Specification: PhiQuest Project, Spring 2022

Ethan Wuitschick, Allen Cheng, Arnaldo Barea
Students
Department of Computer Science and Engineering
University of South Florida
Tampa, FL 33620

Email: wuitschicke@usf.edu
allencheng@usf.edu
ajbarea@usf.edu

1.01
April 15th, 2022

History of document

- **Version 1.00** (February 23rd, 2022) – Initial document created
- **Version 1.01** (April 15th, 2022) – Changes made to document according to feedback received from Dr. Christensen to include a table for risk analysis and mitigation, and to update Design trades-offs specifying what is being sacrificed to improve on a different element.

Table of Contents

1. Introduction

As businesses transition into hybrid operations that demand online and physical systems, demand has grown for online solutions to the flow of work. A major competitor in this market, Accusoft has created a no-code solution for businesses to create automated document procedures in their OnTask.io application. OnTask.io has flexible solutions to many use cases, but one case that went unconsidered was an on-demand software for creating a workflow while browsing online. For this, they contracted PhiQuest to develop a Google Chrome extension for generating a workflow from a document found while browsing.

This extension, DocBridge.io, creates a communication bridge between Google Chrome's browser and the OnTask.io systems that direct work to organization members. As a part of the extension's development, a dashboard for running the extension is needed so that Accusoft may easily manage the extension and its users. Additionally, Accusoft wants to explore the development of a system that can secure the authenticity of a document. Private ledgers, called blockchains, are perfect for collectively endorsing a document's originality and authenticity.

PhiQuest has approached this student team for help in building this administration interface for the DocBridge.io extension in Google Chrome and for building a private blockchain to support the onTask.io application. The problems that this project will be addressing are as follows: (PS1): “Can a private blockchain system be implemented to certify transactions and documents signed via the DocBridge.io user interface?” and (PS2): “Can an administration user interface be implemented so that operations personnel can visualize Docbridge.io metrics, perform user administration, and log selective events and transactions?”

2. Glossary

User: Anyone that uses DocBridge.io extension is considered a user.

Administrator: Someone that works for OnTask.io or someone that has access to the private Google Chrome extension project, DocBridge.io, is considered an Administrator.

Webhook: A webhook is a lightweight API that powers one-way data sharing triggered by events. It allows customization within a predefined process.

3. Constraints

a. Document data is needed for entry into the blockchain. This data may only be accessed through the OnTask.io API, which requires the document's ID and a secure authorization token for retrieval. Accessibility is impacted by the acquisition of these two keys.

b. Access to the ID of a document is restricted to the usage of a webhook, which will send data to the Node.js server for processing and commitment into the blockchain. These webhooks are limited by user choice and knowledge. Users must choose to place a

webhook in the OnTask.io workflow and must know the correct method for building the webhooks into the system.

c. As mentioned in 3.a, access to OnTask.io API calls requires a secure authorization key, called the group access token. Current understanding says that this token must be shared with any API call, thus restricting access to these API functions. These tokens may only be copied at the time of creation and may be revoked.

d. This project must interoperate with the DocBridge.io system. The code for this system was built by previous student projects and is not well-documented for shared usage.

e. Administration of user accounts requires access to Accusoft account administration tools and privileged access to their servers. Accessibility and functionality is restricted due to the smaller scope and external nature of the project.

f. OnTask.io and DocBridge.io are used for document signatures and sensitive work operations that may hold private information that cannot be distributed. This will restrict the blockchains functionality, as it must be privately held since the information it holds should not be publicly available.

4. Applicable standards

IEEE 1233-1998
Developing System Requirements Specifications

Our project/software requirements conform to the required properties set out in section 4.2 of the IEEE 1233-1998 guide for development of system requirements (4). These standards were selected for their dual communication purpose; our requirements need to be understood by the project stakeholder team and by our classmates. The stakeholders have a much more intimate knowledge of the technical specifications for the project, but our class peers need to understand the different choices we have made in development so that our solutions are reasonable to that audience.

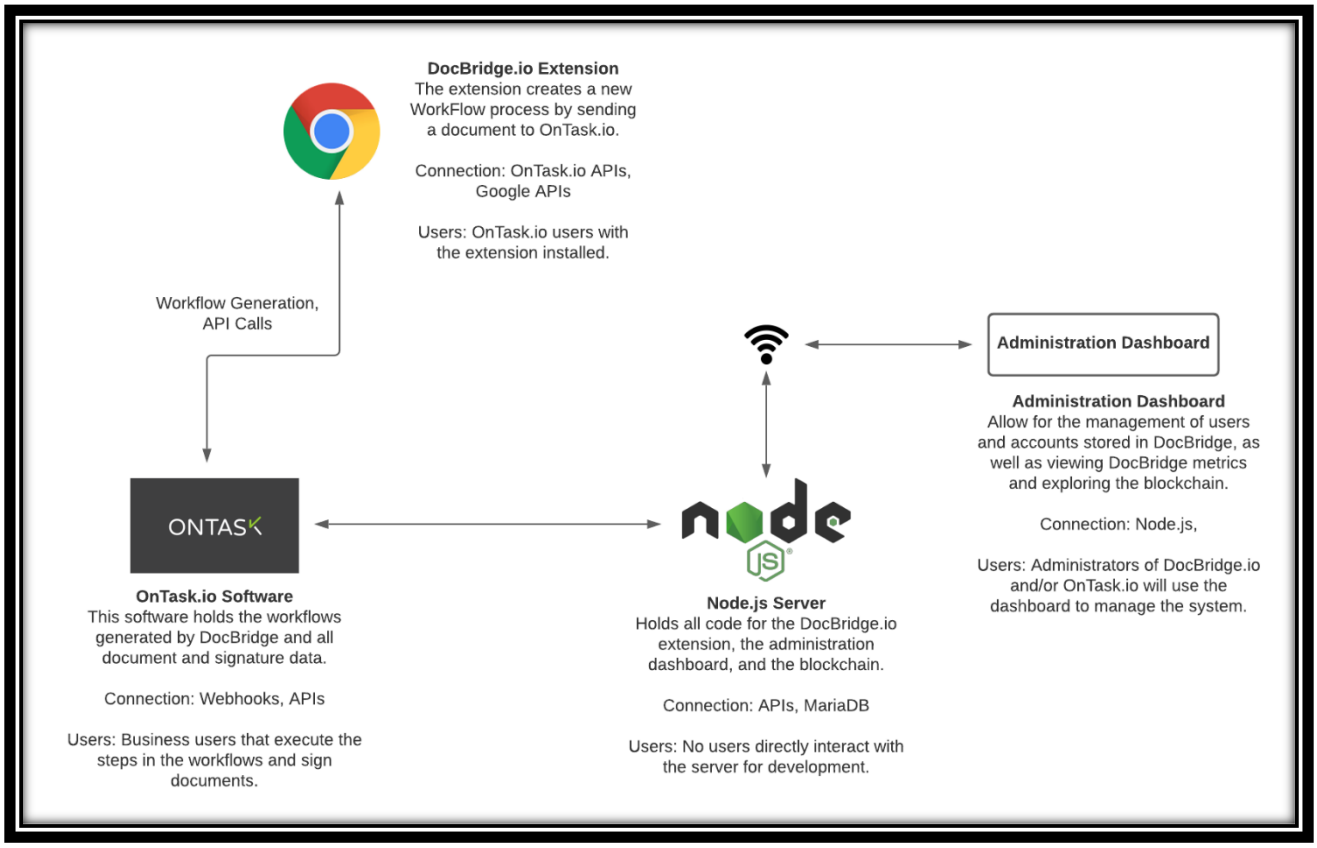
IEEE 730-2014
Standards for Software Quality Assurance Plans

This project follows the agile development methodology, with client meetings occuring twice every week to QA the project software at that stage. Development is also test-driven, with each task being completed once a pre-written set of testing scenarios results in a success. By conforming to these two aspects of software quality assurance, this project conforms to clause 5 of the IEEE 739-2014 standard (5).

As mentioned above, the development of this software is test-driven, with each stage intended to satisfy some set of pre-written tests. Once these tests execute with success, the stage is considered completed. This complies with the IEEE 1008-1987 standard for software unit testing.

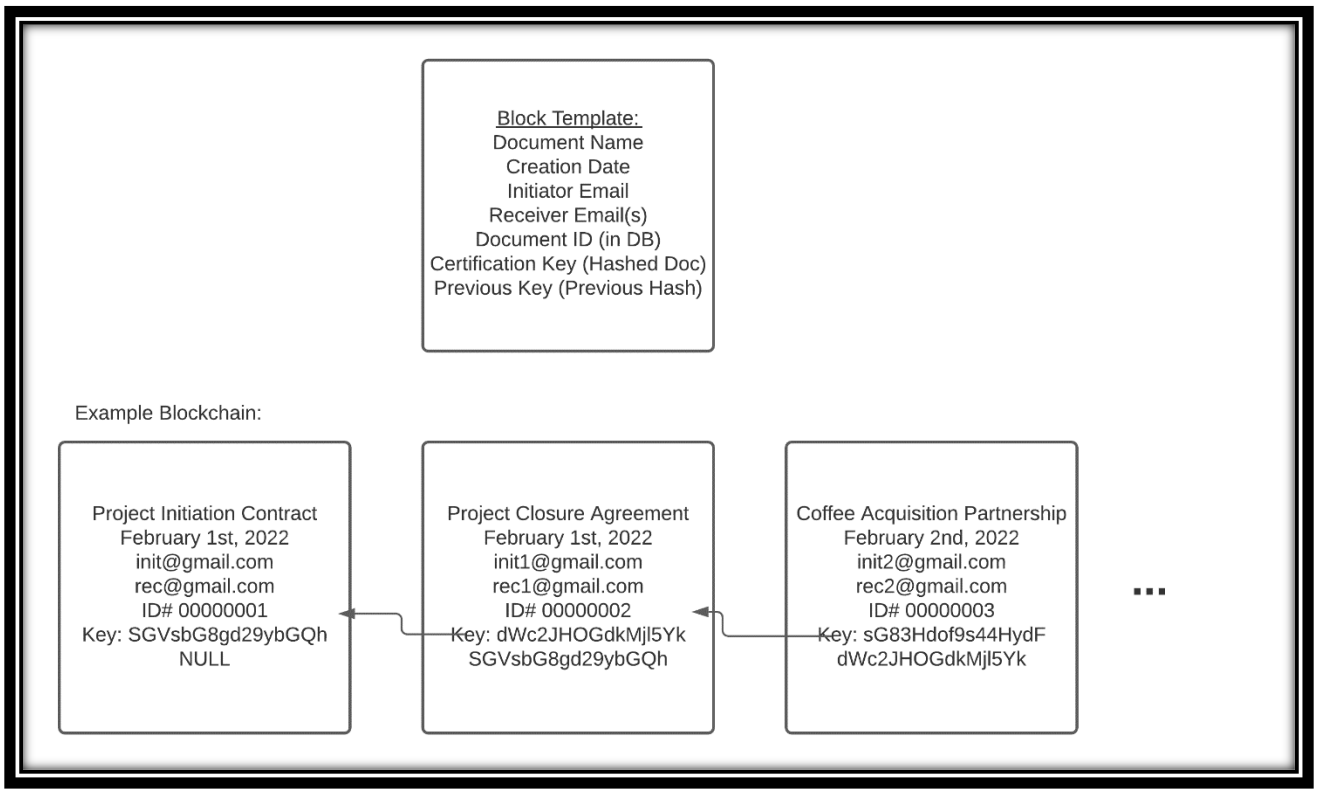
5. Design

5.1 Block Diagram



This block diagram shows a high-level view of the entities of the project and how they are connected. The Docbridge extension is used to initiate a workflow through the calling of Ontask API, Ontask will then process that workflow, and at the end of the workflow a webhook will be linked to a call back to the node.js server, which is where the blockchain, database and the administration dashboard is hosted on. With the payload that the webhook sends the node.js server side will then use the documentID that the payload sends to retrieve an array buffer of the PDF file that was just sent, that array buffer along with other information such as document name, time, sender, receiver will be stored inside the Maria DB database, where the blockchain will also retrieve information from to display the different blocks of the chain. The administration dashboard is hosted on the node.js server that is used for administrators of the Docbridge project to monitor accounts, blocks, transaction logs etc.

5.2 Blockchain Structure

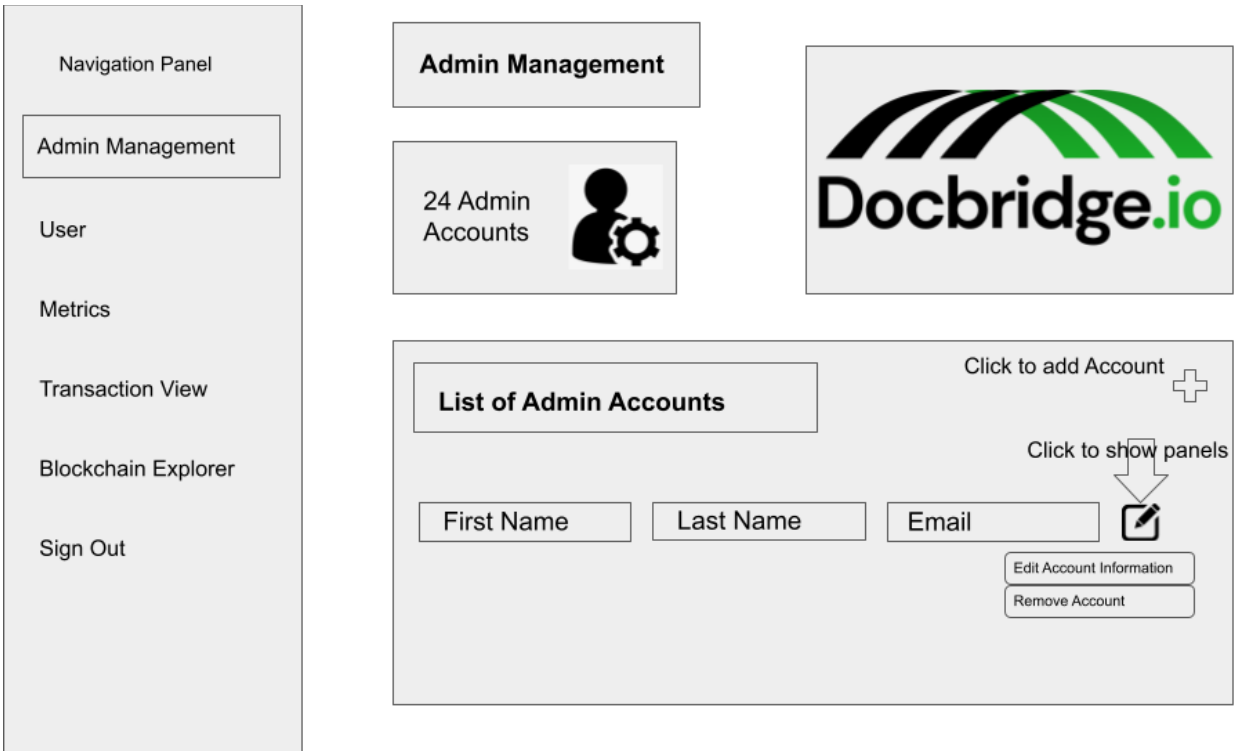


This blockchain structures shows the structure of the blockchain that we will be using, each block will display information on the document Name, creation Data, initiator email, receiver email, document ID, and the generated hash with the array buffer of the PDF.

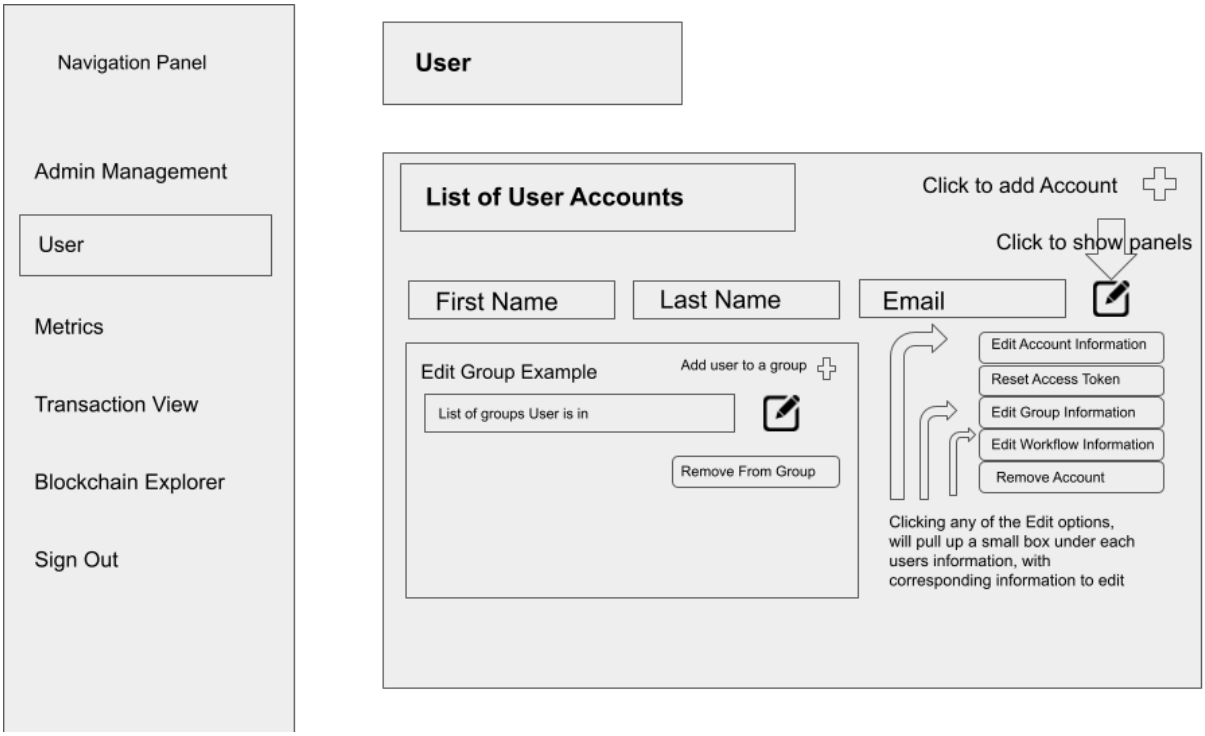
5.3 Administration Dashboard

The administration dashboard is only accessible to admin users of Docbridgeo.io, one can sign into the adminstration dashboard using a google account that is a part of the private google extension “Docbridge”.

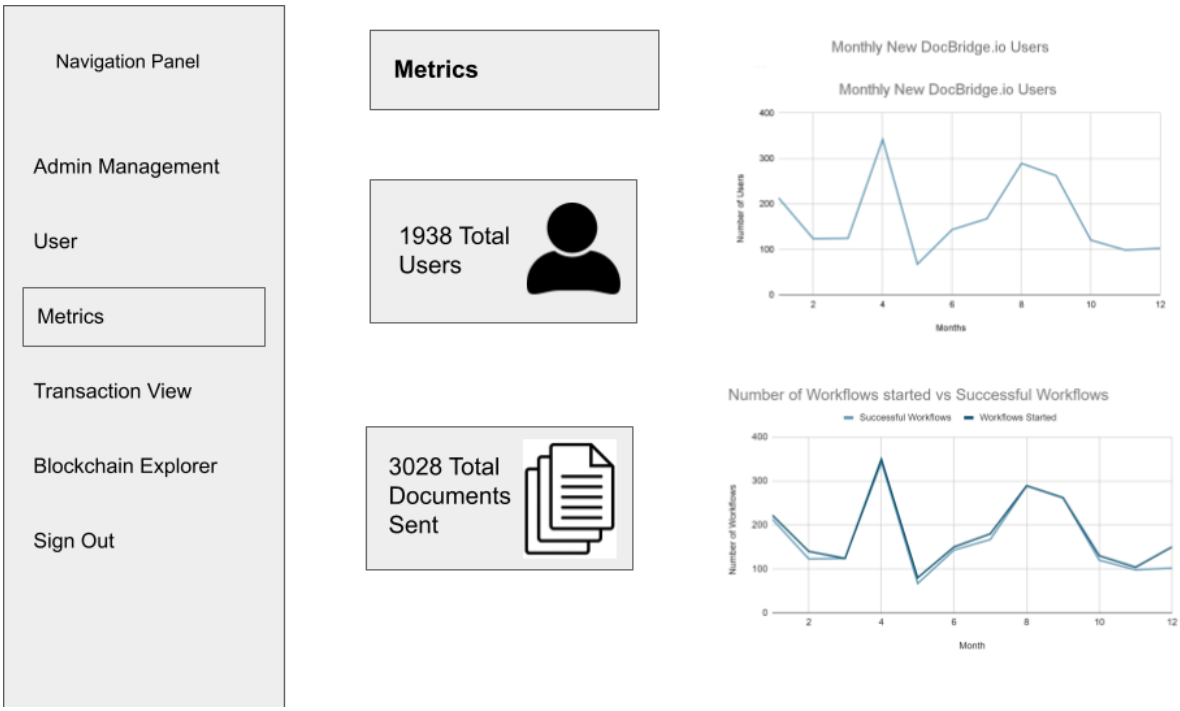
Note: These page wireframes are still a work in process as we work in 2-3 week sprints which has mainly focused on the blockchain side of the project so far, details of the content on each page has not been discussed with the Phiquest Team yet.



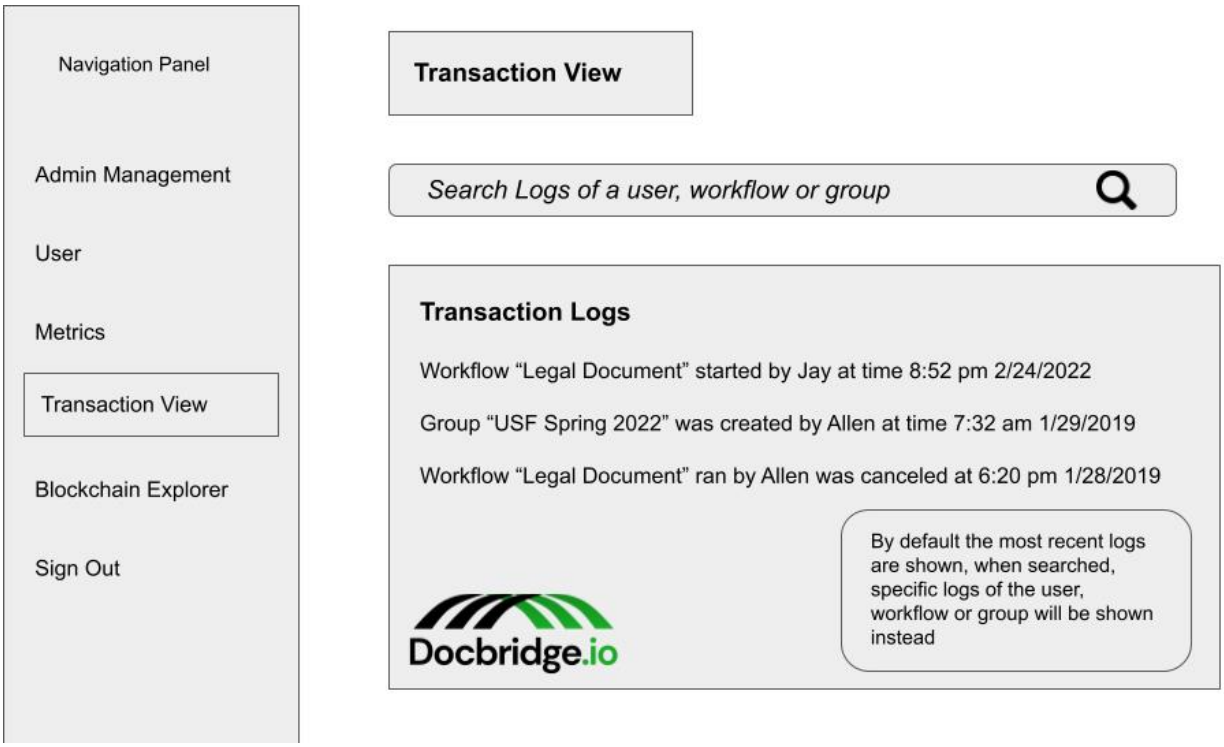
The Admin Management page will show a list of all the admin accounts, where the information of the accounts will be displayed. It will also provide the ability to edit the information of existing accounts. These data are stored into a database currently when one logs into the project through the administration dashboard login page. Interactions on the website such as editing accounts information, adding or deleting an account will be done through accessing the database and changing the information stored in the database.



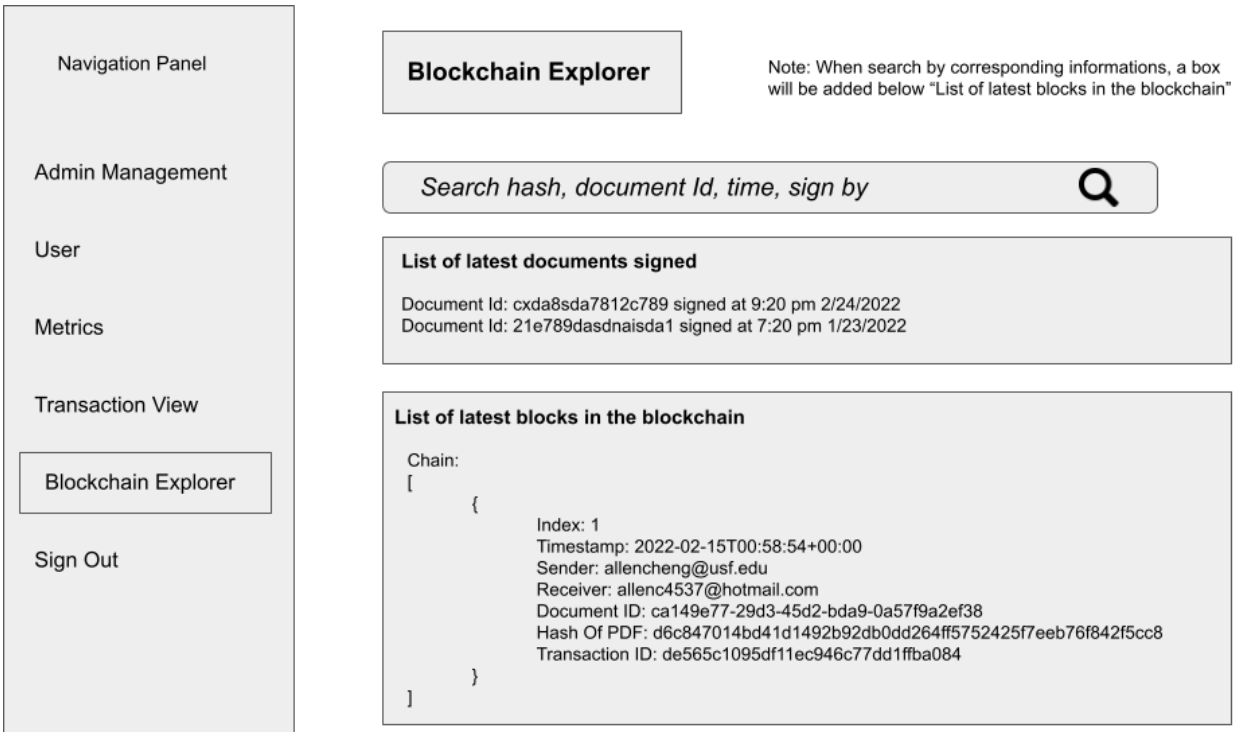
The User page will show a list of all the Docbridge users, where the information of the accounts will be displayed. An admin will have the ability to edit the information related to these accounts such as changing the groups that they are in, the workflows that they have run. Groups and workflows are information related to the accounts from on Tasks side, which are already stored in the database from Docbridge side when workflows or groups are created with Docbridge. Editing this information can be done through just changing data stored in the database. However, API calls to onTask might also be needed to edit this information, but that is something that will need to be discussed with the Phiquest team.



The metrics page will display information about the Docbridge extension, how many Docbridge users there are, how many documents have been sent using Docbridge. When documents are sent using Docbridge, informtion such as the sender, the reciever, the file name, the timestamp are stored inside the database, which is then used for the private blockchain to store information. The databse tables with these information will be read to display the wanted information, a note on the user database table, a timepstamp column will need to be added to that table to include the time the users created the account.



The transaction view page will display recent transaction logs by any user, any workflow or any group. Details on how these will be implemented are lacking as information on these is obtained through the Docbridge side of the project, which we are currently not familiar with this will be implemented in a much later sprint.



The blockchain explorer page displays the most recent documents sent and the latest blocks in the blockchain. An admin can also search by the hash of a pdf, document id, time or sign by to search for specific transactions. This information will be stored in the database every time a document is sent using Docbridge, so this information will be displayed by just searching through the blockchain table of the database to show relevant information.

6. Risk analysis and mitigation

Risk	Mitigation
a. Server issues are a major risk to successful project operations. Testing and implementing the blockchain systems rely on the server being operational to receive the webhook's POST with document access data. After the project is completed, if the Node.js, MariaDB, AWS, Google, or OnTask.io servers experience any downtime, the entire project's functionality is compromised.	The development of the project will continue locally with the use of a localhost port when servers are down to ensure that the project is still progressing even when servers are down.
b. The Docbridge extension requires the user to provide an OnTask-recognized group access token, which is a secure access token necessary for API calls that the extension uses. If this access token is revoked at any time, the API calls will fail, functionality of the Docbridge extension will be broken, which disables the ability to store document information into the private blockchain.	This risk will be mitigated by having tutorial steps when a user downloads the DocBridge.io extension, providing steps for the user on how to retrieve the group access token and reminding them to not revoke the access token.
c. This project shares its codebase with the DocBridge.io extension. Changes in this codebase may affect the operation of any shared functions, which poses a risk to the normal execution of the administration dashboard and to any users using the DocBridge.io extension.	This risk will be mitigated by having the DocBridge.io extension be pushed to production and having their own server that runs a functional version of the code. While the USF team will have a version of the code that runs the extension locally so that changes can be made to the DocBridge.io extension code if needed to implement the administration dashboard.
d. Loss of the point of contact for the project is a major risk. Currently, the team communicates any questions or concerns with Jay Dinicola and Andre Roberts. Should PhiQuest lose these	This is mitigated by having two points of contact; the redundancy of having both available provides extra protection should one become unavailable.

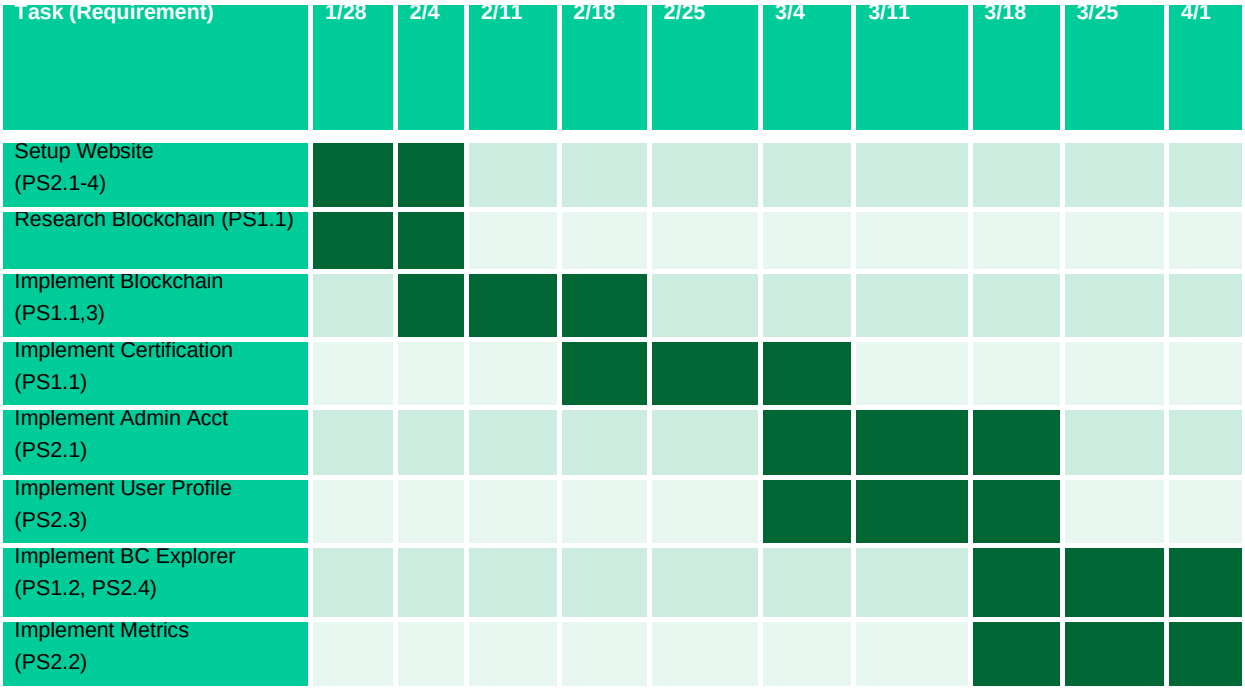
two employees, the project's communication channels would be severely hindered.	

7. Design trade-offs

Security vs reliability/Performance
Extracting PDF metadata: While researching methods to extract a PDF's metadata (this PDF metadata would be what we used to generate the hash for the blockchain), a potential solution was discovered with the use of a npm library function "PDF extraction". The function would take in a PDF file and return with any information related to the PDF file. However, this function required the file to be on the local machine, which was not something we would like to have as part of the process due to security reasons of not wanting to have a document downloaded on the user's machine even if the document were to be deleted immediately after the operation. Along with many other PDF metadata extraction methods that we researched, they all seemed to need the file on the local machine. Since security was considered a very important measure an alternative solution that was found was to generate the hash for the blockchain using the array Buffer of the PDF, which is an array of bytes that makes the PDF itself. While this solution is more secure than the metadata solution where a local download is not needed, the array Buffer is a much longer string compared to the metadata, which results in a slightly higher chance of the hashing system producing an error and the time needed to generate the hash.

Reliability vs Cost
Currently the project has the blockchain itself that stores information about the documents signed, and on top of that there is a database that stores the information stored in the blockchain, which is an example of redundancy to increase the reliability of the project, where if one of the two were to be damaged there is always the other option to backup so that no information is lost. This does, however, increase the cost as a server is needed to host the database.

8. Project plan



9. Traceability to requirements

Design	Corresponding Requirements
5.1 Block Diagram	PS1. 1 The process of starting an Ontask.io workflow with DocBridge.io and then having a webhook respond with essential data needed to generate the hash of the blockchain allows the user to know that the signature is authentic.
5.2 Blockchain Structure	PS1. 1, PS1.3 The blockchain structure shows how the information will be stored inside the private blockchain.
5.3 Administration Dashboard	PS1. 2, PS2.1- 4 The administration dashboard will allow an administrator to browse transactions and related information with a blockchain explorer. The administration dashboard will have backend interfaces for users, administrator accounts, transaction logs and metrics.

10. Traceability to needs and factors

Needs and factors	Corresponding Design
Public safety needs	5.2 The blockchain will ensure that the legal documents sent through DocBridge.io will not be altered, securing critical information.

Environmental factors	5.1 The node.js server that serves as the host of the project is run on the local machine, which minimizes any potential environmental pollution due to power consumption.
Social factors	5.2, 5.3 The private blockchain containing the sending of documents will be monitored authority with the use of the blockchain explorer without any alterations or erasures.
Economic factors	5.1, 5.2, 5.3 The process of retrieving the file after it has been signed and storing it in a private blockchain will ensure that minimal errors occur reducing any economic factors due to system errors.

References

[1] G. Strawn, "BLOCKCHAIN," in IT Professional, vol. 21, no. 1, pp. 91-92, Jan.-Feb. 2019, doi: 10.1109/MITP.2018.2879244.

[2] Cope, J. C. (2002, April 8). What's a Peer-to-Peer (P2P) Network? Computerworld. Retrieved February 2, 2022, from https://www.computerworld.com/article/2588287/networking-peer-to-peer-network.html

[3] Traub, E. (2018). "Learn Blockchain Programming with JavaScript: Build your very own blockchain and decentralized network with JavaScript and node.js". Packt Publishing.

[4] "IEEE Guide for Developing System Requirements Specifications," in IEEE Std 1233, 1998 Edition , vol., no., pp.1-36, 29 Dec. 1998, doi: 10.1109/IEEESTD.1998.88826.

[5] "IEEE Standard for Software Quality Assurance Processes - Redline," in IEEE Std 730-2014 (Revision of IEEE Std 730-2002) - Redline , vol., no., pp.1-231, 13 June 2014.

[6] "IEEE Standard for Software Unit Testing," in ANSI/IEEE Std 1008-1987 , vol., no., pp.1-28, 30 Nov. 1986, doi: 10.1109/IEEESTD.1986.81001.