be integrated into the same Android App as #3 and #4 ☑ After scanning a Certificate, an ID scan must be automatically prompted ☑ The App must scan the MRZ on an ID ☑ The verification criteria must be fulfilled ☑ Victor must be told to compare the photo on the ID

Table 8.5.: Task 5 - Verify Certificate <–> ID correlation by scanning MRZ

#6	Sign issuerKey and export/import it.
Module	Cert
Scope	Mandatory
Description	Bob must be able to sign keys for Alice.
Criteria	☑ Bob must be able to sign Alice's keys ☑ Alice must be able to import and use these signatures (#1)

Table 8.6.: Task 6 - Sign issuerKey and export/import it.

#7	Prepare book entry
Module	Documentation
Scope	Mandatory
Description	Prepare the entry for book.bfh.ch. It should be similar in content to the Poster for the Finaltag.
Criteria	☑ The book entry must contain an attractive picture. ☑ The book entry should consist of some "public friendly" text. (not overly technical)

Table 8.7.: Task 7 - Prepare book entry

#8	Publish RevocationList
Module	Cert
Scope	Mandatory
Description	Bob must be able to publish a Revocation List of Alice's keys that are no longer valid.
Criteria	☐ The CRL should be published in a standard format ☑ The CRL must be signed by Bob ☑ The CRL must be accessible to Victor

Table 8.8.: Task 8 - Publish RevocationList

#9	Verify issuerKey is not on the CRL
Module	Verification
Scope	Mandatory
Description	Victor's verification app must check that the issuerKey in a certificate is not on the CRL.
Criteria	☑ The CRL must be downloaded periodically ☑ The CRL must be considered when verifying certificates ☐ A warning should appear if the local CRL is out of date (older than 48h (configurable)) ☑ The CRL signature should be checked to verify its authenticity

Table 8.9.: Task 9 - Verify issuerKey is not on the CRL

#10	Setup Document
Module	Documentation
Scope	Mandatory
Description	Setup the main report using the BFH LaTeX template.

Table 8.10.: Task 10 - Setup Document

#11	Write F/NF Requirements
Module	Documentation
Scope	Mandatory
Description	The functional and non-functional requirement must be present in the report.

Table 8.11.: Task 11 - Write F/NF Requirements

#13	Compare ARCTIC and Swiss solutions on a conceptual/high level.
Module	Documentation
Scope	Mandatory
Description	The report must contain a high level / conceptual comparison between the proposed solution and the Swiss solutions.
Criteria	☑ The report must explain the conceptual working of the ARCTIC ☑ The report must explain the conceptual working of the Swiss solutions ☑ The report must draw a comparison between the three (ARCTIC, Swiss, Centralized) ☑ The report must draw a comparison to a hypothetical centralized system

Table 8.12.: Task 13 - Compare ARCTIC and Swiss solutions on a conceptual/high level.

#15	Automated Access - Proof of Concept
Module	Access
Scope	Mandatory
Description	An automated system must be able to read the RFID Chip in a passport and scan the certificate using a webcam.
Criteria	☐ A raspberry pi must be able to read the RFID Chip in a passport and have a usable interface ☑ A raspberry pi must be able to read a certificate QR-Code using a webcam ☑ A raspberry pi must be able to read a passport MRZ using a webcam

Table 8.13.: Task 15 - Automated Access - Proof of Concept

#16	Automated Access - open gate automatically
Module	Access
Scope	Mandatory
Description	An automated system must be able to read and verify a certificate + ID, then open a gate. As a stand-in for a gate LEDs via GPIO will be used.
Criteria	☑ An automated system must be able to verify a certificate to the same standard as #4 and #5 (barring facial recognition) ☑ If the certificate + ID combination is acceptable, light a green LED ☑ If there is some problem, light a red LED

Table 8.14.: Task 16 - Automated Access - open gate automatically

#18	Create poster
Module	Documentation
Scope	Mandatory
Description	
Criteria	☑ The poster must be done based on the templates provided in the moodle course ☐ The poster should not be mainly text based, but rather include a good graphic and small amounts of text

Table 8.15.: Task 18 - Create poster

#20	Automated Access - Prettify Prototype
Module	Access
Scope	Optional
Description	The prototype must look nice and should not have any loose components.

Table 8.16.: Task 20 - Automated Access - Prettify Prototype

#21	Create automated specification importer
Module	Documentation
Scope	Mandatory
Description	GitLab issues must be importable using the CSV-Export function and script that generates TeX from it. This is done to avoid problems with consistency between GitLab and the report.
Criteria	☑ GitLab Issues can be entered into the report with minimal manual effort ☑ No more than 90 minutes is spent on prototyping this ☐ Requirements should have their own environment and not just be tables

Table 8.17.: Task 21 - Create automated specification importer

#22	Analyze Requirements based on Project 2
Module	Documentation
Scope	Mandatory
Description	Some requirements were already generated during Project 2, but may no longer be relevant or may be incomplete.
Criteria	☑ The requirements for Project 2 must be re-evaluated ☑ The current requirements must be reflected in the GitLab issues ☑ The requirements must be ready for the report

Table 8.18.: Task 22 - Analyze Requirements based on Project 2

#23	Compare ARCTIC and Swiss solutions in detail
Module	Documentation
Scope	Mandatory
Description	The report must compare the ARCTIC and Swiss solutions in detail, especially in the following areas: Privacy, Soundness, Availability, Usability.
Criteria	☑ The advantages and disadvantages of the solutions must be compared ☑ Privacy aspect is compared ☑ Soundness aspect is compared ☑ Availability aspect is compared ☑ Usability aspect is compared

Table 8.19.: Task 23 - Compare ARCTIC and Swiss solutions in detail

#24	The processes of the proposed solution are described in detail
Module	Documentation
Scope	Mandatory
Description	The report must describe the processes for the various areas of the ARCTIC in detail
Criteria	☑ Issuer Setup is described ☑ Certificate Issuance is described ☑ Certificate Verification is described ☑ Automated Access is described ☑ Issuer Revocation is described

Table 8.20.: Task 24 - The processes of the proposed solution are described in detail

#25	Perform integration tests
Module	All
Scope	Mandatory
Description	Integration tests must be performed.
Criteria	☑ The test execution must be documented ☑ Any discrepancies that are found must be documented

Table 8.21.: Task 25 - Perform integration tests

#26	Write integration test concept
Module	Documentation
Scope	Mandatory
Description	A test procedure for integration tests must be written.
Criteria	☑ The test procedure must be documented ☑ The relation to the requirements must be clear

Table 8.22.: Task 26 - Write integration test concept

#27	Create movie
Module	Documentation
Scope	Mandatory
Description	Create a movie that covers the rough outline of this thesis. Further criteria will be specified.
Criteria	☑ The movie must be shorter than 120 seconds

Table 8.23.: Task 27 - Create movie

#28	Short presentation for Finaltag
Module	All
Scope	Mandatory
Description	Create a short presentation for the Finaltag

Table 8.24.: Task 28 - Short presentation for Finaltag

#29	Presentation for defense
Module	All
Scope	Mandatory
Description	Create a presentation for the defense.

Table 8.25.: Task 29 - Presentation for defense

#31	Rewrite Report in a new Structure
Module	Documentation
Scope	Mandatory
Description	Rewrite the report to conform to a two part structure.
Criteria	☑ The first part must be structured akin to a paper ☑ The second part must discuss the project management

Table 8.26.: Task 31 - Rewrite Report in a new Structure

#32	The project management is described.
Module	Documentation
Scope	Mandatory
Description	The report must describe the project management and compare planned vs. actual.
Criteria	☑ The report describes how the project was executed ☑ The report describes what was planned and executed ☑ The report shows the time frame and planned vs. actual

Table 8.27.: Task 32 - The project management is described.

#12	Enable BFH Card as a form of ID
Module	Verification
Scope	Optional
Description	A BFH Card must work as a form of ID.
Criteria	☐ Alice must be able to register a BFH Card as a form of ID when issuing certificates ☐ Victor must be able to accept a BFH Card as a form of ID when verifying certificates ☐ As a patient identifier, the number in the Code39 barcode on the BFH Card must be used
Reason	As an optional feature, this was rejected due to a lack of time.

Table 8.28.: Task 12 - Enable BFH Card as a form of ID

#14	Carry and present a Certificate
Module	Carrying
Scope	Mandatory
Description	Charlie must have a convenient way to carry and present one or more certificates (e.g. for him and his children).
Criteria	☐ Charlie must be able to import his certificate into an Android App ☐ Charlie must be able to give Nicknames to the certificates ☐ Charlie must be able to delete the certificates ☐ Charlie must not be able to verify his certificate in the Carrier App ☐ The certificate should be verified upon import
Reason	This was rejected because a wallet app is not necessary. The Holder (Charlie) can present their certificate in any way they want.

Table 8.29.: Task 14 - Carry and present a Certificate

#19	Automated Access - provide feedback using a screen
Module	Access
Scope	Optional
Description	The automated gate must provide feedback using a screen.
Criteria	☐ The screen must display instructions ☑ The screen should display a camera feed to help alignment ☐ If a failure occurs, an explanatory error message must be displayed
Reason	This was rejected to make the automated access prototype more distinct from the verification app. The android device used displays the camera feed to help with alignment.

Table 8.30.: Task 19 - Automated Access - provide feedback using a screen

#30	Investigate Swiss CovidCertificate outage of 2021-10-15 in more detail.
Module	Documentation
Scope	Optional
Description	What went wrong and how does our solution prevent it from happening?
Reason	As an optional feature, this was rejected due to a lack of time.

Table 8.31.: Task 30 - Investigate Swiss CovidCertificate outage of 2021-10-15 in more detail.

9. Conclusion

The goal of this thesis was to examine existing solutions to the problem of a proof of vaccination and compare them on the basis of security, privacy, soundness, and usability, as well as conceptualize and prototype ARCTIC, my own proof of vaccination system.

The comparison between the existing solutions and ARCTIC shows that my system has advantages in terms of privacy and availability, as well as usability and soundness over the examined systems. The ARCTIC prototype demonstrates that the concept presented in this thesis can be implemented successfully.

Further investigations into the security of identity documents, as well as how to detect forgeries, should be performed. The ARCTIC system could be optimized in the following ways as a future project: use of a more efficient encoding, improved reliability and speed of the machine vision, and use of standard formats (e.g. for the Issuer Revocation List). Furthermore, the administrative processes surrounding the system could be improved and established, such as how to detect and handle genuine certificates issued by now untrustworthy Issuers.

Declaration of Authorship

I hereby declare that I have written this thesis independently and have not used any sources or aids other than those acknowledged.

All statements taken from other writings, either literally or in essence, have been marked as such.

I hereby agree that the present work may be reviewed in electronic form using appropriate software.

January 20, 2022

M. Schär

Bibliography

[1] C. Bormann and P. Hoffmann. RFC 7049 Concise Binary Object Representation (CBOR). RFC 7049, 2013. URL https://www.rfc-editor.org/rfc/rfc7049.html.

[2] C. Bormann and P. Hoffmann. RFC 8949 Concise Binary Object Representation (CBOR). RFC 8949, 2020. URL https://www.rfc-editor.org/rfc/rfc8949.html.

[3] Carsten Bormann. CBOR | Overview, 2020. URL https://cbor.io/.

[4] European Comission. Union Register, 2022. URL https://ec.europa.eu/health/medicinal-products/union-register_en.

[5] Ecma International. The JSON Data Interchange Syntax, 2017. URL https://www.ecma-international.org/wp-content/uploads/ECMA-404_2nd_edition_december_2017.pdf.

[6] EU eHealth Network. verifiable vaccination certificates - basic interoperability elements, 2021. URL https://ec.europa.eu/health/sites/default/files/ehealth/docs/vaccination-proof_interoperability-guidelines_en.pdf.

[7] EU eHealth Network. Value Sets for EU Digital COVID Certificates, 2021. URL https://ec.europa.eu/health/sites/default/files/ehealth/docs/digital-green-value-sets_en.pdf.

[8] EU eHealth Network. Simple Vaccination Sample, 2021. URL https://github.com/ehn-dcc-development/ehn-dcc-schema/blob/release/1.3.0/examples/vaccination/simple.json.

[9] EU eHealth Network. Interoperability of health certificates - Trust framework, 2021. URL https://ec.europa.eu/health/sites/default/files/ehealth/docs/trust-framework_interoperability_certificates_en.pdf.

[10] Federal Office of Information Technology, Systems and Telecommunication FOITT. Covidcertificate-documents, 2021. URL https://github.com/admin-ch/CovidCertificate-Documents.

[11] Federal Office of Public Health FOPH. Information for checkers and issuers and technical details on the COVID certificate, 2021. URL https://www.bag.admin.ch/bag/en/home/krankheiten/ausbrueche-epidemien-pandemien/aktuelle-ausbrueche-epidemien/novel-cov/covid-zertifikat/covid-zertifikat-pruefer-aussteller-technische-informationen.html#1851413288.

[12] Argentinian Government. Carnet Unificado de Vacunación Digital, 2019. URL https://www.argentina.gob.ar/noticias/carnet-unificado-de-vacunacion-digital.

[13] Argentinian Government. Carnet Unificado de Vacunación Digital, unknown. URL https://www.argentina.gob.ar/miargentina/servicios/vaccine_covid.

[14] Argentinian Government. miArgentina, unknown. URL https://www.argentina.gob.ar/miargentina.

[15] Argentinian Government. Validá tu identidad digital, unknown. URL https://www.argentina.gob.ar/miargentina/validar-identidad.

[16] HL7. Publication (Version) History, 2019. URL https://hl7.org/fhir/directory.html.

[17] HL7. Immunization - FHIR, 2019. URL https://www.hl7.org/fhir/immunization.html.

[18] International Civil Aviation Organization (ICAO). About ICAO, 2021. URL https://www.icao.int/about-icao/Pages/default.aspx.

[19] International Civil Aviation Organization (ICAO). Doc 9303 Machine Readable Travel Documents, 2021. URL https://www.icao.int/publications/pages/publication.aspx?docnum=9303.

[20] International Organization for Standardization. Automatic identification and data capture techniques - QR Code bar code symbology specification. ISO Standard 18004, 2015.

[21] National Security Agency/Central Security Service (via Archive.org). Commercial National Security Algorithm Suite and Quantum Computing FAQ, 2016. URL https://web.archive.org/web/20170830013313/https://www.iad.gov/iad/library/ia-guidance/ia-solutions-for-classified/algorithm-guidance/assets/public/upload/CNSA-Suite-and-Quantum-Computing-FAQ.pdf.

[22] California Department of Technology. California Launches New Digital Tool Giving Residents Convenient Access to Their COVID-19 Vaccine Record, 2021. URL https://cdt.ca.gov/news/california-launches-new-digital-tool-

giving-residents-convenient-access-to-their-covid-19-vaccine-record/.

[23] D. van Gulik P. Faltstrom, F. Ljunggren. The Base45 Data Encoding. Draft base45-02, 2021. URL https://tools.ietf.org/id/draft-faltstrom-base45-02.html.

[24] SJD S. Josefsson. RFC 4648 The Base16, Base32, and Base64 Data Encodings. RFC 4648, 2006. URL https://www.rfc-editor.org/rfc/rfc4648.html.

[25] J. Schaad and A. Cellars. RFC 8152 CBOR Object Signing and Encryption (COSE). RFC 8152, 2017. URL https://www.rfc-editor.org/rfc/rfc8152.html.

[26] Marius Schär and Severin Kaderli. Vaccination Certificates - Project 2, 2021. URL https://gitlab.com/martyschaer/project-2/-/blob/master/ARTICLE.pdf.

[27] Swiss Confederation. Swiss Criminal Code - Art. 251, 2021. URL https://www.fedlex.admin.ch/eli/cc/54/757_781_799/en#art_251.

[28] Swiss CovidCertificate-App-Android contributors. Self verification button used for verification, 2022. URL https://github.com/admin-ch/CovidCertificate-App-Android/issues/295#issuecomment-1005719045.

[29] Denso Wave. QR Code Features, 2013. URL https://web.archive.org/web/20130129064920/http://www.qrcode.com/en/qrfeature.html.

[30] Denso Wave. Patents pertaining to the QR Code, 2019. URL https://www.qrcode.com/en/patent.html.

[31] Denso Wave. QR Code Error Correction Feature, 2021. URL https://www.qrcode.com/en/about/error_correction.html.

[32] Denso Wave. QR Code Information capacity and versions, 2021. URL https://www.qrcode.com/en/about/version.html.

Glossary

ARCTIC **Automatically Readable Chain of Trust-based Immunization Certificate (ARCTIC)** is the proof of vaccination solution proposed in this thesis. Each ARCTIC is a cryptographically signed document which verifies that the individual identified by it and an accompanying official identity document received a vaccination, and is thus entitled to certain privileges.

CBOR **Concise Binary Object Representation (CBOR)** is a binary data format designed for small code size and small message size. It ensures that the same data is encoded in the same way every time, which is important for signature validation. CBOR is specified in RFC 8949.

COSE **CBOR Object Signing and Encryption (COSE)** is a protocol built on top of CBOR which supports signing and encryption. COSE is specified in RFC 8152.

CoT **Chain of Trust (CoT)** allows establishing the integrity and authenticity of a document through a series of cryptographic signatures. In order to verify that the chain is intact, a verifier only needs to know the trust root (or anchor) at the start of the chain.

ECDSA **Elliptic Curve Digital Signature Algorithm (ECDSA)** is an algorithm of Digital Signatures based on elliptic curve cryptography.

EU DGC **European Union Digital Green Certificate (EU DGC)** is the EU solution to the problem of proof of vaccination. Also known as “Digital Covid Certificate”.

FHIR **Fast Healthcare Interoperability Resources (FHIR)** (pronounced "fire") is a health data interchange standard by HL7 which is organized by resources (e.g. Immunization, Patient).

FOITT **Federal Office of Information Technology, Systems, and Telecommunication (FOITT)** one of the Swiss Federal Government’s IT providers and developer of the Swiss Covid Certificate.

FOPH **Federal Office of Public Health (FOPH)** is the Swiss feral government’s centre for public health.

GPIO **General Purpose Input/Output (GPIO)** a GPIO pin is a pin on an integrated circuit that has no pre-defined role, but is controlled by a program at runtime.

HIN **Health Info Net AG (HIN)** is a company which facilitates secure communication in the medical field and can act as an identity provider.

HIS **Health Information System (HIS)** is a computer system which contains medical information.

HTML **HyperText Markup Language (HTML)** is a markup language used to design documents to be displayed in a web browser.

HTTPS **HyperText Transfer Protocol Secure** is a widely used extension of the HyperText Transfer Protocol (HTTP) that ensures privacy and data integrity by using an encrypted channel.

ICAO **International Civil Aviation Organization (ICAO)** is a United Nations specialized agency which sets standards and performs research concerning international air travel.

ID **Identity Document (ID)** is a document issued by a government or other authority which identifies the person carrying it.

IDE **Integrated Development Environment (IDE)** is a program that facilitates software development, e.g. Eclipse, IntelliJ, Netbeans.

IRL **Issuer Revocation List (IRL)** contains Issuers which are no longer to be trusted. Certificates issued by them are no longer considered valid.

JavaFX **JavaFX** is a framework used to build graphical desktop applications using the Java programming language.

MQTT Broker **MQTT Broker** is a program that routes messages from their producers to their consumers.

MRTD **Machine Readable Travel Document (MRTD)** is a travel document in accordance with ICAO document 9303 [19] which contains a machine readable zone.

MRZ **Machine Readable Zone (MRZ)** is a part of the machine readable passport (MRP) as specified by ICAO, which is designed to be read using OCR.

RPi **Raspberry Pi (RPi)** is a cheap, credit card sized, single board computer. It is usually used to run a variety of Linux distributions, and is popular in small server and Internet of Things applications.

SCC **Swiss Covid Certificate (SCC)** is the Covid Certificate used in Switzerland. It is an implementation of the EU DGC.

UTF8 **UTF8** is a character encoding defined by the Unicode Standard, that supports all Unicode code points, while being backwards compatible with ASCII.

UVCI **Unique Certificate Identifier (UVCI)** a unique identifier for each issued certificate as specified in the DGC interoperability document, annex 2 [6]. It allows revocation of individual certificates (e.g. in case the name is misspelled).

X.509 **X.509** is a standard defining, among other things, a format for storage and exchange of public keys.

List of Figures

List of Tables

Listings

Appendices

A. Task Description

Bachelorthesis-Aufgabe

für	Marius Schär
Fachbereich	Informatik
Betreuung durch	Dr. Annett Laube Dr. Reto Koenig

Digitaler Impfausweis

Gerade in Zeiten einer globalen Pandemie ist der Nachweis, dass man durch eine Impfung geschützt ist, wichtig, um Zugang zu bestimmten Orten oder Leistungen zu bekommen. Doch wie kann man sicherstellen, dass nur wirklich geimpfte Personen diesen Nachweis vorlegen können. Das ist mit einen normalen Impfheft oder einem Dokument, das nur Name und Geburtsdatum enthält nicht möglich. Nur durch eine Kopplung an eine überprüfbare, offizielle Identität, z.B. einem elektronischen Reisepass, kann man auch länderübergreifend Betrug vorbeugen und internationale Interoperabilität sicherstellen. In diesem Projekt soll untersucht werden, ob es möglich ist, einen digitalen Impfnachweis zu erzeugen, der eine sichere Überprüfung und vollautomatischen Zugang ermöglicht, die Privatsphäre gewährleistet und auf breiter internationaler Basis einsetzbar ist. Ein Vergleich mit existierenden Lösungen (z.B. Covid-Zert) rundet die Arbeit ab. Der digitaler Impfnachweis und die Prozesse zur Ausstellung, Überprüfung und Zugang sollen prototypisch umgesetzt werden.

Beginn der Arbeit	20. September 2021
Abschluss der Arbeit	20. Januar 2022

Der Betreuer:

Der Fachbereichsleiter:

B. Vaccination Certificate Data Schema

```
{
    "$schema": "http://json-schema.org/draft/2020-12/schema",
    "$id": "bfh.ch/p2/kades2-scham17/vc-data",
    "title": "Vaccination Certificate",
    "description": "A subset of a FHIR Immunization resource.",
    "type": "object",
    "properties": {
        "status": {
            "description": "The status of the vaccination.",
            "type": "string",
            "enum": [
                "completed",
                "entered-in-error",
                "not-done"
            ]
        },
        "resourceType": {
            "description": "Identifies the type of FHIR resource.",
            "type": "string",
            "enum": [
                "Immunization"
            ]
        },
        "vaccineCode": {
            "description": "Which vaccine was administered. Must match with manufacturer",
            "type": "object",
            "properties": {
                "coding": {
                    "description": "Describes the system and ID of the vaccine.",
                    "type": "array",
                    "maxItems": 1,
                    "items": {
                        "type": "object",
                        "properties": {
                            "system": {
                                "type": "string",
                                "enum": [
                                    "https://ec.europa.eu/health/documents/community-register/html/"
                                ]
                            },
                            "code": {
                                "description": "Any code allowed by the system property.
                                     Must match the name.",
                                "type": "string"
                            }
                        }
                    }
                },
                "text": {
                    "description": "A human readable description of the coding-value",
                    "type": "string"
                }
```

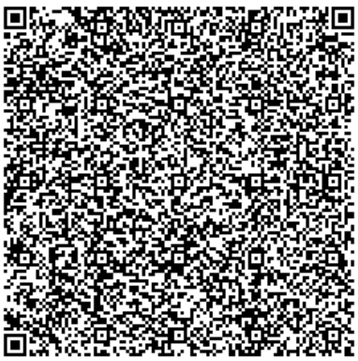
```
            }
        },
        "patient": {
            "description": "Identifies the subject of the vaccination certificate",
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "Patient"
                    ]
                },
                "identifier": {
                    "type": "object",
                    "properties": {
                        "value": {
                            "description": "A passport/id document number.",
                            "type": "string"
                        }
                    }
                }
            }
        },
        "recorded": {
            "description": "When was this vaccination certificate issued (ISO-8601).",
            "type": "string",
            "format": "date"
        },
        "occurrenceDateTime": {
            "description": "When was the vaccine administered (ISO-8601).",
            "type": "string",
            "format": "date"
        },
        "manufacturer": {
            "description": "Who manufactured the vaccine. Must match with vaccineCode.",
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "Organization"
                    ]
                },
                "identifier": {
                    "type": "object",
                    "properties": {
                        "value": {
                            "description": "A manufacturer identifier according to
                                https://spor.ema.europa.eu/v1/organisations",
                            "type": "string"
                        }
                    }
                },
                "display": {
                    "description": "A human readable manufacturer name",
                    "type": "string"
                }
            }
        },
        "lotNumber": {
            "description": "The lot number of the administered vaccine",
            "type": "string"
```

```
        },
        "protocolApplied": {
            "description": "Describes the system and ID of the vaccine.",
            "type": "array",
            "maxItems": 1,
            "items": {
                "type": "object",
                "properties": {
                    "doseNumberPositiveInt": {
                        "description": "The number of doses that were administered
                             already.",
                        "type": "number",
                        "minimum": 1
                    },
                    "seriesDosesPositiveInt": {
                        "description": "The number of doses the the subject of the
                             certificate should receive.",
                        "type": "number",
                        "minimum": 1
                    },
                    "targetDisease": {
                        "type": "array",
                        "maxItems": 1,
                        "items": {
                            "coding": {
                                "description": "Describes the disease that the vaccine is
                                     targeting (COVID-19).",
                                "type": "array",
                                "maxItems": 1,
                                "items": {
                                    "type": "object",
                                    "properties": {
                                        "system": {
                                            "type": "string",
                                            "enum": [
                                                "http://snomed.info/sct"
                                            ]
                                        },
                                        "code": {
                                            "description": "The code for COVID-19 as
                                                 described by the system",
                                            "type": "string",
                                            "enum": [
                                                "840539006"
                                            ]
                                        }
                                    }
                                }
                            },
                            "text": {
                                "type": "string",
                                "enum": [
                                    "COVID-19"
                                ]
                            }
                        }
                    }
                }
            }
        }
    }
}
```

Listing B.1: Vaccination Certificate Data Schema

C. Example ARCTIC PDF

ARCTIC



ID number:	D12027
Vaccine:	Vaccine(Comirnaty)
Vacc. Date:	2021-12-14
Issue Date:	2022-01-19
Doses:	Protocol{2/2}
Lot-Number:	B.897686