THE UNIVERSITY OF SYDNEY

School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - GROUP ASSESSMENT

Unit of Study: COMP5703 IT Capstone Project

Assignment name: Project Group Final Report

Tutorial time: Wednesday 16.00-18.00 **Tutor name:** Hamid Samani

DECLARATION

We the undersigned declare that we have read and understood the *University of Sydney Academic Dishonesty and Plagiarism in Coursework Policy*, an, and except where specifically acknowledged, the work contained in this assignment/project is our own work, and has not been copied from other sources or been previously submitted for award or assessment.

We understand that failure to comply with the *Academic Dishonesty and Plagiarism in Coursework Policy* can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

We realise that we may be asked to identify those portions of the work contributed by each of us and required to demonstrate our individual knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Project team members				
Student name	**Student ID**	**Participated**	**Agree to share**	**Signature**
1. Yuming Jiang	460444981	Yes / No ✓	Yes/No ✓	Yuming Jiang
2. Xinyi Liu	450456633	Yes / No ✓	Yes / No ✓	Xinyi Liu
3. Zhengyang Liu	460161413	Yes / No ✓	Yes / No ✓	Zhengyang Liu
4.		Yes / No	Yes / No	
5.		Yes / No	Yes / No	
6.		Yes / No	Yes / No	
7.		Yes / No	Yes / No	
8.		Yes / No	Yes / No	
9.		Yes / No	Yes / No	
10.		Yes / No	Yes / No	

SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.info@sydney.edu.au
sydney.edu.au/it

ABN 15 211 513 464
CRICOS 00026A

COMP 5703 IT Capstone Project

A Reference Model of Applying IoT (Internet of Things) and Blockchain to Enterprise System

Final Group Report

YUMING JIANG, XINYI LIU, ZHENGYANG LIU

2017-11-17

CONTENT

1. Executive Summary

This report explains the detailed information of how we processed the whole project, in which the IoT technologies, such as the blockchain, are to apply to achieve the objective of eradicating counterfeit pharmaceuticals. A comprehensive introduction of the whole system developed by our group will be illustrated in the system specification section. To fulfil our goals, we developed the system with the assistance of BigchainDB to realize the authenticity and uniqueness of the legal pharmaceutical. In the system, each legal pharmaceutical will be granted a unique transaction ID which is generated by our system. Beyond preventing counterfeit pharmaceuticals, our group also wants to show the effectiveness of using the blockchain system to solve the real-world problem. We think that our system can also be seen as a reference model of implementing blockchain system to enhance the existing enterprise system.

2. Literature Review

2.1. Internet of Things

Internet of things (IoT) also called internet of everything or industrial internet, which is a new technology paradigm(Lee & Lee, 2015). IoT defined an internet environment that can share the data and information, and the 'things' are all the devices in this environment. Therefore, all the devices can exchange and communicate with each other rapidly and effectively in this environment. Due to the characteristics of IoT, this technology is used in different fields. IoT has reached 0.9 billion units in 2009, and Garter predicted that IoT will reach 26 billion units in 2020(Lee & Lee, 2015). This data shows that IOT will have a huge impact on the way that information exchanged to the largest extends.

2.1.1. Essential IoT Technologies

There are five essential technologies to form the IoT system, which are Radio frequency identification (RFID), wireless sensor networks (WSN), middleware, cloud computing

and IoT application software.

Radio frequency identification (RFID) is a technology that is used to identify and collect the data, which includes RFID tags and RFID readers(Gubbi, Buyya, Marusic, & Palaniswami, 2013).The RFID tags have three types: active RFID tag, passive RFID tag and semi-passive RFID tag. Active RFID tags have batteries and external sensors and mainly used in labs and manufacturing industry(Lee & Lee, 2015). The passive and semi-passive RFID tags do not have internal batteries, and this kind of tags are dormant when they are not working, thus passive and semi-passive RFID tags have longer working life than active RFID tags. The passive RFID tags mainly used in passport and electronic passes(Lee & Lee, 2015).

Wireless sensor network (WSN) is a distributed sensing network. This network is consisted of the amount of sensors that can communicate with other sensors by the wireless way(Gubbi et al., 2013).Because of the rapid development of technology, WSN provides a platform that can collect, process, analyze and spread the information and data in a low-power, low-prince and efficient way(Lee & Lee, 2015). Nowadays, WSN has applied to many fields, such as cold chain logistic, smart transportation, environment surveillance etc.

Middleware is a software layer between the operation system and all kinds of distributed application software, which provide a unified operation platform and a friendly environment(Lee & Lee, 2015).The middleware can satisfy a mass of application requirements and also can operate on different kinds of hardware and OS platform. Besides, different application systems can share the data via middleware, and middleware can pre-process the mass data that collect from the perception layer.

Cloud computing is an internet-based computing mode that is a configurable Shared resource pool includes hardware, software resource and information and the resource can be offered for all kinds of computer terminals and other devices(Lee & Lee, 2015).Cloud computing provides a new and high-efficiency compute mode, thus could computing can better process the mass data from the perception layer in IoT.

The last technology of IoT is IoT applications. To better extend the popularity of the IoT to different areas, IoT applications have played an important role. IoT application archive the connection of devices and guarantee the interaction between users and devices(Lee & Lee, 2015).Therefore, IoT applications visualize the data that can provide the information and data for users in an easy to understand way.

2.1.2. IoT for Logistic Systems

The traditional logistic is a way to move the goods to another place and the whole process is controlled by the human, which logistic service is simple, and its's supply chain is short. Therefore, the traditional logistic is not flexible enough and also lack the expansibility(Chen, Chen, & Hsu, 2014). In contrast, modern logistics system effectively overcome the traditional logistic weaknesses and IoT provide a very good platform for the logistic system that will change the process and management of supply chain fundamentally(Chen et al., 2014).

As mentioned before, RFID tags can save mass product information, RFID reader can easily obtain all the information that would be saved in the tags and WSN can exchange and communicate with all the sensors in the network quickly. In this way, products' location and information can be identified, located, monitored and managed intelligently(Chen et al., 2014). Thus, users do not need to write down the product information one by one and all the work will be replaced by the machine, which can save time and manpower.

The middleware can improve the way to collect, share and exchange data and also can help end users to load, modify and deploy the data(Chen et al., 2014).Besides that, middleware can help the RFID tag to filtrate data and enhance the operation of RFID(Chen et al., 2014).

The cloud computing is a resources pool, which user can get the resource from this pool on demand at anyplace anytime. Therefore, cloud computing can help to handle and process plentiful logistic information and reduce the information management workload and time.

The IoT application can deploy on any operating systems, such as Android devices, iOS devices, etc. Thus, end users can check and acquire the data on their laptop or mobile phone.

In conclusion, The Internet of Things can fundamentally change the logistics industry. Using IoT, the logistics process can be automated totally, and transportation can be enhanced efficiency to the largest extent. Moreover, IoT can make information acquisition faster and more accurate. Therefore, this modern IoT logistics system can guarantee transportation of the long distance, a huge number of goods.

2.1.3. Challenges and Risks

However, Companies still need to face some challenges and risks when they deploy IoT in their logistic system. There are two main challenges and risks should be considered.

(1) The most serious problem is security issues. The RFID is the most vulnerable part in IoT that can be tacked by anyone and not intelligent to identify scan devices (Gubbi et al., 2013).Besides, the research shows that 70% of the commonly used IoT devices have very serious vulnerabilities, lack of transmission encryption, insecure network interfaces, inadequate software protection and insufficient authorization that are all reasons lead to data leakage(Lee & Lee, 2015).

(2) When the company use resources from cloud computing pool, they need to choose one suitable model. If the company chooses private cloud, this cloud model can protect the privacy of client and company information, but company need to maintain and repair the services and infrastructures (Chen et al., 2014). On the other hands, if user chooses other kinds model, they need to share services and infrastructures with other companies, which provided by a third-party cloud resource providers(Chen et al., 2014).Therefore, the security and privacy of the information need to be considered.

2.2. Blockchain

The blockchain is a technology which can play an important role in eliminating the multiple spend problem. Blockchain is a decentralized system which involves a large

number of participants who play the role of bookkeeper in the system (Pilkington, 2015). Its working principle is to assign a key pair to each participant, with the public key shared among all the other participants and the private key is kept by the owner like a password. To initiate a transaction, the buyer needs to send the public key to the owner of the asset. The owner of the asset will conduct the transaction, and the transaction will be recorded by all the participants in the system. Meanwhile, enabled by cryptology technology, every single record will be hashed with the transaction id and the public key. Since the cryptographic hash function is extremely difficult to revert, the hash is theoretically impossible to be recreated.

Also, the characteristic of blockchain can guarantee the validation of any completed transaction. When verifying a transaction, the transaction details will be sent to all the participants in the system. Only the consensus is achieved among all the participants can the transaction be processed and recorded to all the agents in the system (Wright & De Filippi, 2015). This can ensure most participants keep the same record of the transaction. Meanwhile, the new block will be attached to the end of previous blocks, and it cannot be deleted from the network. Indeed, the transaction information contained in the blocks can be accessed and verified by all the participants in the system. Furthermore, the “proof of work” mechanism ensures that the granted transactions are legal and valid. The mechanism can realize the ability to anti-counterfeit. Because the transaction has to be agreed by most participants in the system before it is complete. Meanwhile, the participants can be located anywhere in the world, and the number of participants is large. Therefore, completing a fake transaction is impossible since this requires the hacker needs to control more than a half of calculation power of the who network, which cannot happen in most conditions.

Meanwhile, the “proof work” mechanism can guarantee the uniqueness of transaction effectively. Most participants in the network will record the transaction once it is complete. Which means that there already have an existing transaction using the unique hash and transaction ID. If anyone wants to reuse the transaction ID to empower a new

transaction, most participants will deny it since they have recorded the transaction ID, which means that another same transaction ID is fake definitely (Sadouskaya, 2017).

3. Background

3.1. Motivation

As a result of economic globalization, people can buy any products from other countries on the internet. However, more and more counterfeit products flow into the market, among these counterfeit goods problems (Sowder, 2016). Fake medicine deserves to be paid more attention to the public. Counterfeit medicine is a global problem. Due to this problem, the mortality rate is very high in any country around the world per year, especially in some undeveloped countries.

Harmful substances are one of the characteristics of counterfeit medicines. Epidemic meningitis broke out in Nigeria in 1996, but the fake vaccines lead to 2,500 people death(Morris & Stevens, 2006).Besides this feature, insufficient active molecules in drugs are another feature of counterfeit drugs. Lacking active factors would delay the treatment of patients, and the surviving virus would evolve into the drug-resistant virus(Morris & Stevens, 2006).Therefore, counterfeit drugs will lead to serious consequences.

The World Health Organization estimates that approximate 10% of the global pharmaceutical market is counterfeit drugs (Nsimba, 2008). The percentage will up to 25% in some developing countries, even more in individual countries this percentage would increase to 50% (Nsimba, 2008). For example, in China, it is estimated that around 192,000 people died because of counterfeit drugs in 2001, and approximately one-third to one-fifth of antimalarial drugs are fake in southeast Asia(Cockburn, Newton, Agyarko, Akunyili, & White, 2005).

The number of counterfeit drugs is still growing at a significant rate. The US Food and Drug Administration (FDA) published the number of counterfeit drugs from 1997 to 2008 (Figure 0)(Agbaraji, Ochulor, & Ezeh, 2012). This chart shows that the counterfeit

drug is a problem that cannot be ignored.

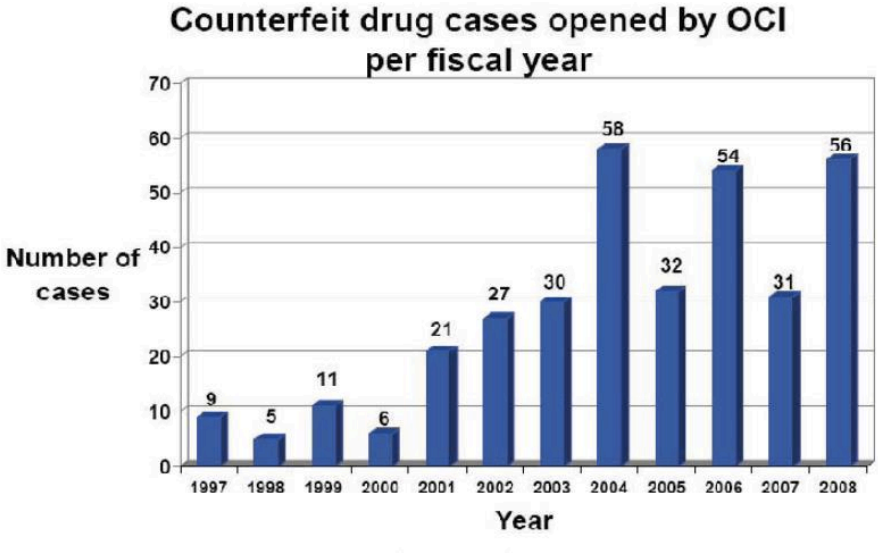


Figure 0 The Counterfeit Drug Published by FDA

In addition, counterfeit drugs have weakened the innovation passion of pharmaceutical research and development companies and reduced the company's revenue as well, which will impact company's researches(Morris & Stevens, 2006). Moreover, the price of most counterfeit medicines is cheap. Thus, counterfeit medicines undermine the healthy competition in the pharmaceutical market.

However, why counterfeit drugs are hard to avoid? Firstly, because the high price of imported medicines that contain high tariffs and the patentability of medicines in some underdeveloped countries cannot be well guaranteed(Morris & Stevens, 2006). Therefore, counterfeit medicines will become more and more in the market. Secondly, because of huge economic benefits, imported medicines may be replaced by counterfeit medicines. Just because of it, the quality and safety of imported medicines cannot be guaranteed(Pecoul, Chirac, Trouiller, & Pinel, 1999). Thirdly, some medicines require very high production conditions, and underdeveloped countries cannot meet the standards to product qualified medicines. Thereby the technical, financial and human resources cannot meet the requirements, which fail to meet the medicine quality standards(Pecoul et al., 1999). Fourthly, in underdeveloped countries, the cheaper medicines from informal channels are more attractive for poor consumers(Pecoul et al., 1999).

3.2. Existing platform and technologies

In order to ensure that consumers can obtain good quality drugs, there are some existing technologies and methods to check the authenticity of drugs.

3.2.1. Idea Anti-counterfeit Technologies

Idea anti-counterfeit technology should have these features: (1) a high level of security (2) rapidly identify rate (3) hard to eliminate and reuse (4) easy to check and identify automatically (5) comply with industry laws and regulations(Bansal, Malla, Gudala, & Tiwari, 2012). Every country tries to develop the ideal anti-counterfeit technology, the following paragraphs will detailed explain some existing verification means.

3.2.2. Mobile Authentication Service(MAS)

Figure 1 The Work Flow of MAS (Anti-CounterfeitingPackaging, 2016)

The National Agency for Food and Drug Administration and Control (NAFDAC) introduced this service to Nigeria to control the counterfeit medicines problem (Ayodokun, 2014). The Mobile Authentication Service (MAS) has three steps to verify the medicines (figure 1). Firstly, consumer scratches the sproxil label on the product package back, which contain the unique twelve digits PIN (Ayodokun, 2014). Secondly, the consumer sends the PIN via SMS to a dedicated number which would automatically reply the information about the authenticity of the product (Ayodokun, 2014).

Nevertheless, this service is not good enough. The sproxil label is too easy to copy, and the service cannot decide if the label has been reused. Therefore, this MAS cannot estimate this situation of fake product with a real sproxil label.

3.2.3. Rapid Alarm and Product Recall System

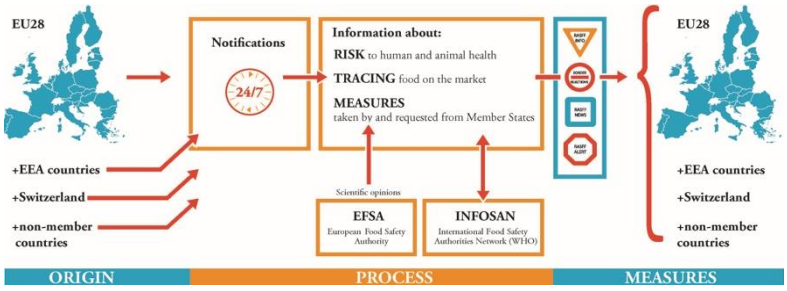


Figure 2 Work Flow of Rapid Alarm and Product Recall System(EuropeanCommission, 2017)

Figure 2 shows a drug tracking and recall system that can exchange of information about dangerous non-food products between 31 European countries and the European commission. If the consumer thinks they have bought a counterfeit drug, they can submit a description of the drug to the system, and then the consumer will receive an expert feedback (Isah, 2012). This system can ensure the approved drugs that can be traded on the market only. Drugs which have side effects would receive a warning, and the illegal drug will be recalled (Isah, 2012).

However, this system still has a shortage. Because the consumers are not the expert of medicine, thus it is hard to guarantee the accuracy of drug description. In addition, the authenticity of expert judgment is another issue that needs to be considered.

3.2.4. ePedigree System

This system is built by The United States Food and Drug Administration (FDA), which use RFID to trace and identify the drugs and every product have a unique ID. This system can trace the whole path from medicine factories to medicine store and avoid being replaced by fake drugs during transportation. Every medicine package has been marked and many marks can be read at the same time. Thus the RFID can trace the medicine in the whole process of the supply chain(Agbaraji et al., 2012). Moreover, the tag application, product package, the whole distribution process and pharmacy class will be traced and identified at the read point(Agbaraji et al., 2012).Therefore, RFID can enhance the drugs supply chain more effective and safe.

However, this system still has some disadvantages. As mentioned before, the RFID is too easy to be traced by anyone, which means that the information in the RFID tags can be read by any RFID readers that can lead to the duplication of RFID tags. Thus, the safety of supply chain cannot be protected.

4. Objective

4.1. User Stories

4.1.1. Definition of User Story

A user story is a natural language, the informal expectation of one function of the software system written by the end user. In this system, there are two applications for the end users, the mobile application and web application. To determine the user stories for these two deliverables, we searched a lot of information on the Internet. All of the user stories are written by our group members in the view of our users with the help of collected information. For the mobile application, there are some functions will only be used by specific types of users. These specific types of users are denoted as drug producer, seller and buyer.

4.1.2. User Stories

<table>
<tr><th>Role in the system \ Deliverables</th><th>Mobile Application</th><th>Web Application</th></tr>
<tr><td rowspan="4">User</td><td>(1) As a user, I can sign up an account or log in with existing account so that I can access to the mobile application.

(2) As a user, I can see if the transaction has succeeded.</td><td>(1) As a user, I can sign up new account or log in with existing account so that I can access to the web application with the browser.

(2) As a user, I can type in the transaction ID so that I can verify if the drug is real.</td></tr>
<tr><td>Drug Producer</td><td>As a drug producer, I can get new transaction ID for newly produced drug.</td><td rowspan="3"></td></tr>
<tr><td>Seller</td><td>As a seller, I can update the transaction ID for my own drug trading.</td></tr>
<tr><td>Buyer</td><td>As a buyer, I can provide my public key to the seller.</td></tr>
</table>

Figure 3 User Stories

4.2. Deliverables

To fulfil the user stories shown in Figure 3, we developed three deliverables, API Server, Mobile Application and Web Application. Mobile Application and Web Application will be used by the end users directly. API servers will support the running of Mobile Application and Web Application by providing APIs with the help of MongoDB and BigchianDB. The solution for the two problems, “Serial Number Reuse” and “Unauthenticated producer”, will be implemented in API Server and will be explained in the Methodology section. The detailed architecture and functions of our three deliverables will be explained in the System Specification section.

5. Methodology

5.1. API Server

5.1.1. Serial Number Reuse Problem

5.1.1.1. Problem Specification

As shown in the background section, nowadays, the most popular way to prevent the fake drug is to attach a unique serial number to each drug. Also shown in the background section, if the serial number is copied by the fake drug producer and is reused for the fake drug. It is hard for existing methods to detect that the drug is fake.

5.1.1.2. Problem Solution

To solve this problem, our system uses transaction ID instead of the serial number as the identification of the drug. Transaction ID is a unique string generated by our system. Each drug will always have one transaction ID.

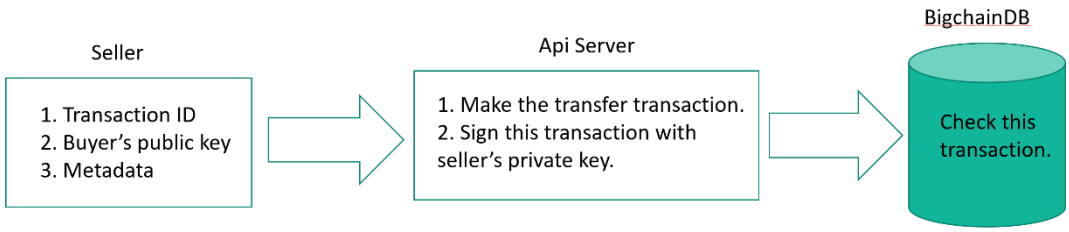


Figure 4 Drug Selling Procedure

As shown as figure 4, if the seller wants to sell the drug to the buyer, the seller should use Mobile Application to send this information, current transaction ID, buyer's public key and metadata, to API Server. API Server will make the transfer transaction, sign this transfer transaction with the seller's private key and push this signed transaction to BigchainDB. With the help of BigchainDB's function called built-in transaction validity checking (GmbH, 2017 May), API Server will send back the different response to seller's Mobile Application based on the drug's current transaction ID.

```
{
    "user_profile": {
        "_id": "59e30e6abd282b06f6199c52",
        "updatedAt": "2017-10-15T07:29:46.419Z",
        "createdAt": "2017-10-15T07:29:46.419Z",
        "email": "alice",
        "password": "1234",
        "private_key": "3xBnQ23nKm6kYTkRmqCYumFqToFsDYnEQo41g9KhE1ZA",
        "public_key": "AQYP5fpZt3ytZ4AtVzxUdxEzpe3TCBQF9LSFuFR7vFL4",
        "__v": 0
    },
    "transaction_id": "143a9b111d9419c95d86dee6463fa02de7f697658557cb49dc2795c4e0b892ff"
}
```


Figure 5 Valid Transaction Response

If the seller sells the real drug which means the real transaction ID is used, the updated transaction ID will be sent back to the seller's Mobile Application which is shown in figure 5.


```
{
    "url": "http://localhost:9984/api/v1/transac
    "status": 404,
    "statusText": "NOT FOUND",
    "headers": {
        "_headers": {
            "server": [
                "gunicorn/19.7.1"
            ],
            "date": [
                "Tue, 17 Oct 2017 06:13:44 GMT"
            ],
            "connection": [
                "close"
            ],
            "content-type": [
                "application/json"
            ],
            "access-control-allow-origin": [
                "*"
            ],
            "content-length": [
                "47"
```


Figure 6 Invalid Transaction Response for Fake Transaction ID

If the seller sells fake drug and uses the fake transaction ID which is made up by the seller randomly, as shown in figure 6, only the error message will be sent back to the seller's Mobile Application. In this situation, the seller cannot show the updated transaction ID to the buyer. If the seller makes up another fake transaction ID and uses it as the updated ID, the buyer can notice this by checking this transaction ID with the buyer's mobile application. In a word, the buyer will find out that the drug is fake and won't buy it.

```
{
    "url": "http://localhost:9984/api/v1/transac
    "status": 404,
    "statusText": "NOT FOUND",
    "headers": {
        "_headers": {
            "server": [
                "gunicorn/19.7.1"
            ],
            "date": [
                "Tue, 17 Oct 2017 06:13:44 GMT"
            ],
            "connection": [
                "close"
            ],
            "content-type": [
                "application/json"
            ],
            "access-control-allow-origin": [
                "*"
            ],
            "content-length": [
                "47"
```


Figure 7 Invalid Transaction Response for Reused Transaction ID

If the seller sells fake drugs and uses transaction ID which is copied by other sold real drug, as shown in figure 7, only the error message will be sent back to the seller's Mobile Application. In this situation, the seller cannot show the updated transaction ID to the buyer. If the seller makes up another fake transaction ID and uses it as the updated ID, the buyer can notice this with checking this transaction ID with the buyer's own Mobile Application. In a word, the buyer will find out that the drug is fake and won't buy it.

5.1.2. Unauthorized Producer Problem

5.1.2.1. Problem Specification

The traditional systems that prevent the fake drug are very easy to be hacked. After the system is hacked, the unauthorized producers can make up a real serial number for their fake drug.

5.1.2.2. Problem Solution

BigchainDB is very tamper-resistant, the tamper can be detected (and rejected) because

of the extensive use of cryptographic signatures, hash chains, and Merkle trees (GmbH, 2017 May). This means that it is very hard to hack our blockchain system - BigchainDB to get the real transaction ID directly. The only way to get the new transaction ID for the newly produced drug is to call the “create_new_asset” API provided by our API Server. Only the authorized user can call this API and get the new transaction ID as the response as shown in figure 8. As shown in figure 9, unauthorized users can only get the error message as the response.


```
{
    "user_profile": {
        "_id": "59eac3f7ae4c8a695818d257",
        "updatedAt": "2017-10-21T03:54:39.627Z",
        "createdAt": "2017-10-21T03:50:15.233Z",
        "email": "alice",
        "password": "1234",
        "private_key": "EqzgZom8BvQXEQs3yXuBGZJPMKzxUdcHRU5g3husHdpr",
        "public_key": "Af5EmfRdvTqExVdPo3QfdVNFLUAsMH13twuWxxixLifY",
        "producer": "yes",
        "__v": 0
    },
    "asset": {
        "drug": {
            "serial_number": "sydney_201710221449",
            "manufacturer": "Jiang"
        }
    },
    "transaction_id": "8b5de85c14ebc75ac19fbcd542b1a82ee80259ecf1115b7e667c285c08252383"
}
```


Figure 8 Response for Authorized Producer


```
{
    "msg": "You are not allowed to create new asset."
}
```


Figure 9 Response for Unauthorized Producer

5.2. Web application

5.2.1. Methodologies Used in The Project

5.2.1.1. React.js Library

React is a JavaScript library, which allows the code to be organized in a more predictable way (React, 2017). React.js is developed by Facebook, which is a framework which is different from Model View Controller framework. This library is

used to build the Instagram web site in the beginning. React.js has a unique design idea and excellent performance, whereas the code logic is very simple. React will automatically manage and update the user interface (UI) when the data changes. React constructs reusable components, hence the components can be repeated used, and every function can be encapsulated that would make the program easy to test and modify(React, 2017).

React.js can easily create UI interaction interface that design a simple view for every state. When the data changes, React.js can update the rendering interface effectively. In this declarative way to build the UI，the code can be more reliable and easier for developers to debug. Moreover, using React.js can build reusable components with their states and these components can be utilized to constitute more complicated interface.

The React.js library is mainly composed of components, JSX and Virtual DOM and Data Flow(React, 2017). The component is the most important part of React application, because react applications are built by components. Each component must implement a render method which can output component. In addition, the component also includes two core concepts: props and state. Props is used to configure the properties of the component. When the component has been called, the properties of the component would be customized by the props, and then display this component on the interface(React, 2017). The state is another concept of components in React. React.js considers that user interfaces have different states and render these states can easily ensure that user interface is consistent with data(React, 2017). Therefore, the user interface would be re-rendered when the state of the component has been updated, which means the user interface will change as the state changes.

In the React applications, HTML is embedded in the JavaScript code directly, and this syntax is JSX syntax(React, 2017). Therefore, JSX supports to embed HTML in JavaScript code that makes the web app development more componentized.

In traditional web applications, if the web app is modified, the entire DOM tree needs to be updated. To directly modify the DOM tree requires a very high-level performance.

In contrast, there is a concept of Virtual DOM in the React application. The component DOM structure is mapped to this Virtual DOM and React implements a diff algorithm on this Virtual DOM(React, 2017). When the component wants to be re-rendered, the diff algorithm would find the Virtual DOM nodes that need to be updated, and then updates the changes to the actual DOM nodes, which means that there is no need to re-render the entire DOM tree(React, 2017). Therefore, the performance of Virtual DOM is better than real DOM.

In React applications, the way to render component from data to view layer is unidirectional, which means one-way data binding. The way of data flow is very suitable for the complex web applications.

5.2.1.2. Redux Framework Used in Web Application

To build a large web application, only have React.js framework to build web UI is not enough. Therefore, Redux is a good option, which is a front-end framework that can organize the code and arrange the web application's internal logic. Redux is a JavaScript state container that provides predictable state management. Redux allows the web application to run in different environments, which make the web app easier to test and improves coding efficiency(Redux, 2017).

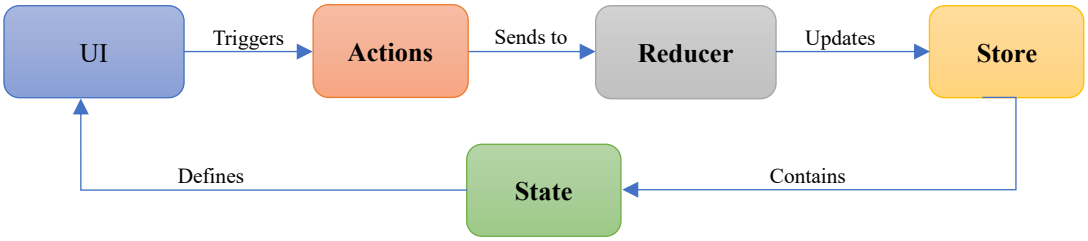


Figure 10 The Work Flow of the Redux

As the figure 10 shown, the data flow unidirectionally in Redux framework. After rendering of the web pages, the user interface would display on the screen. If the user clicks any button on the UI and then there have some actions would send to reducer method. The reducer method will update the states that contain in the store, and the state would define how to render the UI and the view layer.

Therefore, Redux can better manage and operate the states of an application component. Redux can query, change the state and announce the state changes in the same place. Thus, Redux can better manage the application state in a simple way.

5.2.1.3. React-Redux Framework

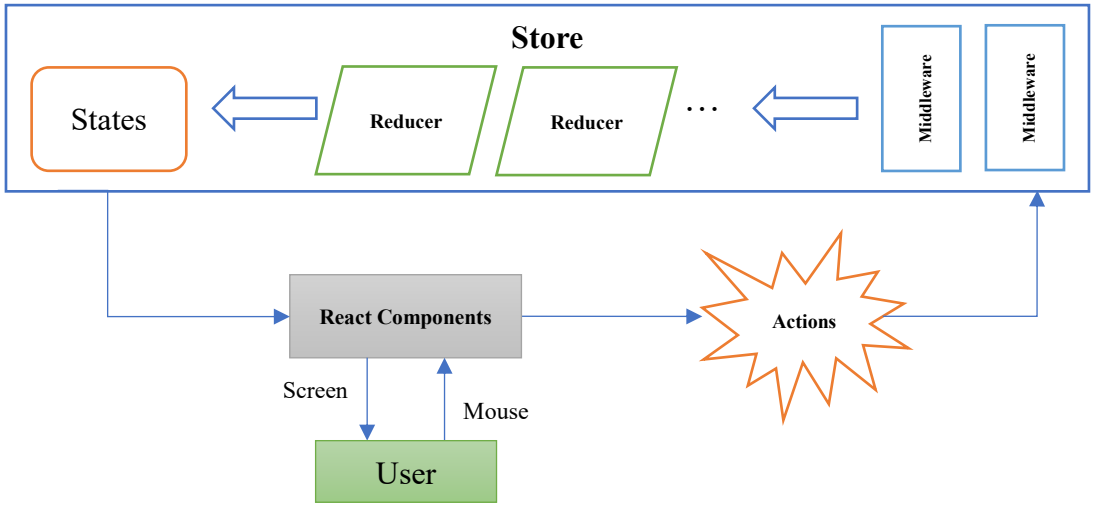


Figure 11 The work Flow of React-Redux Framework(Staltz, 2015)

As the figure 11 shown, how the web interacts with users when the React and Redux work together. When the users use the mouse to click a button on the view layer, there would generate an action if the react component has changed. When the actions flow into the store, one middleware only accepts one action at one time, which will transfer the actions to one reducer. Afterwards, reducers would change the states that save in the store and the states would work on the React components. After components have been changed, the UI will be re-rendered and then display on the screen for the user.

5.2.1.4. Ant Design UI Kit

Ant Design is a react-based UI and front-end development tool kit which is developed by Alibaba company(AntDesign, 2017). Ant Design makes the UI design easier and faster, and it also reduces many unnecessary design differences and costs during the development process. Therefore, that Ant Design can help to release front-end development resources(AntDesign, 2017). Hence, Ant Design has been used by many developers and also deployed in many companies' web application.

In addition, Ant Design also aims to improve the users, developers and other multi-party experience and happiness in using(AntDesign, 2017). Due to the exquisite design of Ant Design, it not only enhances users' experience but also helps the designers and developers to design the web UI in a simple way.

5.2.2. Other Methodologies

5.2.2.1. Model View Controller (MVC) Framework vs React-Redux Framework

Model-view-controller model (MVC) is a software architecture model. MVC model is one of the most common software architectures and is widely used in many types of applications, including web applications(Microsoft, 2017). In MVC model, the software is divided into three basic parts: Model, View and Controller.

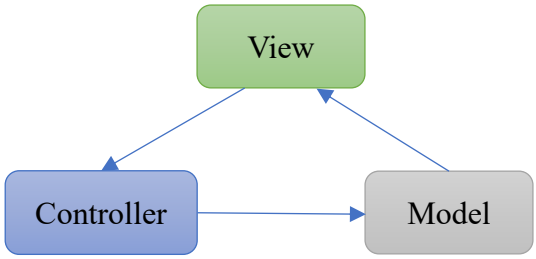


Figure 12 The Communication Pattern of MVC Framework(Microsoft, 2017)

As the figure 12 shown, the view layer provides a user interface (UI) for developers directly. Model layer is the most core part of the MVC model. In the model layer, it saves the data or information of an application. Besides, controller layer base on the user's input from the view layer to select data from the Model layer that correspond to the user's operation. Therefore, the model layer mainly controls software logic.

These three layers interact with other layers, but are independent of each other, which means that one layer has any changes and other layers would not be impacted. Every layer has interfaces to communicate, and exchange data with other layer and all communications are unidirectional. Therefore, using this model, the web app could be better maintained and upgraded.

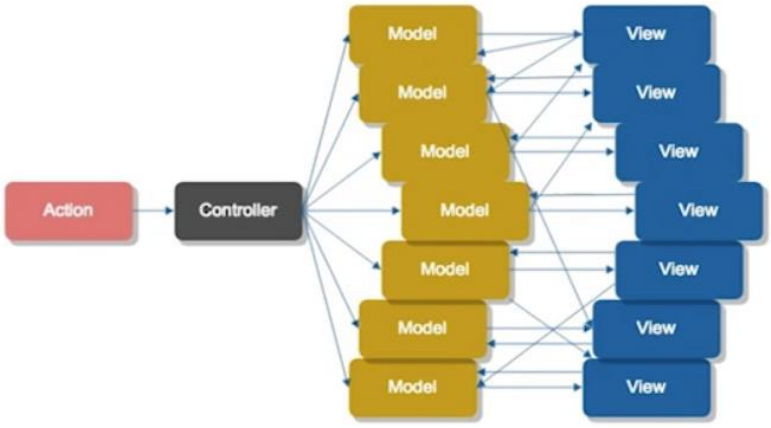


Figure 13 The Data Stream Structure of MVC Framework(Rey, 2016)

However, comparing with React-Redux framework, the MVC framework have to face some problems. If the size of the application is increasing, a series of problems would show up in the application. As shown in Figure 13, if the user submits some data on the website, then the model layer will be updated. After that, View layer will change as well. However, if there is any change in the view layer and the model layer will change with it. Therefore, the web interface will become unpredictable when the next user clicks the web site.

Therefore, using MVC architecture to develop web applications has increased the complexity of the application structure and implementation. In addition, since the view layer cannot directly receive the data, which have to go through controller layer, which will lead to deterioration of application performance.

In addition, due to the limitations of the MVC framework, if a developer needs to add a new view to the web application, the code of model and controller layers would be increased accordingly. Because this modification of MVC framework is top-down, hence the amount of code and developer's workload would increase dramatically.

5.2.2.2. Flux Framework VS Redux Framework

Flux can make the application easier to develop and maintain. In the flux framework, the data is one-way flow, and Flux divides the application into four parts: view layer, store layer, dispatcher and actions.

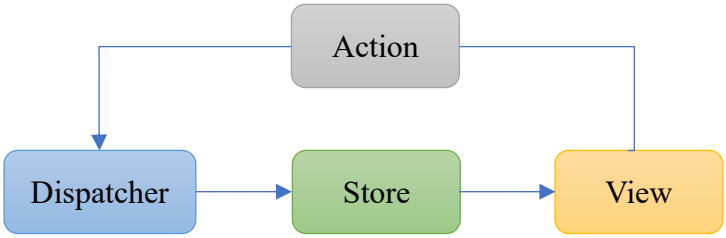


Figure 14 The Work Flow of Flux(Flux, 2017)

The work flow of the Flux is shown in Figure 14 if the view layer (react component) want to change any UI parts which will send a change action to the dispatcher, and the dispatcher will build a correct route for the change action(Flux, 2017). The store will save the status of the whole application. In the end, the view layer will receive a change event and then change the UI.

Comparing Flux and Redux framework, there are amount of differences. However, the Redux has more advantages.

Functionally, Redux has more functions than Flux, which includes logging, hot reloading, time travel, universal apps, record, and replay functions(Redux, 2017).

In addition, Flux does not give detailed explanations for these common problems such as asynchronism and isomorphism aspects. Besides, the store and view in Flux have too much duplicate code. In contrast, Redux removes duplicate code and optimizes Store, Action to make them more flexible.

Furthermore, the Flux complexity is higher, and the code is not easy to read. In contrast, Redux complexity is moderate and easy to understand relatively. Hence, comparing with Flux, Redux is easier for beginner to learn and master, and that is the reason why we choose redux rather than flux in this project.

5.3. Mobile Application

5.3.1. Functional overview

Since the mobile app should work together with the backend server and show the real-time corresponding data from the database, what is followed is the structure of the mobile app from the overview perspective:

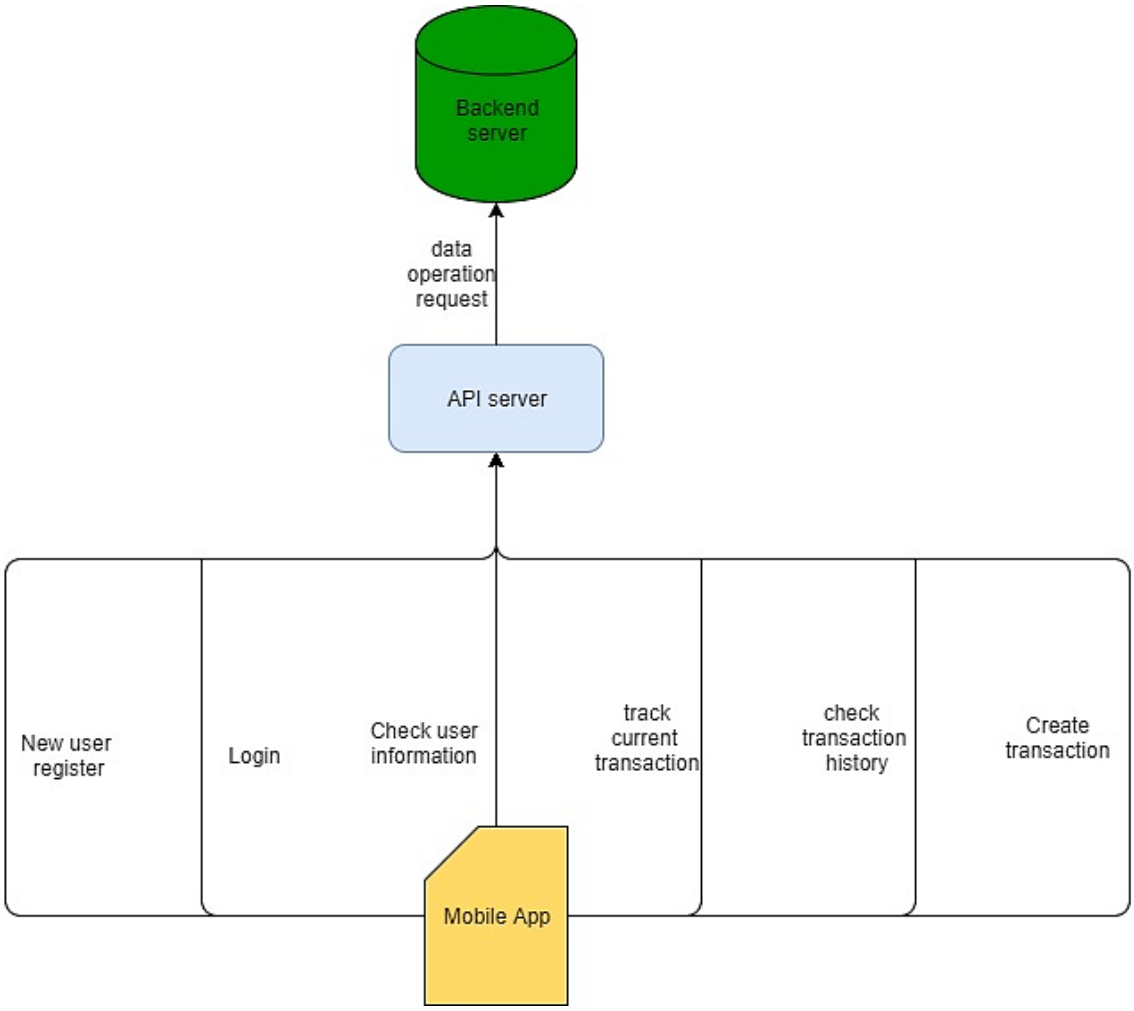


Figure 15 The Overview Perspective of Mobile App Structure

From the figure 15, we can observe that the mobile app is working with the support of API server and the backend server and provides six different functions, which will be demonstrated in more details in the demonstration section. In this mechanism, the mobile app needs to send the request to the API server whether it needs to store new data to the backend or to fetch some corresponding data from the backend. The API will handle the different request and deal with different conditions and return the result to the mobile app.

Besides the overall structure of the mobile app and the whole system, the structure which indicates how the mobile components and functions work together is showed as follow:

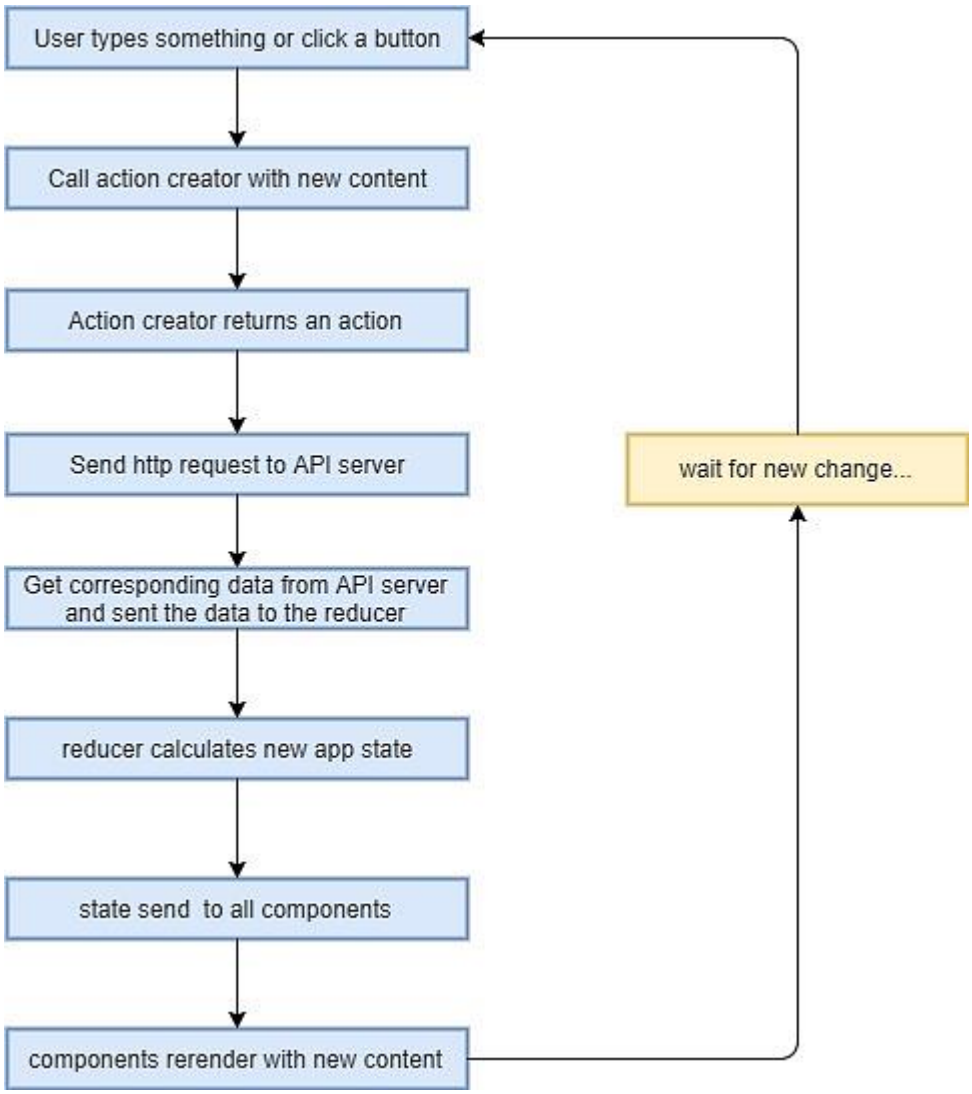


Figure 16 The Work Principle of Mobile Components and Functions

In the flowchart (figure 16), we can find that the action is triggered by the user when the user types some new content or clicks a button in the app. The data will be transported to the action creator where the corresponding action will be called. The returned data will be proceeded to the related reducers once the action is completed, after which the mobile app state will update according to the new content of different components. Finally, the app will re-render the components and show the latest data on the screen.

Apart from the overview structure of the working mechanism of the mobile app and the backend server, the app framework, as well as some related information will be instructed next.

5.3.2. React Native:

Prior to start programming, the framework that is going to applied must be determined firstly. React native is selected to be used in the development of the mobile app mainly because its advantages compared to other languages.

5.3.2.1. Community Driven

The React Native is created by Facebook and supported and sustainably pushed forward by the development community which is composed of a large number of JS and native developers (React-Native, 2017). In this community, the corresponding developers share the components, expertise and the related knowledge with the users. This can ensure the rapid evolvement of this framework.

In comparison, Vue is a framework realized and maintained by a group of individuals sponsored by some investors. Which means the React Native has a competitive advantage than Vue when it comes to the stability, sustainability, and continuity since React Native is empowered by Facebook, a leading technology company in the world, which can guarantee enough efforts and investments for the development and maintenance of React Native. An available evidence from Github is that React has more than one thousand contributors while the Vue only has 120 contributors (Medium, 2017).

5.3.2.2. Code Reuse and Cost Saving

With React Native, the code can be used both on iOS and Android, the leading two mobile app developing platforms (React-Native, 2017). Undoubtedly, this is an outstanding advantage of React Native to attract more companies and teams to adopt it since the reuse between iOS and Android means the time and financial contribution decline when developing a new product.

Ionic, another popular mobile app framework, enables the ability that writing once and running everywhere because principally the app developed with Ionic is a hybrid app

which means a web app in the shell of the mobile app (Codementor.io, 2017). However, the app realized using React Native experiences more like a native app. Meanwhile, React Native enables the ability that learning once and writing everywhere which implies that most components or code can be used both on IOS and Android. This can help the programmers save time and improve efficiency.

5.3.2.3. Live Reload

Live reload is another interesting and attractive feature of React Native has (React-Native, 2017). With the advanced live reload concept, you can do something that you can only imagine in the native frameworks, that is, you can immediately see the change you make to your code. This means the efficiency of your work can be enhanced obviously as you can see what your change is rather than waiting for the compile and other processes.

To be compared, Ionic is also excellent at testing during development processes. The Ionic apps can be previewed on both website and mobile devices (Codementor.io, 2017). This means the programmer can review the result of the modification very soon after the modification is done. The app enabled with React Native can only be previewed on mobile devices, however, the preview of React Native apps is more instant and convenient. Once the coder has done the modification at the source end, the content will update on the screen instantly according to the changes.

5.3.3. Redux Framework Used in Mobile Application

Redux is a state container. This means the data of the state related to the whole app is stored with Redux in the form of reducers (Redux.js.org, 2017).

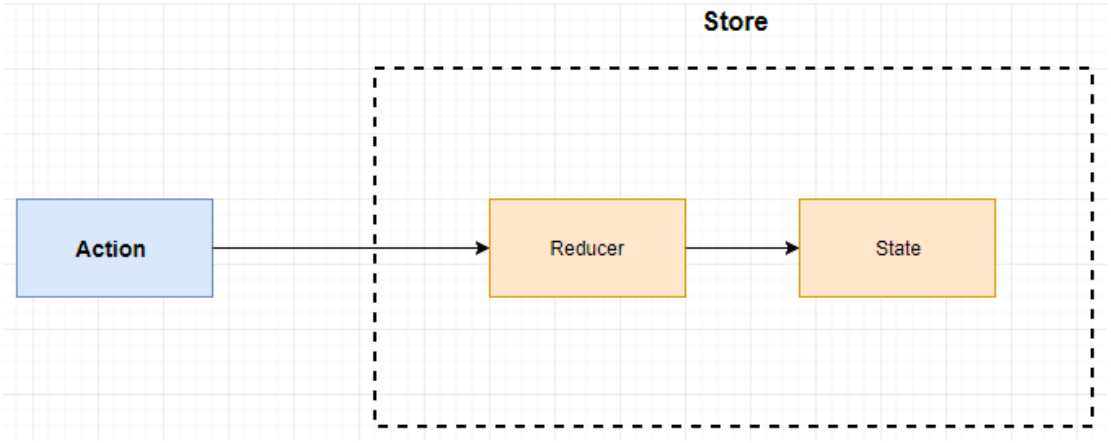


figure 17 Work Principle of Redux

The working principle of redux is simple. When the user calls an action through the action creator, the action will be conducted and return data to the reducer, where the state data is stored. After that, the state will be updated according to the returned action type and the returned data. Afterwards, the new contend of the state will be shown on the screen.

5.3.4. NativeBase

As an open and free UI component library, the NativeBase is developed to assist the React Native app development (Docs.nativebase.io, 2017). Since there is no available native component library in React Native, the coder needs to define some necessary components when developing a new mobile app, such as button or textbox. With the help of NativeBase, the coder can use the available components from NativeBase instead of defining by themselves, which means the coder can save time to do more complex and core work. In our project, we decided to apply some components from NativeBase to boost our efficiency.

6. Implemented Strategy

6.1. Time Plan

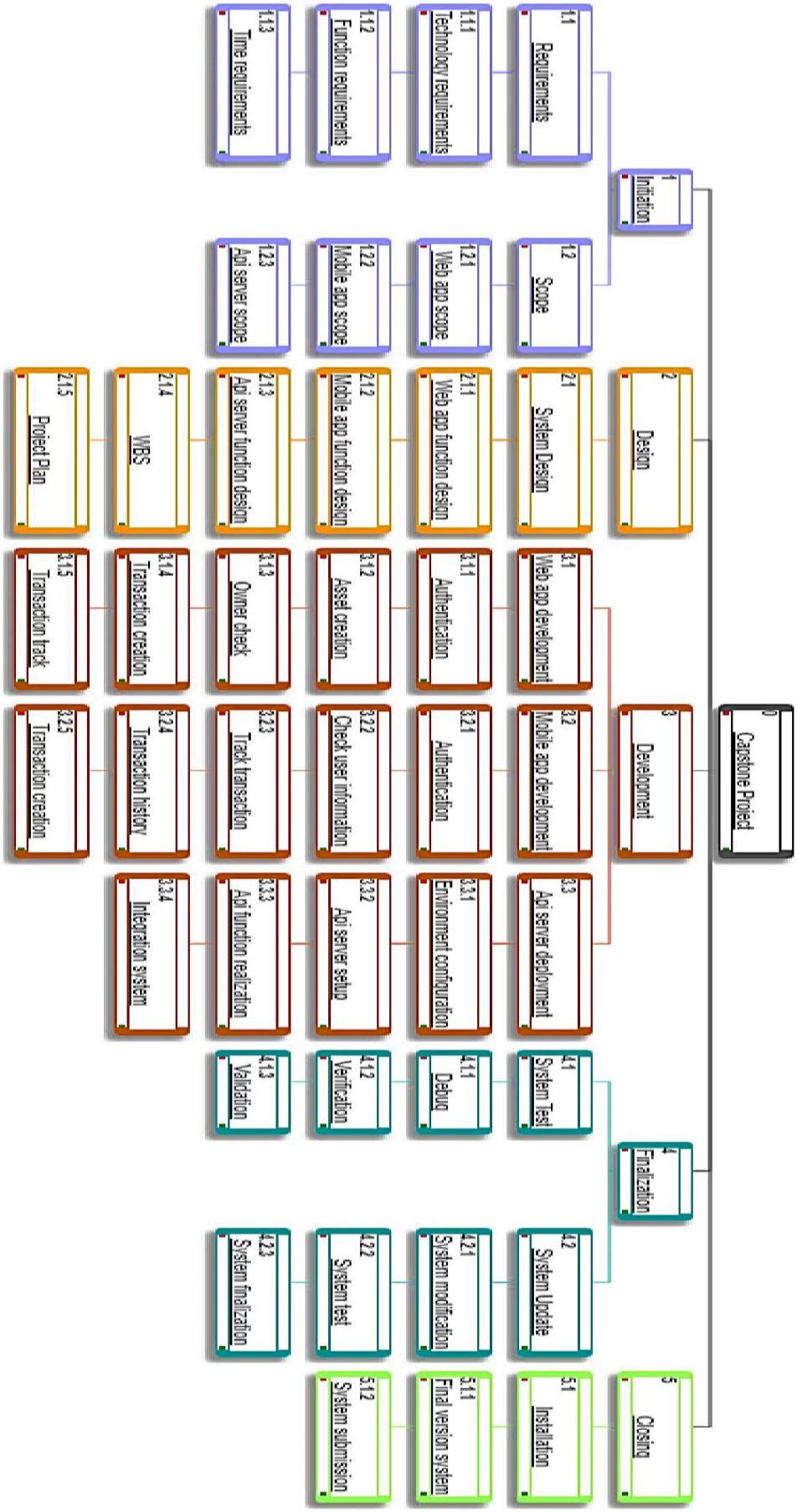
To demonstrate the timeline and monitor the continuous progress status, a detailed timeline is constructed and the tasks have been defined and assigned to different students.

Week No	Task	Students In Charge
Week 1	1. Determined objectives and goals of the project 2. Find out what should we learn 3. Learn something about react	Yuming Jiang Xinyi Liu Zhengyang Liu
Week 2	1. Learning Redux concepts and models 2. Learn React-redux framework	Yuming Jiang Xinyi Liu Zhengyang Liu
	3. Background research on the existing system	Yuming Jiang
	4. Literature review on related fields	Xinyi Liu Zhengyang Liu
Week 3	1. Function designing	Yuming Jiang; Xinyi Liu; Zhengyang Liu
	1.1 Web app function design	Xinyi Liu
	1.2 Mobile app function design	Zhengyang Liu
	1.3 API_Server design and construction	Yuming Jiang
	2. Drafted proposal presentation	Yuming Jiang; Xinyi Liu; Zhengyang Liu
Week 4	1. Literature review for proposal and proposal draft 2. Component design and confirmation	Yuming Jiang Xinyi Liu Zhengyang Liu
Week 5	**1. Complete and submit proposal** 2. React-native study	Yuming Jiang; Xinyi Liu; Zhengyang Liu
Week 6	1. Blockchain-system review 2. Basic structure for API_Server	Yuming Jiang
	3. Configure the environment for React-Native and build the basic framework for mobile app	Zhengyang Liu
	4. Build the basic structure of the web app	Xinyi Liu
Week 7	1. Executable basic web app	Xinyi Liu
	2. Executable basic mobile app	Zhengyang Liu; Yuming Jiang
Week 8	1. First version of web app	Xinyi Liu
	2. First version of mobile app	Zhengyang Liu; Yuming Jiang
	3. Well-function API_Server and BigChainDB	Yuming Jiang
	4. Test and integrate the whole basic system	Yuming Jiang; Xinyi Liu;

		Zhengyang Liu
Week 9	1. Fix bug of the basic system	Yuming Jiang; Xinyi Liu; Zhengyang Liu
	2. Add new features to the basic system 2.1 Add new features to the web app	Xinyi Liu
	2.2 Add new features to the mobile app	Zhengyang Liu
	2.3 Update the API_Server and adapter	Yuming Jiang
	3. Complete and submit Progress Report	Yuming Jiang; Xinyi Liu; Zhengyang Liu
Week 10	1. Add new features to the basic system 2. Do some research and system review on UI design 3. Confirm the basic UI design of the system	Yuming Jiang Xinyi Liu Zhengyang Liu
Week 11	1. Second version system 2. Test and debug for the final version 3. Prepare for the presentation	Yuming Jiang Xinyi Liu Zhengyang Liu
Week 12	1. Complete the presentation	Yuming Jiang; Xinyi Liu; Zhengyang Liu
	2. UI design 2.1 Web app UI design	Xinyi Liu
	2.2 Mobile app UI design	Zhengyang Liu; Yuming Jiang
Week 13	1. Integrate and implement all the parts of the system 2. Whole system update and optimization	Yuming Jiang; Xinyi Liu; Zhengyang Liu
Week 14	**1. Final version system** 2. Drafted the final report	Yuming Jiang; Xinyi Liu; Zhengyang Liu
Week 15	**Final report submission**	Yuming Jiang; Xinyi Liu; Zhengyang Liu

In the project, five milestones have been determined, and they are marked with yellow color: complete and submit proposal, test and integrate the whole basic system, complete and submit the progress report, final version system, and final report submission.

6.2. Work Breakdown Structure (WBS)



7. System Specification

7.1. Architecture

7.1.1. System Overall Structure

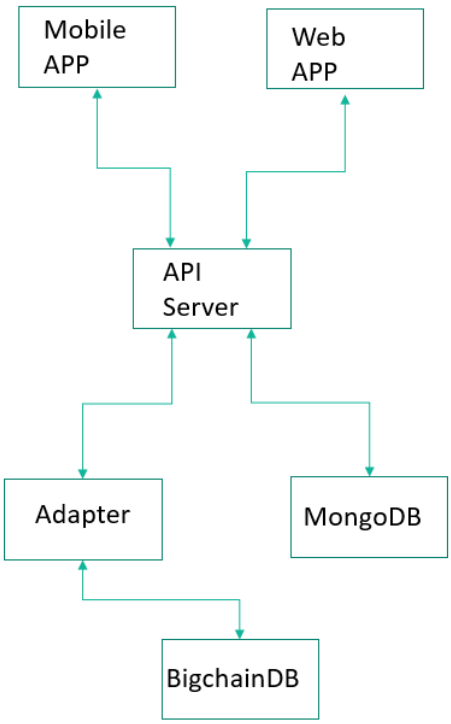


Figure 18 System Structure

As shown in figure 18, the whole system consists of three main parts, Mobile Application, Web Application and API Server. Users will use Mobile Application and Web Application to call the API provided by API Server. API Server implements the functions and solutions that are described in the objective section with the help of MongoDB and BigchainDB.

```
{
  "api": {
    "v1": {
      "assets": "/api/v1/assets/",
      "docs": "https://docs.bigchaindb.com/projects/server/en/v1.1.0/http-client-server-api.html",
      "outputs": "/api/v1/outputs/",
      "statuses": "/api/v1/statuses/",
      "streams": "ws://localhost:9985/api/v1/streams/valid_transactions",
      "transactions": "/api/v1/transactions/"
    }
  },
  "docs": "https://docs.bigchaindb.com/projects/server/en/v1.1.0/",
  "keyring": [],
  "public_key": "CkCoQrxsD4YhNTAV6pDBoWvb24qZDMsDXfsVwDRryWSe",
  "software": "BigchainDB",
  "version": "1.1.0"
}
```


Figure19 BigchainDB Homepage

As shown in the Figure19, the homepage of the blockchain system that we used to demonstrate our system can be visited with the URL “http://188.166.250.43:9984/”.

API_SERVER

COMP5703_BLOCKCHAIN_IOT_GROUP_3

Method	URL	Function
get	/	Show homepage.
get	/user	Show user profile.
post	/user/signup	Sign up with email and password.
post	/user/login	Login with email and password.
post	/blockchain/create_new_asset	Create new asset.
post	/blockchain/transfer_asset	Transfer the asset from the seller to the buyer.
post	/blockchain/transaction_check	Check if the user is the owner of the transaction.

Figure 20 Homepage of API Server

As shown in figure 20, our API Server’s homepage can be visited with the URL “http://188.166.250.43:3000/”.

7.1.2. API Server

7.1.2.1. Overall Structure

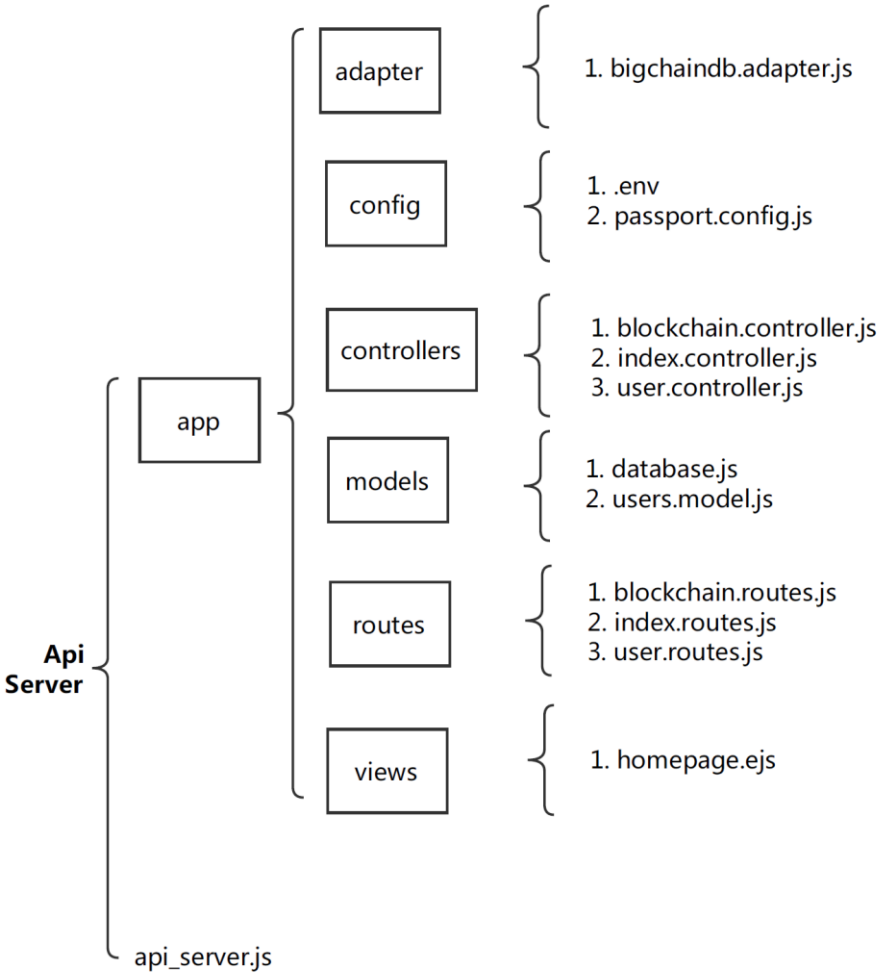


Figure 21 API Server Overall Structure

As shown in figure 21, API Server consists of "app" folder and a JS file called "api_server.js". The overall structure of API Server follows the Model–view–controller (MVC) structure, which means the modification of the function and the future development are very easy.

7.1.2.2. "api_server.js" File

This JS file is the starting point of the whole API Server. The port number used by API Server is defined here. Some middleware packages such as "dotenv" and "passport" are initialized here. The administrator should start the server with this file.

7.1.2.3. "app" Folder

The "app" folder consists of six other folders, "adapter", "config", "controllers", "models", "routes" and "views".

The "adapter" folder contains the JS file "bigchaindb.adapter.js". This file implements the functions related to the communication between API Server and BigchainDB. There are four major functions within this file. "get_key_pair" return a random key pair. "create_new_asset" can add new asset into BigchainDB. "transfer_asset" can add transfer transaction into BigchainDB. "transaction_check" can check the owner of the asset or the authenticity of the transaction.

The "config" folder contains the file ".env" and the JS file "passport.config.js". The file ".env" keeps all the runtime parameters that will be used by "dotenv" package, for example, the port number and the URL of the MongoDB database. The file "passport.config.js" implements the functions related to the user authentication with the help of package "passport". This API Server uses the JSON Web Token standard to dispatch access tokens for Web Application and Mobile Application.

The "controllers" folder contains the JS file "blockchain.controller.js", "index.controller.js" and "users.controller.js". The file "index.controller.js" handles the event to show the homepage. The file "users.controller.js" handles the events related to MongoDB. There are three major handling functions, "post_login", "post_signup" and "show_user_profile" within this file. The file "blockchain.controller.js" handles the events related to BigchianDB. There are three major handling functions, "create_new_asset", "transfer_asset" and "transaction_check" within this file.

The "models" folder contains the JS file "database.js" and "users.model.js". The file "database.js" handles the connection between API Server and MongoDB. The file "users.model.js" implements the functions related to the communication between API Server and MongoDB. There are three major functions in "users.model.js". "user_model_login_check" can check if the inputted user name and password are correct. "user_model_signup" can create a new user and save it into MongoDB.

"user_model_passport_jwt_strategy" support the "passport" strategy construction.

The "routes" folder contains the JS file "blockchain.routes.js", "index.routes.js" and "users. routes.js". The file "blockchain.routes.js" handles the URL related to blockchain API. The file "index.routes.js" handles the URL related to the homepage. The file "user.routes.js" handles the URL related to user authentication spi.

The "views" folder contains the file "homepage.ejs". The file "homepage.ejs" constructs API Server's homepage that is shown to the users.

7.1.3. Web Application

7.1.3.1. Overall Structure

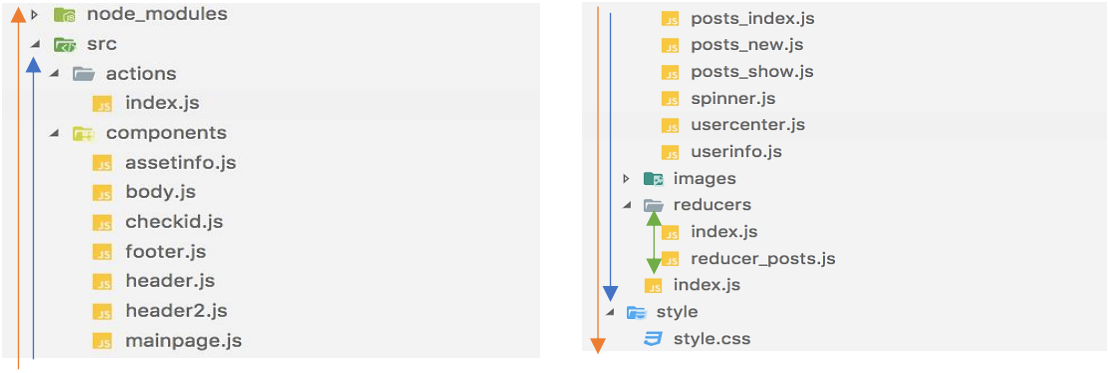


Figure 22 The Overall Structure of Web Application

As the figure 22 shown, the web application structure mainly includes six folders, which play different roles in this application.

7.1.3.2. 'node_modules' Folder

Node.js is a necessary engine when the JavaScript is running, which is an efficient, lightweight event driver and a non-blocking I / O model. Therefore, our web app needs node.js. Besides, the npm is also needed in this application, which is a package management tool and will be installed with Node.js. The npm can solve many problems when the Node.js code is deploying. Hence, the node_modules folder contains all the tools that are installed using the npm command, which includes the React.js library and Ant designs UI kit that is used in our web applications.

7.1.3.3. ‘actions’ Folder

Before to explain what does this folder includes, there has some concept need to be solved.

Due to the Redux is a library of data storage, hence action is the payload that transfers the data from the application to the store, and the action is the only source that passes the data to data. Thereinto, store is used to maintain the state of the application and provide some methods to get and update the state. Therefore, the ‘index.js’ file in action folder stores the methods to connect the API server with the web application.

7.1.3.4. ‘components’ Folder

As mentioned before, the React application mainly reply on components, hence all the React components are contained in this folder. Firstly, the ‘header.js’, ‘header2.js’ and ‘footer.js’ include the logo, the login/register function and the copy right information. Hence if other components want to add header or footer, there is no need to write again and only need to import header.js or footer.js to the component, which can help the developer to reduce the workload.

Secondly, the ‘posts_index.js’, ‘posts_new.js’ and ‘posts_show.js’ achieve the ‘add new product information’ function. Therefore, the code to achieve this function is written in these three js files.

Thirdly, the ‘assetinfo.js’ and ‘userinfo.js’ files contain the code of achieving the ‘create new asset’ function, new asset table and user information table separately. The ‘check owner of transaction ID’ function is achieved by ‘checkid.js’ file that is imported by ‘userinfor.js’ file.

Finally, the ‘usercenter.js’ file import ‘posts_index.js’, ‘assetinfo.js’, ‘userinfo.js’ and ‘header.js’ ‘footer.js’ to consist of the user center that present for users.

7.1.3.5. ‘images’ Folder

This folder contains the advertisement materials and the logo that used in the header of the web page. Therefore, if other components need to use these material or logo file, the component needs to call the image file from this folder and the path should be ‘./src/images/image name.png’.

7.1.3.6. ‘reducers’ Folder

As mentioned earlier, the action simply states that the state needs to update, but the action does not indicate how the application should update state. Therefore, reducer can help the application to update the state.

These two files in the reducers folder mainly explain how to update the state. The 'reducer_posts.js' file is detected the type of the operation that happened in the application, such as deleting and adding product information that would return different values to state. The ‘index.js’ file in this folder declare the posts method that from ‘reducer_posts.js’ file and also declare the redux form that used in the ‘posts_new.js’ file.

7.1.3.7. ‘style’ Folder

This folder contains a style.css file that mainly to work on the UI of the web page. Almost all the UI adjustment code is written in this CSS file. Therefore, in this web application, the position of advertisement card, the size of the logo and this kind of styling work all should be declared in this style.css file.

7.1.4. Mobile Application

The mobile app consists of six different main functions: signup, login, track transition, check history, check user profile, and create transactions. They worked consistently as showed in the next figure:

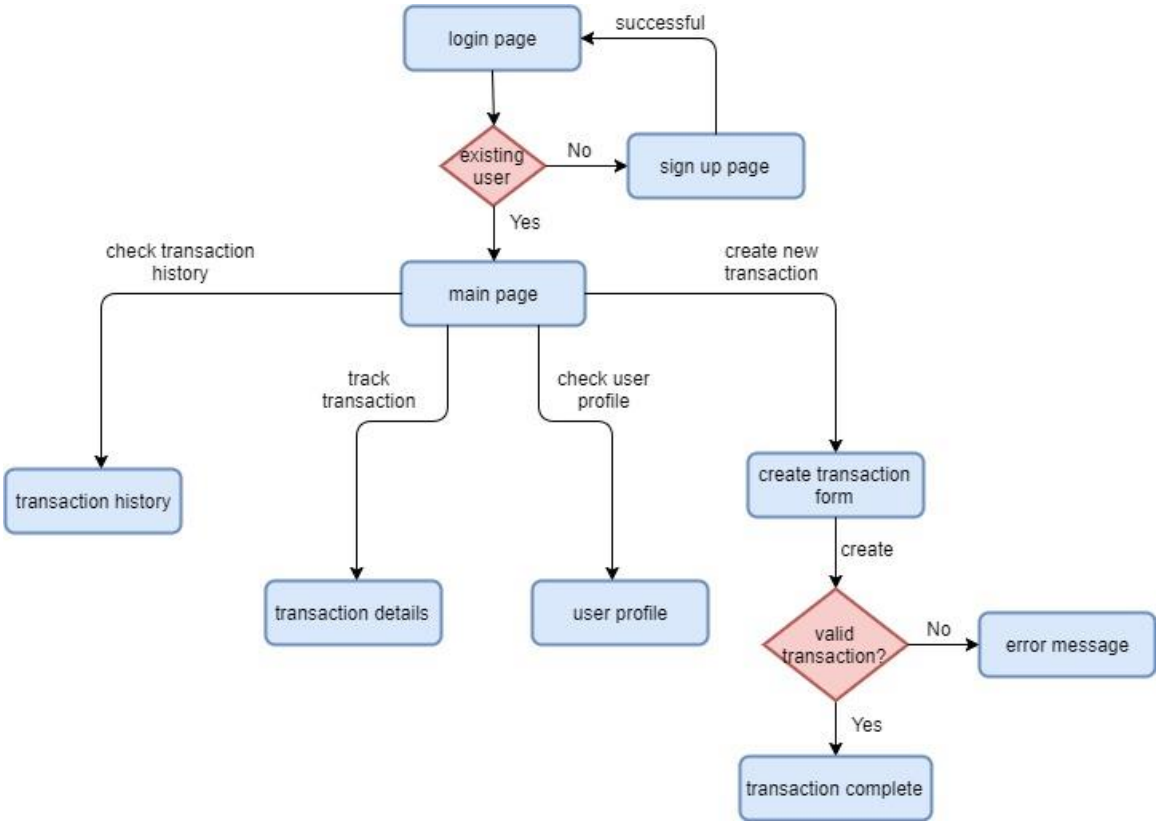


Figure 23 The Work Principle and Interaction of Mobile App Functions

As shown in figure 23 above, the existing user can log into the system with their account number and password. Generally, the account is a unique email address, and the password is defined by the users themselves. If the user is new to the system, a new account needs to be created, with which the user can log into the system.

From the main page of the mobile app, there are different buttons related to different functions and can trigger different pages. When clicking the “history”, the transaction history will be showed on screen. Clicking the “track” on the current transaction will invoke the details of the current transaction. Clicking the “create” button will call the new form to create a new transaction, and the “user” button will return the user profile including the private key, the public key which is used for a transaction, and other related information of the current user.

In the new transaction creation form, the mandatory fields are the transaction id and the public key of the buyer. The transaction complete page will return once the transaction completes successfully, while the error message will be showed om the screen if the transaction is denied by the system, whether the transaction id is incorrect meaning the

product is counterfeit or the transaction id has been used meaning the pharmaceutical is fake.

7.2. Functionality

7.2.1. API Server

7.2.1.1. Provided API

As shown in figure 20, there are mainly 6 APIs provided by API Server. They can be called by the Web Application and Mobile Application to obtain the objective of this project.

7.2.1.2. User Account

There are 3 APIs provided to implement functions related to user accounts. The first API can be called with the URL "/user/signup" with POST method. The request to this API should contain one JSON object which consists of the new user name and password. If the new user account is valid, it will be stored in MongoDB. API server will send back the success message as the response.

The second API can be called with the URL "/user/login" with POST method. The request to this API should contain one JSON object which consists of user name and password. If the user name and password match the pair stored in MongoDB, API Server will send back a JSON Web Token as the response. The token contains information related to the user account and will play the role as a key for the users to access other APIs.

The third API can be called with the URL "/user" with GET method. The request to this API should contain the token got from "login" API. API Server will send back the user account's detail as the response.

7.2.1.3. Drug Producer

The authenticated drug producer can put the newly produced drug into the market by calling API with the URL "/blockchain/create_new_asset" with POST method. The request to this API should contain the token got from "login" API. API Server will send back the transaction ID as the response. The authenticated drug producer can attach this transaction ID to the drug and sell this drug to other people.

7.2.1.4. Drug selling

The drug seller can sell the drug by calling API with the URL "/blockchain/transfer_asset" with POST method. The request to this API should contain the token got from "login" API, the old transaction ID, buyer's public key and the metadata such as the price of the drug. API server will make the transfer transaction, sign this transaction with the drug seller's private key and push this signed transfer transaction to BigchainDB. If the drug is real and the transaction is valid, API Server will send back the new transaction ID as the response. The drug seller can attach this new transaction ID to the drug and sell the drug to the buyer.

7.2.1.5. Drug buying

When the drug buyer receives the drug and transaction ID from the seller, the buyer can call the API with the URL "/blockchain/transaction_check" with POST method. The request to this API should contain the token got from "login" API and the transaction ID attached to the drug. API Server will check if the transaction ID is valid. If the transaction ID is valid, API Server will send back the message about the owner of the drug as the response to the buyer.

7.2.2. Web Application

Figure 24 The Main Page of Web Application

The figure 24 shows that the main page of our web application. This page can show some information and advertisements of TBSx3 company and users also can login their account in this page. The left part of the web page body is a corporate advertising display card, which would show advertising pictures in the carousel style. Moreover, the right part is a bulletin board that shows the introduction or the system update information of the medicine tracking system. The login and register button is on the right side of the web page header, and the user can click this button to log in their account.

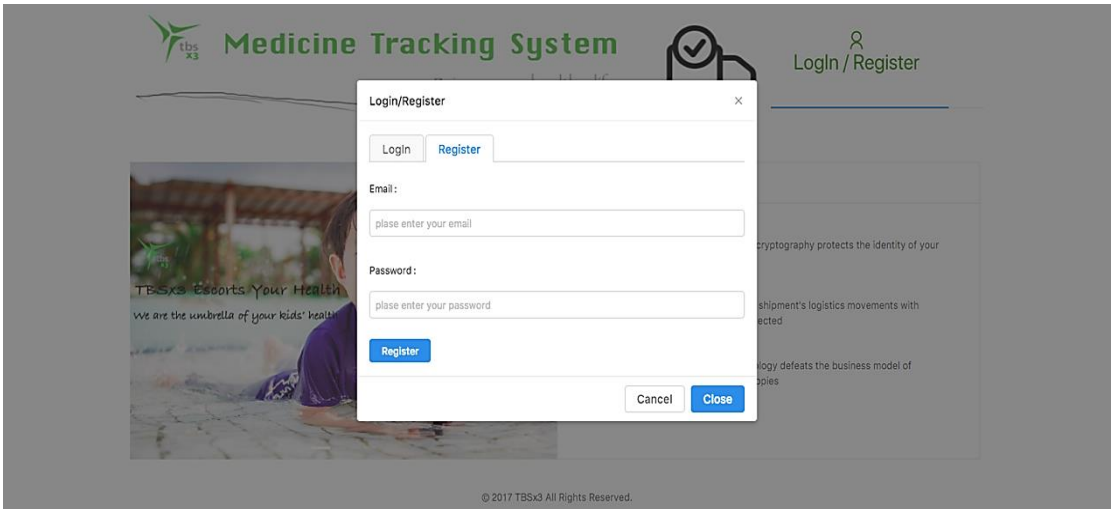


Figure 25 The Register Page of Web Application

As the figure 25 shown, when the user clicked the ‘Login/Register’ button, the login

and register page will show up. And if the user enters a valid register information in the register table and a ‘signup successfully’ message would show up (figure 26 left). However, if the user enters a user name that has already exist in the database, then the warning error (figure 26 right) will show up to remind the user to choose another email or username to register the account.

Figure 26 The Register Message (left is success message and right is warning message)

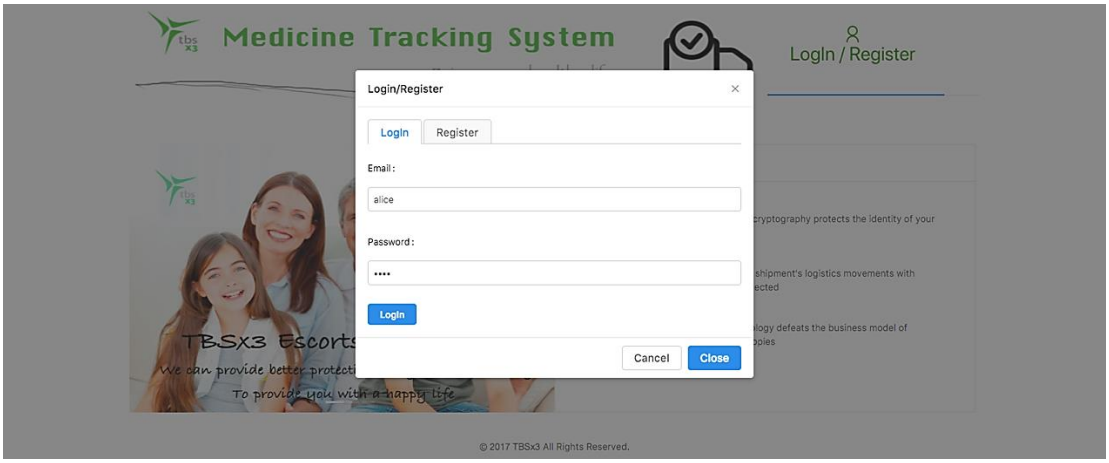


Figure 27 The Login Page of Web Application

After register a user account, the user has to login the existing account to check the account information. If user enters wrong email or password, an error massage will show up to remind the user to check the email or password. On the contrary, if the user enters the correct account information and a ‘login successfully’ message will show up, which is similar to register message. As the figure 27 shown, the user has logged in a manufacturer account, which has more permission than the normal user account.

Figure 28 The Web Page When the User Logged in Successfully

As shown in the figure 28, when the user has logged in the account, the username and the entrance of the user center will display in the right corner of the web page. Besides, the user also can logout the account on this page.

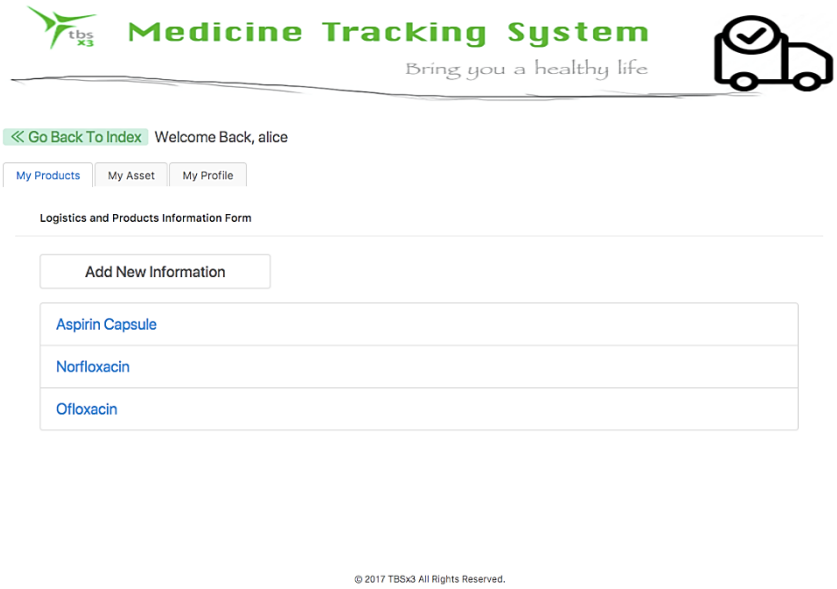


Figure 29 The User Center of the Web Application

In the user center, the user can check and manage the products information and check the owner of the transaction ID and the account information. However, only the manufacturer account can create an asset, and the normal account have no right to add new asset to the blockchain. As the figure 29 shown, in ‘My Product’ section, user can check the product information list and also can add a new product information if the user clicks the ‘Add New Information’ button on top of the list.

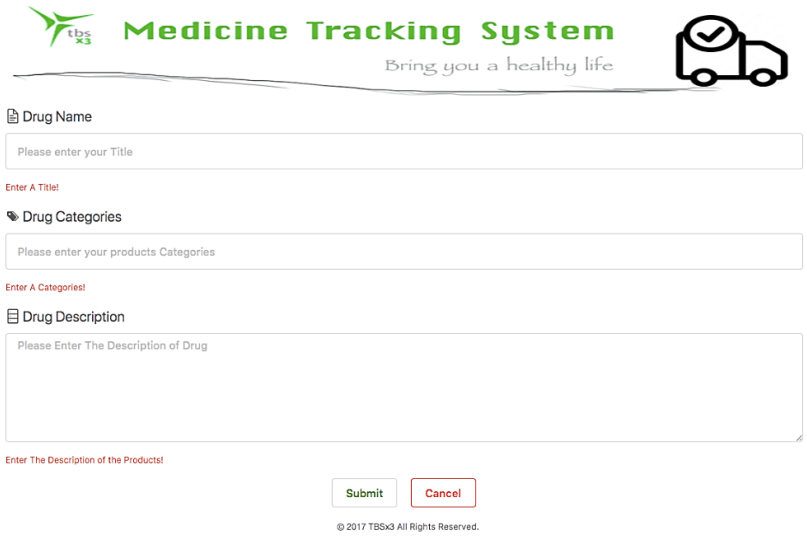


Figure 30 The Page of Adding New Product Information

In the figure 30, if user wants to create new product information, the user has to fill the table with the medicine name, category and description, which have to be filled. If these information does not be filled, the error messages will show up to remind the user to fill up. Only when the user fills all the information about the medicine in the table, they can submit the information table to the database.

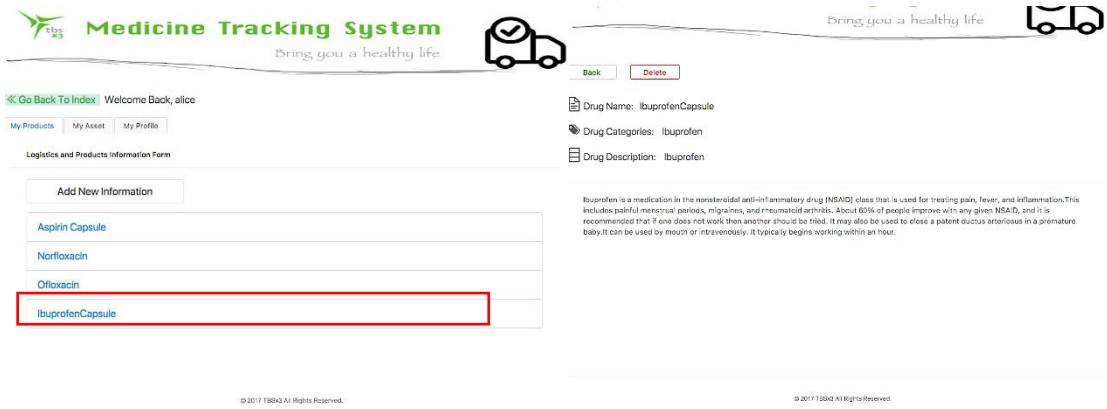


Figure 31 The Page of Adding New Product Information

After the user adds new product information, the new product information will show on the list like right part of figure 31. Besides, the user also can click the list to check the detailed information of the product, which is showed as the figure31 left part. Moreover, user also can delete the product information in the information display page,

and the information record will disappear in the list as well, hence the user does not need to delete the information record from the list.

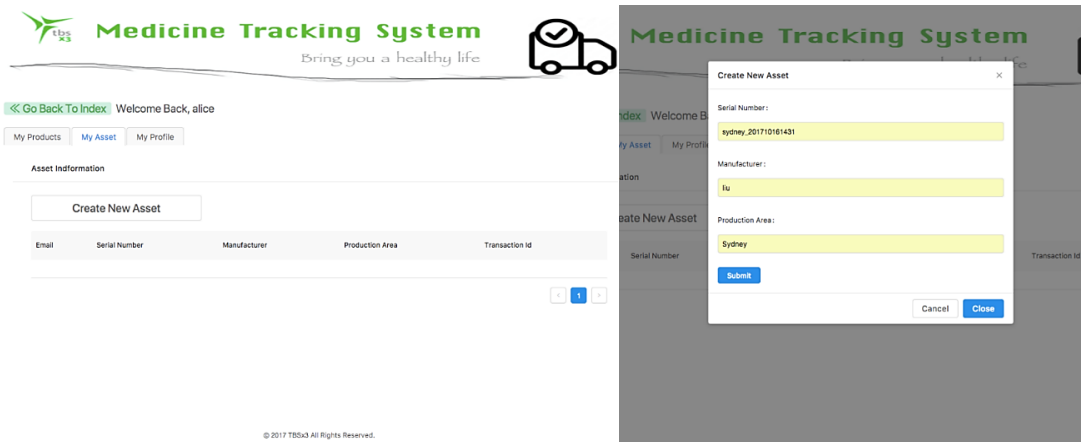


Figure 32 The 'My Asset' Page of the Web Application

In the user center, the user also can add the new asset in the blockchain. However, only the user who has manufacturer account have the right to create a new asset. As shown in figure 32, after the user clicks the 'Create New Asset' button, the new asset table will show up. User need provide the serial number, manufacturer and production area of the medicine.

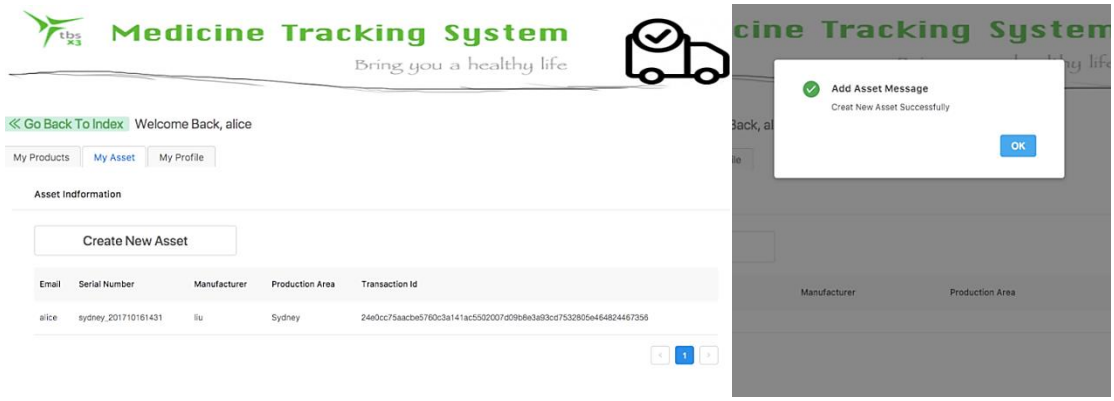


Figure 33 The New Asset Page of The Manufacturer Account

When the user creates a new asset, and submit the asset information table to the blockchain, a successful message will show up, and the asset information will display in the table in the 'My Asset' section, which is shown in figure 33. Besides, the

transaction ID will also show in the table, which would be used in every product transactions. In every transaction, the owner of the transaction ID needs to transfer to the new owner of the product at the same time. Therefore, this transaction ID is very important and guarantee the unduplicatedness of each transaction ID.

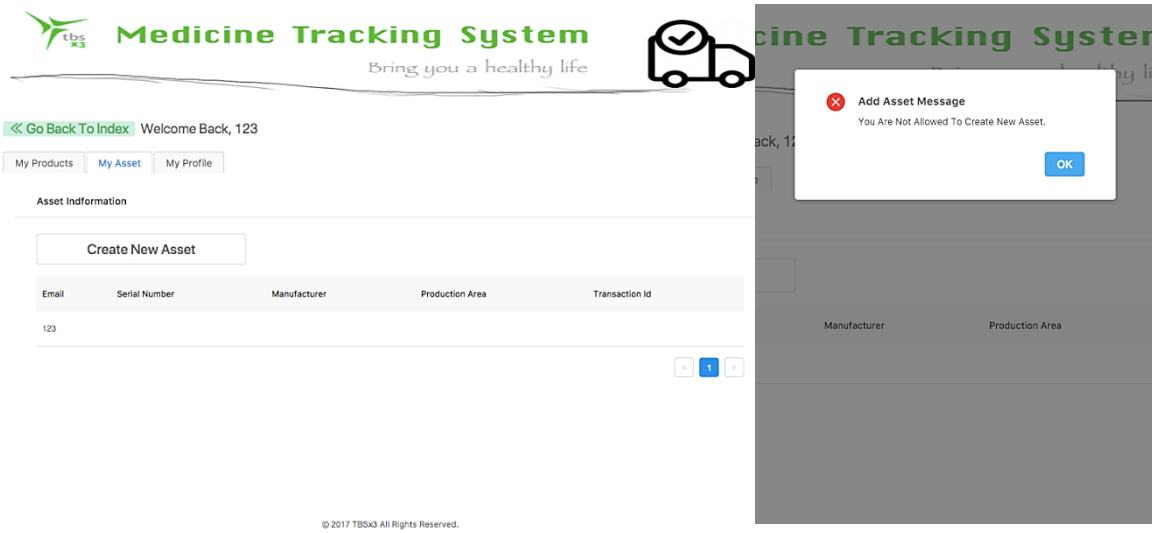


Figure 34 The New Asset Page of The Normal Account

Figure 34 shows the new asset page of a normal account. As mentioned before, the normal account cannot create the new asset, and only manufacturer account has the authority of adding a new asset. If a no-manufacturer account user wants to add a new asset and submit the asset information table to the blockchain, then the API server will return a message that shows the user is not allowed to create a new asset. Therefore, the transaction ID cannot be created by other users, and the buyer and seller in the supply chain only can use the valid transaction ID to conduct each transaction.

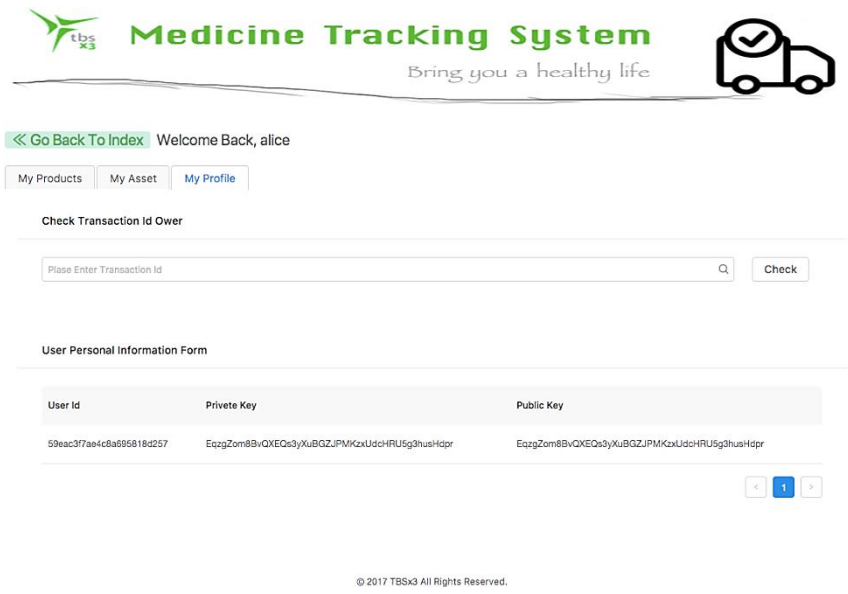


Figure 35 The 'My Profile' Page of Web Application

In order to guarantee that the user can hold the valid transaction ID in every transaction, the user can check the owner of transaction ID, and if the transaction ID that they have is existed or not. As the figure 35 shown, in 'My Profile' section, the user can check their account information, includes user ID, private key and public key. Thereinto, the private key and public key are necessary information that needs to be provided during the transaction. Besides, the user can check the transaction ID that they have in the search bar above the user information table.

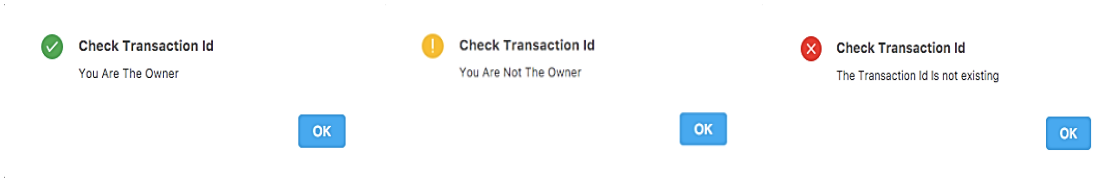


Figure 36 The Reminder Message of Transaction ID

If the user enters the transaction ID that belongs to the user, the successful message will show up on the web page that shows in the left part of figure 36. However, if the user enters the transaction ID that does not belong to the user, the warning message will show up on the web page to remind the user, which means that the transaction is not successful. Besides, if the user enters the transaction ID that does not exist, an error

message will show up in the web page to remind the user that the transaction ID is a fake one or the user make a mistake when they enter the transaction ID.

7.2.3. Mobile Application

7.2.3.1. Login & Signup

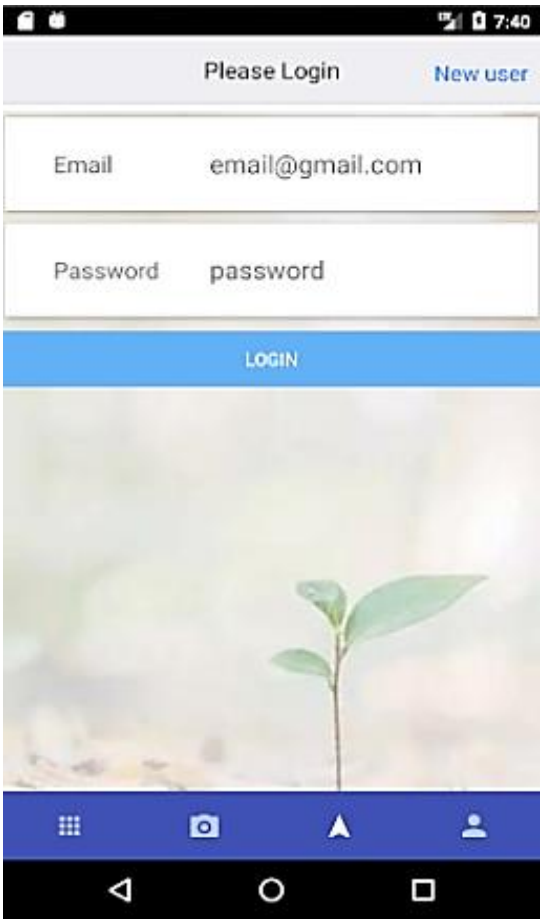


Figure 37 The Login Function of Mobile Application

The existing user can log into the system using the email and the password. The value pair is stored in the blockchain system. The login function will be called once the user clicks the login button, which is realized with the code.

```
onButtonPress() {
    const { email, password } = this.props;

    this.props.loginUser({ email, password });
}
```

The loginUser function is imported from the actions folder, where all the action

skeletons are stored. Since the login function only needs the correct email and the password value pair to process authorization in the backend server, the value pair will be passed to the action with the loginUser function once the user clicks the login button. Afterward, the loginUser function will be executed as the code showed below.

```
export const loginUser = ({ email, password }) => {
    return (dispatch) => {
        dispatch({ type: LOGIN_USER });

        axios.post('http://188.166.250.43:3000/user/login', {
            email: email,
            password: password
        }).then(res => loginUserSuccess(dispatch, res));
    };
};
```

The value pair will be passed to the backend through posting the value to the given API server address with the axios method. After which the returned message will be justified in the function loginUserSuccess, where the dispatch method is used to dispatch the data of res to the reducers. The loginUserSuccess function skeleton is as follow:

```
const loginUserSuccess = (dispatch, res) => {
    console.log(res);
    token = res.data.token;
    dispatch({
        type: LOGIN_USER_SUCCESS,
        payload: res
    });

    if (res.data.msg === 'Success! You are logged in.') {
        Actions.main();
    }

    if (res.data.msg === 'passwords did not match') {
        dispatch({
            type: LOGIN_USER_FAIL
        });
    }
};
```

The function is responsible to determine whether the login is successful based on the message returned from the API server. If the login is complete, the main page of the app will pop out. Otherwise, the login will be denied and the error message will be shown on the screen to indicate the email and password are not eligible to log into the system. Also, the dispatch method will dispatch the corresponding data to the reducers

according to the authorization results. The state stored in the reducer will be updated according to the action type as showed below.

```
case LOGIN_USER_SUCCESS:
    return { ...state, ...INITIAL_STATE, user: action.payload };
case LOGIN_USER_FAIL:
    return { ...state, error: 'Authentication Failed.', password: '', loading: false };
```

7.2.3.2. User Profile

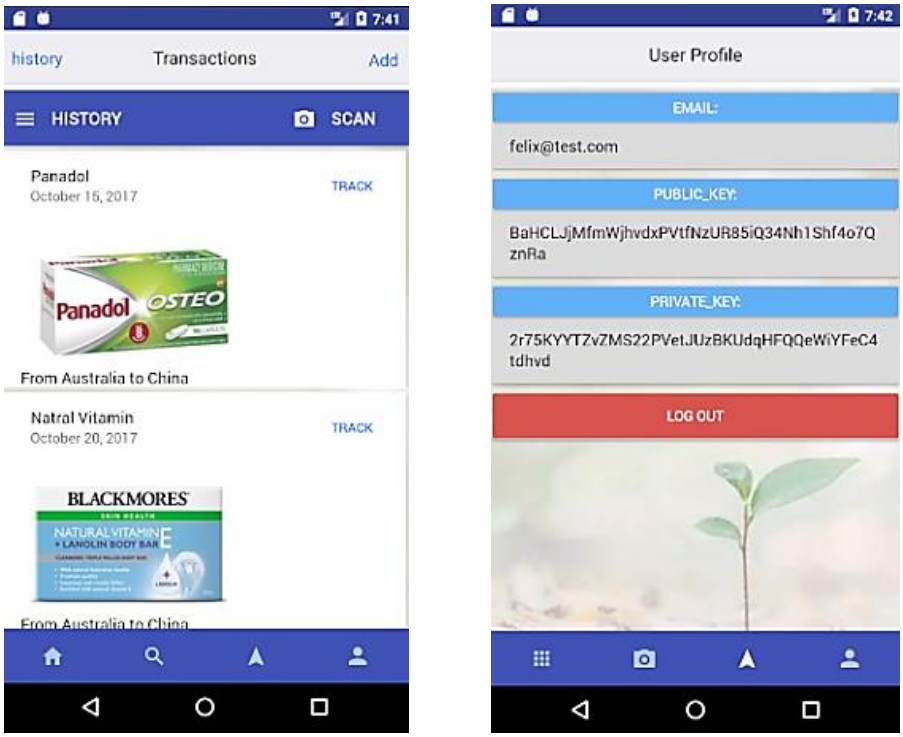


Figure 37 The Main Page (left) and User Profile Function(right) of the Mobile Application

The main page shows the overview of transactions under operation. Also, at the right bottom corner is a button that can invoke the page to show the user information, including the email, public key, and the private key. The working principle behind the scene is assisted by the use of Jason Web Token (JWT). The token is returned from the backend server and stored in the state from the login stage. When the user login successfully, the token will be returned and stored in the state, which means the following functions can use the token to play the role of a proof.

In the function of returning the user information, the token is passed to the API server

once the user clicks the user button:

```
onButtonPress() {
    const { user } = this.props;

    this.props.fetchUserInfo({ user });
}
```

The fetchUserInfo function which is imported from actions folder is responsible for fetching the user information from the backend server. An important point that needs attention is that the token is stored in the user as a sub parameter, thus this function is passing the user to the function skeleton.

```
export const fetchUserInfo = ({ user }) => {
    return (dispatch) => {
        dispatch({ type: FETCH_USER_INFO });
        axios.get('http://188.166.250.43:3000/user',
        {
            headers: {
                Authorization: `${'Bearer '}${user.data.token}`
            }
        }).then((response) => fetchResult(dispatch, response));
    };
};
```

In the function skeleton, the app sends an HTTP request to the appointed API server address with the token to fetch the user information. After which the fetchResult will be called:

```
const fetchResult = (dispatch, response) => {
    dispatch({
        type: FETCH_USER_INFO,
        payload: response
    });
    Actions.user_profile();
};
```

The fetchResult function will dispatch the returned data to the reducers with the given type:

```
case FETCH_USER_INFO:
    return { ...state, user: action.payload };
```

The user state will be updated based on the returned data and the email, public key, and the private key will show on the screen. The method to connect the component of user

profile page and the actual data from the reducer is to use mapStateToProps and connect:

```
const mapStateToProps = ({ auth }) => {
    const { email, password, user, error, loading } = auth;
    return { email, password, user, error, loading };
};

export default connect(mapStateToProps, {
    employeeUpdate, fetchUserInfo, logUserOut
})(UserProfile);
```

mapStateToProps is used to get the data from the specific reducer to enable the further functions. The connect is to combine the data, the actions, and the component together.

7.2.3.3. Create New Transaction

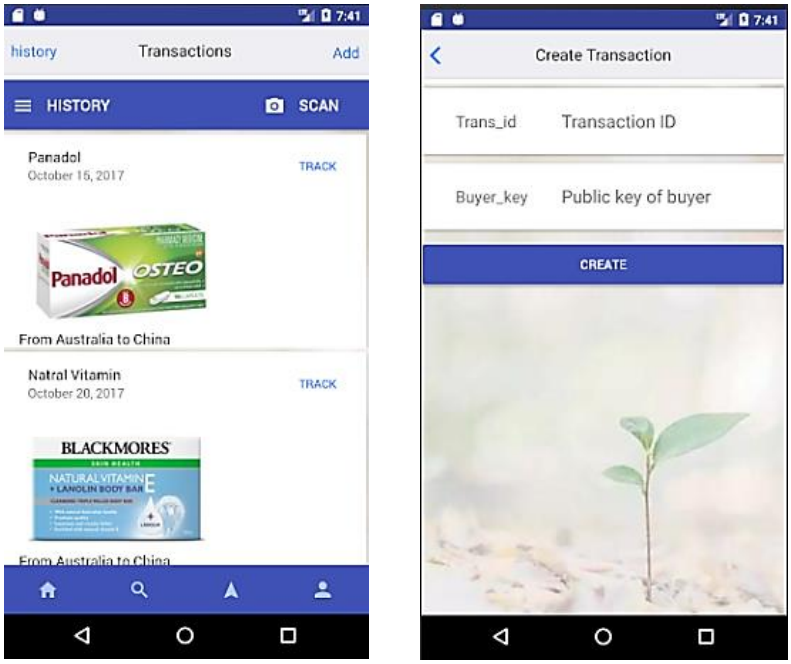


Figure 38 The Main Page (left) and New Transaction function(right) of the Mobile Application

The create transaction page will be popped up when clicking the add button at the top right corner of the main page.

To complete a transaction, the compulsory information consists of the transaction id attached to the pharmaceutical and the public key of the buyer. The createTransaction function imported from the actions folder will be called once the user clicks the create button.

```
onButtonPress() {
    const { user, transactionID, publicKey } = this.props;
    this.props.createTransaction({ user, transactionID, publicKey });
}
```

The createTransaction will take the three variables to the function skeleton, including the user where the token is stored, the transaction ID, and the public key. In the function body, it will send an HTTP request to the API server together with the mandatory variables as shown below:

```
export const createTransaction = ({ user, transactionID, publicKey }) => {
    return (dispatch) => {
        dispatch({ type: CREATE_TRANSACTION });

        axios.post('http://188.166.250.43:3000/blockchain/transfer_asset',
        {
            buyer_public_key: publicKey,
            seller_transaction_id: transactionID,
            metadata: {
                price: '100 dollars'
            }
        },
        {
            headers: {
                Authorization: 'Bearer ' + user.data.token
            }
        }).then(res => createComplete(dispatch, res));
    };
};
```

Then, the createComplete function will handle the returned result to perform the further process.

```
const createComplete = (dispatch, res) => {
    console.log(res);
    dispatch({
        type: CREATE_SUCCESS,
        payload: res
    });

    if (res.data.statusText === 'BAD REQUEST') {
        dispatch({
            type: CREATE_FAIL,
            payload: res
        });
    }

    if (res.data.statusText === 'NOT FOUND') {
        dispatch({
            type: NO_TRANSACTION_FOUND,
            payload: res
        });
    }

    if (res.data.statusText !== 'BAD REQUEST' && res.data.statusText !== 'NOT FOUND') {
        Actions.trans_complete();
    }
};
```

Three possibilities are available after the creating function. The first one will return "BAD REQUEST" which means the transaction id has been used, the second one is "NOT FOUND" which means the transaction id is counterfeit, while the last one means the transaction is created successfully. In terms of the first two possibilities, the error message will show on the screen to indicate that the transaction is invalid to complete, while the last one will trigger the transaction complete page to indicate that the transaction is valid and has been completed and the user can choose further operation.

7.2.3.4. Transaction track

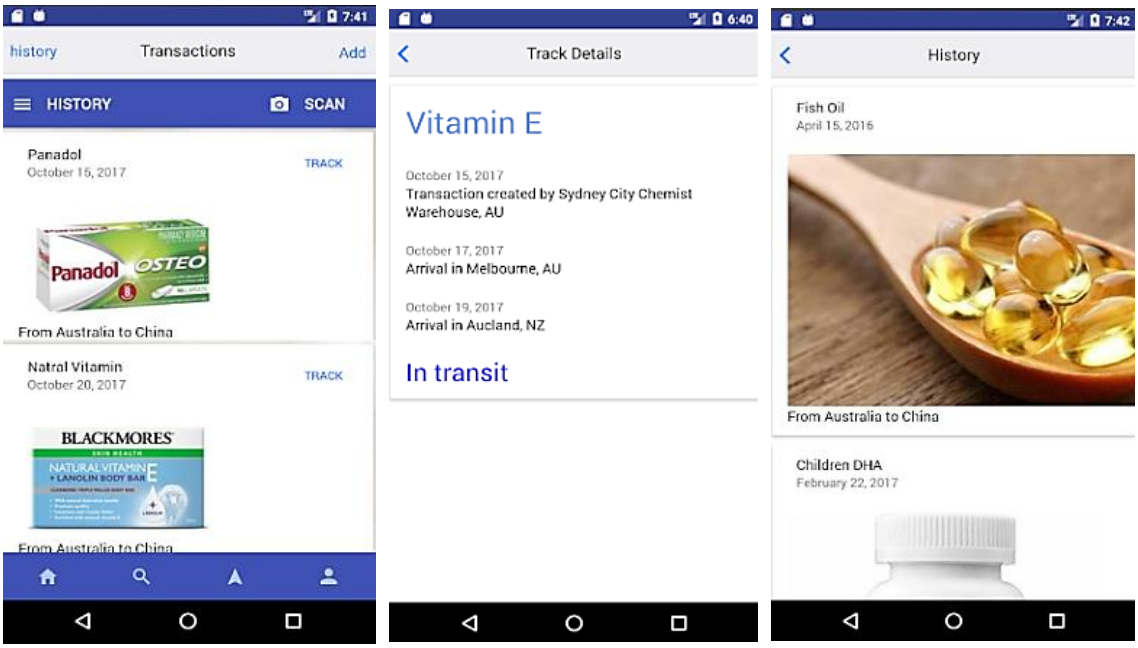


Figure 39 The Main Page (left), Track Detail Page(middle) and Transaction History Page(right) of the Mobile Application

Also, the track function is available on the mobile app. the user can click the track button on the transaction overview to call the detail page of the transaction. On the transaction detail page lie three sections of information: the name of the product, the transition information, and the status of the transaction. Meanwhile, the transaction history can be accessed by clicking the history button at the top left corner of the main page. On the transaction history page, the completed transactions are shown as a preview with the product name, completed data, product image, and the original address and the destination.

8. Resource

8.1. Hardware

8.1.1. API Server Development

Windows 10 PC to run the developing software for NodeJS.

One Virtual Private Server to run demonstration our system's API Server.

8.1.2. Web Application Development

In web development, there is no need to use other hardware to debug and run the web application except a computer. Therefore, we only need a laptop to complete the whole development of this project.

8.1.3. Mobile Application Development

Windows 10 pro

Processor: Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz 2.19GHz

Installed memory (RAM): 12.0 GB (11.7 GB usable)

8.2. Software

8.2.1. API Server Development

Visual Studio Code to modify and test our API Server's NodeJS code.

Ubuntu 14.04 on the Virtual Private Server to run our API Server.

8.2.2. Web Application Development

To develop the web app, we used Visual Studio Code (VS code) to edit the code which is a fully functional code editor. VS code is a free code editor and supports Windows, Linux and IOS operation systems. Besides, VS code also can edit several languages, such as Html, JavaScript, Java and so on. VS code also supports some extended functions to give developer a friendly development platform.

In addition, to adjust and debug the web interface, we also need a browser. The chrome browser is a good choice. Chrome web browser is developed by Google company that

can be applied to the computer, mobile phone and tablet computers. Chrome provides a tools kit for web developers that is a set of web development and debugging tools and built into Google Chrome. Developers can use this tools kit to iterate, debug, and analyze the web. Moreover, Chrome also can provide some utility plug-ins that can better support web development.

8.2.3. Mobile Application Development

Visual Studio Code is used to edit and modify code

Android studio is used to generate Android emulator

9. Discussion and Evaluation

9.1. API Server

9.1.1. Trading Chain Tracking

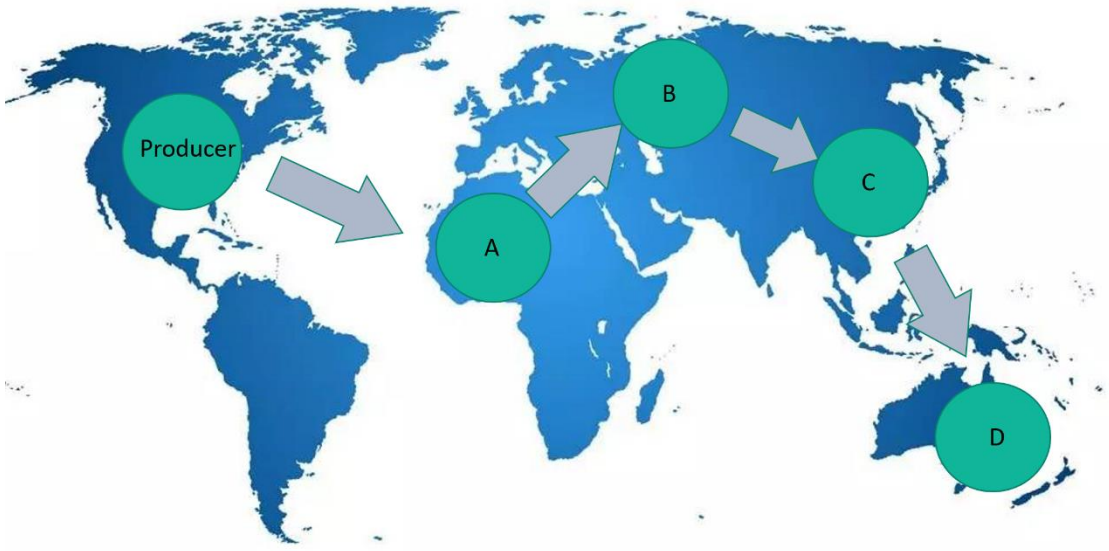


Figure 40 Drug Trading Chain

As shown in Figure 40, if the drug producer sold the drug to distributor A, and then A sold the drug to B. B is another distributor, so B sold the drug to distributor C and then distributor C sold the drug to D. The whole trading chain of this drug is from the producer to A to B to C and then to D. In the current stage, this system's API Server can only provide the information of the current owner of the drug. Users can not know

who is the producer of the drug. In the future, we will provide an API for the users to track the whole trading chain of the drug, so that the producer of the drug can be tracked easily.

9.1.2. Drug Producer Authorization

In the current stage, the “create_new_asset” api and “transfer_asset” API are provided by one same API Server, which means if this API Server is hacked by the fake drug producer, the fake drug producer can call the “create_new_asset” API to put its fake drug into the market with valid transaction ID. This possible situation fails the whole system. In the future, we will build two API servers. The first one only contains the “create_new_asset” API and will only be open to the authorized drug producer. This server is not connected to the Internet so that it cannot be hacked easily. The second API server will contain other APIs and will be open to all of the sellers and buyers.

9.2. Web Application

In this stage, the web application has achieved (1) login and register function (2) add, check and delete medicine information function (3) create and check asset function (4) check transaction ID function (5) check user information function. All these functions are planned in the beginning. However, our group want to optimize the web application.

9.2.1. Logistic Information Optimizing

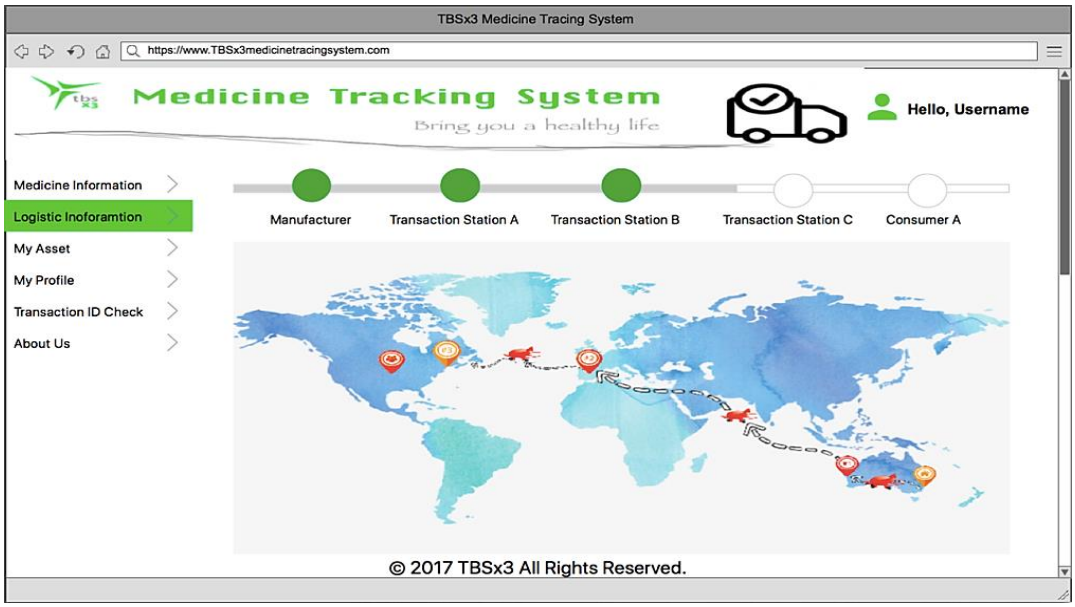


Figure 41 Next Version of Web Application Logistic Information Page

As the figure 41 shown, the new version of the web app. In the new version, we want to display the logistic information as a progress chart and show the logistics route on the map, which can dynamically track logistics progress. Using progress chart and route map, the logistic progress and route can be shown in a straightforward way.

9.2.2. Private Information Display Mode Optimizing

In addition, in the next version of the web app, the way to display the transaction ID, private key and public key should be optimized as well. In the old version, the transaction ID, private key and public key are showed directly in the web pages. Thus, if users need to get these messages, the user has to tap the very long ID and keys one letter by one letter. In the next version, we would show the transaction ID, private key and public key as QR code, which are showed in figure 42. In this way, the user only needs to scan the QR code by their phone or tablet PC, and then they can easily acquire the transaction ID, private key and public key, which can help users to save time to the largest extend. Moreover, showing the transaction ID, private key and public key as QR code can better protect the privacy of this information.

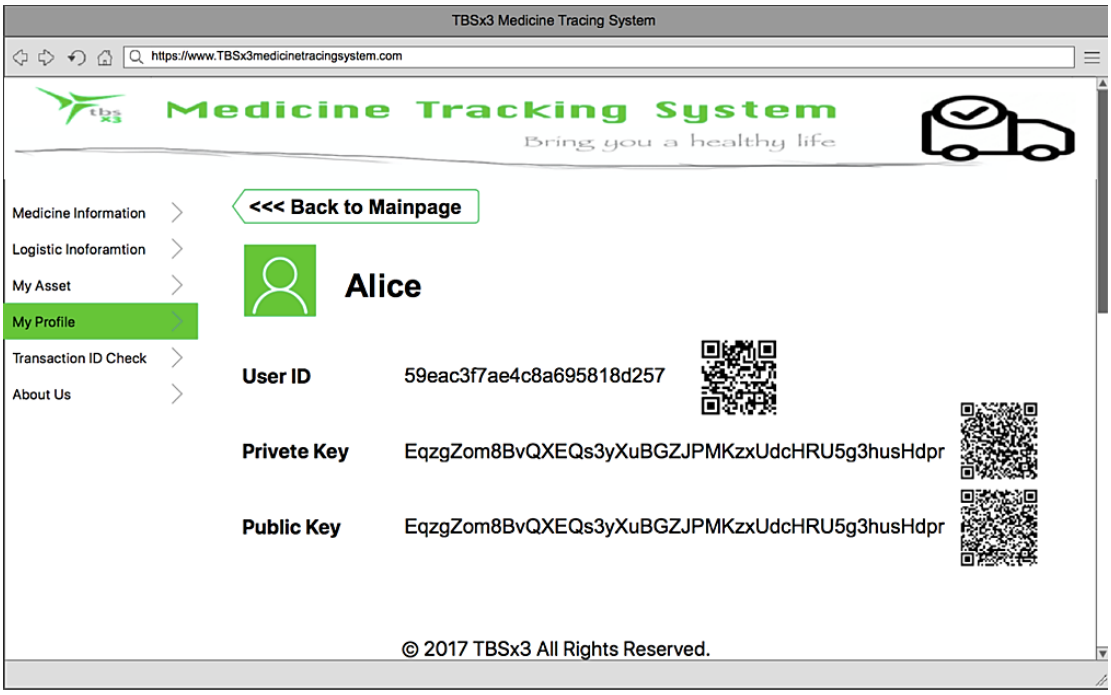


Figure 42 Next Version of Web Application My Profile Page

9.2.3. UI Design Optimizing

Furthermore, as shown in figure 43 we plan to add a side menu bar that can reduce the pages jumps and can help the developer to reduce the workload. Users can select the section, and the corresponding information would show in the main portion.

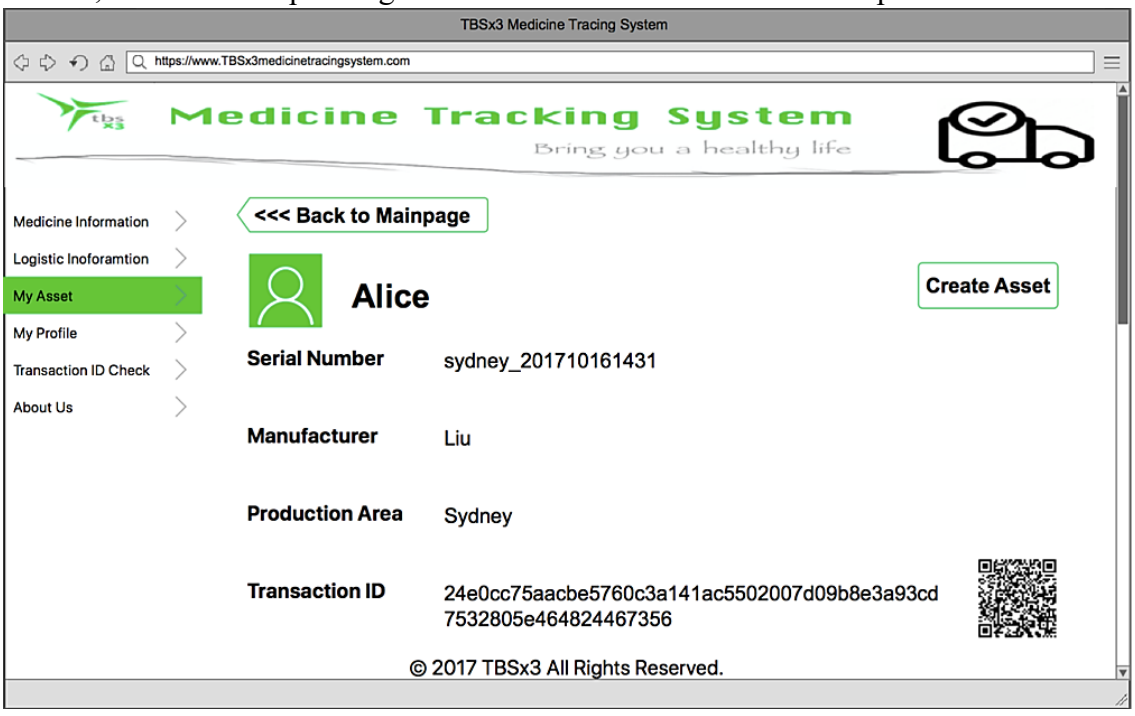


Figure 43 Next Version of Web Application My Asset Page

Besides, the UI design also needs to be redesigned. In the current stage, the UI is too simple and not attractive enough. Therefore, optimizing UI design can enhance the user experience and help users to find what they want to obtain from the web app.

9.3. Mobile Application

9.3.1. Scan Function

Figure 44 The Expected Renderings of Scan Function of Mobile Application

We designed the scan function but have not realized it. The scan function is very important for the mobile app since it can improve the efficiency and feasibility of the app. The transaction id and the public key are the two most critical variables when creating a new transaction. However, the public key is generated automatically in the backend server, and it is very long and complex to be remembered. Meanwhile, the transaction id is composed of capital letters, lower case letters, numbers, which makes it pretty long and difficult to remember as well. With the scan function, the user can scan the barcode attached to the medicine to get the real-time transaction id and can scan the QR code generated by the app to input the public key automatically.

This function can help the user to complete the transaction more efficiently and can help the user to avoid unnecessary mistake caused by manual input. Unfortunately, the function has not been realized in this version of mobile app. It is still under development.

9.3.2. UI Design

React Native comes without UI library and encourages the coders to define their own Components. However, this is unfeasible for us due to the time limitation. We are all new to the React Native, so we need to learn how to use it before the work. Also, the core functions are more crucial for the whole project. Therefore, NativeBase is used in the project to provide some available UI components. The UI of this version of the mobile app cannot satisfy our requirements and further upgrade is needed in the future.

10. Experienced Problems and Solutions

10.1. API Server

10.1.1. Environment Variable Configuration

API Server needs some environment variables which should be defined before the running of the server such as the port number and the URL of MongoDB. These environment variables cannot be hard-coded into API Server because that they may change in different environments. To solve this problem, we store configuration in one file called ".env" which is separated from code. We use a node package called "dotenv" to load environment variables from ".env" file into "process.env" variable at the runtime of API Server.

10.1.2. User Authentication

In the current stage, our API Server can only authenticate users whose user names and passwords are stored in the local MongoDB database. However, in the future, we may need to provide other authentication methods for our users such as authentication with Google account or Facebook account. It is hard to write the code again and again to implement the user authentication function. We use a powerful and famous node package called "Passport" to handle all the functions related to the user authentication which reduces the future work and also tidies the code.

10.1.3. Blockchain System

The blockchain is totally a new topic. Most of the blockchain system projects are also work-in-progress which means there may exist some unknown bugs and the documentation about the usage of the system is not detailed enough. In this situation, it is really hard to understand the whole blockchain system and write an adapter to connect our API Server to the blockchain system. We assessed a lot of blockchain systems and found that BigchainDB's documentations are the most easily to be understood. BigchainDB also provides a JavaScript driver which is highly compatible with this project. Finally, we choose BigchainDB as the blockchain system that we used to implement the whole system.

10.1.4. Deploy API Server on Linux

Before the study of this project, we only focus on learning how to write the code of API Server. We do not have any experience on deploying the server on Linux. To solve this problem, we read a lot of tutorials online to learn how to install NodeJS and MongoDB on Linux. In the end, API Server is deployed successfully on a server that is running ubuntu 14.04. The homepage URL of API Server is http://188.166.250.43:3000/ and this server can also be called by our Web Application and Mobile Application.

10.2.Web Application

We have a lot of problems during the process of developing the web application.

10.2.1. Process of Learning JavaScript

JavaScript is a friendly language for the beginners, because it is relatively simple to get started comparing with other computer languages. However, we also met some barriers When we studied JavaScript.

It is difficult to distinguish element and object which have been defined different meaning in JavaScript. Some concepts of JavaScript box model are hard to understand, but these concepts are important in locating elements in the web pages. The document

object model (DOM) is the based concept in JavaScript, but it is not easy to understand and master the attributes of all the object in DOM for the beginner. Therefore, facing all the problems, we completed online courses and practice to learn how to use JavaScript to build a simple web app.

10.2.2. Process of Learning React.js

After completing the based JavaScript learning, we need to learn how to use React.js, a JavaScript library, to build the web app. In the process of learning React.js, we still use online courses to master React.js as soon as possible. However, we still meet several problems during the process of developing the web app.

(1) In the login function, there should have a prompt message show up when the user login their account. However, we find the prompt message always show an incorrect message. Finally, we find that the state in the React.js is asynchronous, which means that the state cannot update immediately when the state is changed.

(2) When we build we app, we need to use webpack to analyze the web structure and find JavaScript modules and other extension languages that browsers can not directly run, and then convert and package them to a suitable format that browser can operate directly. But webapp cannot package the images in the project, because webpack is not good at analyzing JavaScript. Thus, there will alert a 'target image not found' error when webpack packages '.js' document resources. To solve this problem, we need to add image loader in the 'webpack.config.js' file which is shown in figure 45 as below.

```
module: {
    loaders: [{
      exclude: /node_modules/,
      loader: 'babel',
      query: {
        presets: ['react', 'es2015', 'stage-1']
      }
    },
    {
      test: /\.css$/,
      loader:'style-loader!css-loader'
    },
    {
      test: /\.(png|jpg|gif)$/,
      loader: 'url-loader',
      query: {
        limit: 150000,
        name: '.src/images/[name].[ext]'
    }
    }
  ]
}
```

Figure 45 The solution of solving webpack cannot package image problem

(3) When we collocate the version configures of library and tools, we also met a problem that all the library and tools version have to cooperate. Once there is one tool's version is higher or lower, the features of current version might not work with other tools and library. Therefore, all the version number of library and tools would show in the package.json which is shown in figure 46 as below.

Figure 46 The version number of library and tools

(4) In addition to webpack problems, we also met some barriers when we connect to API server. In the beginning, we chose to use fetch method to connect API server, but fetch method need to transfer the parameters to JSON format, which is too difficult. Therefore, we use the axios method that is very easy to use method and can transfer the parameters to JSON format automatically.

(5) Furthermore, when we try to connect web app with API server at the first time, we find that web app cannot get information form API server after user login the account successfully. To solve this problem, we add a function of API server to return a token value when the user login the account successfully and this token value is used to check if the user has the authority to get the information form API server. Therefore, we

modify the axios method as the figure 47 shown below.

```
43      axios({
44          method:'get',
45          url:'http://188.166.250.43:3000/user',
46          headers:{
47            Authorization: 'Bearer' +' '+localStorage.token
48          }
49      })
```

Figure 47 The axios method with the token checking

(6) After fix the connect problems, we find that we still cannot connect with API server and chrome alter an error which shown in figure 48.

```
Failed to load http://188.166.250.43:3000/user/login: Response to preflight  localhost/:1
request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is
present on the requested resource. Origin 'http://localhost:8080' is therefore not
allowed access.
Error: Network Error                                                    bundle.js:61016
    at createError (bundle.js:60156)
    at XMLHttpRequest.handleError (bundle.js:60008)
```

Figure 48 The cross-domain problem

This error is the cross-domain problem. The reason is the browser with the same source policy restrictions, which means when the current page uses JavaScript to obtain the data from another web site, the cross-domain problem will show up. To solve this problem, the developer only needs to install a chrome plug-in named “Allow-Control-Allow-Origin” that allows the developer to request any site with ajax from any source.

(7) After solving all the functional problems, we still met problems when we develop the UI of the web app. We use Ant Design that we have mentioned before, thus we need to adjust webpack to make the elements from Ant Design can be packaged.

```
1   module: {
2       loaders: [{
3         exclude: /node_modules/,
4         loader: 'babel',
5         query: {
6           presets: ['react', 'es2015', 'stage-1']
7         }
8       },
9       {
10        test: /\.css$/,
11        loader:'style-loader!css-loader'
12      },
13      {
14        test: /\.(png|jpg|gif)$/,
15        loader: 'url-loader',
16        query: {
17          limit: 150000,
18          name: '.src/images/[name].[ext]'
19      }
20      }
21    ]
22  }
```

Figure 49 The Solution of Solving Webpack Work with Ant Design

10.3.Mobile Application

10.3.1. The Instability of Android Emulator

The android emulator was not stable enough and often stopped working at the initial stage of the project, especially when we reload the content. In order to solve this problem, many attempts and efforts were made.

The first one is that to close all the other apps and webpages to save as much ram as possible to support the work of emulator, but it did not work as expected. Also, we tried to change a different version of emulators and change different versions of android studio. But they both did not work as expected. Finally, we dismissed the remote debug function which was used to check the app functionality since we started the app development. This resulted in a good consequence that the testing and the reload became more stable.

10.3.2. The Usage of State

Since we are fresh for the mobile app, especially for the React Native framework. It was a little annoying to understand the structure and the work principles of React Native. It did not affect the work at the first stages when there was no need to show the data stored in the reducers. However, when it comes to the functions that need to show the data stored in the reducers on the screen, we encountered problems.

In order to overcome the problems, we looked in related documents through the Internet firstly. There is no doubt that there are a bunch of corresponding knowledge, while the specific one on how to build the connection between reducers and components are not precise. Finally, we took a class about React Native on Udemy which help me to understand how the state works in the React Native framework.

10.3.3. Session Issue

Initially, the session was selected to work as the proof of identity. Its working principle is to store the session_id in the cookie, which can be used to prove the identity of the user in the next stages. However, this method cannot work on the mobile app for some

unknown reason.

Therefore, we looked for an alternative method to realize the function. Jason Web Token is the most popular one. Its work principle is when the user successfully logs into the system with his credentials, a Jason Web Token will be returned and saved in the local storage (<nico@ponyfoo.com>, 2017). Whenever the user needs to access the backend server, it will send the Jason Web Token as the proof of identity to the server. The server will return the required information according to the given Jason Web Token.

11. Reflections

11.1. API Server

One of the amazing parts of working on this project is that we met a lot of useful NodeJS packages such as “mongoose”, “passport” and “dotenv”. All of these packages not only help me solve the problems but also help me tidy the code and reduce the future development and maintenance work. After working on this project, we know that when meeting some problems in developing NodeJS server, the first thing is to check if there exists any amazing package to solve them.

When developing API Server of our system, we learned that blockchain is actually immature technology. A lot of famous blockchain systems are lack of useful documentations or helpful tutorials. Some blockchain systems are also very hard to be understood or used. As the member of the blockchain technology developing community, we will try to write some tutorials to help other beginners.

11.2. Web Application

Web-related knowledge is a brand-new field for us, thus this project is a good chance for us to start the studying path of web. Moreover, we touch the very new technology via the project, React.js is a very new JS library which is a developer-friendly library. Besides, we also practice our coding skills and experiences through this project development.

11.3.Mobile Application

11.3.1. Structure of React Native is a Barrier for Beginners

The structure of React Native is not similar to other frameworks, with different patterns and concepts form other frameworks. Actually, its structure is a little complex for the small apps since the coders need to build a bunch of different folders, including the actions, reducers, components, to make them the app work as expected. However, with regard to the big app or complex app, the complex structure of React Native definitely can help the coders to get a more hierarchical and clear structure of app.

11.3.2. Learn Once, Write Everywhere

React Native is a cross-platform framework, which means most code can be duplicated between IOS and Android although some slight adjustments may be required. Throughout the entire development process of the mobile app, I referred to some code from the course of React Native and Redux, which deploys the app on IOS. The code is compatible on Android. The feature of React Native can boost the efficiency of the programmers and save development cost of mobile app.

11.3.3. Affluent Third-party UI Libraries

UI design is necessary for the mobile app development. Yet, no corresponding UI libraries is provided by React Native since it encourages the coders to define their own UI components. Fortunately, some high-quality UI libraries are available and can be deployed in the mobile app. In this project, some UI components from NativeBase are used. These UI libraries relieve the pressure of app developers to some extent.

12. Evidence of Collaboration

In this project, our group members use slack, google drive and gitlab to help us to communicate with each other and integrate paper work and code.

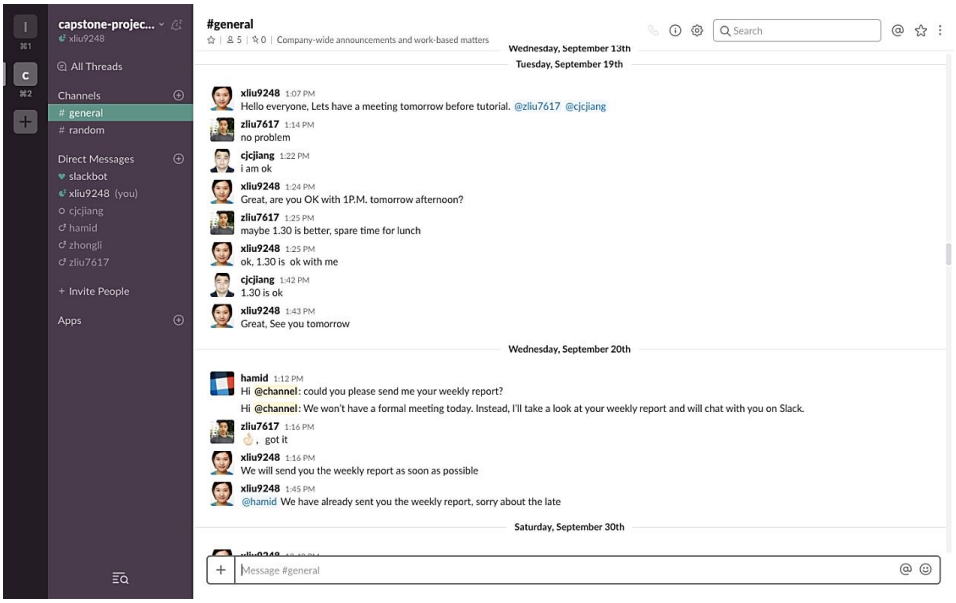


Figure 50 Slack

As shown in Figure 50, our group members communicate via Slack to arrange our weekly meetings. Moreover, our group members also use slack to communicate with our client and tutor. Slack provides a good platform to discuss problems that we meet during the project development. In addition to arranging meetings, we always ask the confused parts of project and requirements of the proposal, progress report and presentation on the slack and our tutor would give us very responsible help every time.

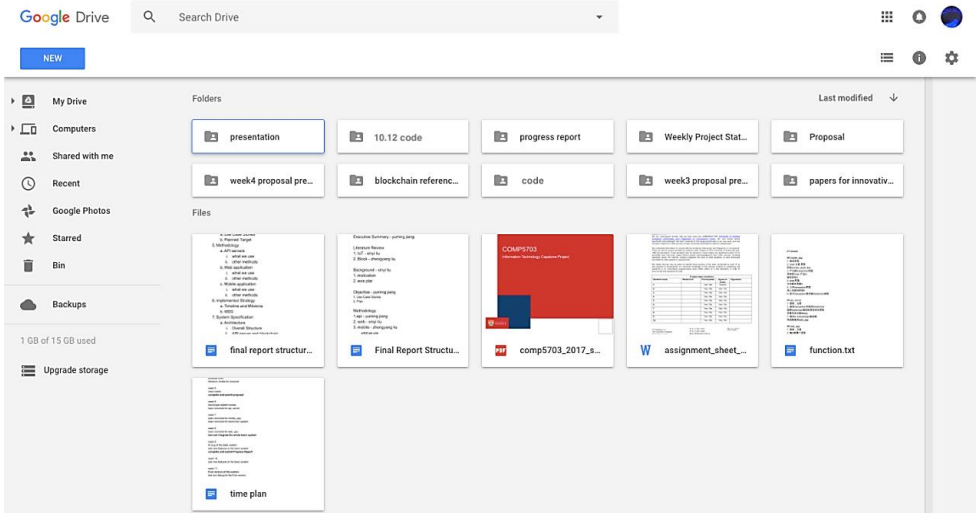


Figure 51 Google Drive

Besides, Google drive is another platform that helps us to record meeting notes and plans of group works. Google drive helps us to share documents that can help us to understand our project and make the project object clear as well. Furthermore, we use google drive to record the content of the weekly meeting and the plan for next week. Because google drive can edit the file by multiple person at same time, thus we also can add new idea about the project in the same file at the same time. In addition, we also use google drive to save all materials of the proposal, the progress report, the presentation and weekly progress reports.

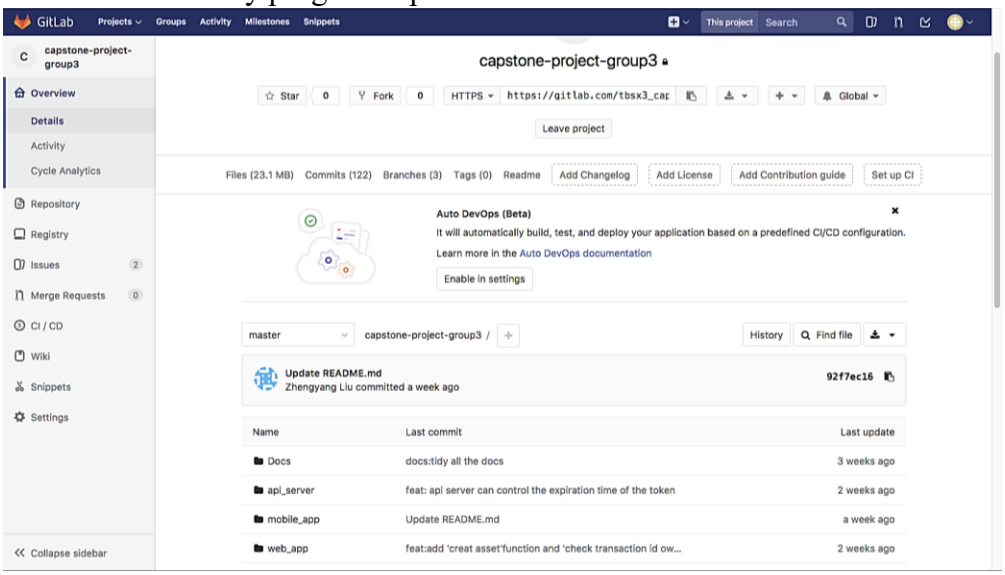


Figure 52 Git Lab

As shown in Figure 52, we also use gitlab to save and integrate our code. Gitlab is a git-based online code hosting service, which is very similar to GitHub. However, comparing with GitHub, GitLab is a closed environment that the repository and code are not open to the public, which guarantee the privacy of our project. Therefore, we uploaded weekly study note and weekly progress report to GitLab in folder Weekly Meeting, which also sent to tutor as weekly project status report via email. Besides, we also upload all the references and related document about the proposal and progress report on the GitLab in corresponding folders. We also use GitLab to manage and integrate our code. GitLab is a very good platform for developers to work together to build system and assign the code work to every group members.

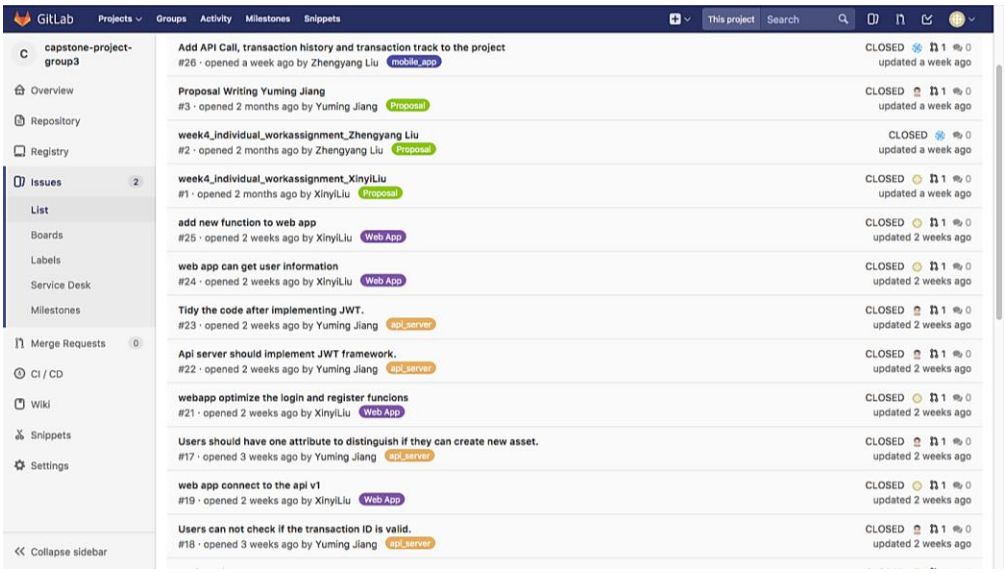


Figure 53. Git Lab Issues

As shown in Figure 53, GitLab can record every changes and upload information. All the records are displayed as issues that also record every group members' workload and involvement degree.

13. Reference

1. Agbaraji, E. C., Ochulor, D. O., & Ezeh, G. N. (2012). Food and drug counterfeiting in the developing nations; the implications and way-out. Academic Research International, 3(2), 24.
2. AntDesign. (2017). Ant Design：A UI Language. Retrieved from https://ant.design/
3. Anti-CounterfeitingPackaging. (2016). Anti-Counterfeiting fighting the good fight. Retrieved from https://anticounterfeitingpackaging.wordpress.com/
4. Ayodokun, O. J. (2014). The Use of Mobile Phone to check for the Authenticity of Pharmaceutical Products in Nigeria a case study of Mobile Authentication Service (MAS).
5. Bansal, D., Malla, S., Gudala, K., & Tiwari, P. (2012). Anti-counterfeit technologies: a pharmaceutical industry perspective. Scientia pharmaceutica, 81(1), 1-14.
6. Chen, S.-L., Chen, Y.-Y., & Hsu, C. (2014). A New Approach to Integrate Internet-of-Things and Software-as-a-Service Model for Logistic Systems: A Case Study. Sensors, 14(4), 6144.
7. Cockburn, R., Newton, P. N., Agyarko, E. K., Akunyili, D., & White, N. J. (2005). The global threat of counterfeit drugs: why industry and governments must communicate the dangers. PLoS medicine, 2(4), e100.
8. Codementor.io. (2017). React Native vs Ionic: A Side-by-Side Comparison | Codementor. [online]Available at: https://www.codementor.io/fmcorz/react-native-vs-ionic-du1087rsw [Accessed 12 Nov. 2017].
9. Docs.nativebase.io. (2017). Introduction · NativeBase. [online] Available at: https://docs.nativebase.io/ [Accessed 12 Nov. 2017].
10. EuropeanCommission. (2017). How does RASFF work. Retrieved from https://ec.europa.eu/food/safety/rasff/how_does_rasff_work_en
11. Flux. (2017). Flux:Application Architecture for Building User Interfaces.

Retrieved from https://facebook.github.io/flux/

12. GmbH, B. (2017 May). A BigchainDB Primer. Retrieved from https://www.bigchaindb.com/whitepaper/bigchaindb-primer.pdf
13. Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future generation computer systems, 29(7), 1645-1660.
14. Isah, H. (2012). Information and Communication Technology in Combating Counterfeit Drugs. arXiv preprint arXiv:1211.1242.
15. Lee, I., & Lee, K. (2015). The Internet of Things (IoT): Applications, investments, and challenges for enterprises. Business Horizons, 58(4), 431-440.
16. Medium. (2017). Angular vs. React vs. Vue: A 2017 comparison – unicorn.supplies – Medium. [online] Available at: https://medium.com/unicorn-supplies/angular-vs-react-vs-vue-a-2017-comparison-c5c52d620176 [Accessed 12 Nov. 2017].
17. Microsoft. (2017). ASP.NET MVC Overview. Retrieved from https://msdn.microsoft.com/en-au/library/dd381412.aspx
18. Morris, J., & Stevens, P. (2006). Counterfeit medicines in less developed countries. International Policy Network, London.
19. Nsimba, S. E. (2008). Problems associated with substandard and counterfeit drugs in developing countries: A review article on global implications of counterfeit drugs in the era of anti-retroviral (ARVS) drugs in a free market economy.
20. Pecoul, B., Chirac, P., Trouiller, P., & Pinel, J. (1999). Access to essential drugs in poor countries: a lost battle? Jama, 281(4), 361-367.
21. Pilkington, M. (2015). Blockchain technology: principles and applications. *Browser Download This Paper*.
22. React. (2017). React;A JavaScript library for building user interfaces. Retrieved from https://reactjs.org/

23. React-Native. (2017). Retrieved from http://facebook.github.io/react - native/releases/0.47/docs/running-on-device.html#connecting-to-the-development- server
24. Redux. (2017). Retrieved from https://redux.js.org/docs/introduction/
25. Redux.js.org. (2017). Read Me · Redux. [online] Available at: https://redux.js.org/ [Accessed 12 Nov. 2017].
26. Rey, J. L. (2016). Modular development with redux.
27. Sadouskaya, K. (2017). Adoption of Blockchain Technologyin Supply Chain and Logistics.
28. Sowder, A. (2016). The Harmful Effects of Counterfeit Goods. Athens State University. Athens State University, nd Web, 1.
29. Staltz, A. (2015). Retrieved from https://staltz.com/unidirectional-user-interface-architectures.html
30. Wright, A., & De Filippi, P. (2015). Decentralized blockchain technology and the rise of lex cryptographia.
31. <nico@ponyfoo.com>, N. (2017). JSON Web Tokens vs. Session Cookies: In Practice. [online] Pony Foo. Available at: https://ponyfoo.com/articles/json-web-tokens-vs-session-cookies [Accessed 13 Nov. 2017].