Icon

Description automatically generated

Computing Department, Donegal Atlantic Technological University, Port Road, Letterkenny, Co. Donegal, Ireland.

COVID-19 Certificate App Which Utilises Blockchain and Off-Chain Storage with Secured QR Codes

Author: Mark McGuinness

Supervised by: Mandy Douglas

A thesis submitted in partial fulfilment of the requirements for “Research in Computing with Emerging Technologies” and “Project Development”

BSc (Hons) in Computing with Computer Security & Digital Forensics

Submitted to Quality and Qualifications Ireland (QQI)  
Dearbhú Cáilíochta agus Cáilíochtaí Éireann May 2022

# Declaration

I hereby certify that the material, which l now submit for assessment on the programmes of study leading to the award of Bachelor of Science Honours (BSc Hons) in Computer Security and Digital Forensics, is entirely my own work and has not been taken from the work of others except to the extent that such work has been cited and acknowledged within the text of my own work. No portion of the work contained in this thesis has been submitted in support of an application for another degree or qualification to this or any other institution. I understand that it is my responsibility to ensure that I have adhered to LYIT’s rules and regulations.

I hereby certify that the material on which I have relied on for the purpose of my assessment is not deemed as personal data under the GDPR Regulations. Personal data is any data from living people that can be identified. Any personal data used for the purpose of my assessment has been psudonymised and the data set and identifiers are not held by LYIT. Alternatively, personal data has been anonymised in line with the Data Protection Commissioners Guidelines on Anonymisation.

I consent that my work will be held for the purposes of education assistance to future students and will be shared on the LYIT Computing website (www.lyitcomputing.com) and Research THEA website (https://research.thea.ie/). I understand that documents once uploaded onto the website can be viewed throughout the world and not just in the Ireland. Consent can be withdrawn for the publishing of material online by emailing Thomas Dowling; Head of Department at thomas.dowling@lyit.ie to remove items from the LYIT Computing website and by email emailing Denise McCaul; Systems Librarian at denise.mccaul@lyit.ie to remove items from the Research THEA website. Material will continue to appear in printed formats once published and as websites are public medium, LYIT cannot guarantee that the material has not been saved or downloaded.

Signature of Candidate Date

Mark McGuinness 11/05/2022

# Acknowledgements

I would like to express my utmost appreciation and thanks towards my mentor & supervisor, Mandy Douglas, whose encouragement, and enthusiasm made the seemingly insurmountable pitfalls much easier to navigate. I knew I could rely on her timely feedback & guidance on every step of the journey.

I would also like to extend my thanks to Thomas Mayfield, who without their knowledgeable contributions of web3 & Blockchain support during development, would not have made this dissertation possible.

To my friends and family, who motivated me & raised my spirits during the challenging times I could not. They helped put my project back on track, where I strayed off.

Finally, I would like to thank all the lecturers I had throughout the years, as well as the President and Head of Computing at Donegal Atlantic Technological University. Without them I would not be where I am at today, and I am very grateful for the opportunities they have provided me these past couple of years.

# Abstract

The emergence of the COVID-19 pandemic has seen QR codes garner attention throughout businesses worldwide, to battle the transmission of the virus via physical media. They can be seen on current-day COVID-19 documentation papers, which allows the opportunity for anyone & everyone to scan the code without any security measures present to prevent such behaviour. This dissertation aims to address this issue, through providing authentication measures by means of an encrypted QR code and by implementing a blockchain system to validate the genuineness of the certificate data. The research investigates the use of the Interplanetary File System, a private blockchain mimicking a Bitcoin iteration, and Python programming language with Flask to develop this web application. The database of vaccine certificate records will consist of dummy data, entirely fabricated by the researcher of the project. After undergoing testing, the validation process offered in tandem by the off-chain storage and blockchain proved to be a success. Though the resulting artefact was less applicable in the real world as Interplanetary File System was circumvented with an SQLite database, it still showcased the improved confidentiality & data integrity blockchain can offer for vaccine certificates.

# Acronyms

|  |  |  |
| --- | --- | --- |
| Acronym | Definition | Page |
| IPFS | Interplanetary File System | 11 |
| HSE | Health Service Executive | 11 |
| QR | Quick Response | 11 |
| DES | Data Encryption Standard | 11 |
| GDPR | General Data Protection Regulation | 13 |
| SQL | Structured Query Language | 12 |
| WHO | World Health Organisation | 15 |
| CPU | Central Processing Unit | 15 |
| PoW | Proof of Work | 5 |
| DLT | Distributed Ledger Technology | 16 |
| HIPAA | Health Insurance Portability and Accountability Act | 21 |
| EU | European Union | 21 |
| CID | Content Identifier | 22 |
| DHT | Distributed Hash Table | 22 |
| DAG | Directed Acyclic Graph | 23 |
| IOS | iPhone Operating System | 27 |
| RAM | Random Access Memory | 29 |
| OS | Operating System | 29 |
| CLI | Command Line | 44 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Table of Contents

[Declaration 2](#_Toc103195787)

[Acknowledgements 3](#_Toc103195788)

[Abstract 4](#_Toc103195789)

[Acronyms 5](#_Toc103195790)

[Table of Contents 6](#_Toc103195791)

[Table of Figures 10](#_Toc103195792)

[Table of Tables 11](#_Toc103195793)

[1. Introduction 12](#_Toc103195794)

[1.1 Background 13](#_Toc103195795)

[1.2 Purpose 14](#_Toc103195796)

[1.3 Aims & Objectives 14](#_Toc103195797)

[1.3.1 Aims 14](#_Toc103195798)

[1.3.2 Objectives 14](#_Toc103195799)

[1.4 Research Question 15](#_Toc103195800)

[1.5 Outline 15](#_Toc103195801)

[2. Literature Review 16](#_Toc103195802)

[2.1 Introduction 16](#_Toc103195803)

[2.2 Blockchain 16](#_Toc103195804)

[2.3 Distributed Ledger Technology 17](#_Toc103195805)

[2.3.1 Public Blockchain 18](#_Toc103195806)

[2.3.2 Private Blockchain 19](#_Toc103195807)

[2.3.3 Hybrid Blockchain 20](#_Toc103195808)

[2.3.4 Consortium Blockchain 21](#_Toc103195809)

[2.4 Data Regulations 22](#_Toc103195810)

[2.5 Interplanetary File System 23](#_Toc103195811)

[2.6 Quick Response Codes 24](#_Toc103195812)

[2.7 Anatomy of a QR Code 26](#_Toc103195813)

[2.7.1 Finder Pattern 26](#_Toc103195814)

[2.7.2 Format Information 26](#_Toc103195815)

[2.7.3 Alignment Pattern 26](#_Toc103195816)

[2.7.4 Timing Pattern 27](#_Toc103195817)

[2.7.5 Version Information 27](#_Toc103195818)

[2.7.6 Error Correction Data 27](#_Toc103195819)

[2.7.7 Data 27](#_Toc103195820)

[2.8 QR Features 27](#_Toc103195821)

[2.9 QR Security Flaws 28](#_Toc103195822)

[2.10 Conclusion 29](#_Toc103195823)

[3. Design & Methodology 30](#_Toc103195824)

[3.1 Introduction 30](#_Toc103195825)

[3.2 Hardware Requirements 30](#_Toc103195826)

[3.3 Software Requirements 31](#_Toc103195827)

[3.4 Functional Requirements 32](#_Toc103195828)

[3.5 Non-Functional Requirements 32](#_Toc103195829)

[3.5.1 Security 32](#_Toc103195830)

[3.5.2 Reliability 32](#_Toc103195831)

[3.5.3 Fault Tolerance 32](#_Toc103195832)

[3.5.4 Usability 32](#_Toc103195833)

[3.5.5 Confidentiality 33](#_Toc103195834)

[3.6 Use Case Diagram 33](#_Toc103195835)

[3.7 Use Case Descriptions 33](#_Toc103195836)

[3.8 Storyboards 37](#_Toc103195837)

[3.8.1 Login 37](#_Toc103195838)

[3.8.2 Main Menu 38](#_Toc103195839)

[3.8.3 Vaccine Certificate Details 38](#_Toc103195840)

[3.8.4 QR Code Generator 39](#_Toc103195841)

[3.9 Blockchain & IPFS Node Placement 41](#_Toc103195842)

[3.10 Conclusion 43](#_Toc103195843)

[4. Implementation 44](#_Toc103195844)

[4.1 Introduction 44](#_Toc103195845)

[4.2 Challenges 44](#_Toc103195846)

[4.3 Setup 45](#_Toc103195847)

[4.4.1 ‘Block’ Class 46](#_Toc103195848)

[4.4.2 ‘Blockchain’ Class 47](#_Toc103195849)

[*4.4.3* Blockchain Established 49](#_Toc103195850)

[4.5 QR Codes 50](#_Toc103195851)

[4.6 Processes 50](#_Toc103195852)

[4.6.1 Login 51](#_Toc103195853)

[4.6.2 View Vaccine Details 53](#_Toc103195854)

[4.6.3 View Chain 57](#_Toc103195855)

[4.7 Conclusion 58](#_Toc103195856)

[5. Testing & Results 59](#_Toc103195857)

[5.1 Introduction 59](#_Toc103195858)

[5.2 White Box Testing 59](#_Toc103195859)

[5.3 Black Box Testing 59](#_Toc103195860)

[5.4 Testing Strategy 60](#_Toc103195861)

[5.5 Certificate Verification Evaluation 62](#_Toc103195862)

[5.6 Non-Functional Requirements Evaluation 64](#_Toc103195863)

[5.7 Conclusion 64](#_Toc103195864)

[6. Conclusion 65](#_Toc103195865)

[6.1 Introduction 65](#_Toc103195866)

[6.2 Summary of Findings 66](#_Toc103195867)

[6.3 Limitations 67](#_Toc103195868)

[6.4 Recommendations for Future Work 68](#_Toc103195869)

[6.5 Conclusion & Self-Reflection 68](#_Toc103195870)

[Appendix A: References 69](#_Toc103195871)

[Appendix B: Code 72](#_Toc103195872)

[B.1 blockchain.py 72](#_Toc103195873)

[B.2 storeHash.py 75](#_Toc103195874)

[B.3 pdf.py 75](#_Toc103195875)

[B.4 models.py 76](#_Toc103195876)

[B.5 auth.py 76](#_Toc103195877)

[B.6 \_\_init\_\_.py 78](#_Toc103195878)

[B.7 main.py 78](#_Toc103195879)

[B.8 base.html 79](#_Toc103195880)

[B.9 login.html 80](#_Toc103195881)

[B.10 vaccineDetails.html 81](#_Toc103195882)

[B.11 admin\_menu.html 82](#_Toc103195883)

[B.12 blockchain.html 83](#_Toc103195884)

[B.13 certRecords.html 84](#_Toc103195885)

[B.14 verify.html 85](#_Toc103195886)

[Appendix C: Initial Artefact Code 87](#_Toc103195887)

[C.1 Certs.sol 87](#_Toc103195888)

[C.2 Certs.test.js 88](#_Toc103195889)

[C.3 Migrations.sol 89](#_Toc103195890)

[C.4 1\_initial\_migration.js 89](#_Toc103195891)

[C.5 2\_deploy\_contracts.js 89](#_Toc103195892)

[C.6 App.js 90](#_Toc103195893)

[C.7 Main.js 91](#_Toc103195894)

[C.8 Navbar.js 92](#_Toc103195895)

[C.9 index.js 92](#_Toc103195896)

# Table of Figures

[Figure 1 - Simple Illustration of a Blockchain 7](https://d.docs.live.net/9dd6de4466b88500/Desktop/Dissertation.docx#_Toc90472631)

[Figure 2 - Venn Diagram of Blockchain Types & Categories (E. Wegrzyn and Wang 2021) 7](#_Toc90472632)

[Figure 3 - IPFS File Handling 12](https://d.docs.live.net/9dd6de4466b88500/Desktop/Dissertation.docx#_Toc90472633)

[Figure 4 - Table Comparing the Attributes of 1D Barcodes to 2D Barcodes (Powell, 2019) 14](#_Toc90472634)

[Figure 5 - Basic Structure of a QR Code (Hong, 2020) 15](#_Toc90472635)

[Figure 6 - Use Case Diagram 22](#_Toc90472636)

[Figure 7 - Login Page 25](#_Toc90472637)

[Figure 8 - Main Menu 26](#_Toc90472638)

[Figure 9 - Vaccine Certificate Details Page 26](#_Toc90472639)

[Figure 10 - QR Code Generator 27](#_Toc90472640)

[Figure 11 - QR Code 27](#_Toc90472641)

[Figure 12 - High Level Overview of Proposed System 28](#_Toc90472642)

[Figure 13 - Peer-to-Peer Connections and Validity Testing with Blockchain & IPFS 29](#_Toc90472643)

# Table of Tables

[Table 1 - Hardware Specifications 19](#_Toc102941533)

[Table 2 - Software Specifications 20](#_Toc102941534)

[Table 3 - Login Use Case 23](#_Toc102941535)

[Table 4 - View Certificate Use Case 23](#_Toc102941536)

[Table 5 - Generate Encrypted QR Code Use Case 24](#_Toc102941537)

[Table 6 - View Certificates Use Case 24](#_Toc102941538)

[Table 7 – Verify Certificates Use Case 25](#_Toc102941539)

[Table 8 – View Blockchain Use Case 25](#_Toc102941540)

[Table 9 - Logout Use Case 26](#_Toc102941541)

# Introduction

Blockchain is an emerging venture in the health industry. Forecasts predict that the use of blockchain for healthcare data exchange will have a market value of $1.89 billion by 2025 (BIS Research, 2018). However, recent concerns over data security in healthcare have invoked distrust from the public especially when an organisation like the HSE is reported to still employ the use of 30,000 Windows 7 machines in 2021, and is attacked with data encrypting ransomware (Weckler, 2021). Although blockchain could decentralise & encrypt the data as a preventive measure, it simply is too expensive to solely rely on it for storage, with 1MB of storage on the Ethereum network costing roughly $17,000 (Pinto, 2020).

Since the beginning of the Covid-19 pandemic, hospitals and clinics have seen a rise in cyberattacks (Mitchell, 2021). In addition to having to store individual medical records, now they also must store individual covid vaccine details with the rollout of vaccine passports, which brings additional security/data privacy threats (Abid *et al.* 2021). On top of that, many medical services make use of camera scannable QR codes to store vaccine certs. This can also prove a threat too due to how the static barcode can be altered to carry out harmful activities on the user’s scanner device and the lack of credential input generally found to access this encoded information (Krombholz *et al.* 2015). Unauthorised access can be mitigated using cryptography. Take Yaoqiu Hong who applied DES encryption on QR codes to act as access authorisation to workplaces, whose passwords changed regularly (Hong, 2020).

In 2020, a company called Vaxiglobal worked on a COVID-19 immunity verification system built using blockchain technology, which incorporated QR codes via a mobile app or paper document as travellers saw fit (U4 Anti-Corruption Resource Centre, 2021). This system, although competent in preventing fraudulent behaviour, has no mention of ensuring end users’ documentation to remain hidden from others, both on the network & people with viable QR scanners.

This dissertation will build on the aforementioned study; with the introduction of decentralisation tech like blockchain, Interplanetary File System (IPFS), and secure encryption of QR codes. The use of which could achieve complete user confidentiality & mitigation of malicious activity.

## 1.1 Background

A blockchain is a persistent, transparent, publicly visible, append-only ledger that allows parties in the network to trade without needing to trust one another, by advertently making the data immutable. Blockchain was first conceived by a person/group under the pseudonym of Satoshi Nakamoto, as a means of creating a complete peer-to-peer exchange system for the cryptocurrency known as Bitcoin (Nakamoto, 2008). It has been adapted over the years to operate as a data exchange system outside of the virtual currency market, and into fields like banking & healthcare.

Key issues with blockchain when storing files on-chain are latency & scalability, which will bring the system to a halt as every user will have to download a copy of all files uploaded to the chain (Kumar *et al.* 2020). Therefore, files containing sensitive information cannot reside here.

Conceptualised by Juan Benet, the Interplanetary File System is a distributed, peer-to-peer network that enables devices access to the same files. This is done through a process referred to as content-based addressing, with files being retrieved using their assigned hash value (Benet, 2021).

Quick Response codes are encoded pieces of data, that take the form of matrix barcodes. They utilise the scanner found in most mobile phone cameras today, to decode & display the data to viewers. The barcodes of QR codes are near indistinguishable from one another, which makes it difficult to tell a genuine one from a hostile one. In the wrong hands, genuine QR codes found in the real world can be slightly tweaked or outright replaced to cause those to scan it to initiate unwanted events on their devices like SQL injections or similar attacks (Krombholz *et al.* 2015).

Blockchain has the key characteristics of decentralisation & immutability to improve secure data sharing in the medical sector. Since blockchain data is transparent by nature, a correlated IPFS could strengthen the weaker aspect of the system. Abid et al(2021) proposed a system where an off-chain IPFS held & encrypted COVID-19 results, with the hashes of which kept on a private blockchain.

## Purpose

The purpose of this dissertation is to produce an IPFS to store personal health information, in conjunction with a blockchain, to reinforce confidentiality. The decision to store records on an off-chain IPFS ensures compliance with GDPR and protects sensitive data from potentially appearing to unauthorised users on the network.

In this application, users will log in with their credentials to view their record details for COVID-19 immunity. On that page, they will have the option to generate a QR code to store this data, with a password to accompany it. This will address the potential ease of access malicious attackers may have had when scanning the code with their mobile camera. Producing an app as mentioned, will hopefully promote institutions to integrate their systems more often with blockchain and break down the scepticism users may have when engaging with QR code & blockchain systems.

No data belonging to individuals will be used for this study, it will be dummy data generated by the user for the purpose of testing and analysis of the developed application.

## 1.3 Aims & Objectives

### 1.3.1 Aims

This dissertation aims to disprove some misconceptions surrounding the safety of QR codes & systems currently in-place to handle COVID-19 certification data, as well as learn about the possibilities that a blockchain application could offer through a working artefact.

### 1.3.2 Objectives

* Research & investigate blockchain, IPFS & QR code security concepts.
* Examine what features the discussed tech can offer to COVID-19 immunity tracking.
* Construct a blockchain in the Python language.
* Develop code that will generate password encrypted QR codes for personal info regarding COVID-19 immunity tracking.
* Create off-chain storage to house all files, and have it store metadata on the blockchain.
* Evaluate & analyse the system’s functionality & methods of securing data.

## 1.4 Research Question

Can blockchain and encrypted QR codes improve protection against malicious activity & reduce cases of unauthorised access for COVID-19 certificates?

## Outline

The dissertation from here is divided into 5 main sections. The sections are as follows:

1. Chapter 2 - Literature Review: Extensively covers published information relevant to the project.
2. Chapter 3 – Design: Details the intended design & research that will go into the development of the project.
3. Chapter 4 – Implementation: Informs the reader how the ideas presented were formulated into a working application, with snippets of code to accompany the text.
4. Chapter 5 – Testing & Results: Demonstrates how the application can be tested effectively and the testing of Functional Requirements. Relevant tests are documented.
5. Chapter 6 – Conclusion: This chapter summarises the overall process of the project along with problems faced and recommendations for future work.

# 2. Literature Review

## Introduction

On March 11th, 2020, the World Health Organisation (WHO) officially stated that the COVID-19 disease, which was first identified in China in 2019, was a global pandemic. The disease has since claimed over 5 million lives across 200 countries worldwide (Our World in Data, 2021). SARS-CoV-2 has not only posed a global crisis for health services but socioeconomics in general, which have had major changes to our day-to-day lives. The measures taken to curb the spread of the disease has introduced new challenges to the public. The obligatory vaccination certificates for attendance to venues/events has seen many people turned away from the premises, due to misplacing their print certificate prior. Digital certificates alleviate this burden, as well as offer a more secure and faster verification process (Gruener, 2020).

The following chapter will provide a technical breakdown of permissioned & permissionless blockchains, IPFS & QR codes. It will also briefly address rules & regulations related to data handling, and its importance in instilling private & secure measures in an organisation. Topics like privacy, integrity & security of sharing sensitive data are at the forefront of this research, with strengths & flaws of these roles being identified in each technology mentioned. In the sum of its parts, this work will aid in the development of a decentralised, secure means of storing & sharing COVID-19 certificates for individuals.

## 2.2 Blockchain

The first prominent iteration of the tech was developed as a peer-to-peer solution for the Bitcoin cryptocurrency. It was conceptualised to make double-spending of the currency near impossible without the involvement of a third party. This was achieved by timestamping & hashing transactions in tandem with one another using the proof-of-work (PoW) consensus method, which forces peers to exhaust computational power & time to meet requirements. The majority decision on recording what currency was spent on what transactions, boils down to where the greatest CPU power resides (Nakamoto, 2008). Presuming it works as intended with honest peers withholding the CPU advantage, attackers’ attempts to surpass the honest chain of transactions with PoW with less computational power will be in vain.

The technology has seen several enhancements & improvements since. Besides cryptocurrencies, the technology has been adapted to go as far as securing the transfer of virtual assets like artwork or music as seen in non-fungible tokens (NFTs) and has laid out the foundation for what we know as Blockchain today. A blockchain in and of itself is a digital ledger, documenting transactions between computer systems in a series of blocks. By distributing & duplicating this ledger across every system in a network, it forms what is called a blockchain network. Each block in the chain contains several transactions and every time a new transaction occurs, a record of that transaction is added to every participant’s ledger. The decentralised database managed by numerous participants here is known as Distributed Ledger Technology (DLT).

## 2.3 Distributed Ledger Technology

Distributed Ledger Technology (DLT) describes a peer-to-peer network in which transaction details between users are recorded and cryptographically secured, in a decentralised fashion (BlockstreetHQ, 2018). This decentralisation refers to how records are stored across multiple entities in the network, rather than a prone central location with a single point of failure. In addition, entities/nodes on a DLT network can only view & update transaction information on the ledger, not delete them, and nodes can see who modified what in the ledger. That transparency coupled with the immutable nature of the data reduces the chances of fraud & manipulation from occurring.

Over the years, various takes of the DLT formula have appeared in the form of Hashgraph, R3 Corda & IOTA Tangle. Neither of these DLTs have ever played a more prominent role in the market as much as Blockchain.

A blockchain inhibits the previously mentioned core principles of a DLT while breaking away from traditional data formatting. In typical databases data is stored in rows, columns and files. As the titular technology is concerned data is broken up into blocks, with each one connected via cryptographic hashes:

**Block *N+2***

**Timestamp:**

**Hash:** 96b3TYU

**Previous Hash:** A71vIHS

**Block *N+3***

**Timestamp:**

**Hash:** JXp21Q

**Previous Hash:** 96b3TYU

**Block *N+1***

**Timestamp:**

**Hash:** A71vIHS

**Previous Hash:** 0EAD532Z

Figure 1 - Simple Illustration of a Blockchain

There are four main types of blockchain networks in the field, with each one catering to aspects better than others. As shown in Fig. 2, a blockchain network can fall under one of two categories: permissioned or permissionless blockchains; with an exception given to the hybrid blockchain type:

Figure 2 - Venn Diagram of Blockchain Types & Categories (E. Wegrzyn and Wang 2021)

Diagram, venn diagram

Description automatically generated

### 2.3.1 Public Blockchain

Most famously used for mining & exchange of cryptocurrencies, a public blockchain allows anyone with internet access to sign onto the network to become an authorised node who can verify & engage in transactions, as well as suggest changes to the system. Each node has an identical copy of the public ledger, so every transaction initiated on the network is recorded on every node in the blockchain. Transactions are handled by the system and only digital signatures are used for identification and authentication, hiding the real-life identity of nodes (Kumar *et al.* 2021).

In an environment as such, anonymity is valued here due in large part to the independence of a public blockchain. The authority that comes from a single body overlooking a network, is not present; so even malicious individuals are treated to the same permissions as neutral users (Cullen *et al.* 2021). A security risk that is of major concern is the infamous 51% takeover attack, where if over half of the blockchain’s computing power belongs to attackers, they can essentially deny any transaction from occurring. One example of the occurrence happened to a fork of the cryptocurrency Bitcoin Cash, known as Bitcoin SV, in August 2021. As Nikita Zhavoronkov, lead developer of Blockchair stated, 570,000 transactions were wiped out from the blockchain by the attacker (Avan-Nomayo, 2021).

On a network scaled to house possibly thousands of nodes situated all over the globe, performance & scalability are bound to be hindered somewhat. So many nodes, means so many users to obtain consensus from, plus more transactions that the blockchain will have to store.

### 2.3.2 Private Blockchain

A private blockchain is a limited network of nodes, that work under the control of a single authoritative company. Features like transaction of assets & public ledger duplicates for nodes remain intact from the previous iteration, the difference being it is applied on a much smaller scale. The controlling company achieves this via authorisation processes e.g., login credentials & permission levels, specific to an individual node (TechTarget, 2021). As a result, anonymity is all but present in this setup.

The authoritative approach & small pool of vicinity-based nodes makes this system a centralised approach as opposed to the other, more distributed offerings. This goes against the core value of DLTs, however as Chair Professor Kai-Lung Hui put:

*“..blockchain does not have to be distributed – although its practical value is very much in question without it.”* -(BlockstreetHQ, 2018)

Hui indicates that by a blockchain being centralised, it fails what it sets out to achieve. Eradicating any singular points of failure in a private blockchain is not possible, as the parent company nodes are typically situated geographically close to one another. This leaves it more prone to data failures than public blockchains.

Limiting the number of nodes to interact with on a network brings both positive & negative aspects to the system. On the one hand, performance can only flourish as the system can only scale to a certain user threshold. On the other hand, fewer nodes reduce data redundancy & the number of nodes needed to agree on a consensus, attribute to the hinderance of network security (Read BTC 2021). If a handful of nodes were to turn rogue, the network consensus could be manipulated to appeal to those orchestrating the attack.

Another concern is the lack of trust among nodes. The organisation overlooking operations can decide what transactions are valid or invalid and can even go as far as to reverse transactions so that data can be removed from the blockchain. This is a possibility, as *Zhang et al* iterates, only a single participant such as the organisation can have write permissions on the blockchain (Zhang *et al.* 2019).

### 2.3.3 Hybrid Blockchain

As the name suggests hybrid is a middle ground between the public & private blockchain types. It grants organisations access to both a private & public system, allowing them to hide information they deem confidential amidst hundreds of nodes. Nodes are granted access to the network by pre-existing members, who confirms if they are suitable for the network. Once verified, the nodes have permission to read & write to the blockchain and can carry out transactions. This authorisation process, inherited from the private blockchain aspect, ultimately protects the network from a 51% attack.

Missteps from having an authority figure in a private blockchain like data mutability, are sidestepped since the power of the authority here is limited to declaring which transactions are made public or private (Read BTC, 2021). Having said that, sectioning off data from specific users will see a drop in transparency from that found in a public blockchain. Anonymity from the public iteration of blockchain though is retained here to a certain extent. Nodes remain anonymous up until they carry out a transaction, which then results in the exchanging party learning of their identity (TechTarget, 2021).

Hybrid shares the same scalability & swift performance offered by a private system, without falling into the similar single point of failure trappings. This is prevented thanks to the further distribution of computing power derived from a public blockchain. The scalable nature & fast processing times of the system is a by-product of retaining transactions in their own private chains, and only committing to the system’s public chain when verification is required (Geroni, 2021).

### 2.3.4 Consortium Blockchain

Considered a variation of the private blockchain, consortium differs from private by distributing the system throughout multiple organisations (Kumar *et al.* 2021). This eliminates concerns over potential security issues that come with placing nodes within proximity to each other and retains the decentralised aspect seen in both the public & hybrid chains.

Although multiple organisations are now in control of the blockchain, data is still somewhat vulnerable to alterations because transactions are overlooked by a small group of nodes. Known as validators, these nodes are solely responsible for validating every transaction on the network (Parizo, 2021). If they were to fall in the wrong hands, attackers would have influence over all transactions in the system. Thankfully, decentralisation paired with the delegation of authority reinforces a more secure, consensus-driven environment than that offered by a private blockchain.

Other similarities with a private chain include the lack of transparency to that offered in a public network, the fast transaction times, lack of anonymity & scalable form factor. Overall, a consortium system does not want to remain public nor private. Instead, it opts to create an environment open enough for organisations across the globe to collaborate, all the while remaining hidden to everyday individuals.

Blockchain technology has already proven itself to be a viable option for dealing with issues associated with the pandemic with processes like supply chain management in healthcare (Kalla *et al.* 2020). Taking from that, the inherent transparency of a blockchain network, which instils integrity & accountability of data, could circumvent instances of contradictory COVID-19 certificate data from various sources. This factor would also establish a more trustworthy platform for citizens to engage in, as fears of misinformation are significantly reduced (Parizo, 2021).

## 2.4 Data Regulations

Considering COVID-19 certificates contain sensitive info specific to an individual, adherence to the HIPAA & GDPR would be mandatory in a system storing such content. Failure to do so would counteract much of the safeguards in place and breach the privacy & security of individuals who entrusted the system. The Health Insurance Portability and Accountability Act of 1996 (HIPAA) applies to entities within the United States, even people not considered to originate from there. Whereas the General Data Protection Regulation (GDPR) covers organisations that process personal EU citizen information, regardless of their whereabouts in the world. Though jurisdiction for the two is vastly different; with GDPR covering all personal data of an individual and HIPAA narrowing in on protected health information, both overlap & help form the basis for data handling in the healthcare sector.

Adhering to these regulations with just a blockchain alone would prove to be somewhat difficult to achieve. First off, the immutable factor of the technology would challenge the ‘Right to be Forgotten’ & ‘Right to Rectification’ articles exclusive to GDPR since once blocks are stored on a chain, they cannot be altered or deleted. An issue that would affect both HIPAA & GDPR would be to directly store sensitive info on the blockchain. Doing so would possibly let sensitive info leak via blockchain scans to other users on the network (Abid *et al.* 2021).

In accordance with the aforementioned regulations, it would be in best interest for a system to pair the data integrity properties of a blockchain alongside an off-chain storage. This would only come to better the security etiquette of the data, which HIPAA & GDPR both hold in high regard. The next few sections will discuss various systems/methods for storing and sharing data off-chain.

## 2.5 Interplanetary File System

Development of the Interplanetary File System (IPFS) was led by a small team of developers, headed by Juan Benet, who strived to create a peer-to-peer based, decentralised file platform similar to BitTorrent. Git also influenced the team during development, which is mirrored in the way in which IPFS handles file changes. Known as versioning, alterations made to file data are saved through separate, subsequent versions (Benet, 2021). Meaning even if data was to get overwritten, previous states of that data can be retrieved before the mishap, a significant contributor towards the data redundant & immutable nature of the system.

The Interplanetary File System (IPFS) is a distributed means of storing & sharing data, that works on a peer-to-peer network. A file uploaded to IPFS that is above 256KB is divided into chunks or data objects. Each chunk is hashed and assigned a unique content identifier (CID); a technique used over the typical file location method to map to files based on their data make-up. This form of file lookup is known as content-based addressing.

Figure 3 - IPFS File Handling

Merkle DAG

Qmx2t35woby

3da541559918

011011110110

Icon

Description automatically generated

Qmhyt2ev16

Qmp78scF49z

**File**

cef7e59218e3a

81b06facd90fe

110001110110

101001101101

Qmjmhn2drty53vws9csd4ewa32boivv7

**Base CID**

**CIDs**

**Hashed Chunks**

**256 KB Chunks**

As depicted in Figure 3 above, the CIDs of these data objects come together to develop an indicator of the entire file, in the form of a Base CID. It is worth noting that the CID does not indicate where the content is stored on the system. Instead, that duty is designated to the Distributed Hash Table (DHT), which maps what the user is looking for to the peer that is storing the content (Benet, 2021). It is divided amongst all the peers in the network, storing details like CIDs and their corresponding owner with it.

The content-based addressing that IPFS offers can be attributed to the data structure it inherits, the Merkle Directed Acyclic Graphs (DAG). A Merkle DAG takes after a Merkle tree, storing hashes of numerous files/folders (CIDs) in nodes under a branching path formation, that all map the large volume of data together under the root node (Base CID). This structure lends itself to an immutable nature. Meaning if any changes are made to a subsequent file, they will be reflected in the ascendant hashes of that node (Chen *et al.* 2017).

The IPFS concept is highly promising and is a beneficial alternative to storing data. No nodes on an IPFS are privileged & there is no single point of failure, akin to the decentralised blockchain. There are several drawbacks to be aware of when dealing with IPFS. One such limitation is retaining the permanent presence of a file on the network when no other node wants to host said file. The original node can host it, but there is no guarantee that there will be redundant copies of it unless the network is made private. Another issue arises when dealing with the likes of identification documents, as new files must be uploaded to replace the old ones before manually ridding each outdated one (Tabora, 2020). This is a huge hassle with public networks, as an unknown number of obsolete files can be spread across multiple nodes.

## 2.6 Quick Response Codes

A Quick Response (QR) code is a two-dimensional (2D) barcode created by Denso Wave in 1994, to originally track automobile parts for the vehicle manufacturer Toyota (Denso Wave, 2014). The prevalence of QR codes amongst the public today is a phenomenon, inhibiting a wide range of purposeful data. From URLs to WiFi credentials, to their more current venture of storing vaccine data for verification purposes. QR codes are not the first or even the most prominent application of barcodes in the real world.

Prior to 2D barcodes, one-dimensional (1D) barcodes were the only option for encoding data into machine-readable shapes. Even though 1D barcodes still has their uses in retail & parcel tracking to name a few, 2D barcodes trumps it in almost every aspect as outlined below in Fig. 4:

Figure 4 - Table Comparing the Attributes of 1D Barcodes to 2D Barcodes (Powell, 2019)

Table

Description automatically generated

What makes QR codes the more viable option amongst other 2D offerings like Data Matrix & PDF417, is the increased storage capacity. QR codes can hold up to 4,296 alphanumeric characters, Data Matrix 2,335 and PDF417 only 1,850 (Dynamsoft, 2020). In recent years, sub-categories of the QR barcode such as IQR & SQR codes have emerged to increase storage limitations, error correction & data encryption even further.

Not only that but the contemporary widespread adoption of this matrix barcode for services like venue tickets & restaurant reservations, cannot be understated. Blue Bite reported that from 2018 to 2020, there was a 94% growth in QR interactions alone (Blue Bite, 2021). The default feature of most smartphone cameras having a built-in QR scanner only exemplifies the tech’s prominence in the current enterprise.

## 2.7 Anatomy of a QR Code

Figure 5 - Basic Structure of a QR Code (Hong, 2020)

Diagram, schematic

Description automatically generated

The basis for any QR code’s structural makeup is an array of squares coloured either black or white, to represent a binary number. Black being equal to ‘1’, and white equal to ‘2’. Every square or segment of squares have a defined role to play in the decoding process of a scanner:

### 2.7.1 Finder Pattern

The three-corner position/finder patterns provide the position & orientation of the code, allowing the scanner to detect it at any angle or direction. These are made distinguishable to the lens by their bigger black & white enclosures and their slight isolation away from the main body of data.

### 2.7.2 Format Information

Right next to the finders, format information of the QR regarding error correction level & data mask pattern utilised are kept. Error correction in 2D barcodes, enable data to still be readable from a dirty or slightly damaged QR. Masking ensures an equal distribution between black and white modules (Kieseberg *et al.* 2010). The mask pattern is automatically chosen by the encoding software, to ensure an equal number of both black & white modules appear.

### 2.7.3 Alignment Pattern

What looks to be a miniature finder segment, an alignment pattern corrects issues concerned with image distortion that may appear in big codes to a software scanner.

### 2.7.4 Timing Pattern

Timing pattern strips consist of black & white pixels, which run horizontally & vertically between the three corner symbols. They provide synchronicity for pixels throughout the code and help software to determine the width of a module (Meetheed 2017).

### 2.7.5 Version Information

As it implies, version information encodes which iteration out of the 40 possible options the code corresponds to.

### 2.7.6 Error Correction Data

As discussed prior, error correction code has helped strengthen the overall readability of code in varying degrees. It can range from Low (7%), Medium (15%), Quarter (25%) or High (30%), depending on the creator’s preferences. The higher the level, the more damage-resistant a QR is. Though as the levels climb higher, capacity for the readable data symbols drops (QRStuff 2011).

### 2.7.7 Data

The sole existence of a QR code is the information that may resemble a message, document, webpage link or alike converted to a bitstream, and segmented into 8-bit codes (Meetheed 2017). This desired information is then derived from a scanner, made possible to decode thanks to the help of the previously mentioned QR properties.

## 2.8 QR Features

QR codes are designed to provide users with information as quickly and efficiently as possible. They can be scanned with ease, either through the built-in camera of a mobile phone or with third-party scanning applications. The economical nature of these barcodes makes it a viable option for a business to direct potential customers to their website, without the need for advertisements. It is inexpensive both in terms of marketing for the business and in navigation for the consumer, as it is all handled by the camera of an associated device.

Another plus is the ease of portability that comes with these types of codes. A person can just as easily take a photo or screenshot of the original QR and show them to an individual, who can then scan it using their device. This can get content quicker and to more people than having to deal with print media. There are many advantages to using QR codes, and while they are very versatile, they are not without their limitations.

## 2.9 QR Security Flaws

As with any piece of technology, Quick Response codes are not without their flaws. Given the black & white pixel format this data is represented in, alterations could be made to the barcode to re-interpret the original legitimate data into something more harmful without much notice given to it. This can yield substantial results for attackers, who can then carry out malware or phishing attacks on any unaware user who scans it. QRs can be made into an attack vector in one of two ways; either by outright replacing a QR with an ill-intended one or by manually modifying a QR pixel structure. Modifying pixels can alter several different factors including character counter & character mode, which can lead to buffer overflows and potentially SQL injections (Kieseberg *et al.* 2010).

The boundaries of QR code manipulation have exceeded both in scope & complexity in recent times. Referred to as ‘Barcode-in-Barcode’ attacks Dabrowski *et al*. demonstrated how various 2D barcodes could be embedded into QRs, including QRs themselves, to target not only specific barcode reader apps but also certain mobile OSs. Research for this was conducted by taking twelve popular scanners across Android & IOS and scanning a self-made QR in QR barcode, alternating the position of the inner malicious QR each time to see what yielded the best results. The centre positioning paired with the white spacing surrounding the inner QR finder markers was the most promising, with all twelve apps only capable of decoding the malicious one (Dabrowski *et al.* 2014). A countermeasure claim to this is to stick to low data size and error correction levels where possible, as it will reduce the unattended space for attackers to take advantage of.

A more common concern with sharing sensitive data through QR codes is that anyone could use another individual’s code or access the content of a QR code without having to worry about verification of identity. Though this could be a prominent issue, a blockchain environment could counteract this problem. Mainly with the use of public-private cryptographic keys, as demonstrated by Eisenstadt et al. research with an implementation found in a COVID-19 antibody testing app (Eisenstadt *et al.* 2020). Here, peers on the network were able to validate an antigen test encrypted with a private key by using the corresponding public key to view & compare details of the owner, like their device’s digital signature. Thus, reducing potential threats of identity fraud.

## 2.10 Conclusion

After conducting this literature review, the research indicates that storing private data on QR codes alone is not a secure, viable mechanism in and of itself. The data that resides in the barcodes would need to be tethered to an immutable form of data storage, that makes use of cryptography to secure the data.

An ideal candidate for this would be a blockchain, more specifically a private blockchain. That way it could avoid having sensitive data transparent to all users on the network by some form of access control. This would be a sufficient choice if there were no restrictive storage limitations with such tech. Even if the blockchain had no such hinderance peers could unknowingly stumble upon the file data of other nodes by performing a scan on the chain, which would go against GDPR standards.

This is where IPFS comes in, to provide the storage capabilities for the data where blockchain could not. IPFS will cryptographically hash & store the file data within its ecosystem, while also storing the hash of the data on the blockchain for nodes to verify. These decentralised technologies come to form a secure, protective means of sharing confidential files with QR codes. The next chapter will discuss and detail the proposed system.

# 3. Design & Methodology

## 3.1 Introduction

This part of the paper will cover the design process behind the desired system. As established, the system is intended to securely store COVID-19 vaccine certs with the help of DLTs & encrypted QR codes. The sections to follow will detail specifications like the hardware & software requirements and functional & non-functional requirements. Diagrams like the use case & storyboards will be accompanied with text to elaborate on the concepts presented.

The system will not include a registration page. Instead, the data will be hardcoded into the database, assuming that every user has got a vaccination at one point in time. The data in question will be ‘dummy data’, personal details of fictional people. This cuts out any potential ethical issues that may arise.

## 3.2 Hardware Requirements

Table 1 below displays the hardware used to manufacture the system and its specifications:

Table 1 - Hardware Specifications

|  |  |
| --- | --- |
| **Hardware** | **Specifications** |
| Laptop | Intel Core i5-1135G7 CPU, 8GB RAM, Windows 10 OS |
| Android Mobile Device | Android Version 11 |

## 3.3 Software Requirements

The software and versions used to create the system are outlined in Table 2:

Table 2 - Software Specifications

|  |  |  |
| --- | --- | --- |
| **Software** | **Version** | **Description** |
| Visual Studio Code | 1.60.2 | A multilanguage integrated development environment, which will house the coding for this project. The reason for choosing this over other outings like Eclipse is that there are fewer hurdles to getting languages up & running. In Eclipse, the PyDev plug-in must be installed to effectively code in Python. |
| Python | 3.10.3 | A high-level programming language that does more, in less lines of code. This will be used to create the likes of the blockchain and the mobile application. Apart from its efficiency with lines of code, python was chosen due to previous experience with it. An additional benefit with python is the ease of installing packages that comes with ‘pip’. |
| Flask | 2.0.3 | A micro web framework for Python, that can help build a web application with even a single python file. It also packages in the bootstrap toolkit which can make it work well across both PC and mobile device web browsers, without having to write code for it. |

## 3.4 Functional Requirements

**FR-1:** The user must be able to login to the app with an account number & password.

**FR-2:** A logged-in user must be able to view their vaccine cert information upon logging in.

**FR-3:** The application must be able to generate a unique, scannable QR code of the certificate.

**FR-4:** An admin can view & verify each cert record stored on the database, to the data stored on-chain

**FR-5:** An admin can examine the metadata of each block on the blockchain

**FR-6:** A user & an admin should have the ability to log out.

## 3.5 Non-Functional Requirements

### 3.5.1 Security

**NFR-1:** Authenticity of a user will be validated through credential login.

**NFR-2:** That certification info will be stored in the database, the hash of which is then stored in the blockchain.

**NFR-3:** The hash of both on-chain & off-chain data should be viewable for the admin to manually cross-examine and validate similarity.

### 3.5.2 Reliability

**NFR-4:** The system should function without any abhorrent bugs or faults.

**NFR-5:** The system should have a fast response time in terms of general performance & QR scanning, to ensure a smooth user experience.

### 3.5.3 Fault Tolerance

**NFR-6:** Since a blockchain & IPFS must have multiple nodes, there must be multiple copies of the ledgers.

### 3.5.4 Usability

**NFR-7:** The app should be easy for users to navigate through.

### 3.5.5 Confidentiality

**NFR-8:** The data must be encrypted before being stored on the chain, so as to comply with GDPR.

## 3.6 Use Case Diagram

Figure 6 shows a use case diagram for the proposed system. The user of the system will be an everyday citizen who has had at least one COVID-19 vaccination to date. The admin will be a node with high-level permission:

Figure 6 - Use Case Diagram

Diagram

Description automatically generated

## 3.7 Use Case Descriptions

Every use case has a specific purpose and the prerequisites needed to achieve it. The following tables describes the intended flow and alternative flow for each use case in the event of an error:

Table 3 - Login Use Case

|  |  |
| --- | --- |
| **Use Case** | Login |
| **Objective** | To have the user login to the app |
| **Precondition** | The user has their credentials on the IPFS. |
| **Main Flow** | 1. User inputs account number & password 2. User clicks the login button |
| **Alternative Flow** | 1. Invalid login credentials – display error message |
| **Post Condition** | User successfully logs in and is greeted by the main menu selection – which varies depending on what type of account logged-in |

Table 4 - View Certificate Use Case

|  |  |
| --- | --- |
| **Use Case** | View Certificate Information |
| **Objective** | Output the details of the user, including vaccine info |
| **Precondition** | The user has selected the titular option from the main menu |
| **Main Flow** | 1. Upon selecting the choice, the app displays all info pertaining to the individual using the hash from the blockchain. |
| **Alternative Flow** | 1. The user can also just show the QR code, which the recipient can scan, and view by providing them their password |
| **Post Condition** | The user can view their personal COVID-19 vaccination details |

Table 5 - Generate Encrypted QR Code Use Case

|  |  |
| --- | --- |
| **Use Case** | Generate Encrypted QR Code |
| **Objective** | To create a secure QR code, initialised with the personal details of the user |
| **Precondition** | The user must have a vaccine certificate registered with their account |
| **Main Flow** | 1. The user must input a password of their choice into the text field provided 2. The user clicks ‘Generate QR Code’ 3. The QR code is then displayed to the user |
| **Alternative Flow** | 1. The password does not meet the desired properties outlined by the app, and an error appears. |
| **Post Condition** | A password-protected QR code containing the certificate data is created |

Table 6 - View Certificates Use Case

|  |  |
| --- | --- |
| **Use Case** | View Certificates |
| **Objective** | To be able to view each record present on IPFS |
| **Precondition** | The admin must have knowledge of the account-specific password to display details |
| **Main Flow** | 1. Admin logs in and selects ‘View All Certificates’ 2. The admin clicks the ‘Unlock’ button of the desired record to view 3. Admin inputs password for record |
| **Alternative Flow** | 1. Incorrect password is entered – error message displayed |
| **Post Condition** | Vaccine certificate details of relevant record holder is displayed |

Table 7 – Verify Certificates Use Case

|  |  |
| --- | --- |
| **Use Case** | Verify Certificates |
| **Objective** | To be able to verify that the certificate record replicates the data on blockchain |
| **Precondition** | The admin must have ‘unlocked’ a record from ‘View Certificates’ |
| **Main Flow** | 1. Admin selects ‘Verify’ to view hashes on & off-chain |
| **Alternative Flow** | 1. Upon ‘unlocking’ record, if the off-chain hash does not match the on-chain, an error message will be displayed |
| **Post Condition** | Pop-up window detailing hashes, are shown |

Table 8 – View Blockchain Use Case

|  |  |
| --- | --- |
| **Use Case** | View Blockchain |
| **Objective** | The admin can analyse metadata from blocks in the chain |
| **Precondition** | The admin must be logged in, and the blockchain instantiated with mined blocks |
| **Main Flow** | 1. Admin selects ‘View Chain’ from the main menu |
| **Alternative Flow** | 1. They can navigate back to main menu with ‘Back to Admin Menu’ |
| **Post Condition** | Admin can do a deep dive into each block of the blockchain |

Table 9 - Logout Use Case

|  |  |
| --- | --- |
| **Use Case** | Logout |
| **Objective** | To sign out the user from the application |
| **Precondition** | User must be at the main menu of the application |
| **Main Flow** | 1. The user clicks the ‘Logout’ option. 2. A notification reiterating that the user wants to sign-off. 3. The user confirms it. 4. They are signed out of their account and are back at the ‘Login’ screen |
| **Alternative Flow** | 1. The user denies the notification of signing out |
| **Post Condition** | The user has signed out of the application |

## Storyboards

The sample GUIs to follow, aim to visualise the look & functionality of each intended page of the application. This only covers the interface for the user actor.

### 3.8.1 Login

Figure 6 below starts off with the login page, the page they will be greeted by when booting up the application. Here users will need to enter the account number & password, in order to access any one of the other features of the app:

Figure 7 - Login Page

Graphical user interface, text, application

Description automatically generated

### Main Menu

Figure 7 depicts the various options available to the user once a successful login is achieved. There are three options to choose from. ‘View Vaccine Details’ will bring the user to GUI outlined in Figure 8. ‘Generate Encrypted QR Code’ will land the user at ‘QR Code Generator’ seen in Figure 9. ‘Logout’ will bring the user back to the ‘Login’ screen of the application:

Figure 8 - Main Menu

Diagram

Description automatically generated

### Vaccine Certificate Details

Figure 8 shows the vaccine certificate page. Here, details about the user such as date of birth, the date the vaccination was administered to them and status against COVID-19 is displayed. Additionally, specifications of the vaccine they received are shown here as well. The user can transition back to the menu, with a button situated at the bottom of the page:

Figure 9 - Vaccine Certificate Details Page

Graphical user interface, text, application

Description automatically generated

### QR Code Generator

Clicking ‘Generate Encrypted QR Code’ will bring the user to the page seen in Figure 9. The user must adhere to guidelines when forming a password for their QR code, otherwise, they will not proceed to creating it. If the user wishes, they can backtrack to the main menu via the ‘Cancel’ button.

When the user does input an appropriate password into the text field and clicks the ‘Generate QR Code’ button, their request will be met and a code will be displayed for scanning purposes:

Figure 10 - QR Code Generator

Graphical user interface, text, application, chat or text message

Description automatically generated

When the user does input an appropriate password into the text field and clicks the ‘Generate QR Code’ button, a code will be displayed for scanning purposes as visualised in Figure 10:

Figure 11 - QR Code

Qr code

Description automatically generated

## Blockchain & IPFS Node Placement

Figure 12 - High-Level Overview of Proposed System

Diagram

Description automatically generated

Figure 12 gives a general overview of the proposed system above. It tracks the timeline of the system from its initial scan with a QR code reader, through to the retrieval of data via content-based addressing in IPFS. The private blockchain network will handle the verification of users with a combination of validator nodes & smart contracts, as shown below in Figure 13. Without going too in-depth, smart contracts are a series of statements that are coded & stored on a distributed ledger. They essentially act as ‘if-else’ statements to an extent, executing when certain conditions are met.

Figure 13 - Peer-to-Peer Connections and Validity Testing with Blockchain & IPFS

Diagram

Description automatically generated

## 3.10 Conclusion

The various structural aspects of the proposed COVID-19 vaccine certification system were discussed in full here. The use cases detailed the processes involved with logging in, viewing and eventually generating barcodes of their data. These activities were illustrated with the help of mock-up interface designs, further cementing the user experience throughout their interaction. The use case diagram & functionalities help shape the roles of what the blockchain & IPFS are going to adopt within the system.

# Implementation

## 4.1 Introduction

This chapter will detail the development process of ‘COVID-19 Certificate App’; a data immutable, blockchain based web application with secured QR codes, in-depth. In doing so, the implementation chapter will fulfil its purpose of meeting objectives 3, 4 & 5 identified in Chapter 1. Additionally, challenges that were had in adapting the artefact through numerous iterations will also be discussed.

## 4.2 Challenges

The project underwent multiple reworks through its development lifecycle. To start, the artefact itself veered away from mobile app to web app development, with the Python language being side-lined in favour for JavaScript & Solidity. The reasoning behind this decision at the time was based on how many of the Solidity compatible libraries were built with JavaScript in mind, and because of the vast amount of documentation coverage there was for JavaScript over Python.

By essentially adopting two new programming languages, and a new framework in the form of Truffle, there was quite a learning curve to understand & adapt to. This artefact got as far as to have a React DApp up & running, with Ethereum test networks to house & handle the smart contracts developed to create & add certificates to a Ganache blockchain. This had much more lofty ambitions behind it, than what was presented with the proof of concept.

The screengrabs & files for this initial concept can be found in Appendix C of this document. It provides an insight to what was encountered software wise, and skills that were refined throughout the procedure. But for delivery upon the final artefact, this approach ultimately proved to be fruitless.

Severe changes were made considering the roadblocks with linking a Moralis database to the DApp. Due to budgetary & time constraints, this regressed the developments back to a Python Web App which opted for an off-chain SQLite server, and self-built blockchain to store, process, hash, and timestamp transactions within the user’s machine. The blockchain itself takes after the methodology set out by a Bitcoin chain, implementing the proof-of-work consensus to determine the nonce of each block before having the privilege to mine it.

## 4.3 Setup

I began by creating up a virtual environment for the code, in order to manage the dependencies for the project much easier and avoid any potential conflicts with libraries later down the line. In the project directory via the CLI (Command Line) I used ‘*python3 -m venv venv’*, to create the environment, then proceeded to boot up said environment through the ‘activate’ file in the *‘bin’* subdirectory. Figure 14 visualises the CLI of Visual Studio Code when inside the project virtual environment:

Figure 14 – Indication of Presence in Virtual Environment ‘venv’



Once up & running, there were packages that had to be configured for the project within the environment. The list of packages, and corresponding versions of them set up were as follows:

Figure 15 - Package List

Text

Description automatically generated

**4.4 Blockchain**

The ‘blockchain.py’ file creates & instantiates a local iteration of a Bitcoin blockchain. The python file imports the SHA-256, datetime, time & JSON dependencies. JSON & SHA-256 will help in the computing of the hashes of the blocks. Datetime & time will timestamp each block in the respective chain.

Figure 16 - Imports for ‘Blockchain.py’ file

Text

Description automatically generated

The python file is divided into two classes: the ‘Block’ Class, and the ‘Blockchain’ Class. Both classes have two identical methods defined: ‘\_\_init\_\_’ & ‘\_\_str\_\_’. ‘\_\_init\_\_’ simply outlines the parameters & values for both classes, and ‘\_\_str\_\_’ returns their defined class back as a String object.

### 4.4.1 ‘Block’ Class

The ‘compute\_hash’ method converts the python dictionary, or transaction contents, into a JSON string and stores it within ‘block\_string’. The block’s transactions contents are then encoded with the SHA-256 algorithim, with the results stored in ‘first\_hash’ which itself is then hashed to return our block hash.

Figure 17 - ‘Block’ Class Method ‘compute\_hash’ as Shown in ‘Blockchain.py’

Text

Description automatically generated

### 4.4.2 ‘Blockchain’ Class

A genesis block is the first block that a cryptocurrency ever mines, and in turn forms the foundation for the rest of the blocks to follow. As shown below, there is a method to generate such a block and append it to the blockchain:

Figure 18 - ‘genesis\_block’ & ‘getLastBlock’ Methods from ‘Blockchain’

A screenshot of a computer

Description automatically generated with medium confidence

As to mirror Bitcoin’s blockchain, the proof of work consensus mechanism has been replicated so that the computer running this method must exert processing power to solve a mathematical puzzle, the nonce. The value on the ‘difficulty’ value determines how hard the puzzle will be. For my workspace I chose a difficulty of 2, to attempt to spread out the timestamps in which the blocks are added to the chain.

Figure 19 - ‘proof\_of\_work’ Method Which Takes After Bitcoin’s Consensus Mechanisim

Text

Description automatically generated

When called with an appropriate Python dictionary object parsed as a parameter, the length of the chain will be taken as the index for this block, and the previous hash derived from the last object placed in the chain. The content of the dictionary will go under the ‘current\_transaction’ section of the block or labelled as ‘data’ here. From there, the block hash will be derived from the result from the proof of work consensus carried out on the block, which is then appended to chain followed by its own transaction.

Figure 20 - ‘add’ Method to Append a Block to the Chain

A screenshot of a computer

Description automatically generated with medium confidence

To display the blocks in the command line, ‘getTransactions’ is called. Here, each label is checked to see if they match with a block on the chain. If so, they can be displayed either all together separated by their labels or by themselves if the method is parsed with the block’s Integer ID. The blocks do not contain the data of the certificate explicitly, but rather the hash of the concatenated string of data from the record of a fictional individual. The hardcoded data of each block can be seen in Appendix B: Code Listing.

Figure 21 - ‘getTransactions’ to Print Blocks of the Chain

Text

Description automatically generated

### *4.4.3* Blockchain Established

Before closing off the ‘blockchain.py’ file, each block defined in the file is added to the blockchain ‘*B’* for the ease of reference for the various other files in the application. The *‘main’* method depicted here will be called from the *‘main.py’* file, to outline the populated mock records on the blockchain system upon start-up of the application each time.

Figure 22 - Methods of ‘Blockchain.py’ Called to Populate the Chain ‘B’

Text

Description automatically generated

## 4.5 QR Codes

The dynamic QR codes were created through the website ‘QR Code Chimp’, and allows for scanners to download the PDF, but unable to access the content if not knowledgeable of the password safeguarding it.

Figure 23 - QR Codes of Cert PDFs Supplied by ‘QR Code Chimp’

A screenshot of a computer

Description automatically generated with medium confidence

Although the QR codes were not developed in-house, the advantage with this outsourcing is the unique & unconventional makeup of these QR codes. This paired alongside the decision to use small 512px sized QRs, would make it quite resistant against malicious alterations being made to the barcode.

## 4.6 Processes

The COVID-19 Certificate Apphas a various number of processes to contend with. These processes are integral to maintain so that the system is to remain fully operational. Such processes are listed below:

1. Login
2. View Vaccine Details – Plaintext (User)
3. View & Verify All Certificates (Admin)
4. View Chain (Admin)

Each process will have a corresponding HTML file found in the ‘templates’ directory and a dedicated route in the ‘auth.py’ file, which will define & carry out most of the functionality of the process in question. The contents of the ‘templates’ directory is demonstrated in Figure 24:

Figure 24 - HTML files That Make-Up the ‘templates’ Directory

Text

Description automatically generated

### 4.6.1 Login

Figure 25 - ‘login.html’ Webpage

Graphical user interface, text, application

Description automatically generated

Upon entering the web app, the user will be greeted by the login section where they will be required to input their account number and password that are registered on the off-chain database. The functionality of this process is carried out in the ‘/’ route seen in the ‘auth.py’ file.

Figure 26 - ‘/’ Route

Text

Description automatically generated

After checking if the user credentials exist on the system, the account number of said user is examined to see if it falls within one of two ranges. 1 – 4999 indicates that they are a user. This then directs them to the titular ‘Vaccine Certificate Details’. Whereas account numbers falling within 5000 – 9999 implies it is an admin account, sending them to the ‘Admin Menu’.

Figure 27 - ‘adminMenu.html’ Webpage

Graphical user interface

Description automatically generated

### 4.6.2 View Vaccine Details

Figure 28 - ‘vaccineDetails.html’ Webpage

Qr code

Description automatically generated

A regular user will have the vaccine certificate information relevant to them loaded and outputted on the HTML page as plaintext. Alternatively, there is a QR code that users could save to avoid the hassle of logging in through the web app every time. The substitute file stored within the QR code is password encrypted, the key to which is equivalent to the user’s account password. Each PDF file was password protected with the 128-bit encryption offered by the PyPDF2 library.

The code written to apply this password-encryption takes the declared file parsed to the ‘open()’ function through the ‘pdf’ variable, and has every page of the file read & encrypted with the password specified. This pdf then overwrites the unencrypted pdf, restricting unauthorised access to sensitive certificate data.

Figure 29 - ‘pdf.py’ Code

Text

Description automatically generated

* + 1. **View & Verify All Certificates**

Figure 30 - ‘certRecords.html’ Webpage

Diagram, qr code

Description automatically generated

Moving onto the Admin processes, the first action this actor can perform is to view & verify the records. If the admin attempts to ‘Unlock’ any of the certificates, they are prompted by a pop-up screen to enter the appropriate password to gain access, as seen in Figure 31 below:

Figure 31 - Pop-Up Screen Prompted by ‘Unlock’ Button

Qr code

Description automatically generated

Upon confirmation of password, the record will unlock and lead into the verification process of the record. The verification stage involves fetching the sha-256 hash of the concatenated certificate data from the off-chain storage and comparing it to see if it holds up with the ‘data\_hash’ found in its associated block in the blockchain:

Figure 32 - ‘storeHash.py’ Code

Text

Description automatically generated

Any alteration in the off-chain data will reflect in the hash, which in turn will signal a warning with a banner at the top of the webpage. The verification code handled by the ‘*/verify’* route of ‘*auth.py’* is showcased in Figure 33 below, followed by the user interface representation of a successful verification window in Figure 34:

Figure 33 - ‘/verify’ route in ‘auth.py’

A screenshot of a computer

Description automatically generated with medium confidence

Figure 34 - Verify Pop-Up Showcasing the SHA-256 Hashes of the Data On-Chain & Off-Chain Respectfully

Qr code

Description automatically generated

### 4.6.3 View Chain

Figure 35 - ‘blockchain.html’ Webpage

Qr code

Description automatically generated

The admin can partake in viewing the blockchain of the system, segmented block-by-block via HTML card elements. The QR codes previously seen in the view certificates process of both actors make a return, with them signifying what certificate corresponds to which block on the chain. The admin can analyse significant pieces of information about each block on the chain, such as timestamps & nonce of the mined block from here.

## 4.7 Conclusion

To summarise, this chapter recounted the development of the ‘COVID-19 Certificate App’, from the setup of the virtual environment, through to both back & front-end functionalities of the application. Some alterations between this chapter and features proposed in Chapter 3 were altered or outright cut. This was due to the vision of the artefact changing after undergoing various trials & tribulations with its first iteration. Such features to suffer were the user’s main menu and QR code Generator, the latter of which was replaced by an online QR code generator.

Though online QR code generators are a controversial pick for potential security vulnerabilities, the dynamic QR codes address the main concerns outlined by this paper. Those being the lack of credential input required from the user for accessing QR code documents and the danger of malicious attacks performed on static QR barcodes, as mentioned in Chapters 1 & 2.

# 5. Testing & Results

## Introduction

The following section will cover the strategies that will be used to test the developed system proposed in previous chapters, which will meet the criteria for objective 6 outlined in section 1.3.2 of this document. Black box and white box testing will be the supporting pillars of the testing strategy, with functional and acceptance testing evaluating the functional and non-functional requirements laid out in the design chapter. Adoption of these methodologies will provide a look at both sides of the same operations.

## White Box Testing

White box testing is concerned with assessing the internal elements of the application. A developer’s point of view is adopted here, meaning internal knowledge of components is readily available and tested. Testing scenarios for this specific system will go behind the scenes on the web app, to verify the efficiency of the data immutability and security measures present.

Acceptance testing will be the subset of white box category used. This testing technique can be seen as a validation activity for the users of the artefact (Thomas Hamilton, 2022). The feedback from the customer will be the final indicator on whether the system is considered complete.

## Black Box Testing

Black box testing opts to focus in on the core functionality of the system, as opposed to every facet of the overall system. The perspective here will not know the inner workings of the system. Instead, the testers will take on & examine the processes undertaken by an unsuspecting user engaging with the blockchain & off-chain database via the web application interface. Functional testing will the prominent method used in this testing.

Functional testing concentrates on the requirements that were set out at the inception stage of the project. The main concern with this form of testing is ensuring that the output correlates with the specific output that is expected by the tester (Sten Pittet, 2022).

## Testing Strategy

Table 10 - Testing Strategy Results

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Testing** | **Steps** | **Expected Result** | **Actual Result** |
| **R001** | Application is launched and blockchain is initialised | White Box | 1./ User starts application | Application starts & blockchain initialised in < 30 seconds | **Pass** |
| **R002** | User Login | White Box/Black Box | 1./ User inputs registered credentials  2./ User clicks ‘Login’ button | Credentials are verified. Logged in user will have their certificate details displayed. Logged in admin will be directed to the menu page. | **Pass** |
| **R003** | Scan Encrypted QR code | Black Box | 1./ QR code is scanned with smartphone camera | Password protected PDF downloaded. Unlocked once correct account password is entered | **Pass** |
| **R004** | View & Verify Cert Records Stored Off-Chain, to Those On-Chain | Black Box/White Box | 1./ Navigate to ‘View all Certificates’ webpage as admin  2./ ‘Unlock’ record with account-specific password  3./ Click ‘Verify’ button | ‘Hash of Data on Blockchain’ should be the exact same as ‘Hash of Data on Off-Chain Storage’ | **Pass** |
| **R005** | Examine Metadata of Blockchain | White Box | 1./ Navigate to ‘View Chain’ as admin | Information like ‘nonce’, ‘timestamp’, ‘previous\_hash’, etc. should be prevalent for each of the six blocks, and be a mirror of what’s shown in the CLI. | **Pass** |
| **R006** | Logout | Black Box | 1./ User selects ‘Logout’ | User is logged out, and re-directed back to login page | **Pass** |

**R001 –** The blockchain is initialised with the mined blocks, the result of which is printed in the CLI. The Flask server boots up shortly after, navigating the user to the login page when the localhost IP address is entered into a web browser

**R002 –** When a valid account number & password are entered into their corresponding fields, a request is sent to pull the details from the off-chain database. Once verified and depending on the account logged-in, the app will transition to either the vaccine certificate for the user account, or the main menu page for an admin.

**R003 –** As either admin or user, QR codes for certificates are displayed for scanning in ‘View All Certificates’ or ‘Vaccine Certificate Details’ respectfully. Scanning with a smartphone will result in the default browser of the smartphone to open, requesting to download PDF of certificate. Although anyone can freely download it, access to seeing it is restricted on an account-specific, 128-bit encrypted password.

**R004 –** Exclusive to the admin account, each certificate record has the capability to viewed. Like the QR code, it is locked behind a password which is unique for each record and is only known by the user of that account. Once given the password, if the record in the database has been altered, a message will flash indicating so to the admin. The admin can double-check this through the ‘Verify’ button to cross-examine the hashes. The functional testing of this event will be elaborated upon in section 5.7.

**R005 –** Another functionality limited to admin usage is the ability to the view the blockchain itself. By traversing to ‘View Chain’, the admin can view statistics of each block, which would help in providing further evidence to malicious activity occurring on the off-chain database.

## 5.5 Certificate Verification Evaluation

This section will demonstrate the reliability of how the system checks for data interference between the database and blockchain. For this, the testing will be carried out on an admin account, so as to have access to the ‘Verify’ functionality. Proceeding that, the ‘database.db’ file and ‘storeHash.py’ file will be tested to understand how the hashes of data are computed. For this evaluation, the cert record with the ID of 1 will be the use case, parallel with its subsequent block on the blockchain.

Figure 36 - Pop-Up Window for Matching Hashes

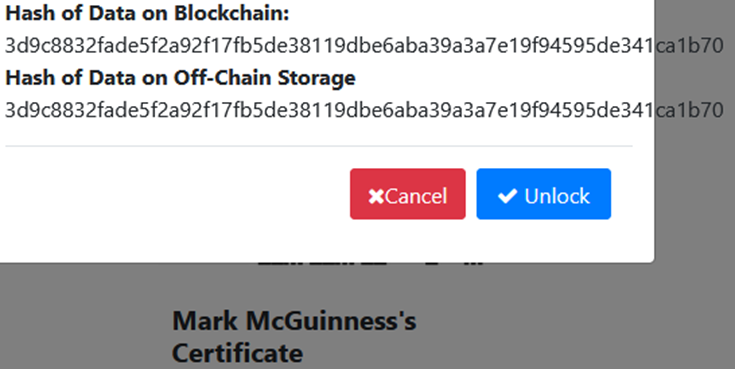


Figure 36 shows the subsequent verification pop-up window which appears for ‘Mark McGuinness’ Certificate’, illustrating that the two hashes are identical. The hash on-chain is hardcoded, whereas the off-chain hash is dynamically created based on the data present in the certificate record. To validate that any change made will reflect in the off-chain hash, the cert record had one letter deleted from it, and restarted the application to see what would occur.

Figure 37 - Pop-Up Window for Mismatching Hashes

Graphical user interface

Description automatically generated

By just removing the letter ‘k’ from ‘Mark’, we see a huge change in the hash. But alongside this change, the admin is notified with a warning message illustrated in Figure 38 instead of the regular “Record Unlocked”, to highlight the changes made.

Figure 38 - Flash Warning Message for Mismatching Hashes



As mentioned, this is because hash of the certificate on the ‘database.db’ is computed, by pulling the data from each column in a row with an SQL query into a list. The list is converted into a single string, concatenating the values together with ‘.join()’ function. Sha-256 hashing is applied to the string, by the sought after ‘hashlib’ module.

Figure 39 - 'select\_cert\_by\_id' Method in 'storeHash.py'

Text

Description automatically generated

To summarise, any changes made to the off-chain data that does not equal exactly to that found on the subsequent block of data, will be notified to the admin. This is due to how the hashing of the off-chain data occurs dynamically, and the procedures in place to raise red flags when such a thing happens.

## 5.6 Non-Functional Requirements Evaluation

This section concerns itself with evaluating if the requirements defined in section 3.5 were met to the utmost standard. Each webpage in the application takes a minimalist approach, with little to no flashy styles to distract the user and no more than one or two pieces of functionality per webpage so as to not overwhelm the user. The application is slow upon start-up due to booting up Flask, and mining & instantiating a blockchain simultaneously. Though after that it is a smooth, instant experience moving from webpage-to-webpage. Feedback is also provided to the user in the form of flash messages, which will indicate positive or negative responses. Like in the case of logging in, a green flash declaring “Logged in successfully!” will occur after entering correct credentials. This application is suitable for use by almost anyone, despite their lack of technical knowledge.

## 5.7 Conclusion

This chapter has outlined and completed the testing strategy that was formulated in section 5.6 of this chapter. The testing strategy not only evaluated the functional requirements but was also able to assess the non-functional requirements. The authenticity of the verification of certificates was analysed in-depth, which demonstrated & reinforced the reliability of such a process.

# 6. Conclusion

## 6.1 Introduction

The surrounding aim behind this project was to construct an application which adopted blockchain technology & off-chain storage solutions, to improve the integrity & authorisation of COVID-19 certificates.

The end-product is a web app that stored records on IPFS, and the hash of that data onto the blockchain. Users can login and share their details via plaintext or with a dynamically generated QR code, which was password protected by the user themselves, to share their vaccine certificate details.

To recap, the objectives were:

* Research & investigate blockchain, IPFS & QR code security concepts.
* Examine what features the discussed tech can offer to COVID-19 immunity tracking.
* Construct a blockchain in the Python language.
* Develop code that will generate password encrypted QR codes for personal info regarding COVID-19 immunity tracking.
* Create off-chain storage to house all files, and have it store metadata on the blockchain.
* Evaluate & analyse the system’s functionality & methods of securing data.

The following sections will outline how these objectives were met, plus any limitations set on the artefact, and considering the additional work that could be carried out for further improvement.

## 6.2 Summary of Findings

The research question that formed the foundation of this paper was “Can blockchain and encrypted QR codes improve protection against malicious activity & reduce cases of unauthorised access for COVID-19 certificates?”. The answer of such lies within on how each objective was met and how well the outcome supports the overall aim of this activity

The literature review covered the grounds for the first two objectives. There it was established that a private blockchain, paired with IPFS’s off-chain storage & hashing capabilities of records were the ideal candidates for handling data in the artefact. The initial plan of IPFS implementation had to be scrapped, as problems with the API interaction arose between the Infura IPFS and the web app during the implementation process.

The third objective was attended to, but with one omission: cryptographic encryption of QR codes. Instead, the PDF files in the QR code were password encrypted with the help of the library PyPDF2. The QR codes themselves, which housed these password-encrypted files, were dynamically generated on the website known as ‘QR Code Chimp’. The details of which is covered in the implementation chapter.

An SQLite database was used for the certificate records, meeting the responsibilities that were initially set out for IPFS. Like the cryptographic hashes, which were used to obtain a SHA-256 hash of a record and store it within the corresponding block in the chain. Although not the original vision of the system, this still met the milestone set out by objective five for the dissertation.

An element of the artefact that did remain intact since its concept phase was the self-constructed blockchain. The ‘blockchain.py’ file mimicked the operations of a Bitcoin chain, ensuring the integrity of data through the proof-of-work consensus and trustworthy hashes, which cryptographically chained the blocks together, satisfying the third objective of the dissertation.

The testing chapter outlined within this document, encompassed the sixth and final objective of analysing the security & functionality of the system. Aspects of the system like data immutability across the off-chain storage to the blockchain were analysed here, as well as assessing potential security flaws.

## 6.3 Limitations

Unforeseen circumstances between the researcher contracting COVID-19, and mismanagement of time stemming from the initial concept were huge factors that had the artefact taking on less & less features throughout development. These instances of time constraints did bleed into and affected every aspect of the final artefact; but they were only partially to blame for missing pieces of functionality.

The complexity of not just working with the hashing & linking capabilities of blockchain, but creating one from scratch in a Python programme, reduced the scale of the project quite drastically. The working artefact, that intended to incorporate multiple virtual machines to act as various nodes in a blockchain network, could not be made possible for various reasons. Those being the lack of hardware capabilities on the laptop to run two or more virtual machines at once, plus the lack of budgetary options for getting such a network up & running in the first place. This scaled the artefact back to build, run & interact with the blockchain on one node essentially.

A feature that was heavily implied to appear in the artefact, encrypted QR codes, as projects like *Abid et al.* ‘NovidChain’ and *Yaoqiu Hong’s* DES QR encryption was talked throughout the paper, was scrapped. The only semblance to this is that the QR codes store a 128-bit password-encrypted PDF file, produced by the python library PyPDF2. The reasoning for this admittedly come down to the lack of knowledge on how to construct a QR code generator to store files, let alone one that incorporates encryption standards.

With that said, every other aspect both within the research question and the objectives themselves were deemed to be successful.

The artefact in question delivered on what the research had promised; an application more adept way at protecting data against malicious modification or unauthorised access via QR codes than other offerings on the market. Without a doubt an encrypted QR code generator would push this ideal further forward. But even with the significant absences of some requirements, this project has had more than enough takeaways on my end to consider it a successful, but arduous learning experience.

## 6.4 Recommendations for Future Work

The blockchain constructed here was restricted to work solely on the user’s laptop when booting up the application. This is due to how Python Flask operated, with the browser sending an API request to Flask on the same machine. For a decentralised solution this would not suffice in the real world as there are no other nodes to verify data or run the blockchain so when the machine turns off, the blockchain goes with it. The obvious step forward would be to have more than one machine running the ledger of the blockchain, to ensure it does not have a single point of failure.

On that topic, the off-chain database would need a similar upgrade to the blockchain. This would come in the form of IPFS, which would have been implemented in this artefact, were it not for the problems with linking it to the Infura API. Again, parallel to the blockchain network, multiple machines would be needed to attend to keeping the IPFS branch up & running.

The final area where enhancements can be made to is adapting the QR functionality to get to what it was originally set out to do, have written Python code to dynamically generate an encrypted QR code, whose cryptographic key is based from what the user inputted as the password for the QR.

## 6.5 Conclusion & Self-Reflection

Upon reflecting on this project from concept all the way through to having a working application there were big missteps made on my part, from a development aspect of it. Th misjudgement of trying to take on the already complicated task of building a blockchain paired with learning JavaScript for the first time in the initial venture, took away half of the development time for the Python application that was ended up with. In that regard having the foresight to drop that development sooner rather than later would have benefitted, as the absence of promised functionality in the final artefact reflects this.

In saying that, the experience had with that initial concept provided invaluable knowledge & skills with Solidity smart contracts, Web3.js & MetaMask, that participating in such processes was not a regretful decision whatsoever. Although none of this appeared within the artefact showcased for this dissertation it was a learning experience, nonetheless.

# Appendix A: References

Abid, A., Cheikhrouhou, S., Kallel, S., and Jmaiel, M. (2021) ‘NovidChain: Blockchain-based privacy-preserving platform for COVID-19 test/vaccine certificates’, *Wiley*, Journal of Software: Practice and Experience, 27, available: https://doi.org/10.1002/spe.2983.

Avan-Nomayo, O. (2021) ‘Bitcoin SV rocked by three 51% attacks in as many months’, *Coin Telegraph*, 8 Jul, available: https://cointelegraph.com/news/bitcoin-sv-rocked-by-three-51-attacks-in-as-many-months [accessed 11 Jun 2021].

Benet, J. (2021) ‘IPFS Documentation’, available: https://docs.ipfs.io/ [accessed 26 Oct 2021].

BIS Research (2018) *Global Blockchain in Healthcare Market- Analysis and Forecast, 2017-2025*, available: https://www.prnewswire.com/news-releases/global-blockchain-in-healthcare-market-to-reach-561-billion-by-2025-reports-bis-research-680230953.html [accessed 16 Oct 2021].

BlockstreetHQ (2018) ‘Before Blockchain, There Was Distributed Ledger Technology’, available: https://medium.com/blockstreethq/before-blockchain-there-was-distributed-ledger-technology-319d0295f011 [accessed 11 Jul 2021].

Blue Bite (2021) ‘The State of QR in 2021’, available: https://www.bluebite.com/nfc/qr-code-usage-statistics.

Chen, Y., Li, H., Li, K., and Zhang, J. (2017) ‘An improved P2P File System Scheme based on IPFS and Blockchain’, *IEEE*, IEEE International Conference on Big Data, 2652–2657.

Cullen, G., A. Mayfield, T., F. Keys, J., Curran, K., Leeney, M., and Bailey, K. (2021) ‘Link-by-Link: A Practical Review of Blockchain’, *IEEE*.

Dabrowski, A., Krombholz, K., Ullrich, J., and R. Weippl, E. (2014) ‘QR Inception: Barcode-in-Barcode Attacks’, *Association for Computing Machinery*, SPSM ’14: Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices.

Denso Wave (2014) ‘QR Code Development Team Becomes First Japanese Winner of European Inventor Award’, available: https://www.denso-wave.com/en/adcd/info/detail\_\_251.html [accessed 11 Nov 2021].

Dynamsoft (2020) ‘What is the Difference Between QR Code, PDF417 and DataMatrix?’, available: https://www.dynamsoft.com/blog/insights/qr-vs-pdf-417-vs-datamatrix/ [accessed 11 Nov 2021].

Eisenstadt, M., Ramachandran, M., Chowdhury, N., Third, A., and Domingue, J. (2020) ‘COVID-19 Antibody Test/Vaccination Certification: There’s an App for That’, *IEEE*, IEEE Open Journal of Engineering in Medicine and Biology, 148–155, available: https://doi.org/10.1109/OJEMB.2020.2999214.

Geroni, D. (2021) ‘Hybrid Blockchain: The Best of Both Worlds’, available: https://101blockchains.com/hybrid-blockchain/ [accessed 11 Sep 2021].

Gruener, D. (2020) *Immunity Certificates: If We Must Have Them, We Must Do It Right*, Harvard University, Edmond J. Safra, available: https://ethics.harvard.edu/files/center-for-ethics/files/12immunitycertificates.pdf?m=1588342659 [accessed 12 Nov 2021].

Hong, Y. (2020) ‘Design of Intelligent Access Control System Based on DES Encrypted QR Code’, *IEEE*, 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications( AEECA), available: https://doi.org/10.1109/AEECA49918.2020.9213475.

Kalla, A., Hewa, T., Anand Mishra, R., Ylianttila, M., and Liyanage, M. (2020) ‘The Role of Blockchain to Fight Against COVID-19’, IEEE Engineering Management Review, 48(3), 85–96.

Kieseberg, P., Leithner, M., Mulazzani, M., Munroe, L., Schrittwieser, S., Sinha, M., and Weippl, E. (2010) ‘QR Code Security’, *SBA Research*, MoMM ’10: Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia, 430–435.

Krombholz, K., Fr ̈uhwirt, P., Rieder, T., Kapsalis, I., Ullrich, J., and Weippl, E. (2015) ‘QR Code Security -- How Secure and Usable Apps Can Protect Users Against Malicious QR Codes’, 2015 10th International Conference on Availability, Reliability and Security, 230–237, available: https://doi.org/10.1109/ARES.2015.84.

Kumar, R., Marchang, N., and Tripathi, R. (2020) ‘Distributed Off-Chain Storage of Patient Diagnostic Reports in Healthcare System Using IPFS and Blockchain’, *IEEE*, 2020 12th International Conference on Communication Systems & Networks (COMSNET, 1–5, available: https://doi.org/0.1109/COMSNETS48256.2020.9027313.

Kumar, S., Kumar Bharti, A., and Amin, R. (2021) ‘Decentralized secure storage of medical records using Blockchain and IPFS: A comparative analysis with future directions’, *Wiley*, 16, available: https://doi.org/10.1002/spy2.162.

Meetheed (2017) ‘What Are The Different Sections In A QR Code?’, available: http://qrcode.meetheed.com/question14.php [accessed 14 Nov 2021].

Mitchell, H. (2021) ‘Healthcare data breaches up 55.1% in 2020, report finds’, available: https://www.beckershospitalreview.com/cybersecurity/healthcare-data-breaches-up-55-1-in-2020-report-finds.html [accessed 23 Oct 2021].

Nakamoto, S. (2008) ‘Bitcoin: A Peer-to-Peer Electronic Cash System’, Journal for General Philosophy of Science, available: https://bitcoin.org/bitcoin.pdf.

Our World in Data (2021) ‘Coronavirus Pandemic (COVID-19) – the data’, available: https://ourworldindata.org/coronavirus-data [accessed 12 Nov 2021].

Parizo, C. (2021) ‘What are the 4 different types of blockchain technology?’, available: https://searchcio.techtarget.com/feature/What-are-the-4-different-types-of-blockchain-technology [accessed 11 Sep 2021].

Pinto, R. (2020) ‘Costs Of Storing Data On The Blockchain’, available: https://www.linkedin.com/pulse/costs-storing-data-blockchain-rohan-pinto/.

QRStuff (2011) ‘QR Code Error Correction’, available: https://blog.qrstuff.com/2011/12/14/qr-code-error-correction [accessed 14 Nov 2021].

Read BTC (2021) ‘Public, Private and Hybrid Blockchain Explained’, available: https://www.readbtc.com/blockchain-101/public-private-hybrid-blockchains [accessed 11 Sep 2021].

Sten Pittet (2022) ‘The different types of software testing’, available: https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing [accessed 5 Sep 2022].

Tabora, V. (2020) ‘Using IPFS For Distributed File Storage Systems’, available: https://medium.com/0xcode/using-ipfs-for-distributed-file-storage-systems-61226e07a6f [accessed 12 Dec 2021].

TechTarget (2021) *Blockchain for Businesses: The Ultimate Enterprise Guide* [online], TechTarget, available: https://searchcio.techtarget.com/Blockchain-for-businesses-The-ultimate-enterprise-guide [accessed 11 May 2021].

Thomas Hamilton (2022) ‘What is User Acceptance Testing (UAT)? with Examples’, available: https://www.guru99.com/user-acceptance-testing.html [accessed 5 Sep 2022].

U4 Anti-Corruption Resource Centre (2021) ‘Covid-19 vaccine certificates: the digitalisation of fraud’, available: https://medium.com/u4-anti-corruption-resource-centre/covid-19-vaccine-certificates-the-digitalisation-of-fraud-2ea84426ec18 [accessed 23 Oct 2021].

Weckler, A. (2021) ‘Ireland’s cyber security branded “a joke” at Oireachtas Committee hearing’, *Irish Independent*, 22 Sep, available: https://www.independent.ie/business/technology/irelands-cyber-security-branded-a-joke-at-oireachtas-committee-hearing-40878363.html [accessed 10 Sep 2021].

Zhang, R., Xue, R., and Liu, L. (2019) ‘Security and Privacy on Blockchain’, *ACM Computing Surveys*, 52(3), 34.

# Appendix B: Code

## B.1 blockchain.py

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

## B.2 storeHash.py

Text

Description automatically generated

## B.3 pdf.py

Text

Description automatically generated

## B.4 models.py

Text

Description automatically generated

## B.5 auth.py

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

## B.6 \_\_init\_\_.py

Text

Description automatically generated

## B.7 main.py

Graphical user interface, text

Description automatically generated

## 

## B.8 base.html

Text

Description automatically generated

Text

Description automatically generated

## B.9 login.html

Text

Description automatically generated

## B.10 vaccineDetails.html

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

## B.11 admin\_menu.html

Text

Description automatically generated

## B.12 blockchain.html

Text

Description automatically generated

Shape

Description automatically generated with medium confidence

## B.13 certRecords.html

Text

Description automatically generated

Text

Description automatically generated

Shape

Description automatically generated with medium confidence

## B.14 verify.html

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

# Appendix C: Initial Artefact Code

## C.1 Certs.sol

Text

Description automatically generated

Text

Description automatically generated

## C.2 Certs.test.js

Text

Description automatically generated

Text

Description automatically generated

## C.3 Migrations.sol

Text

Description automatically generated

## C.4 1\_initial\_migration.js

Text

Description automatically generated

## C.5 2\_deploy\_contracts.js

Text

Description automatically generated

## C.6 App.js

Text

Description automatically generated

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

## C.7 Main.js

Text

Description automatically generated

## C.8 Navbar.js

Text

Description automatically generated

## C.9 index.js

Text

Description automatically generated