OCTOBER 31, 2017

FUNCTIONAL SPECIFICATION

MID IDENTITY ENGINE

CILLIAN MC NEILL - 14352621
SCHOOL OF COMPUTING
Supervisor: Geoff Hamilton

Table of Contents

Introduction **1**

Overview **1**

Business Context **1**

Glossary **2**

General Description **3**

Product / System Functions **3**

Mobile Application 3

Evaluation UI 3

Backend Server 3

Blockchain Network 3

Additional System Functionality 4

User Characteristics and Objectives **5**

Operational Scenarios **7**

Mobile Application Scenarios 7

Web Application Scenarios 8

Constraints **9**

Security 9

Mobile Operating System Support 9

Speed 9

Ease of use 9

Functional Requirements **10**

High Level Design **12**

Application – High Level Overview 12

Sequence Diagram – Verification Implementation 13

Object Diagram – Blockchain Models 14

Preliminary Schedule **15**

Introduction

Overview

The project I am developing is an identity engine that will aid in proving someone's identity. It will store information on the user's mobile device locally and will only be sent out with the user's permission. They can be verified by respected institutions and the proof of that verification will be stored with a distributed, tamper-resistant solution called the "Blockchain".
This system will tackle the problem of identity fraud, where the need for someone to prove who they are in the modern, social-networks driven society is incredibly difficult, as many people leave themselves vulnerable, by voluntarily exposing very personal information. For me to streamline this into a safe and secure service will not only provide peace of mind to potential users but will save companies and even countries a lot of money in the long run.

The project will be made up of a backend and two types of clients. These clients consist of an "Individual" and an "Identifying Party".

- The **Individual** will store personal information in a secure manner. Verification of this information will be initiated by the individual and the appropriate Identifying Party will be contacted. Requests to verify this information in the future will not require any contact with this party. All records of any verification request will be stored on the Blockchain. This will allow the user to prove who they are in a secure and streamlined manner.
- The **Identifying Party** represents the organisation working with the current identity infrastructure (e.g. Issuers of passports, driver licenses, etc.). They will make use of the APIs available in the system to verify information submitted by the user. Information that comes in can be compared with what's currently available in their system.

Business Context

While I came up with the project idea during my time spent in MasterCard, I will be developing the system by myself. While working in the company I noticed the need for an identity platform that would allow for companies like MasterCard to trust the end-user's while using their services. This extends beyond financial companies.
To have a universal form of identity, recognised by any institution, saving time and effort for everyone, especially the issuing institutions. Having a service like this would allow them to work in conjunction, eventually phasing out and replacing different identity documents by a single "master" one. On a case by case basis the system is saving the user hundreds in processing fees and on the large scale it removes the need for numerous institutions, saving the government large quantities of money.

We can see this benefit in a simple example of one person and the need to have a passport, public services card and driver license, alongside many other possible types of identifying documents. A lot of the information on these cards are mirrored and only serve to relay the same information but to a different institution. If we were to combine the information on these cards and store only what's necessary (e.g. name, date of birth, address and relevant ID numbers) then remove the need for many different forms of identity. Currently institutions do not allow for this functionality, however if the relevant information was provided at birth and updated through a person's life then the system will operate at peak efficiency.

Glossary

- **Blockchain**
    - A continuously growing list of records, called blocks, which are linked and secured using cryptography. Each block typically contains a link to a previous block along with any relevant data. Thanks to this and the overall design, blockchains are inherently resistant to modification of its data.
- **Distributed Ledger**
    - Replicated, synchronized and shared digital data that is spread across multiple locations.
- **Distributed Systems**
    - A model of computer network in which the systems pass messages to one another to complete a task. The distribution of these systems allows for concurrent work to be done and no one single point of failure
- **JSON**
    - A format that is easy for humans to read/write and computers to parse. Used in the transmission of messages across a network.
- **Hyperledger Fabric**
    - A branch of the hyperledger project (open source blockchain tools & distributed ledgers) originally contributed by IBM. It is a permission blockchain infrastructure using concepts such as roles between nodes and "smart contracts" to facilitate trading.
- **Node**
    - A JavaScript based development platform
- **SAFe – Scaled agile framework**
    - A form of the agile methodology employed in large organisations. It promotes collaboration and scalability with numerous teams.

General Description

Product / System Functions

The programme will be split into 4 main components. These components will carry out the various tasks necessary to facilitate the identity engine. The overall system will be composed of a backend component linked into the blockchain network. Clients will use the mobile application and evaluation UI to connect to the backend server which will in turn interact with the block chain. With this, we can facilitate communication between devices through the backend as well as any necessary interactions with the blockchain.

Mobile Application

The user will store and access their identifying information through this application. They will be able to register any of the supported document types (based on the Identifying Parties that are registered with the service) and verify them.
Once verified they can use the application in place of the original identity card. Requests for verification/pieces of their identity can be facilitated through this application. Every request, regardless of type will have to be accepted by the user before it's carried out. This ensures the user has total control of what information people have access to.

Evaluation UI

Users of this application will be part of an Identifying Party. Requests to check a user's validity will come through this system, containing the original document information and any other necessary information. Using this information, it is up to the identifying party to verify that user. Once verified, they can accept the user's request and the required transaction between the two parties will be added to the blockchain.

Backend Server

This server will facilitate communication between an Individual and an Identifying Party. It will also allow for the request of information from an individual and return it back (assuming they've accepted this request). This will be the applications bridge into the blockchain network, storing and querying requested information.

Blockchain Network

Serves as the primary storage location for interactions between Individuals and the Identifying Parties. Any Individual registration or verification requests are stored and updated in the blockchain to allow for future queries.
The distributed, tamper resistant nature of the block chain will allow for safe storage of these transactions. Having the transaction public but not the actual information will keep the user safe while still allowing for easy verification.

Additional System Functionality

There are extensions we can make to this application to allow it to be more widely adopted. While they are not critical, it would make the system more appealing in the eyes of the users.

Pets, much like people, require some form of identification. This means that the owner must carry that identification when necessary. To further increase the adoption of the application we can include an option to add the user's pet.
The inclusion of pets will come in the form of an additional profile a user can add. It would function in the exact same way that the original users profile would but it would contain information relating to the pet.

A further extension to the verification process would be to have a type of programmed processing of the submissions. We can match submitted documentation to what's on file and make a decision based on that.
To further improve the security of such an extension we can compare a submitted photograph with what's on file. Using photo recognition, we can ascertain whether this is the same person to a certain percentage.
A criterion can be set whether to accept it at a certain percentage or flag it for a human to process or drop the submission completely. Extensions such as these would decrease the workload for an identifying party significantly.

While an identifying party will be able to consume the systems APIs to receive submissions requests it might not be feasible for them to construct a whole new UI to consume them. To allow larger adoption, we can provide a sample UI that they can receive the requests from and just compare it to the original data from their system. This would not be ideal but in some scenarios, it may be necessary to allow them to transition easily without a complete overhaul to their current infrastructure.

User Characteristics and Objectives

The idea behind this system is to provide a quick and easy way for someone to prove who they are in both the digital and real world as well as allowing Identifying Parties to verify these users. This means that the system will have to be easy to use and understand. While this is true for the average user, extra steps will have to be made to ensure that Identifying Parties have easy access to this system so that they may carry out their job efficiently. With these ideas in mind, I have listed the features that are key:

- **User Friendly**
    - The mobile interface will need to be as simple as possible so that the user understands exactly what they're doing at any point. This will need to be reflected in the layout of the app along with how the features are described to the user. At no point should the user be confused about what action they need to take.

    - An Identifying party will ideally have their own system in place already and all we will provide is another stream of data from another location. While this avoids some of the problems of UI design we must ensure that the data they're receiving is well formed and easily understood so that they may incorporate it into their current system. This data will ideally be well-formed JSON.
- **Unobtrusive**
    - For the individual user, using this application should be very intuitive. No extra steps should be made in showing somebody your identity. It should be as close to showing a physical card as possible. This will have to be brought across in how the mobile application is laid out.

    - For Identifying Parties, the system should ask for no additional work, beyond connecting to the system. In the same way that they would process a new identity for someone, it would ask them to simply compare information of a current user to what is being submitted for verification. By not changing the workload, we can provide a smooth transition to a new system.

- **Speed**
    - Above all else, this must be as quick as possible for the individual user. For it to truly replace a physical card it must be as quick as showing that card. Further verification will require extra time, but a request for information from a user should be easily and effortlessly accepted and sent on with next to no delay. By doing this, the system will be as close as possible to having the physical card.

    - Identifying Parties will be dealing with hundreds of requests at any one time. It would be extremely important for us to ensure that the workload is manageable and processed in as quick a time as possible.
- **Security**
    - This application will be handling very sensitive data for the individual user so it is imperative that they know it's done in a secure manner. The application must allow the user to secure it through a password or biometric lock.
      Having the additional step to get into the app and their information will go a long way in comforting the user. Steps will also need to be made to ensure the user knows that no information leaves their device without them knowing.

    - Identifying Parties will have a deep concern for the security of their data and what goes in and out. We will need to ensure that submitted requests are secured in such a way that external access to this data would be extremely difficult to obtain.

    - Data stored within the servers of the application will need to be encrypted and unreadable by anyone but the intended party. Once requests have been submitted and processed there will be no reason to keep the data beyond what the outcome was.

The above characteristics are not unrealistic to expect from such a system. While there are no specific choices that will dictate how the system will incorporate these features, they will need to be at the forefront of every design choice made.

Operational Scenarios

The system, from an end user perspective, is divided into two front end applications. The mobile application for the individual to store their information and the web based, API consuming web interface for Identifying Parties to process validation requests. Due to the division outlined above, the proposed scenarios will be divided in that manner.

Mobile Application Scenarios

1. **Registration**
    - A user, when first setting up the application, will need to register an initial form of identification. No verification is necessary at this point but for further use of the application it is required. The user must enter the information on the relevant identification card. Once saved it can be accessed at any point.
2. **Verification**
    - If the user wishes to use a form of identification through the application then they must verify it. They will submit the entered information of that form of identification along with a current image (if applicable)
3. **Additional Identification**
    - If the user wishes to store more than one form of identification then they may follow the same steps as they did during the initial registration. Again, verification must be completed before actual use of that identification.
4. **Extensions to Individual**
    - In a scenario where the user has a pet that requires some form of identification then, if supported, they can add this information into their own application. Similar steps will be required to validate the information submitted but will allow for a wider adoption of the application
5. **Requests for information**
    - With an Id of a user given to an application looking for personal information, the mobile application will receive a request for a set amount of information. It is up to the user to accept or reject that request.
6. **Requests for validation**
    - If a user has a form of verified identity they may be challenged to prove that verification. This can be done through a further transaction on the Blockchain to show that the original transaction to verify the identification is using the same id as the one being used now.

Web Application Scenarios

1. **Submission Processing**
    - Requests for validation can be viewed through the web application. Information tied to that submission can be viewed and processed in a way suited to that identifying party. This can be in the form of comparing submitted information to what is stored such as the card id numbers or the picture submitted compared to what is on file.
2. **Submission Acceptance**
    - If a submission has met the Identifying party's acceptance criteria then they can mark the submission as verified. This verification is stored publicly and the end user is notified.
3. **Submission Rejection**
    - If, for some reason, the submission does not meet the party's acceptance criteria then they can mark it as rejected and return that result to the user.
4. **Submission Automation**
    - To speed up the process of submission verification, automation can be implemented in the form of information matching and photo recognition. Criteria can be set for the default actions of that automation or stopped at any time.

Constraints

Below lists some of the constraints that are placed upon the system. These points should be included in any system altering decision.

Security

Regardless of where the data is within the application (stored locally, as part of a verification submission or in transit from a request) it's important that the data may only be read by its intended recipient. Anything less than that is a failure due to the sensitive nature of the data. This will need to be done through a form of cryptography such as public key.

Mobile Operating System Support

As this is an application that will be used by a wide variety of devices it's important that we support a majority of them. Android support will be for version 5.0 and upwards due to the large market share of about 75%. Due to time constraints, there will only be an android version in the initial release and an IOS version can be released at a later date.

Speed

While validation requests are out of the application's hands, it is important that any other requests made to the system are handled in a timely manner. Requests for information are one of the more important examples here. It's important that a user see's little to no delay from clicking a button on another app and receiving a notification on their phone.

Ease of use

This application is handling very sensitive data so it's important that the user knows what they're doing at any point throughout the app. Descriptions and error messages should be concise and address exactly what the user is doing and what they should do next. This will allow for a more enjoyable experience within the application.

Functional Requirements

This section lists the functional requirements in ranked order. It describes what the system must accomplish overall.

ID	1
Description	System must be able to communicate/store data in a secure manner
Criticality	This is a key requirement as the data being communicated is very sensitive
Technical issues	This involves implementing cryptography methodologies relevant to the data and how it's being communicated/stored
Dependencies	N/A

ID	2
Description	System must be able to update appropriate data stores whether local or remote
Criticality	Being able to store the information in the application is second to being able to transmit it securely.
Technical issues	It's imperative to ensure the database used is correctly formatted and accessible. It will conform to at least the 3rd normal form. In addition to this, there will need to be work done to ensure data stored on the mobile device is placed in a logical location along within being secure.
Dependencies	This depends on requirement 1 as data being stored must be secured

ID	3
Description	Backend must be able to communicate with an individual
Criticality	To allow the user to be notified correctly in any way it's important that the system can contact them
Technical issues	Using googles firebase messaging service to contact a device when required will be a must here due to the preference for android on the initial version.
Dependencies	This depends on requirement 1 and 2 as the data transmitted must be secure and data will need to be stored in some way throughout this communication, whether locally or on the backend.

ID	4
Description	Mobile application must be able to store multiple forms of identification
Criticality	While the application should work with one form of identity, it's important that it can allow the user to have multiple forms in case there is extra information that's needed.
Technical issues	Ensuring the data is correctly divided and displayed in a logical order to the user is the only issue here
Dependencies	This depends on requirement 1 and 2 as multiple forms of identity must be stored securely and logically on the device

ID	5
Description	Identifying party must be able to retrieve all applicable submissions
Criticality	To allow for quick and easy retrieval and review of submissions the backend must be able to return submissions applicable to each identifying party.
Technical issues	This can be accomplished by ensuring the data is correctly labelled for the correct party and access control is implemented on any API call.
Dependencies	This depends on requirement 1 and 2 as submissions must be stored/returned securely

ID	6
Description	Identifying party must be able to mark submissions stored as valid/invalid
Criticality	For the overall application to function as intended it's important that identifying parties can say whether information submitted is valid or not.
Technical issues	Ensuring the data is available to the party and calls are in place to edit individual submissions will allow the application to carry this task out.
Dependencies	This depends on requirement 1 and 2 as submissions must be stored/updated in a secure manner

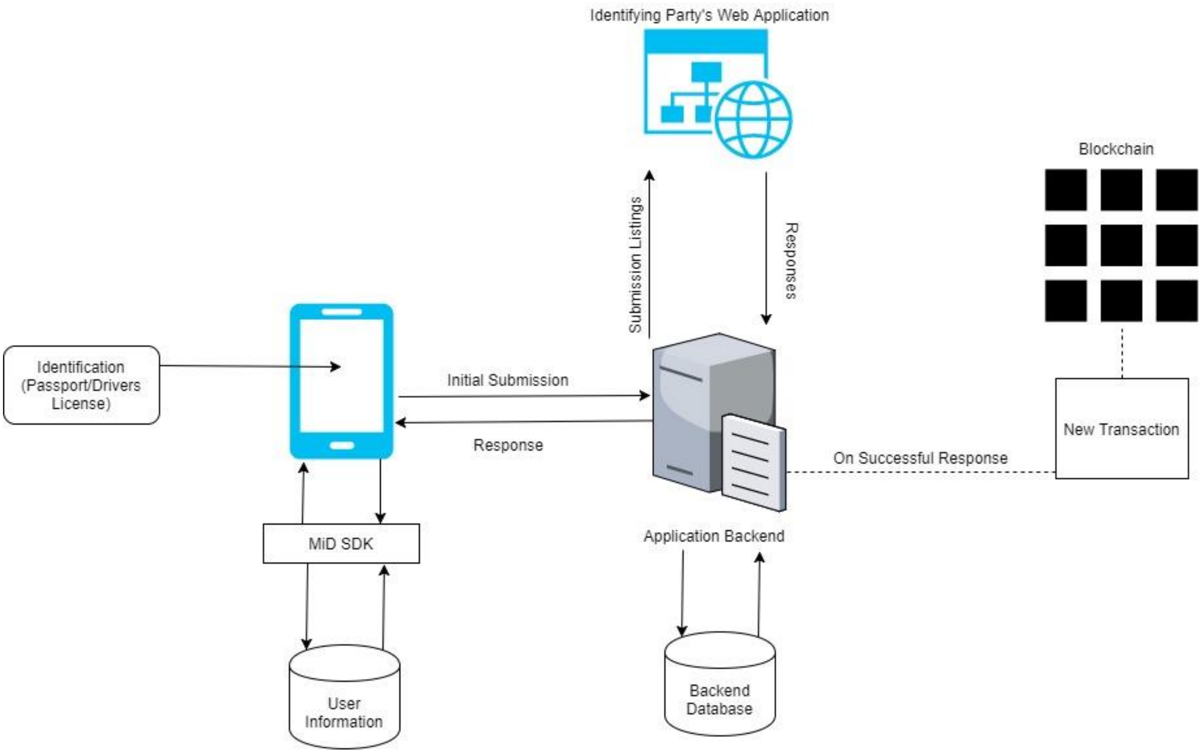
ID	7
Description	Blockchain implementation must have correct models and interfaces in place
Criticality	To track any successful verification transactions, models and interfaces must be in place on the blockchain.
Technical issues	The creation of node interfaces along with the javascript implementation of the models in the hyperledger fabric ledger will allow any transactions to be created, updated and queried.
Dependencies	This depends on requirement 1 and 2. The blockchain itself is public but data transmitted to the blockchain must be secure.

ID	8
Description	Support for a variety of Operating Systems
Criticality	This isn't critical but to allow for a wide adoption of the system it's important that it can run on as many devices.
Technical issues	Ensuring correct code convention is followed in regard to backwards compatibilit.
Dependencies	N/A

High Level Design

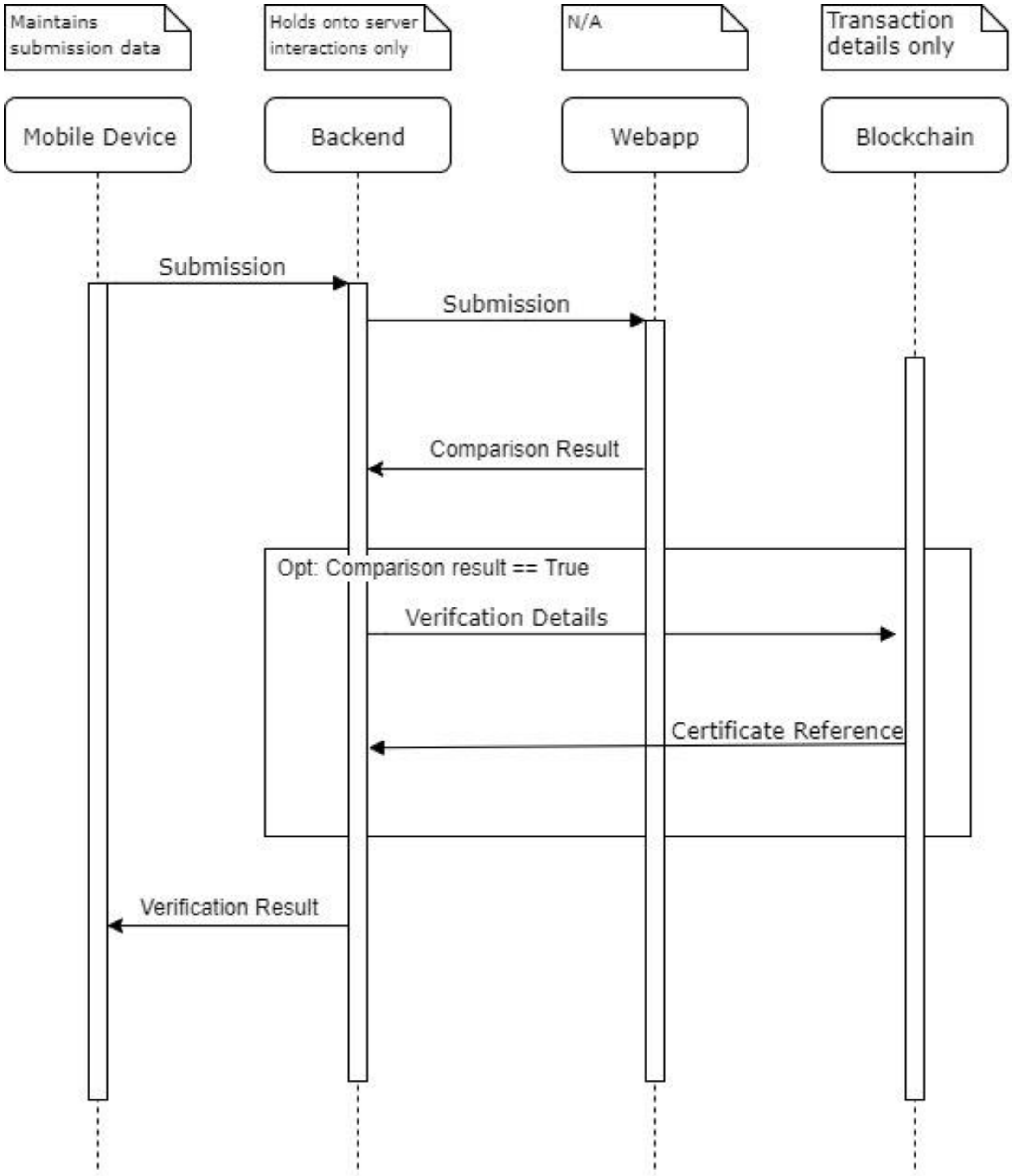
Below lays out a high-level overview of how the system functions. This will include the main system layout and a sequence diagram of the key point of the system, the verification.

Application – High Level Overview



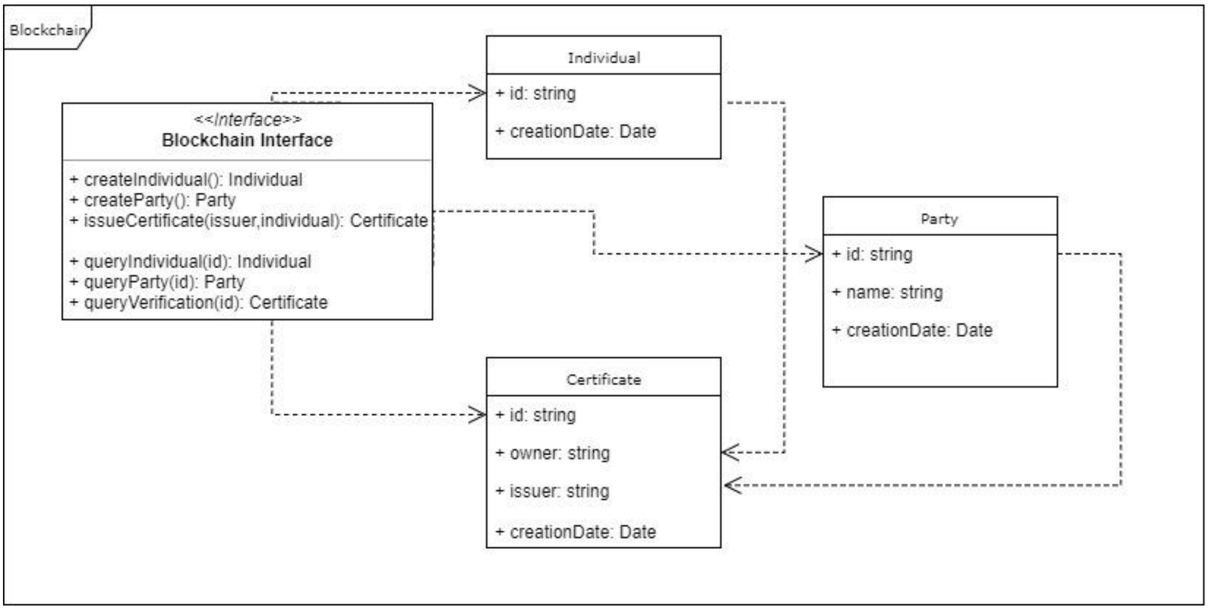
Above draws out how the system will function and how the components of the system will communicate. The mobile device will the main channel through which requests are made and information revolving around any successful request are stored on the blockchain.
Any other relevant information is stored on the application's backend. This will include information that will allow the system to communicate with an individual's device as well as information linking to an individual's currently pending submissions.

Sequence Diagram – Verification Implementation



Above is the sequence in which a verification will be carried out. This is the main form of communication that will occur in the system as a user will need to do this before using any form of identification to verify themselves.
We see that a submissions travels from the mobile device, through the backend, to the identifying party's web application. There, they will make the decision whether to accept or reject the submission. If accepted then the details of that verification are submitted to the blockchain and a certificate of identity is created. Regardless of the certificate creation the user will receive a result of the verification process, whether it's successful or not.

Object Diagram – Blockchain Models



The above diagram illustrates the way in which data will be stored and moved interacted with. The use of an interface will allow for the various interactions to be carried out smoothly. The data itself is replicated across many nodes and access control of the interface will be implemented to allow for legal interactions to take place.

Preliminary Schedule

For the duration of the project I will be following the Scaled Agile Framework (SAFe) approach. This is a form of agile development that I'm familiar with from my time at MasterCard. It will allow me to work quickly and in a structured form, while also providing updates in the form of blog posts and meetings with my supervisor.

To ensure structure my agile approach I'm using trello. The cards in this application will act as the stories I'll be working on. Each story will contain updates and any relevant information that I can point back to or build upon.

Due to my college assignments/work I will not have the full time available to me. I have approximated that I have 25 weeks of work time that I can put towards the system. I will be working on as many tasks in parallel as possible so that I can complete it within the time given. I will also be reducing the number of tasks near the end of the timeline so that I may push forward tasks if there are difficulties/other priorities at the time. Below you will find the Gant chart visualising this.

Project Starting 09/10/17	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15
Requirements Planning	🟩														
Design Analysis	🟨														
Project Planning	🟩														
Project Proposal	🟨														
GUI Prototype		🟩	🟩	🟩											
Functional Specification		🟨	🟨	🟨	🟨										
Mobile Prototype		🟩	🟩	🟩	🟩										
Web App Prototype					🟨	🟨	🟨								
Backend Prototype					🟩	🟩	🟩								
Blockchain Models						🟨	🟨	🟨	🟨						
Blockchain Implementation/Linkage						🟩	🟩	🟩	🟩						
Overall Prototype Build								🟨	🟨	🟨	🟨				
Initial Testing										🟩	🟩				
User Testing										🟨	🟨				
Mobile Build												🟩	🟩		
Web App Build												🟨	🟨		
Backend Build												🟩	🟩	🟩	🟩
Blockchain Build/Linkage												🟨	🟨	🟨	🟨
Overall Build															🟩
Code Review															
Full Implementation Testing															
User Testing															
Testing Writeup															
Documentation	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨
Blogging	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩

Project Starting 09/10/17	Week 16	Week 17	Week 18	Week 19	Week 20	Week 21	Week 22	Week 23	Week 24
Overall Build	🟩	🟩							
Code Review		🟨	🟨						
Full Implementation Testing				🟩	🟩	🟩			
User Testing						🟨	🟨		
Testing Writeup								🟩	🟩
Documentation	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨
Blogging	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩