SIM GLOBAL EDUCATION

UOW AUSTRALIA

School of Computer Science & Software Engineering

Bachelor of Information Technology (Computing)

CSCI321- Project

Technical Manual
12th November 2022

Group: FYP-22-S3-11

Daryl Low Ze Lin	7349026	dzllow001@mymail.sim.edu.sg
Goh En Wei Mervyn	7233292	ewmgoh001@mymail.sim.edu.sg
Low Wei Chern	6656250	wclow002@mymail.sim.edu.sg
Terence Tay Jia Hao	6859136	tjhtay001@mymail.sim.edu.sg
Foo Min Zhan	7058810	mzfoo002@mymail.sim.edu.sg

Supervisor: Sionggo Japit

Assessor: Tian Sion Hui

Document Control

Title: **Project Document**
Document Name: **Technical Manual**

Owner	Current Version	Last Change on		Approved by
		Date	Time	
Min Zhan	1	29th October 2022	1pm	Min Zhan

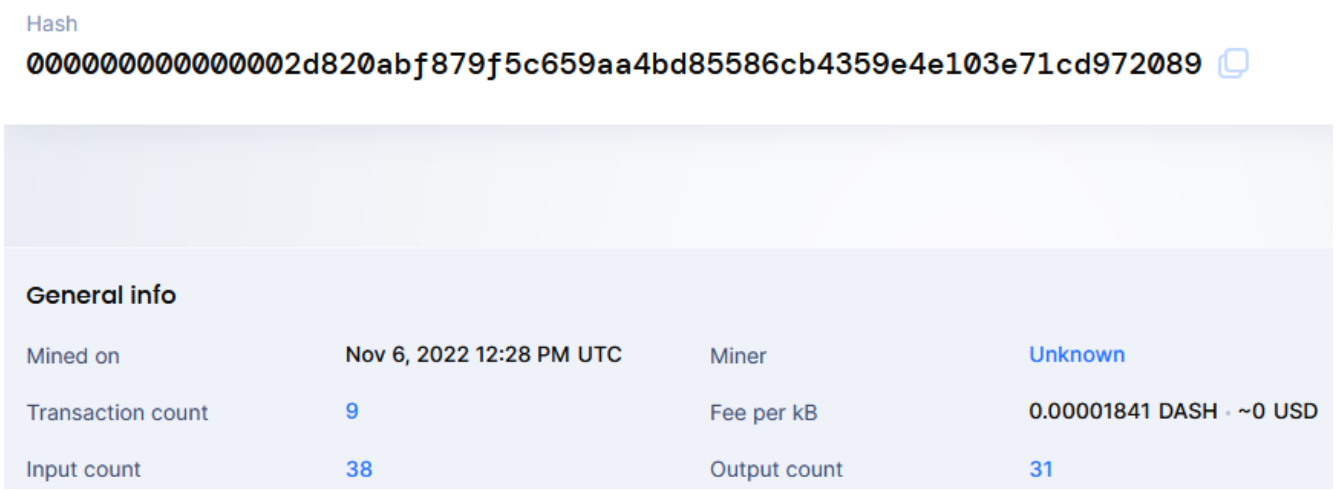
Record of Revision

Revision Date	Description	Section Affected	Changes Made by	Version after Revision
29th October 2022	Continuation of entirety Technical Manual	All	Min Zhan	1.1
3rd November 2022	Development Methodology	Contents	Min Zhan	1.1
4th November 2022	Testing Documentation	Testing	Wei Chern	1.2

Table of Contents

Introduction

Blockchains are Distributed Ledgers, systems that keep records over multiple computers instead of a single computer. Records, transactions are recorded by cryptographic hash functions.



The Goals of this project are to Understanding the Blockchain security mechanisms, Identifying their Cryptographic Libraries and possibly Enhancing the Security of these Blockchain Systems.

Line no.	Security Leak
--	NO
15	YES

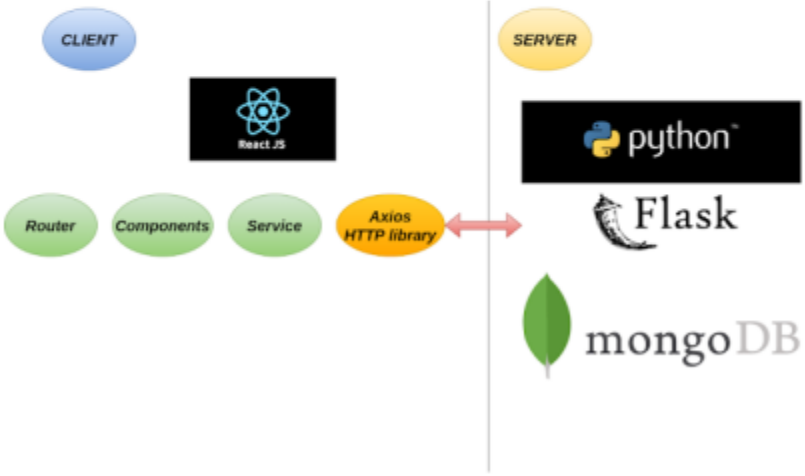
Project Background

With the advancement of the current and upcoming blockchain system emerging in this ever-changing world, it lacks a platform for storing, processing and analysis of blockchains.

The proposed project is built upon the MERN stack, Which is made up of: **MongoDB**, ExpressJS, **React Framework** and **NodeJS**.

In addition, we will be leveraging **Python** Language as the Backend Language, alongside with **Scrapy** – python scraping framework. A background process/scheduler known as Celery. Finally for deployment of the project, we will be using AWS Serverless Architecture.

Technology Stack	
Backend Language	Python
Scrapping Framework	Python - Scrapy
Frontend	React
GUI	HTML - Bootstrap
Background Process/Scheduler	Celery
Deployment	AWS Serverless Architecture



Project Scope

After weeks of researching, we have decided to go on to create a web application which will process files, folders or a Blockchain hash and automatically identify the given hash alongside with its performance and security of that particular cryptographic hash function.

Scripts or applications shall be developed for user interaction to upload files or folders to the web application which will perform analytics.

The main objective is to develop an application tool that automatically detects cryptographic functions in Blockchain systems.

Other scope includes carrying out experiments to compare on the performance of open source cryptographic libraries with conventional enterprise cryptographic libraries.

Stakeholder

Project assessor will be Mr Tian Sion Hui

Project supervisor will be Mr Sionggo Japit

The team consists of 5 members.
Mervyn will be the technical lead in developing the project.
Min Zhan, Daryl, Wei Chern and Terrence will be developers developing the project.

Potential stakeholders could be consumers such as cryptocurrency enthusiasts, security researchers or even developers intending to utilize Blockchain technologies for their future application.

Project Team

Technical Lead - Mervyn Goh

The technical lead has strong experience in full stack development, apart from that he has worked as a Junior Developer at Cardano (ADA) therefore Blockchain and cryptography will not be an issue for him.

Developers

Daryl

Have experience in software development and know about Blockchain technology.

Min Zhan

Have strong experience in software engineering, having worked as a Software Engineer at Keysight Technologies and knows about Blockchain technology and cryptography.

Terence

Have experience in Front-End development and web designing.

Wei Chern

Have experience in Java development.

Project Website

Project Development site

Development website:

https://fyp-22-s3-11.github.io/

Source control website:

https://github.com/FYP-22-S3-11/FYP-22-S3-11.github.io

The development website contains information of the current progress of the project.

The main page of the website will have a brief introduction of the project's objective and the roadmap.

Clicking the "Application" will redirect you to the project allowing you to interact with the application.

Clicking the "Learn more" will redirect you to the project's source control allowing you to view the source codes.

Under the Documentation directory, it contains documents pertaining to this project.

- Literature Review
- Progress Reports
- Requirement Specification
- Technical Manual
- User Manual
- Gantt Chart
- Testing Documentation

Clicking either of the documents will present you with documents regarding each specific information.

Under the About directory, it contains information about the team developing this project.

Project Web Application

Web application: https://stgphys.appsndevs.com/crytopcompare/

The web application allows users to input Blockchains either in their fully qualified Blockchain names or in their currency symbols.

E.g Bitcoin or BTC

Clicking the “Blockchain analysis” button will redirect you to another page which allows you to upload a file or folder in zip, which will analyze and display relevant information such as Visualisation, Security Leak, Dominant functions and other miscellaneous information.

Development Methodology

Given the tight time constraints, compliance of expectations, we have decided to use **SCRUM** as our development methodology.

After the exam break (11th September 2022), we have started a 30-day sprint for the 2 months straight, ending on the 12th November, just in time for our project deliverables.

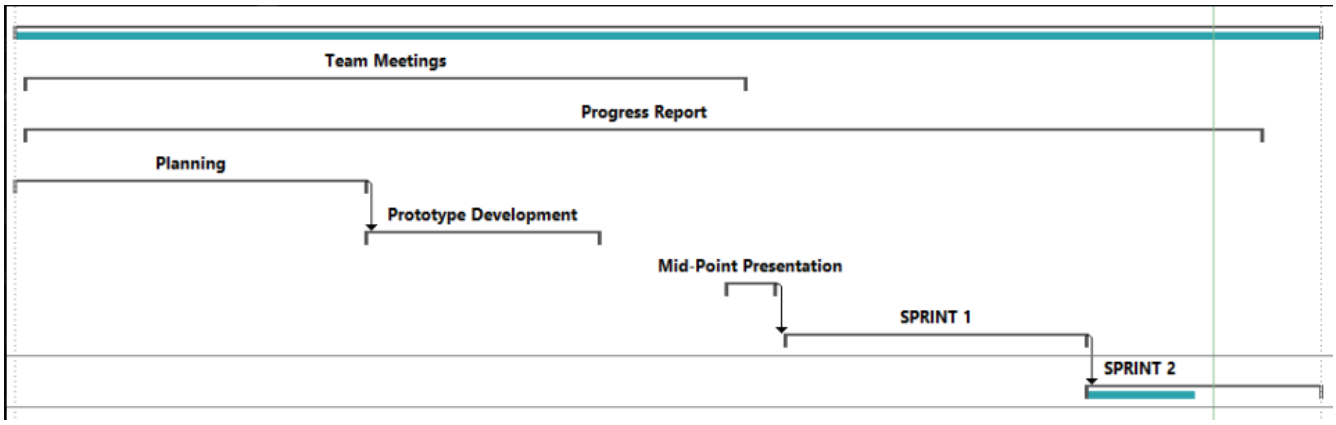
Each Sprint Planning and Sprint Retrospective will fall on every Saturday of the week, from there we will evaluate and decide on our given backlog for the upcoming week and important points that happen during the week.

Each Sprint Review will fall on every Saturday biweekly. From there we will be able to have an overview of how far we have accomplished and possible backlog for future development.

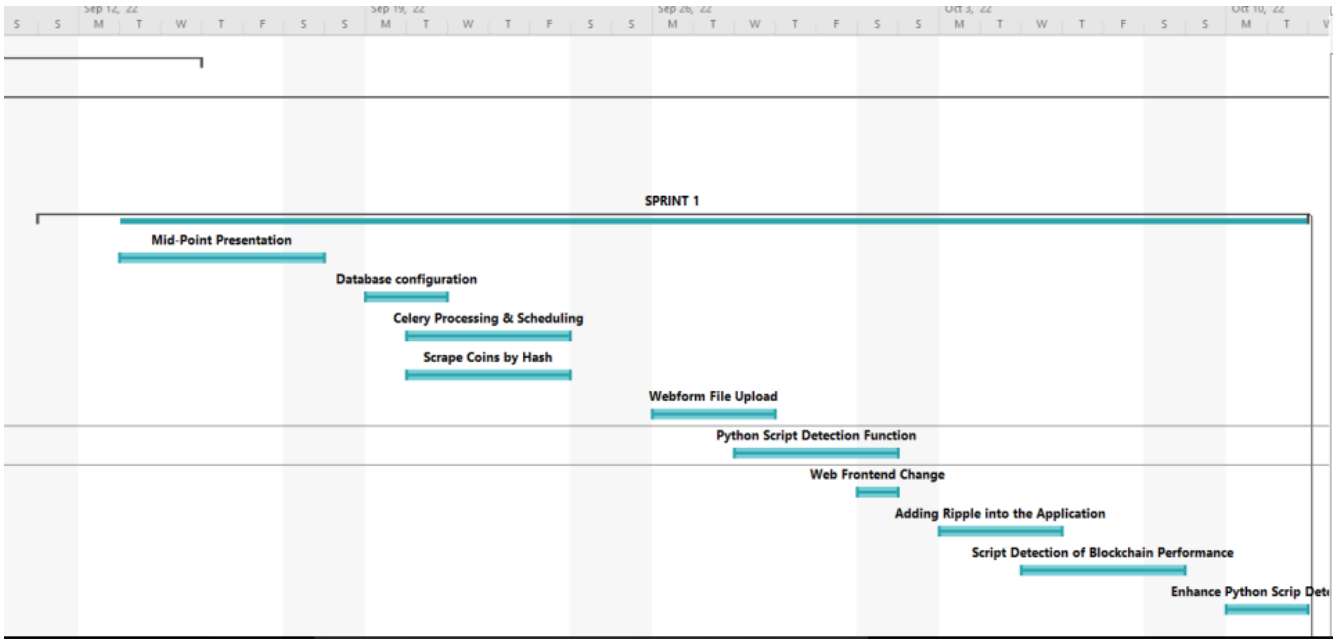
Given our differences in school schedule and work commitments, our team decides to host a weekly standup meeting instead of the traditional daily standup meeting.

During the weekly standup meeting, every single detail will be heavily scrutinized to ensure the software has met the expectations despite severe time constraints.

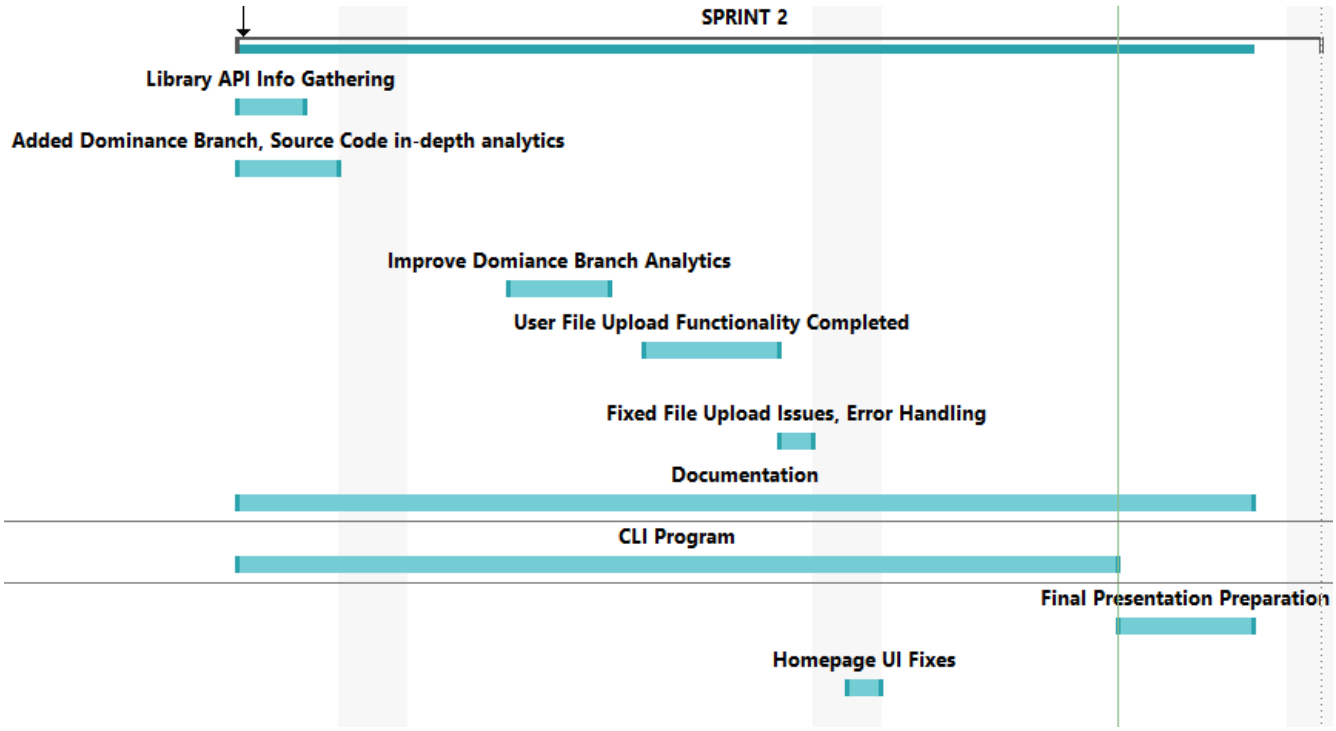
This is the overview of the entire project scope.



On our first sprint of 30 days, the team has a specific task to complete per week, ideally the purpose of the first sprint is to ensure that we have base functionality of our web application up and running.



Moving on towards the second sprint of 30 days, the team is working on improving the web application and fixing any bugs found along the way on our first sprint.



Risk List

Given the tremendous time constraint and huge deliverables, the main risks we faced were the shortage of time and failing to deliver our product as expected. Under such circumstances, the team will be forced to work under immense pressure to meet the deliverables.

Apart from that, we also identified risks such as shortage of manpower, as some of us are working while some of us have school and other commitments.

Another potential identified risk will be the limitation of knowledge. Given that Blockchain is constantly evolving, it is challenging to keep this project to fit everyone’s needs.

Further risk identified is the capability of our programming skills and knowledge on different native cryptographic libraries as well as well-known open source cryptographic libraries.

Creating 2 different applications; Web application and static binary, that supports several operating systems also proved itself to be another challenge.

Test Plan

Test Environment

As the system is developed in a web interface, the testing will be done on browsers over PCs and Notebooks. The web application testing will be done on Chrome for Windows. As for the static binary application, it will be tested on Windows, Linux.

Overview

There are **3 components** that need to be tested for their functionalities. The Webapp has 2 components, the **Compare Crypto** page and the **Blockchain Analysis** page. These 2 components will undergo the same set of conditions of testing.

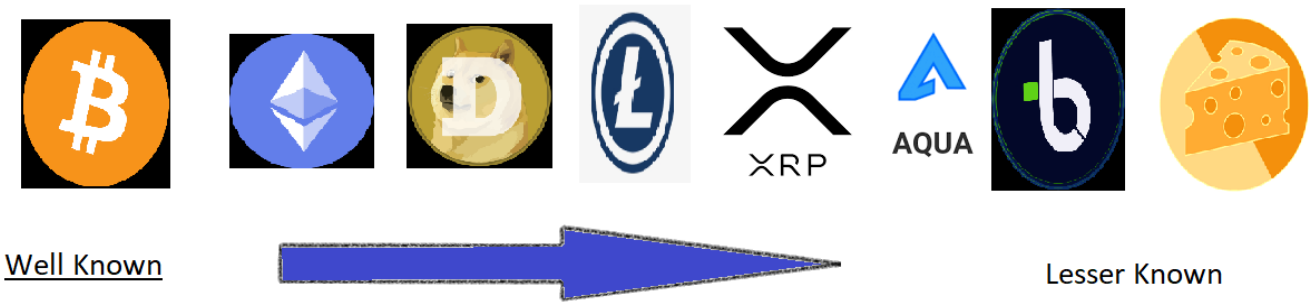
The **Command Line Interface** is the third component and this will undergo its own set of conditions of testing. Overall, 2 sets of tests will be carried out on these 3 components.

Objectives

The objective of the tests run on the Webapp is to observe if the Comparison and Analysis can be performed on a wide array of Cryptocurrency Blockchains.

The objective of the tests run on the Command Line Interface is to determine if the program can accept proper inputs from the user and perform **quick** analysis on named Blockchains and identifying hashes.

Approach

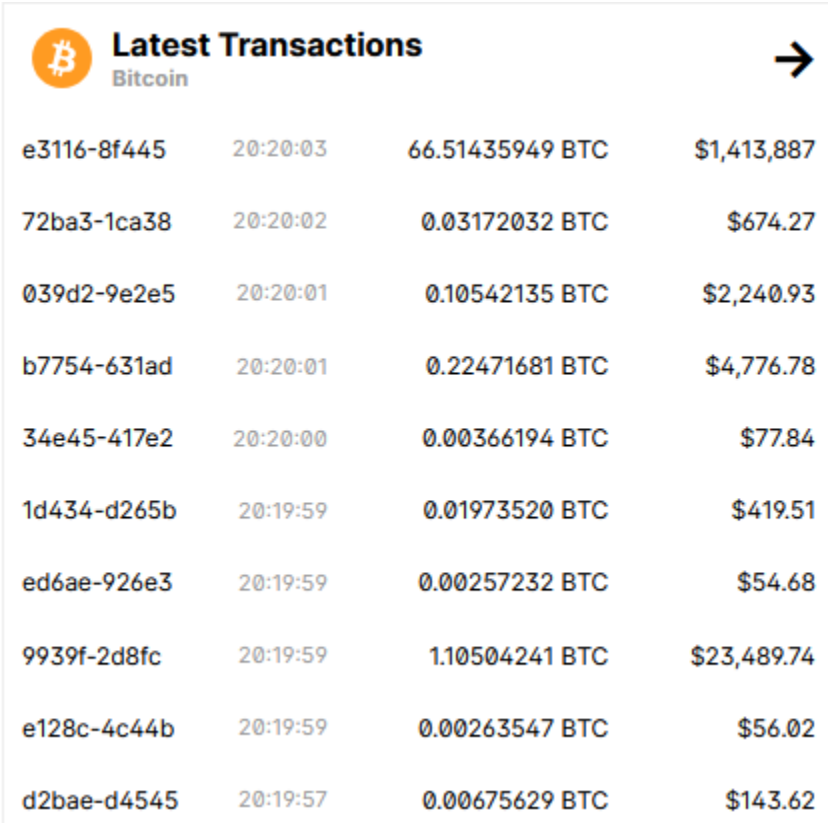


The **Compare Crypto** and **Blockchain Analysis** components of the Web app will be tested on this selection of Cryptocurrency Blockchains. The chosen cryptocurrencies range from being very well known to the obscure and unknown.

All chosen Cryptocurrencies **must** also have their open source code **available** for download and to be used in the Blockchain Analysis component.

The lesser known cryptocurrencies were selected from the **autofill** suggestions on the Compare Crypto Webapp, with inputs of **A, B and C.**

The Command Line Interface program will be tested on sites like blockchain.com/explorer.



Latest Transactions
Bitcoin

e3116-8f445	20:20:03	66.51435949 BTC	$1,413,887
72ba3-1ca38	20:20:02	0.03172032 BTC	$674.27
039d2-9e2e5	20:20:01	0.10542135 BTC	$2,240.93
b7754-631ad	20:20:01	0.22471681 BTC	$4,776.78
34e45-417e2	20:20:00	0.00366194 BTC	$77.84
1d434-d265b	20:19:59	0.01973520 BTC	$419.51
ed6ae-926e3	20:19:59	0.00257232 BTC	$54.68
9939f-2d8fc	20:19:59	1.10504241 BTC	$23,489.74
e128c-4c44b	20:19:59	0.00263547 BTC	$56.02
d2bae-d4545	20:19:57	0.00675629 BTC	$143.62

Here, the tests will involve grabbing the hashes from the latest transactions of these cryptocurrencies.
The program will then be used to determine if the hashes grabbed by the tester is a **Wallet Hash Address or a Hash**.
If it is a Hash, then the program should display the hash function involved or the list of **possible hash functions** in the given hash.

The program will also undergo a separate test on its function to identify the hash of Named Cryptocurrency Blockchain, based on information on online libraries.

(**Refer to the Testing Documentation Folder for the Test Reports and Test Cases**)

Test Report Summaries

Test Report 1

Testing was carried out on the Compare Crypto page. Tests were carried out smoothly, Test Cycle 1 was **a success**.
XRP failed to show the hash function used, this is summed up to it being removed multiple times from the cryptocurrency exchanges due to the United States Securities Exchange Commission's probes into whether it should be considered a security and its regulation.

Test Report 2

Testing was carried out on the newly finished Blockchain Analysis Page. Tests were abruptly halted after just trying 2 test cases of the bitcoin and litecoin source files.
The Webapp's function is still filled with bugs, it fails to show any analysis results. Test Cycle 2 was considered **a failure**.

Test Report 3

Testing was carried out on the Command Line Interface Program.
bitcoin, **ethereum**, **cardano**, **polkadot**, **litecoin** and **dogecoin** were chosen for the test, due to these Cryptocurrency Blockchains having proper documented transactions on display and available for viewing with the hash blocks and wallet addresses.
These Cryptocurrency Blockchains are also selected for Test Cycle 4.
The command Line Interface only struggled with identifying polkadot, which is not identified by the django Rest framework.
Test Cycle 3 **was a success**.

Test Report 4

Testing was carried out on the Command Line Interface Program with the different Hash Blocks and Hash Addresses from the selected Cryptocurrency Blockchains chosen in Test Cycle 3.
The ethereum and polkadot cryptocurrency blockchains posed a small problem for the Program.
Due to the number of Cryptocurrency Blockchains **modeled after Ethereum**, various blockchains also have their hash blocks starting with **0x** and having a **66 length hash value**. Keccak-256 and polkadot's Blake-2b hashing algorithms were unable to be properly differentiated due to both having this aforementioned characteristic.
Overall, Test Cycle 4 **was a success**.
however, if given more time, some revisions are considered to try to identify the ethereum hash block or wallet address properly, due to its popularity as a Cryptocurrnecy

Test Report 5

Testing was once again carried out on the Blockchain Analysis Page.
The tests were carried out successfully, the **bugs have been fixed** by the lead programmer, Mervyn. Performance of the Blockchain Analysis feature was massively improved after several rounds of tweaks.

The first version of our product took **20 minutes** for the bitcoin source file analysis. After all the improvements made before this round of testing, the time our product requires now has been reduced to **5 minutes**.
Resounding improvements have been made and observed.
Overall Test Cycle 5 **was a success**.
Bugs have been fixed and the product now works as intended.

Requirement Specification

User Stories

#1, As a User, I want to view the Hash function used on a Cryptocurrency Blockchain.

Name: View Hash Function used on a Cryptocurrency Blockchain
Stakeholder and goals: User, obtain information on the Hash Function used on a Blockchain
Description: The user enters the name of the Cryptocurrency Blockchain and the Product displays the main Hash Function used on it to the User
Actor: User
Trigger: User enters Cryptocurrency Blockchain into our Product CLI/GUI/API/WebApp
Normal Flow: 1. User enters the Cryptocurrency Blockchain into the Product 2. Product retrieves the main hash function used 3. Product displays the main hash function used to the user 4. User reads the information displayed to them
Sub-flows: None
Alternative flows: 1. User enters a wrong Cryptocurrency Blockchain 2. Product informs User of a wrong input 3. Product shows that no information was found

Requirement Definition

User Stories (Cont.)

#2, As a User, I want to view the Hash Function used from a given Block Hash

Name: Identify Hash Function on a Block Hash as a User
Stakeholder and goals: User, Identify Possible Hash Functioned used
Description: The user enters a block hash into the product. Product analyses it and displays the possible Hash Function used
Actor: User
Trigger: User selects view the Hash Function from the CLI/GUI/API/WebApp
Normal Flow: 1. User enters a block hash into the product 2. Product performs analysis 3. Product displays possible hash functions used based on input 4. User reads the information displayed by the product
Sub-flows: None
Alternative flows: 1. User enters a block hash file that is not identifiable 2. System fails to perform analysis 3. System shows that no Hash Functions can be identified

Requirement Definition

User Stories (Cont.)

#3, As a User, I want to be able to perform analysis on a Cryptocurrency Blockchain source file for dominant cryptographic functions.

Name: Conduct Analysis on Cryptocurrency Blockchain source files
Stakeholder and goals: User, Analysis of Cryptocurrency Blockchain source file
Description: The user uploads a source file onto the program, program performs analysis on this source file
Actor: User
Trigger: User selects Source File analysis from the Product, User uploads a file onto the product
Normal Flow: 1. User enters source file onto the Product 2. Product accepts the file 3. Product performs analysis on the file given 4. Product outputs the analysis to be displayed to the user 5. User views the analysis done by the program and view the dominant cryptographic function
Sub-flows: None
Alternative flows: 1. User uploads an invalid file (not .zip file) 2. Product accepts the file 3. Product detects that an invalid file type was uploaded 4. Product displays error message to the user

User Stories (Cont.)

#4, As a User, I want to be able to compare the performance between different Cryptographic Functions on different Cryptocurrency Blockchains.

Name: Performance Comparison between different Cryptographic Functions on different Cryptocurrency Blockchains
Stakeholder and goals: User, Performance Analysis comparison between different cryptographic functions
Description: The user selects the Compare Crypto feature on our Product
Actor: User
Trigger: User selects the Compare Crypto feature on our Product .
Normal Flow: 1. User selects the Compare Crypto Feature 2. User selects 2 different Cryptocurrency Blockchains to compare 3. Product perform analysis 4. Product displays comparison to the user
Sub-flows: None
Alternative flows: 1. User selects the Compare Crypto feature 2. User makes an invalid selection 3. Product registers the invalid selection 4. Product informs user of the invalid selection

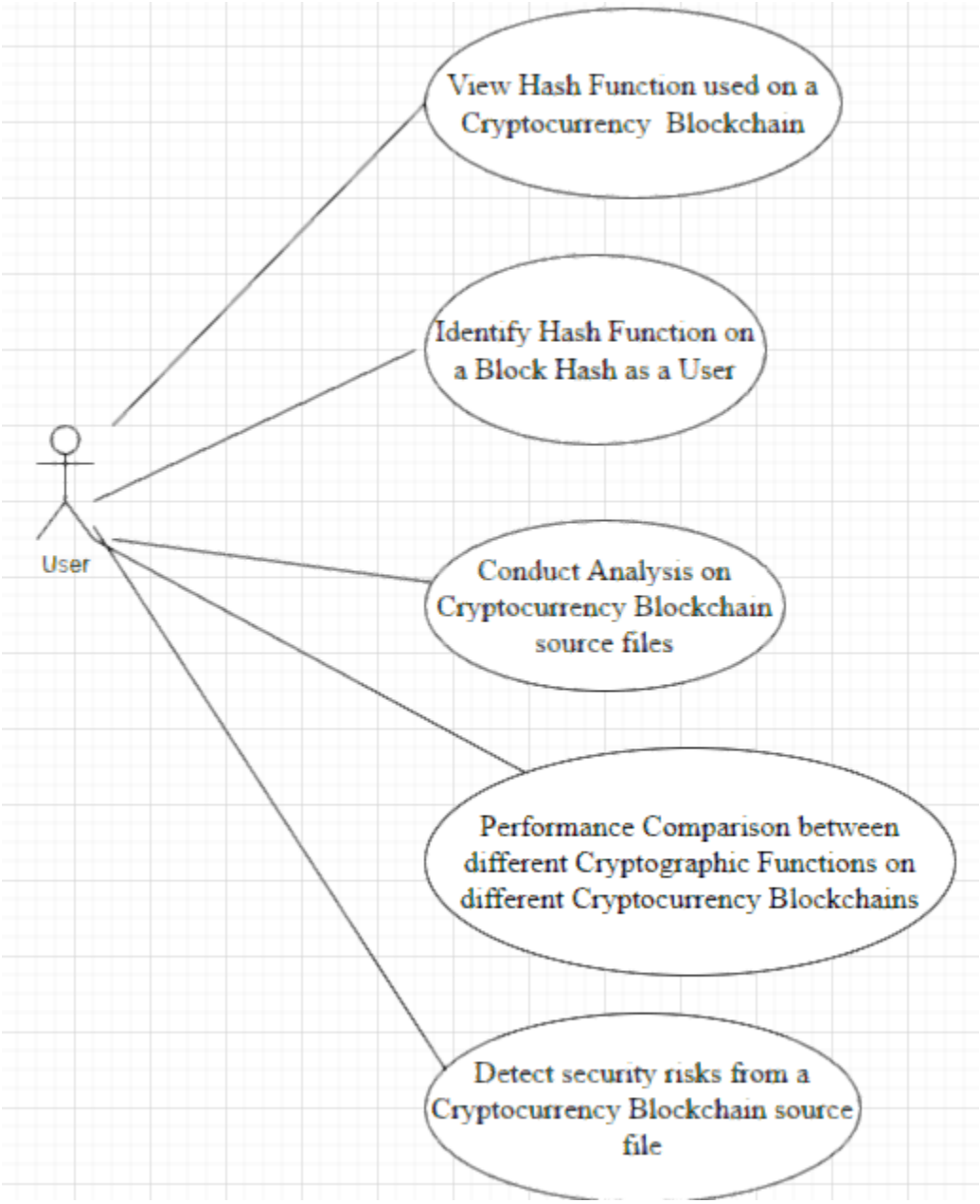
User Stories (Cont.)

#5, As a User, I want to be able to see if there are any security risks from a Cryptocurrency Blockchain source file

Name: Detect security risks from a Cryptocurrency Blockchain source file
Stakeholder and goals: User, Detect any security risks or weakness
Description: The user uploads a source file onto the program, program performs analysis on this source file
Actor: User
Trigger: User selects Source File analysis from the Product, User uploads a file onto the product
Normal Flow: 1. User enters source file onto the Product 2. Product accepts the file 3. Product performs analysis on the file given 4. Product outputs the analysis to be displayed to the user 5. User views the analysis done by the program and view the dominant cryptographic function
Sub-flows: None
Alternative flows: 1. User uploads an invalid file (not .zip file) 2. Product accepts the file 3. Product detects that an invalid file type was uploaded 4. Product displays error message to the user

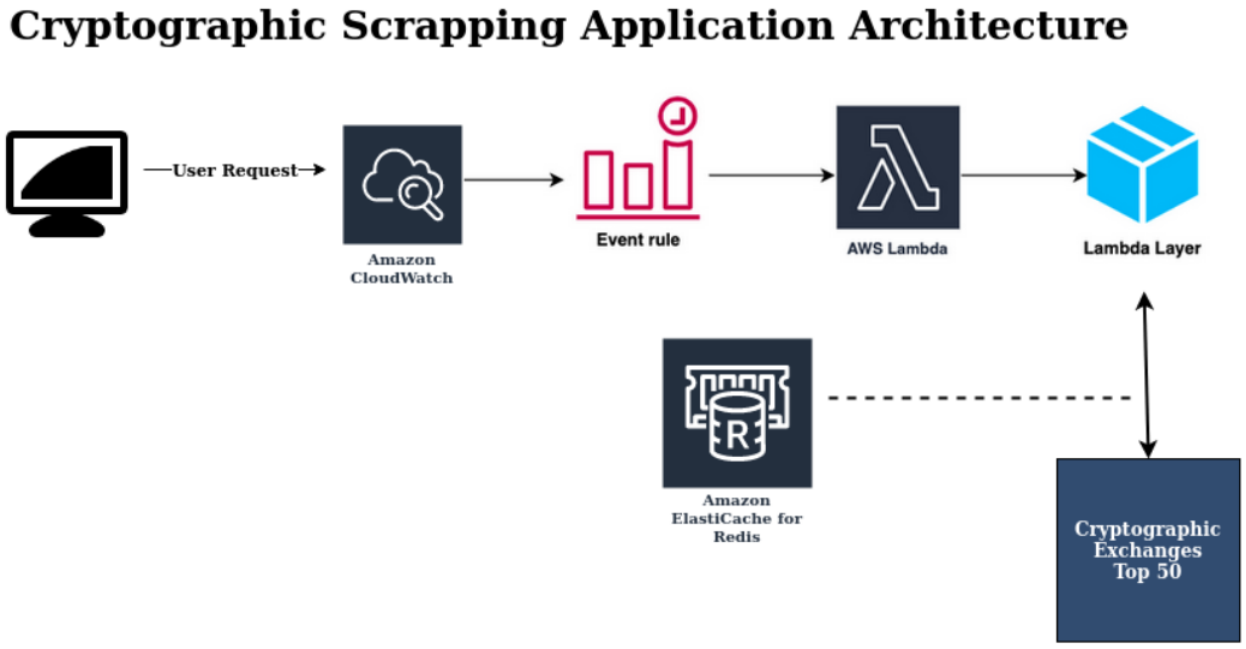
Requirement Definition

Use Case Diagram

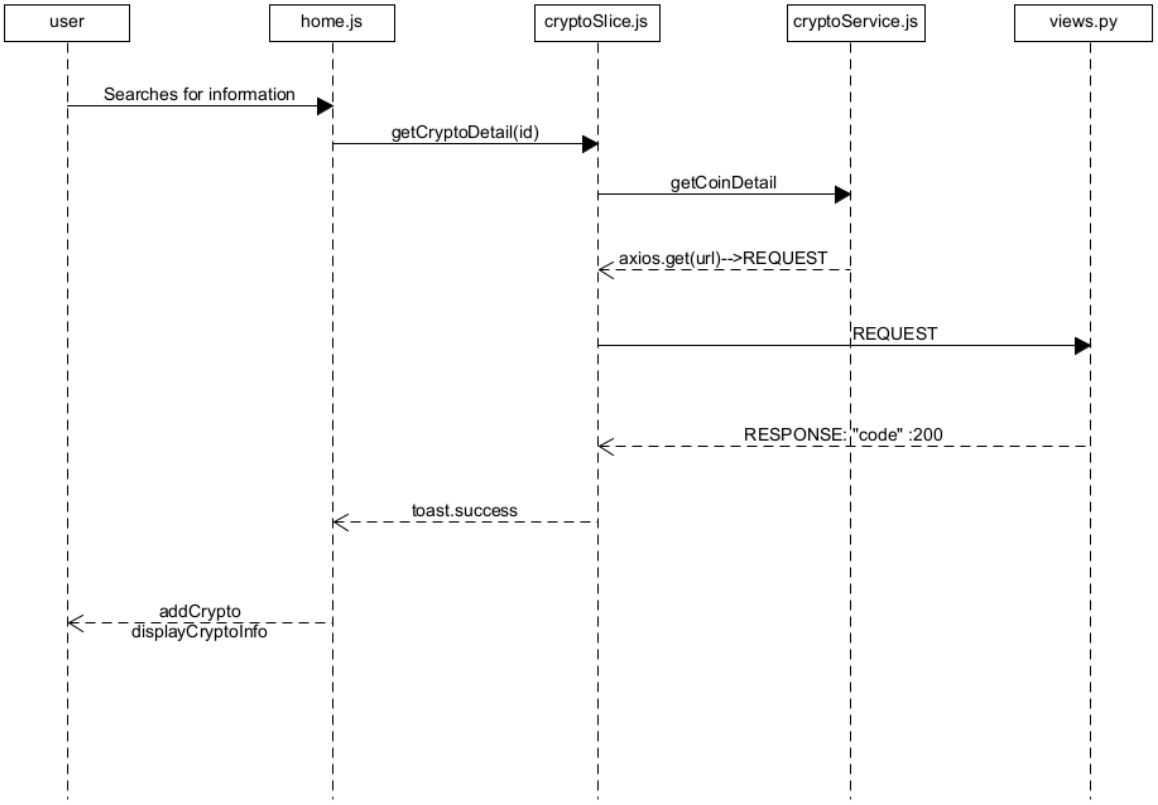


Design Specification

Webapp (Compare Crypto) Application Architecture



Webapp (Compare Crypto) Sequence Diagram



home.js contains the Javascript program that displays the tables and the autofill help function for the users typing in the search field.
When the Compare Now button is clicked with the corresponding Cryptocurrency to search for, home.js sends the id, the Cryptocurrency name in the search field, to **cryptoSlice.js**.
cryptoSlice.js then gets a **REQUEST** from **cryptoService.js** with the getCoinDetail function. Once a **REQUEST** has been received, the site then does an **API Call** on the **views.py** file

views.py is the file that contains all the web scrapping programming, all the BeautifuSoup Python programming that scraps **coinlore** and **blockchain.com** for information regarding the input cryptocurrency name, fetches then and returns them to cryptoSlice.js

Upon successful return, cryptoSlice.js does toast.success, a React JS notification, to the main page, home.js

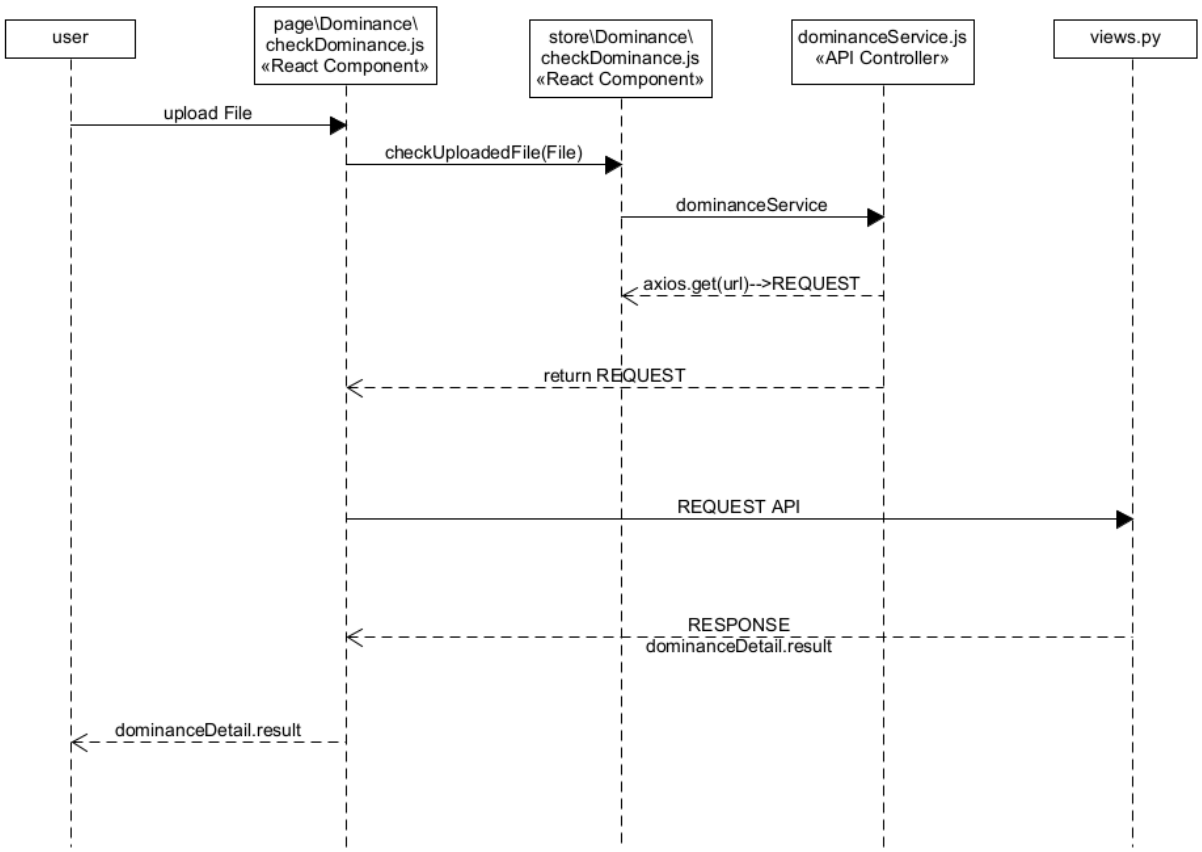
home.js, upon successfully receiving the toast.success, then notifies the user that the process was successful, cryptocurrency successfully added and displays another row in the table with the information scrapped by the views.py file.

```
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
Scheduler: Sending due task coin_15_sec (api.tasks.get_list_coin_celery_demo)
```

Nested within the webpage, is the Celery Task Scheduler. These processes update the requests and continuously update the information displayed on the page.

These update the Price, Market Cap and Features Fields on the table displayed on the Compare Crypto Page.

Webapp (Blockchain Analysis) Sequence Diagram



The Blockchain Analysis page of the Product is made of 2 files, both aptly named checkDominance.js.

page\Dominance\checkDominance.js is the file that is displaying the webpage of the Blockchain Analysis web page.

Dominance\views.py is a different file from the views.py used on the Compare Crypto page.

For the Blockchain analysis, upon receiving an API Call, Dominance\views.py carries out all its functions. First it checks if the file type is of type .zip or .js, if the file is of wrong type, it immediately notifies **store\Dominance\checkDominance.js**, before this page notifies **page\Dominance\checkDominance.js** to notify that user that the file is of wrong type.

If the file is of the correct type, all of the views.py functions will be carried out, mainly the examination of all the source files uploaded by the user.
views.py scans through all the source files, packages all the analysis done and **stores it in a response** to be sent back to the main page, displaying all the analysis done to the user.

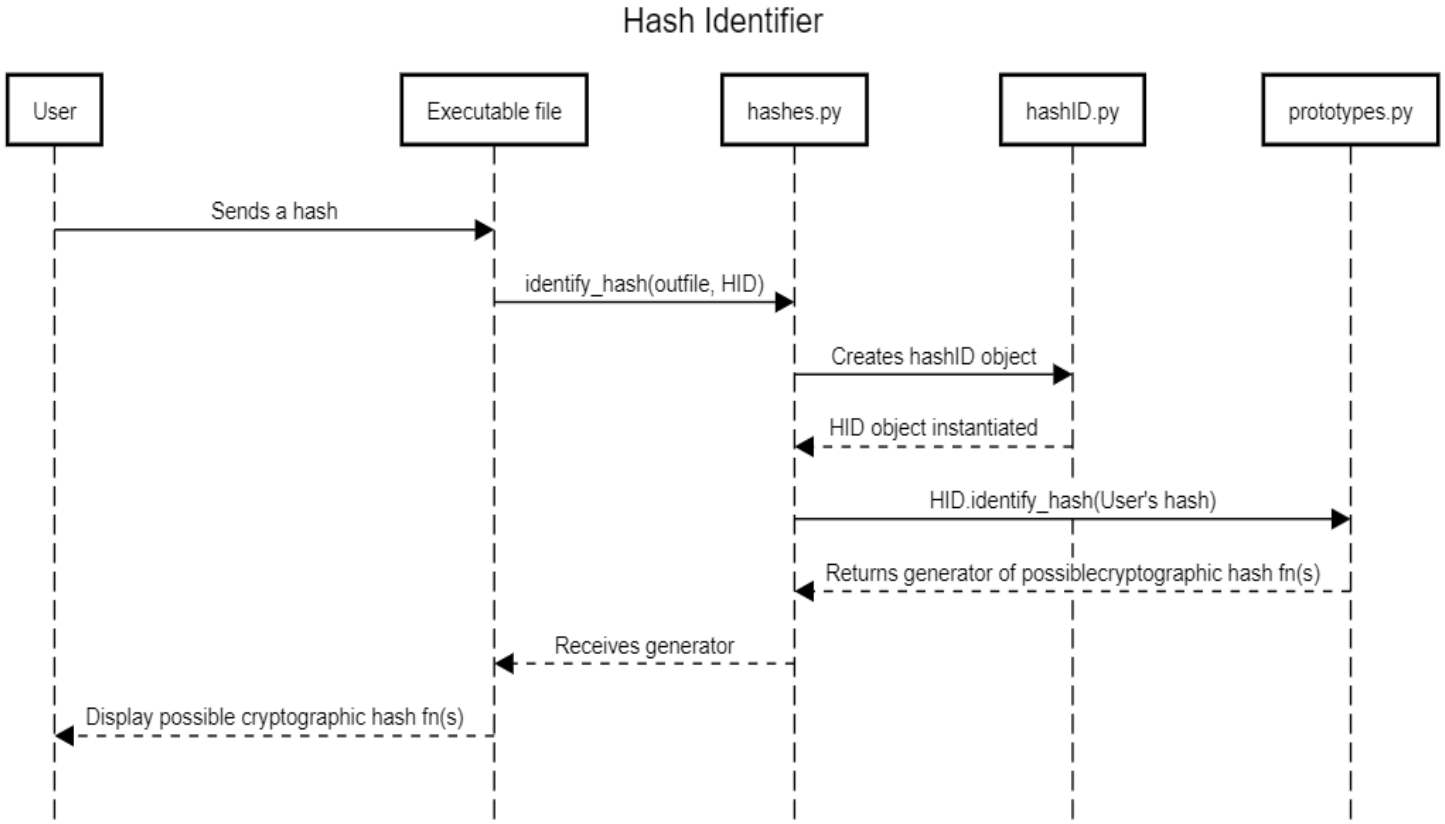
```
def isJsFile(filename):
    if '.js' in filename:
        return True
    elif '.py' in filename:
        return True
    else:
        return False

def isZippedFile(filename):
    if '.zip' in filename:
        return True
    else:
        return False

def optimize_zip(filename,folder):
    response = {}
    with zipfile.ZipFile(folder+filename, 'r') as zip_ref:
        zip_ref.extractall(folder)

    for path, subdirs, files in os.walk(folder):
        for name in files:
            if '.js' in name and '.json' not in name and '.map' not in name:
                # print("--------path", path)
                # print("------- subdir", subdirs)
                if len(path) > 10:
                    res = optimization_response(path+ '/' + name, name, path)
                    response[path+ '/' + name] = res
                else:
                    res = optimization_response(path+name, name, path)
                    response[path+name] = res
```

Command Line Interface Design

The design specification for the static binary is as follows:

The intended idea for this application is to provide users a quick, portable, reliable way of identifying hash without the need of any development environment. Given that there are many stakeholders involved, some of them may not be tech savvy enough to follow through a series of installations of the development environment needed.

An example of how this program works on a higher level using a Sequence Diagram.



Based on this Sequence Diagram, it is possible to implement other functionalities such as output terminal output into an output file, allowing users to quickly search for cryptocurrencies such as Bitcoin, Ethereum, etc.

Hashes.py will be the main program which drives the whole program, while hashID.py and prototypes.py will be the supporting python files which will assist the main program.

The prototypes.py contains most used cryptographic hash functions. As all cryptographic hash values have their own unique string of pattern and fixed output length, by using regular expressions to evaluate the user's input hash value, it is possible to give a best estimate of what that potential cryptographic hash function used.

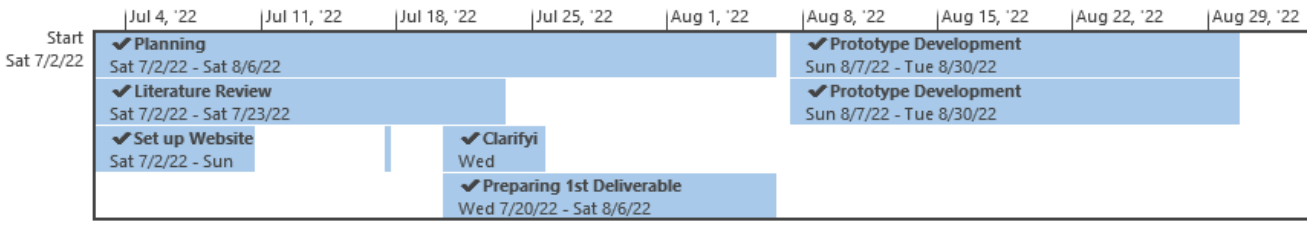
The hashID.py is a file which creates a class called HashID. This allows hashes.py to instantiate an object, giving the object to call the function "identify_hash()". The identify_hash() function will return a generator of possible cryptographic hash function, retrieved from prototypes.py

From here, the hashes.py will then receive the generator from the object and print out or save it into a textfile allowing the user to view the information.

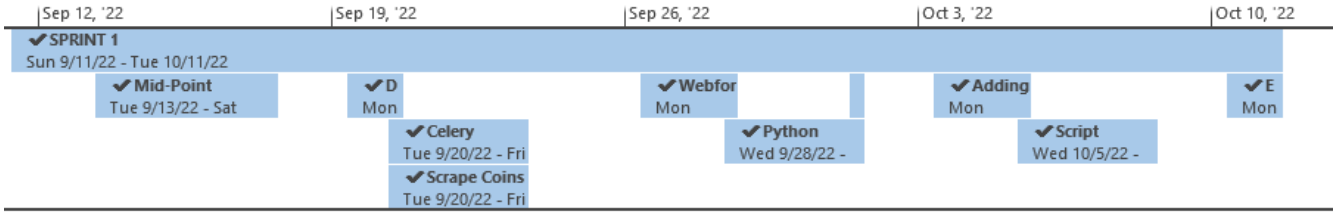
The hashes.py also have a function which allows the user to retrieve information of Cryptocurrencies. This function is supported with the API calling to a website which is generated by the Web application. The results of this API call will allow users to quickly view the type of cryptographic hash function used in the Blockchain.

Project Workflow

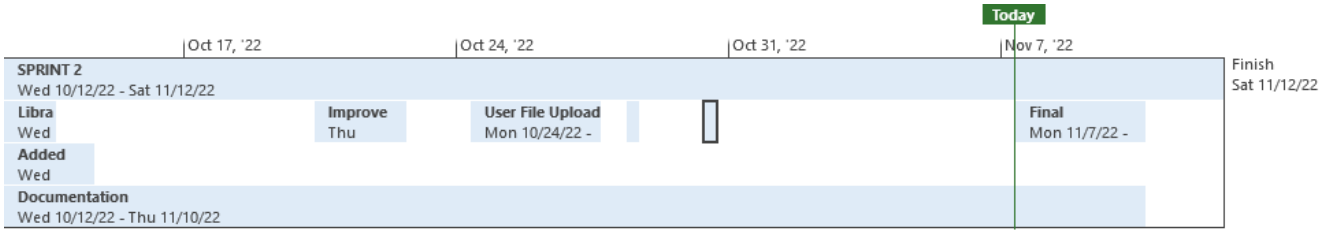
Project Start - Planning



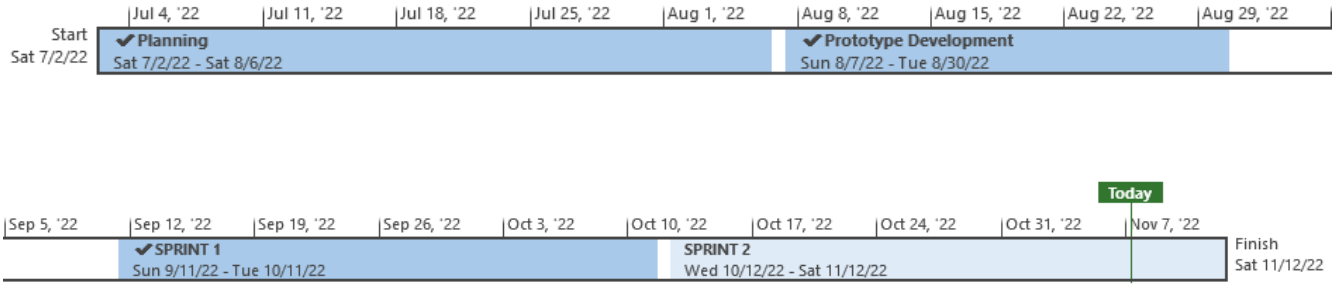
Sprint 1 Timeline



Sprint 2 Timeline

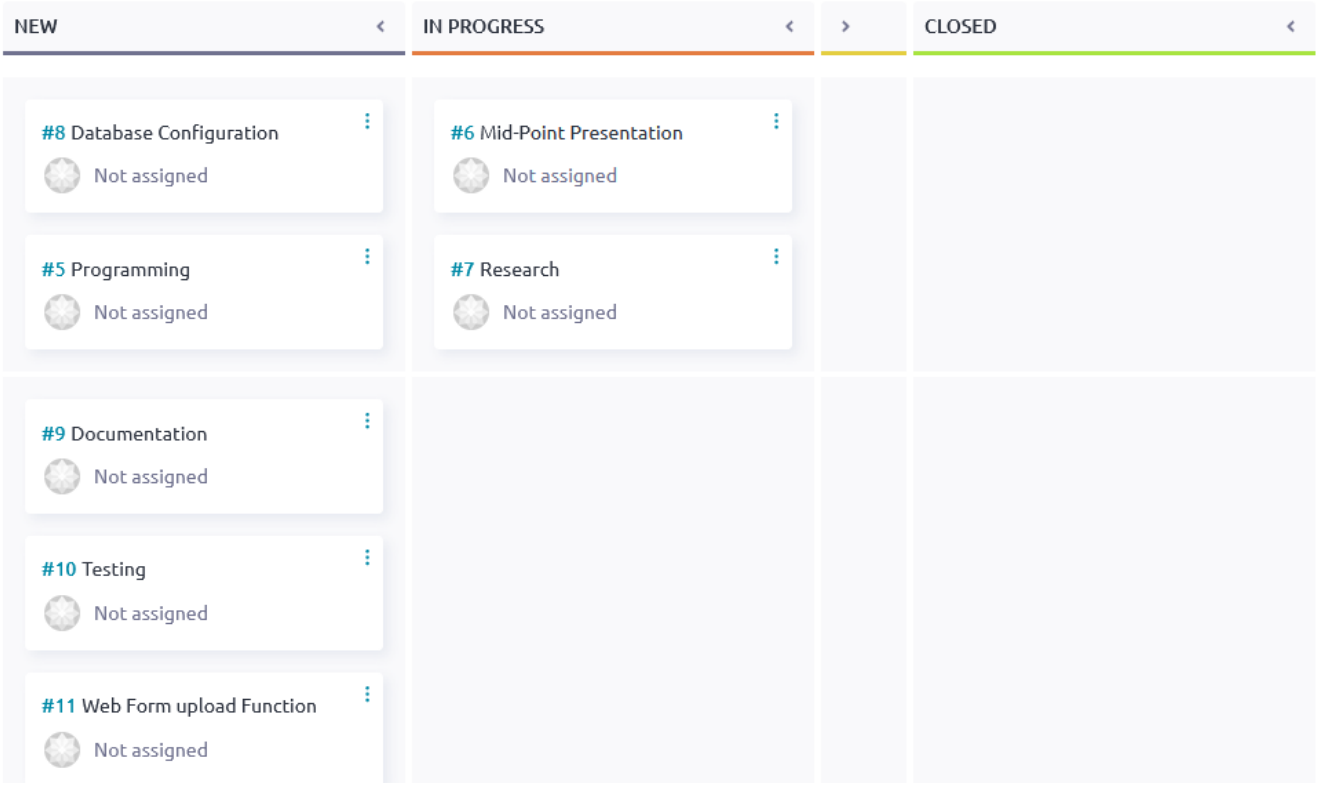


Overall Project Timeline

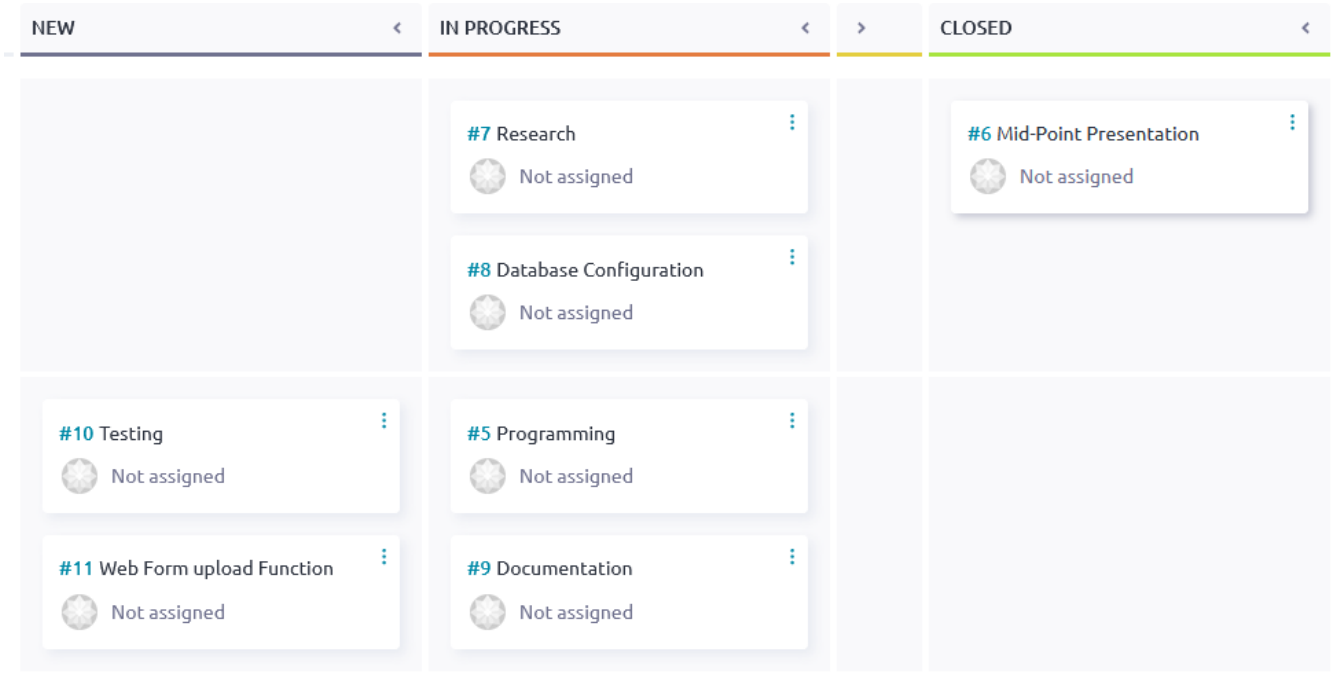


Sprint 1

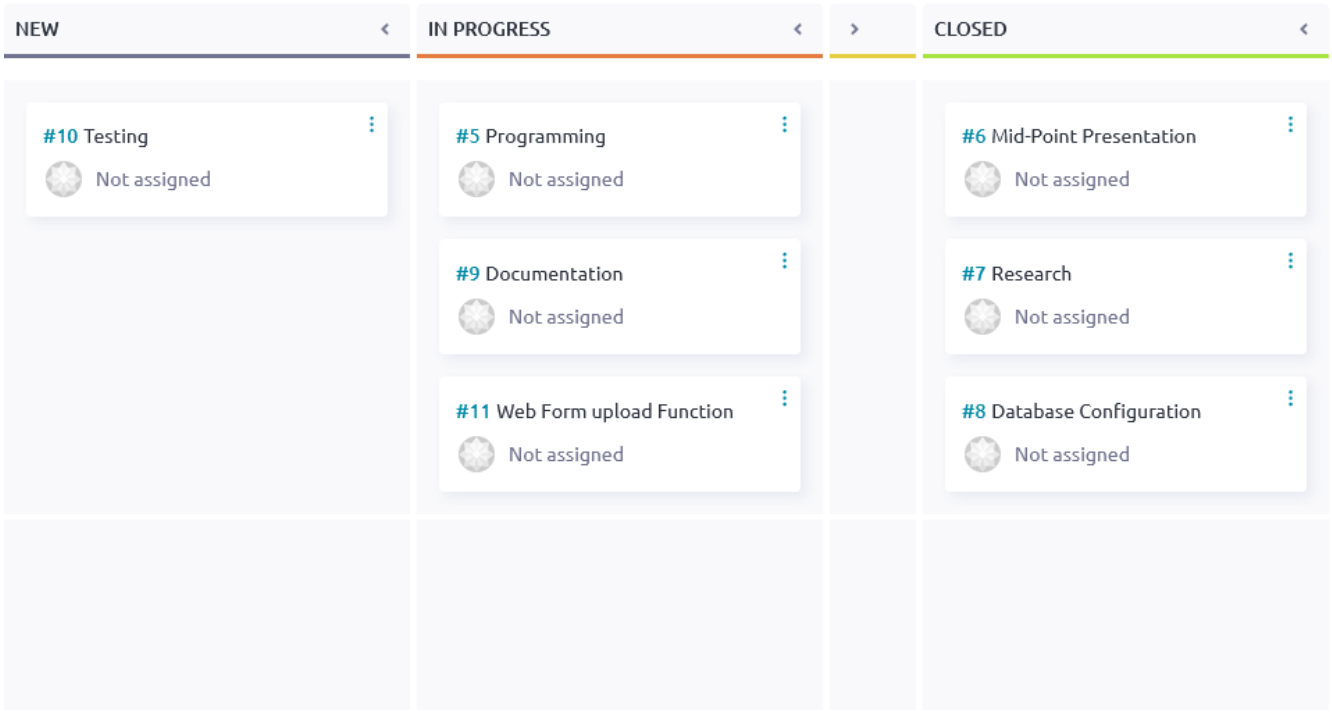
Week 1



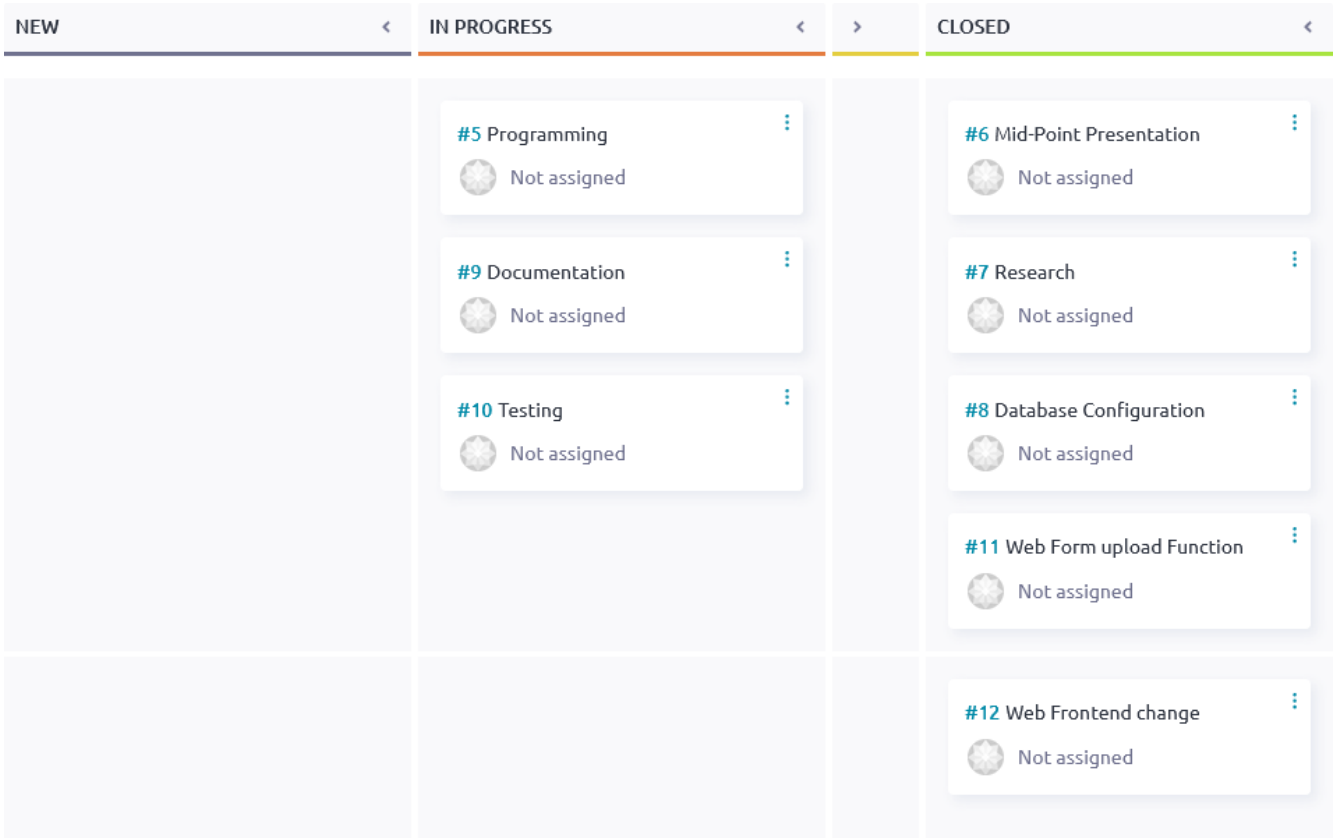
Week 2


NEW
IN PROGRESS
CLOSED
#7 Research
Not assigned
#8 Database Configuration
Not assigned
#6 Mid-Point Presentation
Not assigned
#10 Testing
Not assigned
#5 Programming
Not assigned
#11 Web Form upload Function
Not assigned
#9 Documentation
Not assigned

Week 3

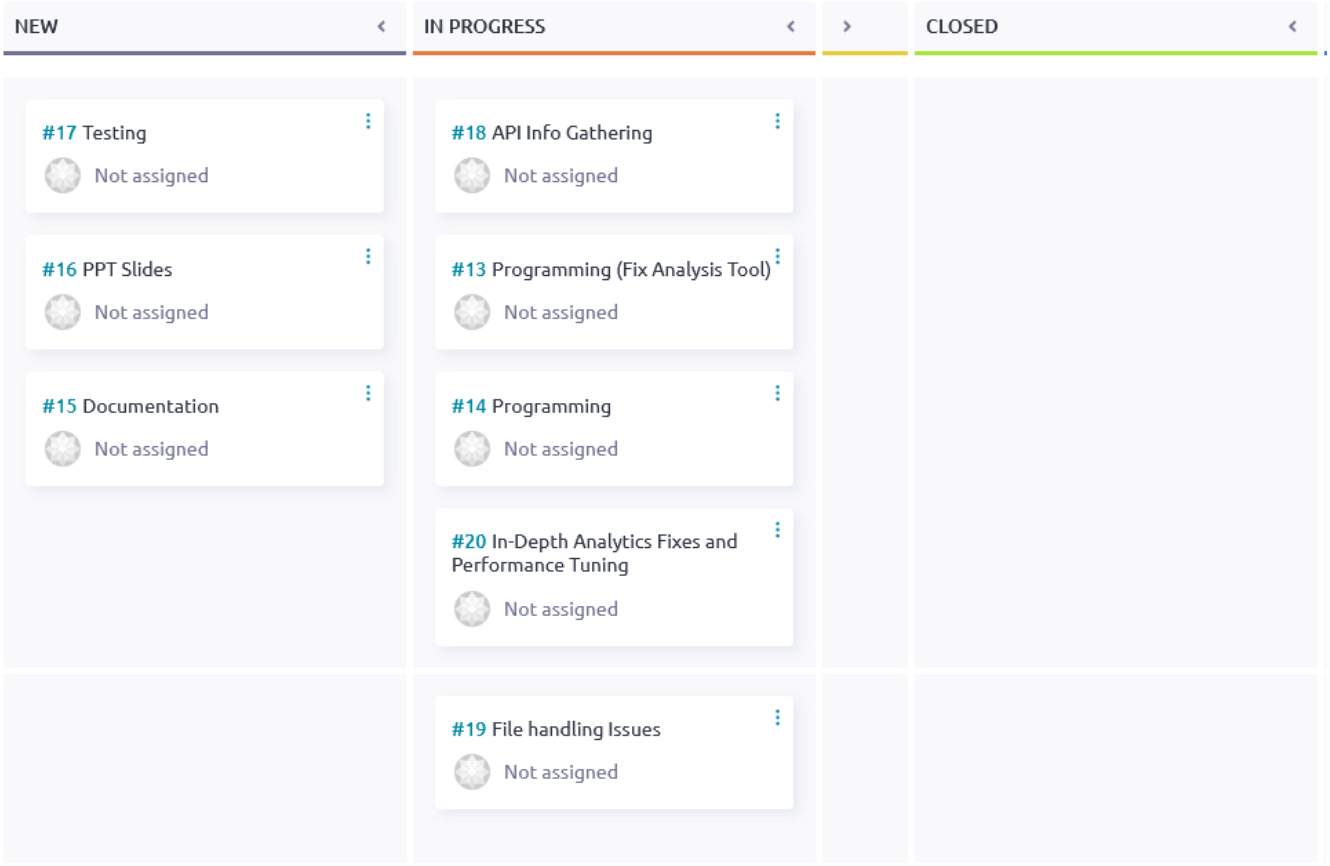

NEW
IN PROGRESS
CLOSED
#10 Testing
Not assigned
#5 Programming
Not assigned
#9 Documentation
Not assigned
#11 Web Form upload Function
Not assigned
#6 Mid-Point Presentation
Not assigned
#7 Research
Not assigned
#8 Database Configuration
Not assigned

Week 4

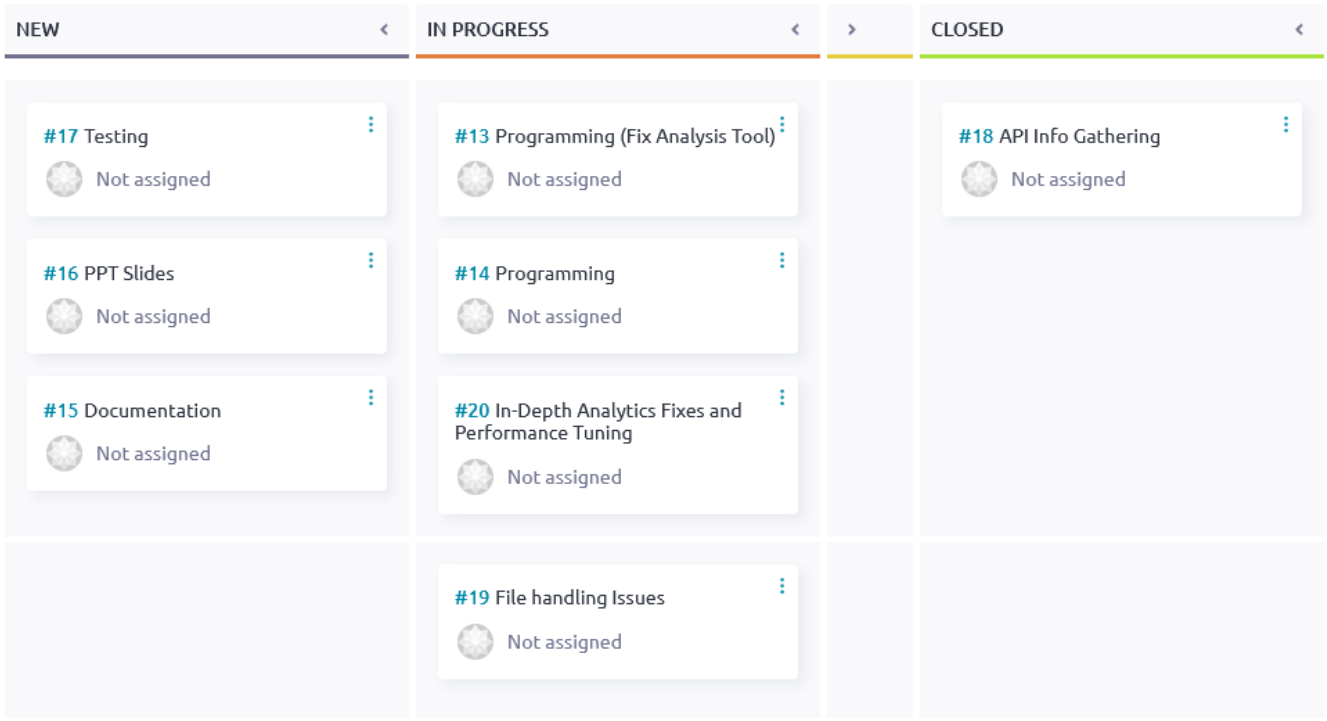

NEW
IN PROGRESS
CLOSED
#5 Programming
Not assigned
#9 Documentation
Not assigned
#10 Testing
Not assigned
#6 Mid-Point Presentation
Not assigned
#7 Research
Not assigned
#8 Database Configuration
Not assigned
#11 Web Form upload Function
Not assigned
#12 Web Frontend change
Not assigned

Sprint 2

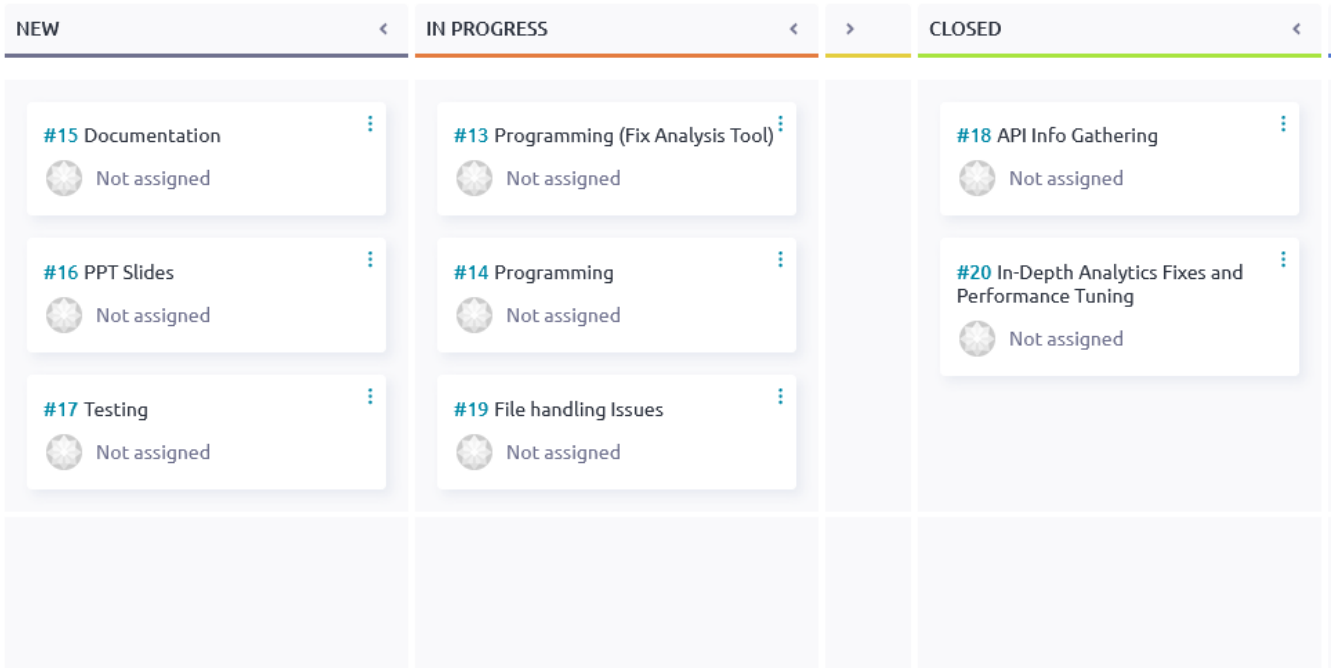
Week 1



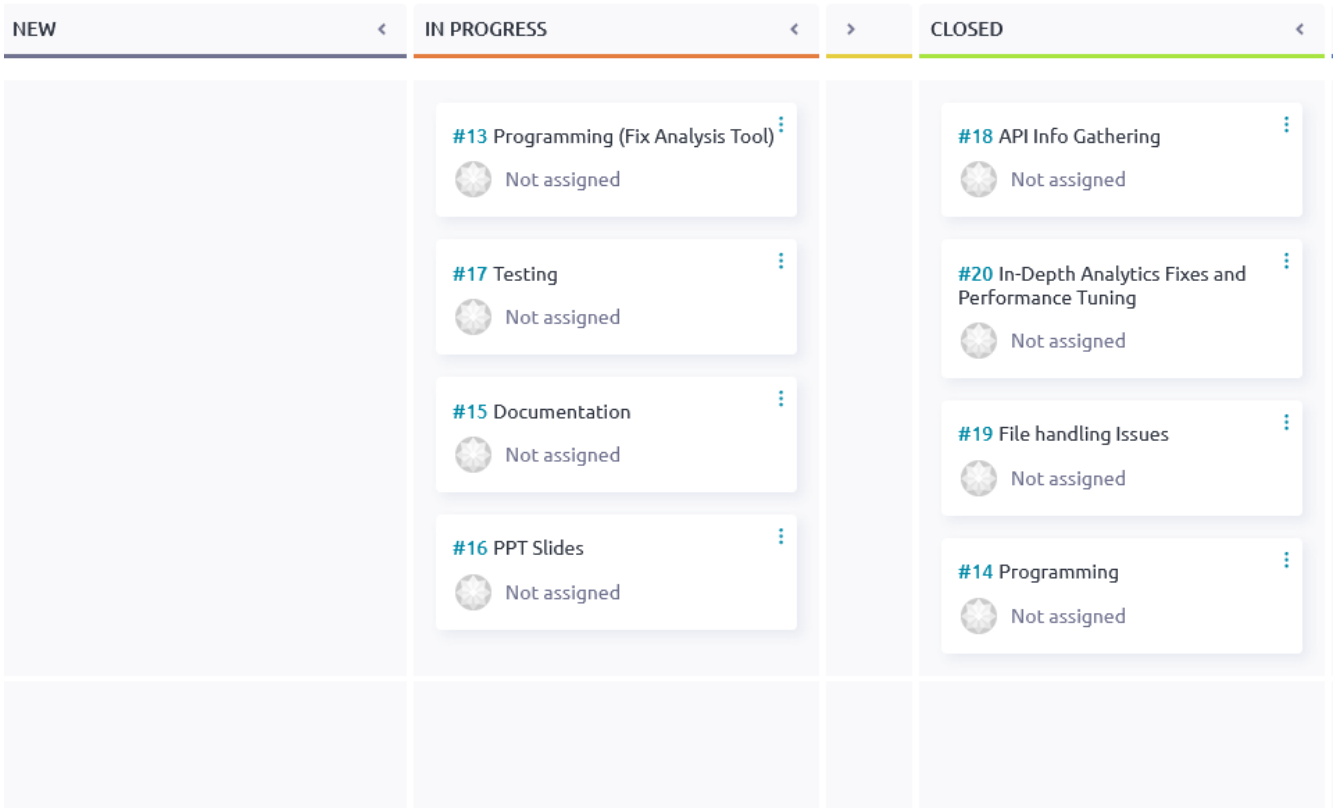
Week 2


NEW
IN PROGRESS
CLOSED
#17 Testing
Not assigned
#16 PPT Slides
Not assigned
#15 Documentation
Not assigned
#13 Programming (Fix Analysis Tool)
Not assigned
#14 Programming
Not assigned
#20 In-Depth Analytics Fixes and Performance Tuning
Not assigned
#19 File handling Issues
Not assigned
#18 API Info Gathering
Not assigned

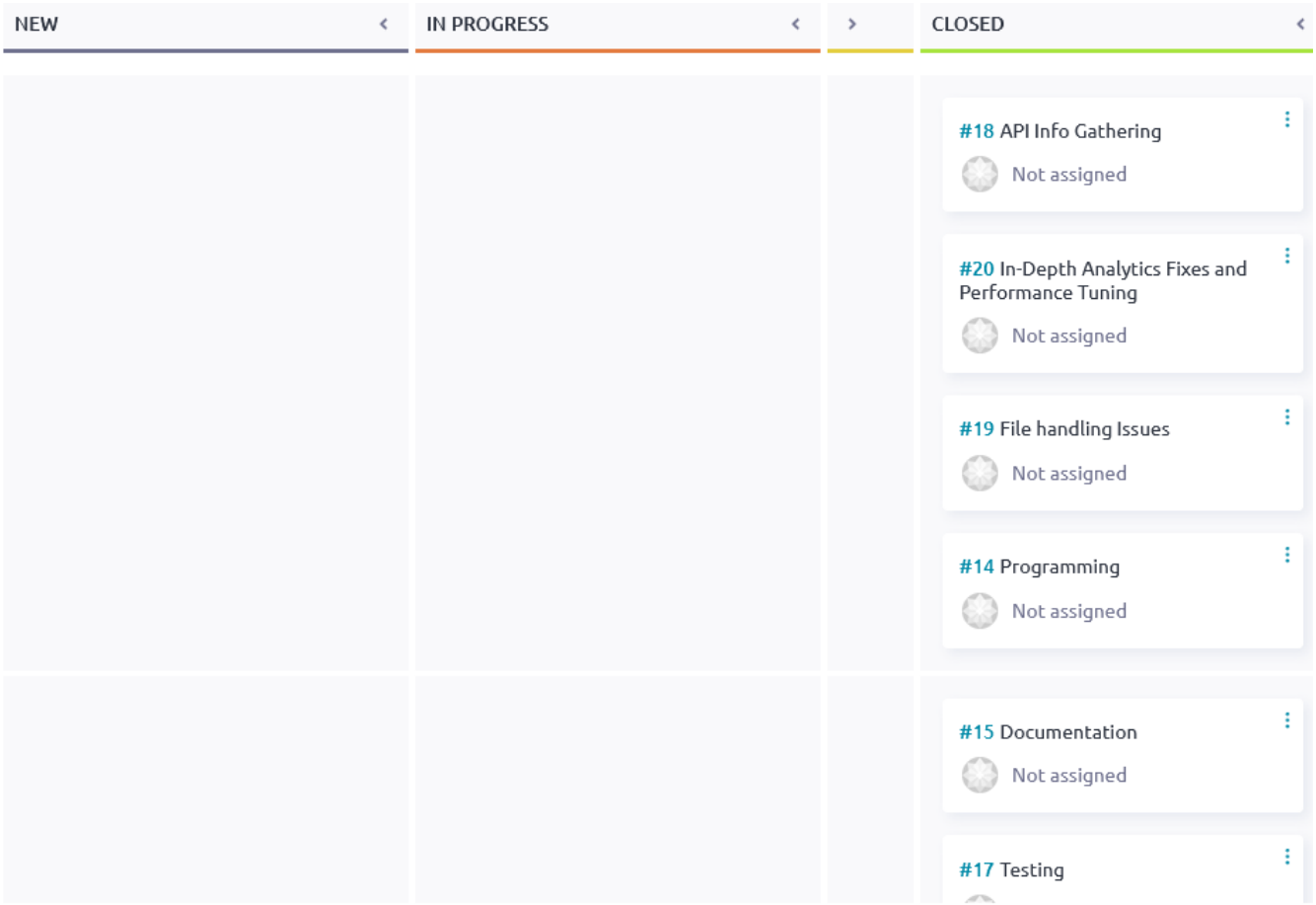
Week 3


NEW
IN PROGRESS
CLOSED
#15 Documentation
Not assigned
#16 PPT Slides
Not assigned
#17 Testing
Not assigned
#13 Programming (Fix Analysis Tool)
Not assigned
#14 Programming
Not assigned
#19 File handling Issues
Not assigned
#18 API Info Gathering
Not assigned
#20 In-Depth Analytics Fixes and Performance Tuning
Not assigned

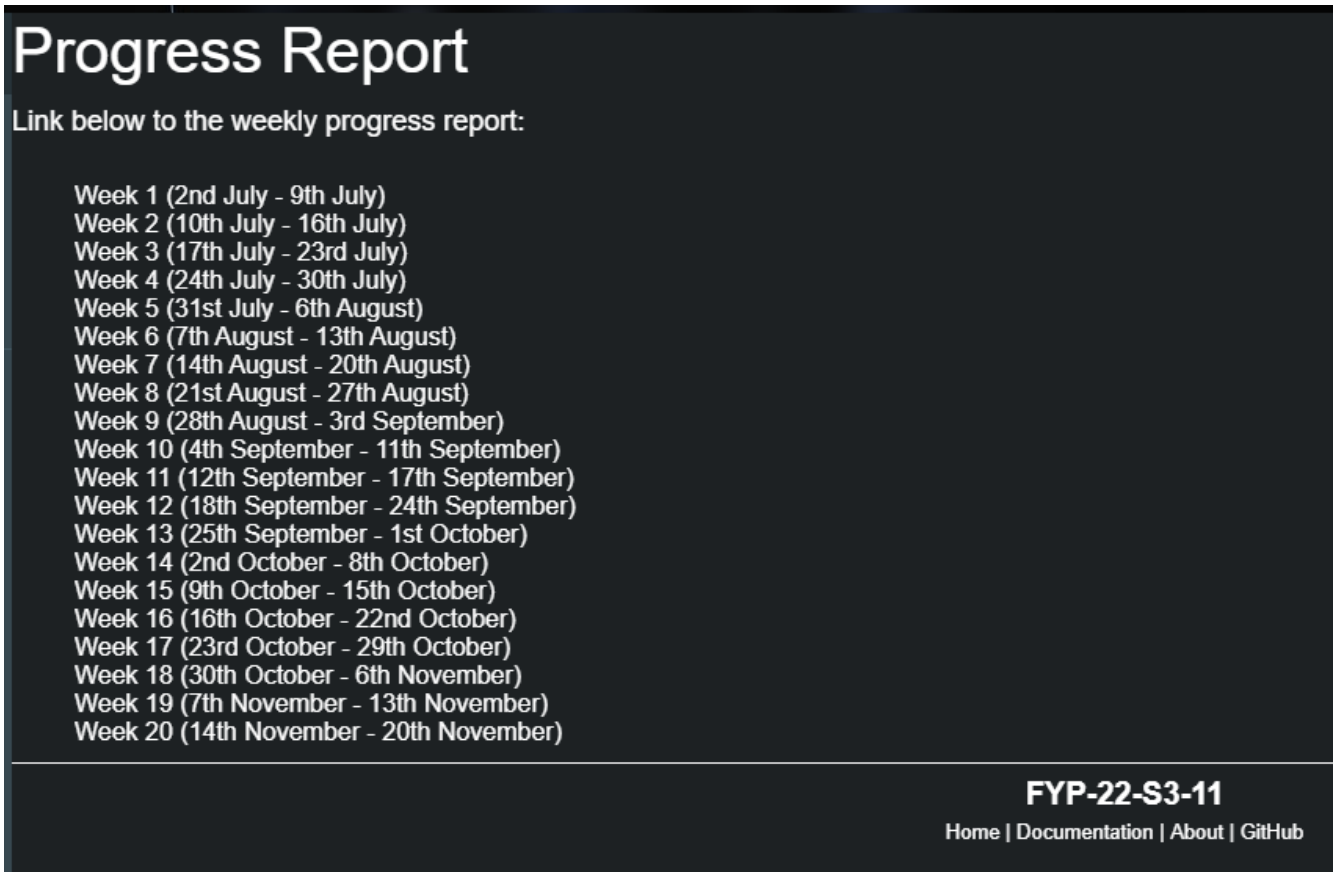
Week 4


NEW
IN PROGRESS
CLOSED
#13 Programming (Fix Analysis Tool)
Not assigned
#17 Testing
Not assigned
#15 Documentation
Not assigned
#16 PPT Slides
Not assigned
#18 API Info Gathering
Not assigned
#20 In-Depth Analytics Fixes and Performance Tuning
Not assigned
#19 File handling Issues
Not assigned
#14 Programming
Not assigned

Week 5


NEW
IN PROGRESS
CLOSED
#18 API Info Gathering
Not assigned
#20 In-Depth Analytics Fixes and Performance Tuning
Not assigned
#19 File handling Issues
Not assigned
#14 Programming
Not assigned
#15 Documentation
Not assigned
#17 Testing

Progress Reports



Do Refer to the project's website for the weekly reports regarding this project.

Project Milestones

Date	Deliverables
6th August 2022	Project's Requirement Specification document
17th September 2022	Prototype Demonstration
17th September 2022	Progress Report submission
1st October 2022	Blockchain Analysis Page
30th October 2022	Fixed Blockchain Analysis Issues
3rd November 2022	CLI Program stable release
12th November 2022	Final product & documentation submission
12th November 2022	Reflective Diary submission
19th November 2022	Final Presentation

Appendix

Language processing:

The ability of a program that breaks down human language for machines to understand how human language as it is spoken and written.

Tokenization - Breaking down of text into single clauses.
Part-of-speech-tagging - Process of correctly marking words as nouns, verbs, adjectives, adverbs, pronouns

Stemming and Lemmatization - Reducing words to their root forms

Stop word removal - Filter common words that add no meaning or provide no information

Open-Source Development

This refers to software with their **source codes available to the public** which allows for communities to help and participate in continuous development.
The end product of these software might not be free but the source codes are still made available freely for communities to develop their own versions of the program.

Open-Source developments are published under **Open-Source Licences** for all to have read and write access and distribute the source code. This results in a form of **Decentralised Collaboration.**

Command Line Interface

A text-based user interface which users use it to interact with a machine. This provides a way for users to update setting parameters for environment, invoking executables or issuing commands, providing information to users as to what actions they perform

Graphical User Interface

Another form of user interface which allow users to interact with executables, programs through graphical icons and/or audio indicators such as primary notation, instead of text-based interface as mentioned above.

Application Programming Interface

A software intermediary that allows applications/products to communicate with each other without the need to know how they're implemented. API gives developers flexibility; simplifying design, administration and usage, hence allowing developers to develop applications faster and simpler, saving time and cost of developing applications.

Dominance

Occasionally, some documents reference the word Dominance or in other forms such as the dominance.js file. Dominance is the given name by our lead programmer to the folder and files responsible for the Blockchain Analysis features of our product.