Department of Computer Science and Engineering
The Chinese University of Hong Kong

Final Year Project Report (Term 1)

Anonymous Online Course Evaluation

Written by
Yik Fung Yau
Yuk Lung Lui

Supervised by
Prof. Lyu Rung Tsong Michael

1 December 2019

Contents

Table of Figures

Abstract

In this project, we design and develop an anonymous online course evaluation system with Hyperledger Fabric. The system can provide verifiable course evaluation platform. Anonymity of the system can protect the user privacy and evaluation process.

Traditional handwritten course evaluation has numerous problems. Massive paper usage, course hours occupation and data analysis time, etc., these problems can be avoided if we do our course evaluation on an online platform. We implement our course evaluation system with Hyperledger Fabric to improve anonymity of evaluation process. So that the online course evaluation system can be an advanced alternative option for course evaluation.

In this semester, we only implement the simplified version of designed system with simple user interface. Next semester, we will complete the whole designed system and try to improve the data encryption by using homomorphic encryption. The final version of our system should be user-friendly and every user can have better course evaluation experience.

Acknowledgments

We would like to express our gratitude to our supervisor Prof. Michael Lyu and advisor Mr. Edward Yau for providing guidance, feedback and resources. Without their help, this report will not be completed.

We are grateful to everyone who has supported us in this project or read this report.

1 Introduction

1.1 Motivation

Course evaluation in CUHK is always paper-based. For completing the course evaluation, we must spend some lecture time for distributing evaluation forms, filling evaluation forms and collecting evaluation forms. If we even count the preparation before the course evaluation and the time for data analysis, it is not difficult to notice that the whole course evaluation procedure spends an unreasonable amount of time and resources. In some cases, filling the form can be quite annoying when we need to fill the basic course information by ourselves. To save the cost of evaluation and improve the student experience, digitalize the evaluation process is an inevitable trend.

If we type and search “online course evaluation” on Google, we can find out that some schools in other countries already have their online course evaluation systems. This shows that the online course evaluation is feasible. The only problem of introducing an online course evaluation system is getting trust from users. For students, we need to prove that their submitted forms are counted and the submitting process would not reveal their identities to teachers. For teachers, we need to promise that results are correct and every counted form is verified.

To provide a reliable and trusted online platform, we choose to deploy the system on a blockchain network. Hyperledger Fabric, the blockchain framework we

used, promises permissioned membership in the network and the channels in the network to allow data only go to the parties who need to know. This provides a safer and reliable environment to store the evaluation information. As our project continues, we hope that the course evaluation system can be more reliable and secure, such that it can be used by our schoolmates.

1.2 Background

1.2.1 Online course evaluation consideration

Evaluation of a student is verifiable but anonymous

An evaluation system should be end-to-end verifiable such that the student can verify his submitted encrypted form, his saved form on server and his form is counted in the results [1]. Besides, only eligible students for that course evaluation can join the evaluation. The submission of the form should be anonymous so that the owner of the submitted form and form content should be unknown to everyone except the owner.

Allow each student to cast a course evaluation just once

To prevent inaccurate results because of spamming forms from students, the evaluation system should have a mechanism to prevent multiple forms from being sent from a student in the same course.

Accurately records the evaluations

The evaluation records should be always accurate. Under cyber-attack, the

system should be able to filter the wrong data from malicious attack and prevent the records from being corrupted

Accurately counts the evaluations

The evaluation results should be always correct and the only submission from eligible students will be counted. Any corrupted results should be noticed and removed.

1.2.2 Blockchain

Blockchain is a list of records and block is the basic storing unit. A simple block contains a timestamp, stored data, hash of current block and hash of the previous block. These blocks are linked by the hash and this makes the data modification difficult since modified data will change the hash of block and hash of all successor blocks need to be recalculated.

In addition, the blockchain network is distributed and sometimes decentralized. Everyone with permission (For public blockchain, no permission is required) can become a node in the network and access the data. Every data modification or block generation is required approval from the majority of network nodes. Unless more than half of the network nodes are hacked, it is impossible to change the data saved in the blockchain.

There are two major kinds of blockchain, public blockchain and private

blockchain.

In public blockchain, everyone can join the network and every node can generate a new block. Therefore, public blockchain usually uses proof of work mechanism [2] to control the block generation and enhance data security. When a node tries to generate a new block, the node first needs to calculate the nonce, which can be hashed to a value with a specific feature. Then the node broadcasts the new block with the nonce. After the validation from majority of nodes, the new block will be accepted. The nonce calculation is computationally intensive. As a result, the block generation is slow and further increase the difficulty of data modification.

In private blockchain, only the users with permission can participate and only trusted nodes can generate new block. To secure the block generation and keep the performance of the network system. Some kinds of private blockchain (Hyperledger Fabric) use Practical Byzantine Fault Tolerance [2] instead of proof of work mechanism. The block generator will be determined by majority. Once there are at least two-third honest nodes in the network, this consensus approach will work.

1.2.3 Hyperledger Fabric

Hyperledger Fabric is an open source business blockchain framework. Hyperledger Fabric is a framework for permissioned (private) blockchain network [3]. As identities of all members are known, the anonymity of private blockchain is weaker than the public blockchain but it is easier to do user authentication in private

blockchain.

When we join the blockchain network, we set up nodes for participating in the network. These nodes are called peers. Peer is the basic element of the network. A peer should have its own chaincode and ledger. Chaincode (smart contract) is the code which can access the ledger. Ledger is responsible to record the data by blockchain and world state which is the current value of stored objects. To update the ledger, an orderer is required for generating new block and providing ordering service. If we want to have private data sharing, then we can set up channels between peers or create private data collections, which are subsets of organizations share private data in channel.

The workflow of Hyperledger Fabric can be explained as below.

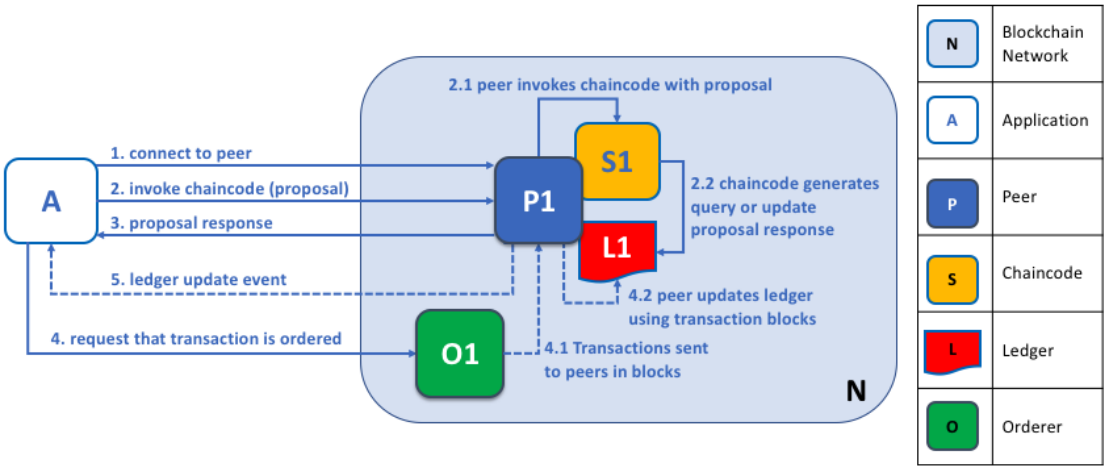


Figure 1: An example of Hyperledger workflow (Source: hyperledger-fabric.readthedocs.io)

Assume we want access the data in ledger, we first invoke the chaincode of the peer through an application which is connected to the peer. Then the chaincode makes a response and the application receives the response. If we just read the data

in ledger, the response will be returning requested data. If we want to update the ledger, we make a transaction from the response and send it to the orderer. Then the orderer generates new blocks. If there are multiple channels in the network and orderer receives multiple update requests, the orderer will order the requests chronologically by channel. The peer updates its ledger with the blocks. Once the update is completed, the peer sends an event to the application for notification [4].

1.3 Objective

The goal of our project is to design and build an online anonymous course evaluation which can protect the privacy of students and the ownership of submitted evaluation form cannot be known to teachers. Hyperledger Fabric and other technologies would be used to improve the anonymity, end-to-end verification and user authentication. The final version of the evaluation system should be the replacement of current paper-based course evaluation.

The whole project is expected to be completed in 2 terms. In the first term, we have these objectives:

1. Study the related research and implementation of end-to-end verifiable voting, blockchain voting and Hyperledger Fabric.
2. Design the flow of a course evaluation, from preparation to releasing results.
3. Implement the basic version of application and we can do a course evaluation and view the results.

2 Related Work

2.1 Online course evaluation system

When we search “online course evaluation” by Google, we can find out that some colleges in the other countries use different approaches to do their online course evaluation.

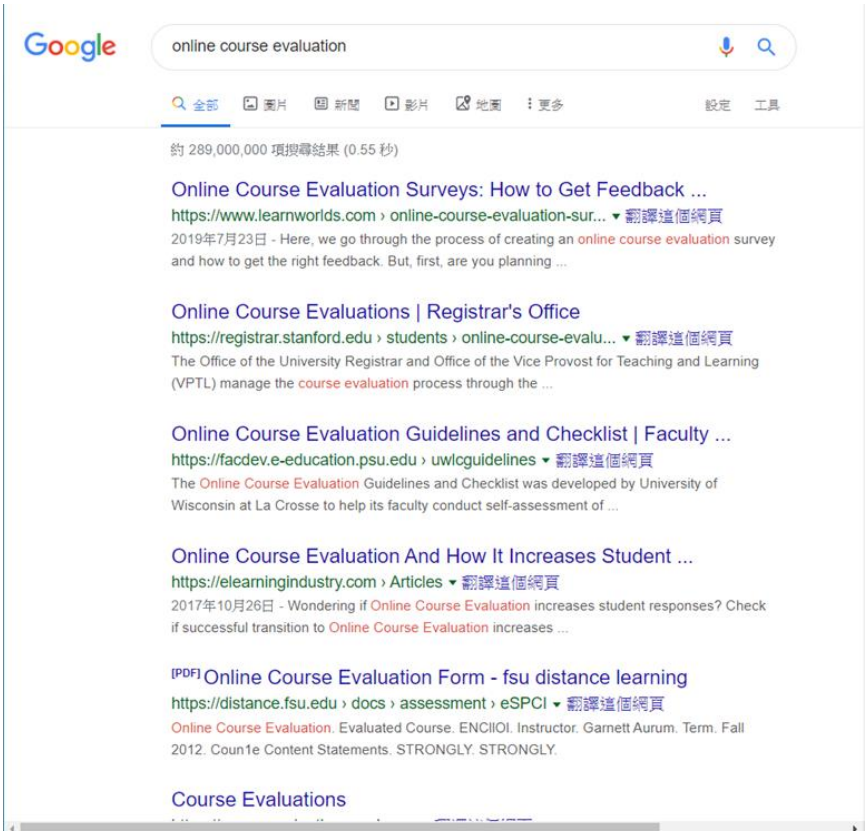


Figure 2: A screenshot of Google search (Source: Google)

Some schools may find outside vendor to manage the evaluation data. Stanford University is one of the examples [5].

Figure 3: A screenshot of Stanford University website (Source: registrar.stanford.edu/students/online-course-evaluations)

However, the websites of those schools seldom mention how the online course evaluation system works. For the anonymity of the systems, those schools may only mention that the identities of students are confidential. The students who use the systems can only trust their school without any proving.

2.2 Online e-voting system

As we cannot try the online course evaluation of other schools and understand how those systems work, we try to study the implementations of online e-voting system, which are quite similar online course evaluation system.

2.2.1 Helios

Figure 4: A screenshot of Helios website (Source: https://vote.heliosvoting.org/)

Helios is an open-source verifiable online voting system [6]. To start a vote, we can log in with their Facebook or Google accounts. Then we fill in some setting to generate an election. We can find a group of people who we trust to be the trustees, or Helios will be the trustee by default. Each trustee holds a unique key pair and the public keys of trustees are used for encrypting the votes.

When we finish our votes, our votes will be encrypted with those public keys and some random values. We will also get ballot trackers which can be used for vote verifying. There will be a bulletin board which list all submitted ballot trackers. If our trackers are shown on the board, then our vote is counted in the tally.

All encrypted votes are combined into an encrypted tally by homomorphic

encryption and only the tally will be decrypted for showing the voting results. To decrypt the tally, all private keys of trustees are needed [7].

2.2.2 Follow My Vote

Figure 5: A screenshot of Follow My Vote website (Source: https://followmyvote.com/)

Follow My Vote is an open-source voting system which is implemented with blockchain technologies [8]. The data of voting is stored in blockchain and the online voting platform uses Elliptic Curve Cryptography (ECC), which is a kind of asymmetric cryptography, to create votes.

Before we start voting, we have to create two ECC key pairs. One pair (identity key pair) is for checking voter identities and another pair (voting key pair) is for voting. When we join a voting, we have to show our identities to a verifier and the verifier will record our identities with our identity key pair. Then we register our voting key pair with one of the identity keys anonymously. Then we can make a vote and sign the vote with our private voting key. Everyone in the Follow My Vote can use the public voting key to verify our vote [9].

2.2.3 Polys

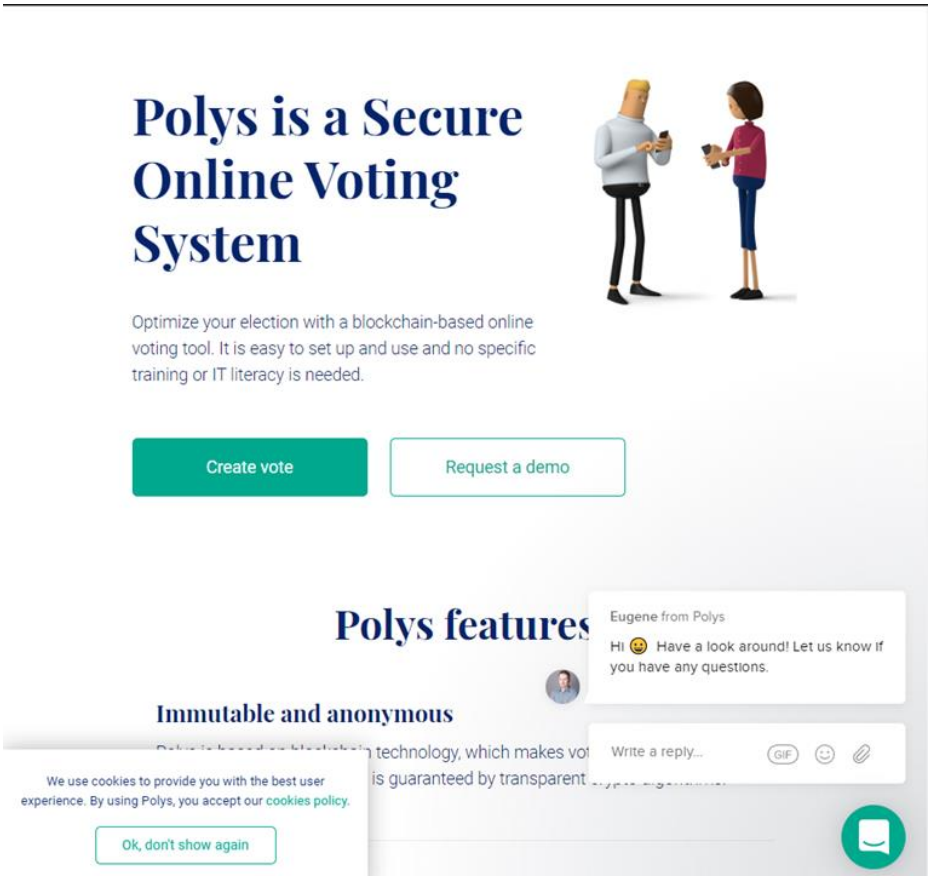


Figure 6: A screenshot of Polys website (Source: https://polys.me/)

Polys is another blockchain-based online voting platform [10]. In a voting,

each voter has own token and a random Ethereum account with permission. To make a vote, the voter submit the token to registry. The registry will find an alias and the registry will return the address of alias to the voter. Then the voter can ask the alias to cast a vote. The alias will check the Ethereum account. If the Ethereum account is allowed to vote, the alias will cast the vote [11].

3 Design

3.1 Overview

We decided to use Hyperledger Fabric as our system network framework. The main reason is that Hyperledger Fabric is a permissioned blockchain network framework and only the users with permission can use the network. With the feature of passing data through channels and private data collections (PDC), the data only pass to the related parties. With these properties, Hyperledger Fabric provides a reliable environment to store data.

3.2 Principle of system

3.2.1 User authentication by token

Before starting a course evaluation, we have to make sure that only the students who study the course can participate the course evaluation. Therefore, the students who study the course should receive a token from the faculty. This token is generated by our designed evaluation system. To join the evaluation, students have to prove their identities with their tokens.

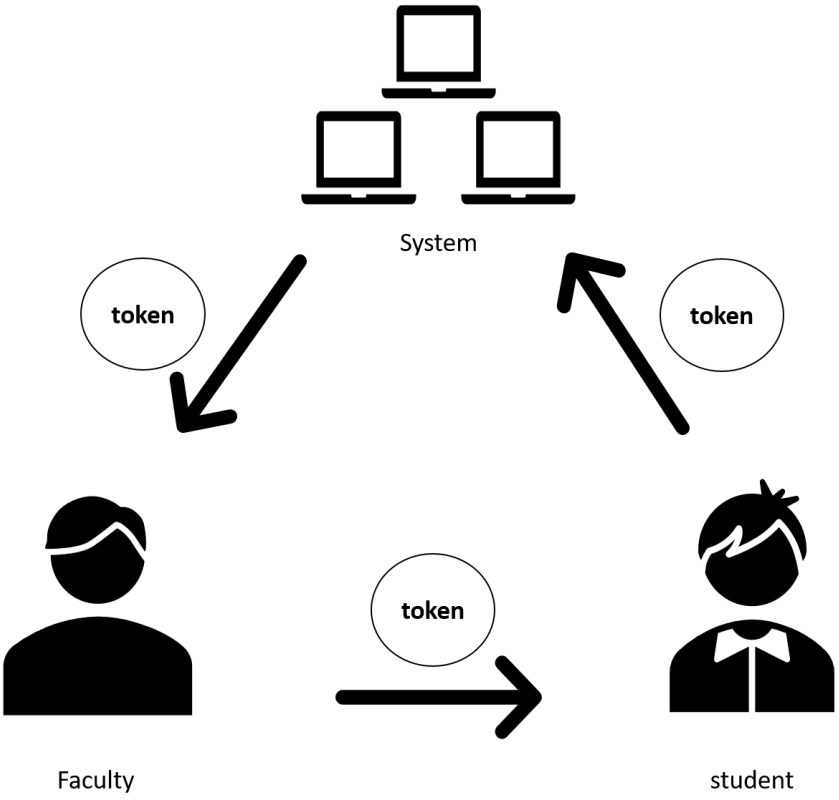


Figure 7: User authentication by token

3.2.2 RSA signature

When we use the system generated tokens to verify identities of students, the faculty would have a record of token ownership. If we directly use the tokens for end-to-end verification, the ownership of submitted evaluation forms will be exposed.

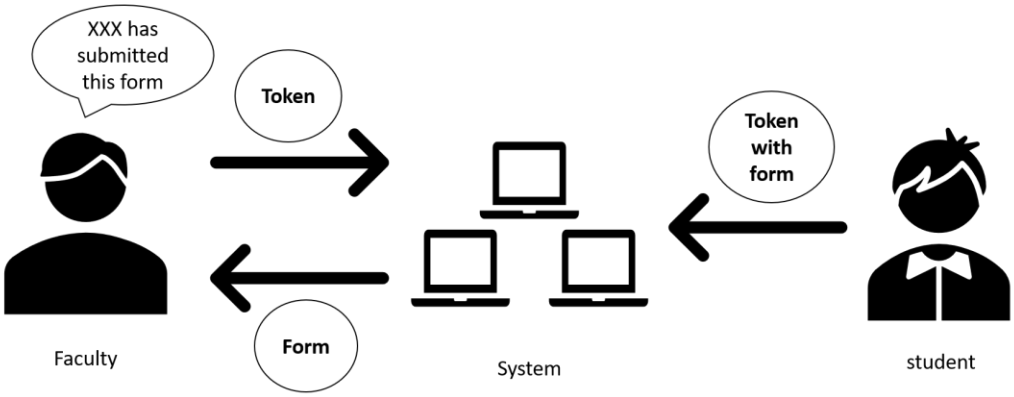


Figure 8: Token reveals user privacy

To prevent above situation, we use RSA signature for end-to-end verification. Before doing the evaluation, student has to generate own key pair and keep the private key safe.

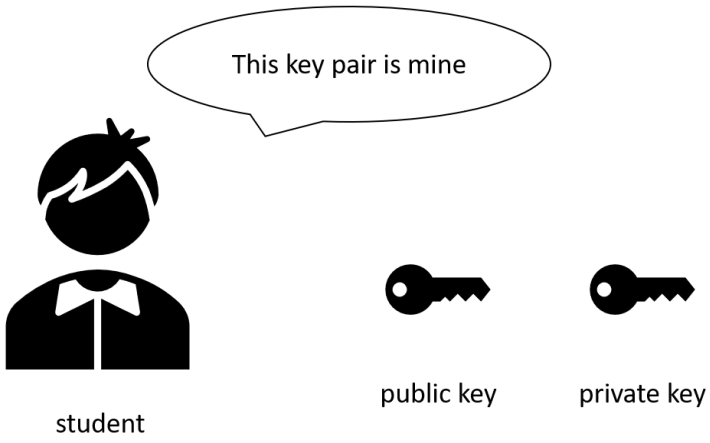


Figure 9: User has unique key pair

Then the public key, with the token, will be sent to the system. The token will be checked by the system. If the token is a valid token, the public key will be saved in the system and the token will be marked as used. No public key can be accepted with the used token.

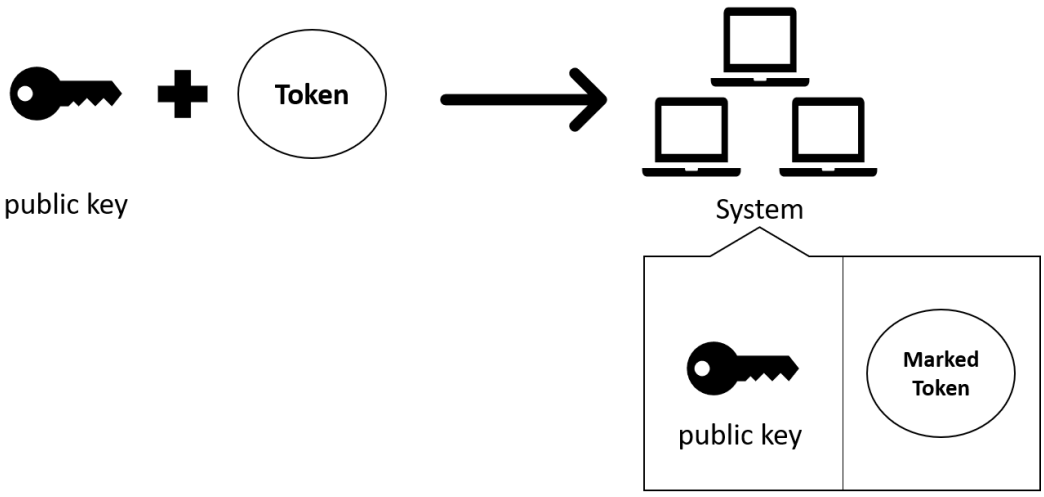


Figure 10: Public key registration

When a student submits evaluation form, a hash of that form is created and the hash is encrypted by private key.

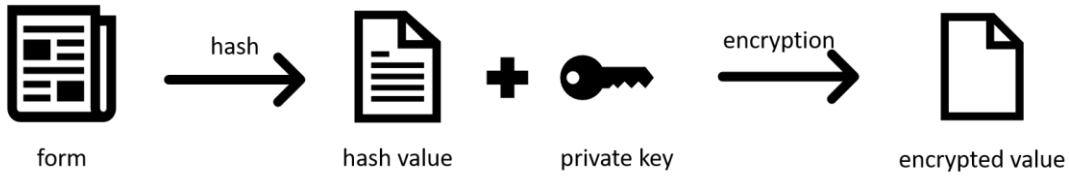


Figure 11: Signature creation

Then the student can send the evaluation form with the encrypted value to the system.

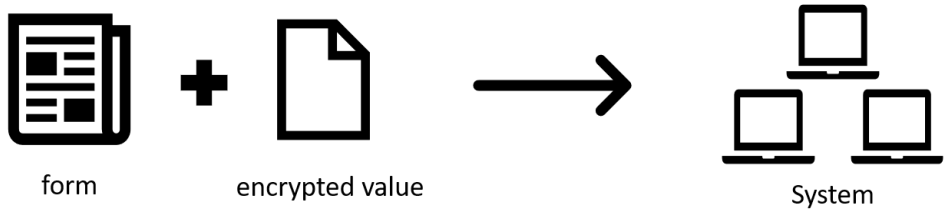


Figure 12: Submit the form with signature

The system decrypts the encrypted value with the stored public keys and

compares the decrypted value with the hash value of the form. If there is decrypted value which equals to the hash value, the submitted form will be saved. The relation between the form and public key will be saved as well.

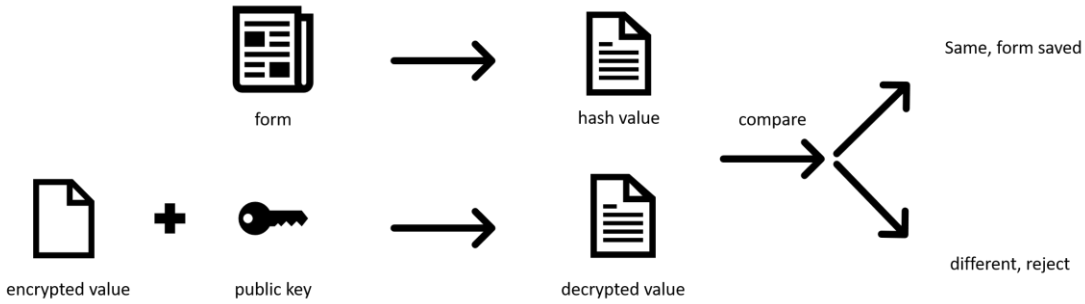


Figure 13: Processing submission

As the relation between public key and token has no record in the system, people who only hold the tokens cannot know corresponding public keys and they cannot know the owner of submitted form.

	faculty	student	teacher
owner of token	know	know (own token)	know
owner of key	do not know	know (own token)	do not know
owner of form	do not know	know (own token)	do not know

Figure 14: Table of owner identity permission

3.3 Proposed system architecture

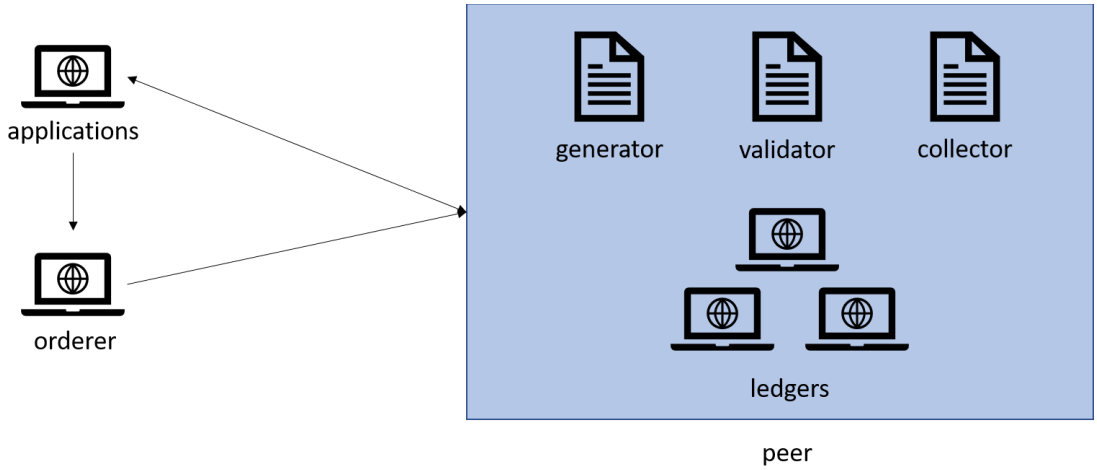


Figure 15: Proposed system architecture

In the proposed system, there are multiple ledgers and three kinds of smart contract in peers. The ledgers are responsible for storing token information, public keys provided by the verified users and verified evaluation forms with corresponding public key and evaluation results. Validator is the smart contract which is used for updating used token information and update stored public key information. Collector is the smart contract which queries valid public keys and update stored evaluation information with valid public key. Generator is the smart contract for updating new token information.

3.3.1 Workflow of the system

3.3.1.1 Token generation

The faculty provides issuer identity, course ID, number of tokens and token expiration date to an application. The application sends update token information

request and the system returns the tokens. Then the faculty get the tokens from the ledger and distributes the tokens to the students.

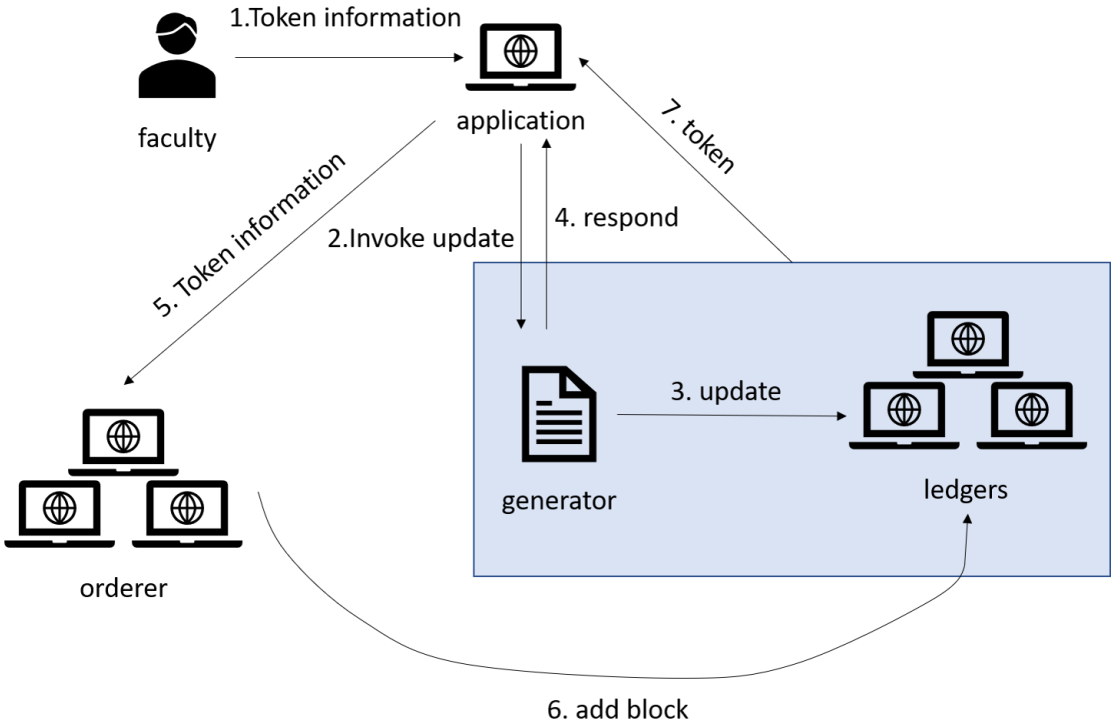


Figure 16: Workflow of token generation

3.3.1.2 Public key verification

Students have their tokens and own key pairs. The key pair is proposed to be generated by the application. Then students submit the token and public key to the system for updating valid public key list.

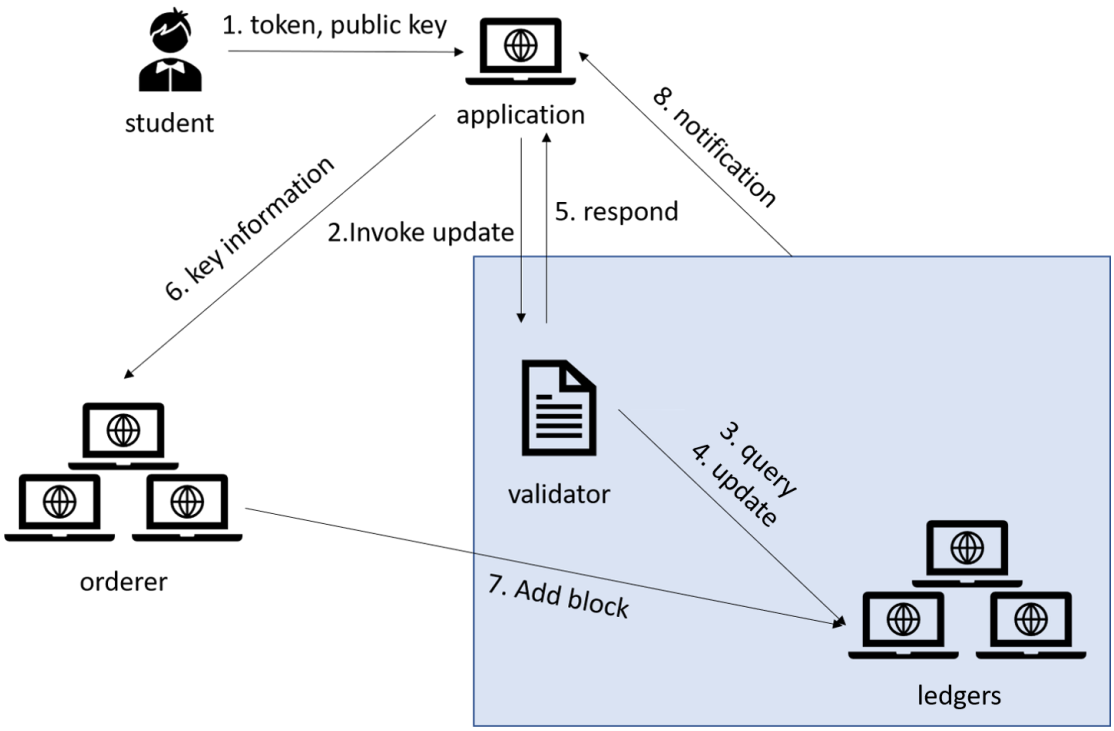


Figure 17: Workflow of public key verification

3.3.1.3 Evaluation form submission

After submitting the public key, students can fill their evaluation forms and submit the forms with RSA signature. Then the system checks the signature and updates evaluation information if the signature is valid.

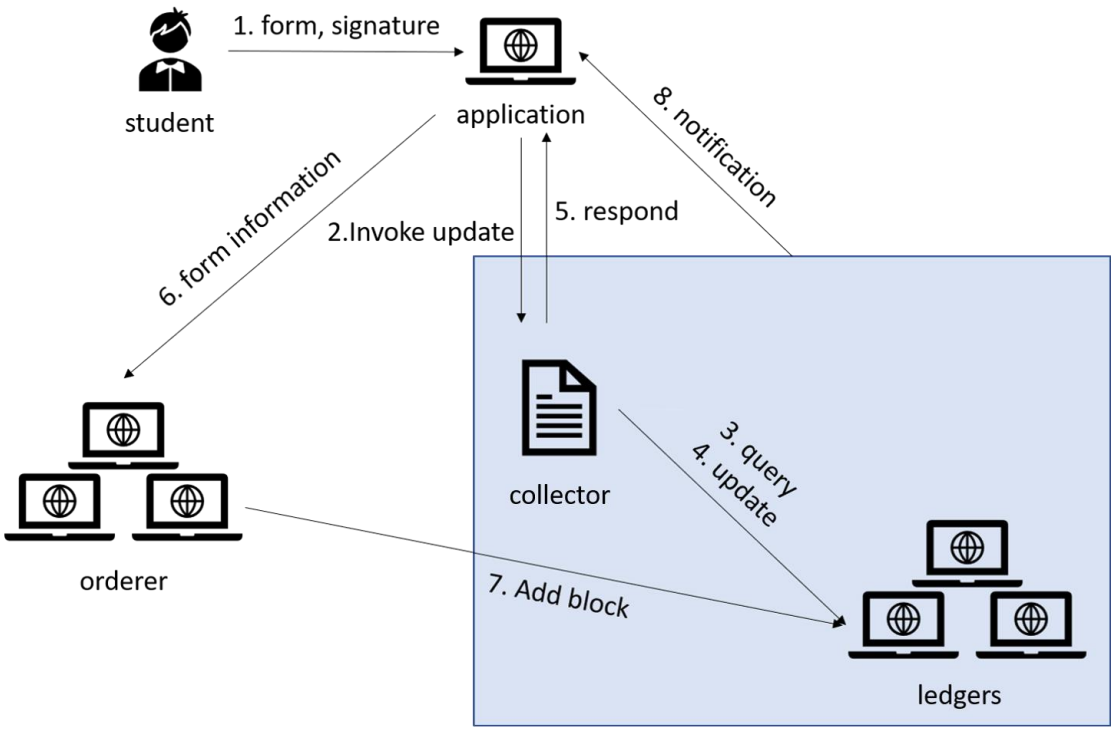


Figure 18: Workflow of evaluation form submission

3.3.1.4 Viewing results

When we want to see the results of evaluation, we can submit the course ID to see the overall results or submit the course ID and public key to see the particular form.

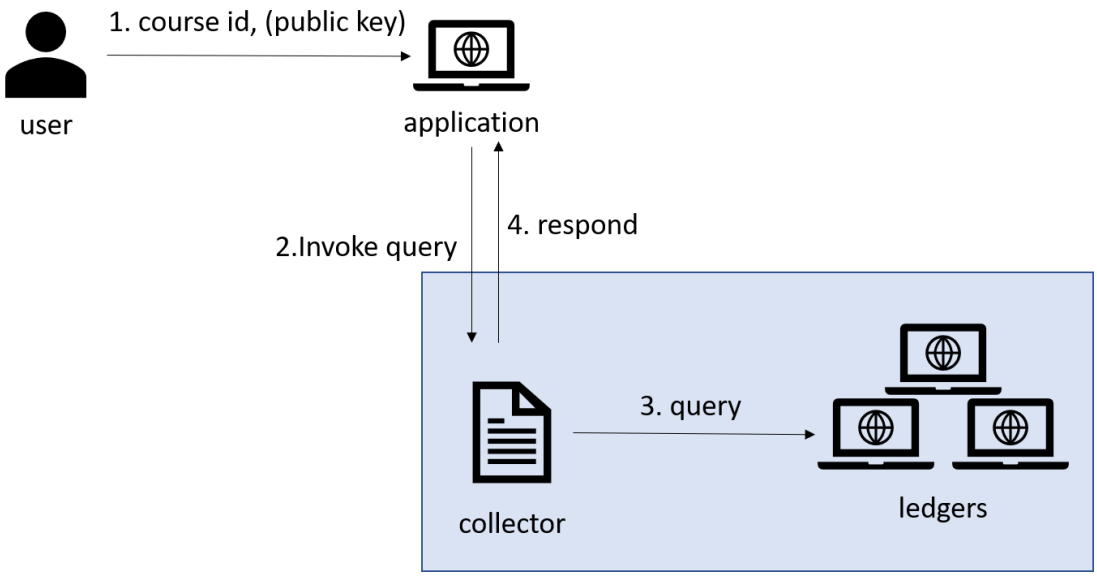


Figure 19: Workflow of viewing results

3.4 Expectation for proposed design

For our current design, the stored keys and evaluation information are not linked to any token. Therefore, we expect the token holders cannot use their tokens to figure out the owners of keys and evaluation forms.

When we show the results of evaluation on an application, the related public keys can be shown with the results. Students can check list of keys to ensure their keys and forms are involved in the evaluation.

3.5 Assumptions

When we design our system, we have made some assumptions:

3.5.1 The party who hold the system is honest

This assumption is the most important assumption. If this assumption fails, then our system will be failed.

For example, the faculty controls the token generation and distribution. If there is an unhonest staff try to affect the evaluation, he or she can make extra tokens and joins the evaluation. At the same time, if there are some students who do not do the evaluation, then the cheating may not be discovered as the number of received form is reasonable.

We do not have solution to solve this problem at this moment. However, making this assumption is acceptable since current traditional paper-based evaluation also makes this assumption.

3.5.2 Majority of peers are honest

As the consensus mechanism of Hyperledger Fabric is based on the endorsement of peers. If majority of peers are malicious, the orderer will provide incorrect ordering service.

We can increase the number of peers to ensure most peers are honest. Moreover, the peers should be hold by the faculty and the peers should be honest in this case.

3.5.3 The evaluation progress can be view at any time

In current design, the overall results of submitted form can be checked at any time. This may prove that results is not corrupted if all the students who have finished the evaluation check the results together. However, the students who have not submit their forms may be affected by the current results. This may not be fair to the teachers.

3.5.4 The identities of used token owners can be revealed

As students use their token, the faculty can know their identities by checking the used tokens. We think this is acceptable for an evaluation system. Even if we can achieve zero-knowledge proof [12] and proof that the participants are students without knowing their identities, this level of anonymity is not necessary for our course evaluation as all students should finish the evaluation.

We focus on the anonymity of form submission that no one knows who submit the form. The token and key pairs validate the identities of participants and keep the identities of form owner as a secret.

4 Implementation

4.1 Overview of 1st term implementation

In the 1st term, our goal is to implement a simplified version of the online evaluation system utilizing Hyperledger Fabric as the blockchain framework for the system. The system consists of two major parts: the blockchain network that handles the data transactions, and the clients and web interfaces for different roles in the system.

Hyperledger Fabric blockchain network is the core component that handles all the data transactions includes: adding tokens, verify tokens, adding evaluation results and querying evaluation results. All of these features are implemented by chaincodes to ensure the integrity and reliability of the transactions. For simplicity, the current version of the network only consists of one peer, but it can be easily expanded and will be done in 2nd term.

In order to communicate and interact with the network, different clients are needed to perform certain actions such as invoking different chaincodes. Web interface is provided for certain clients for ease of use.

4.2 Blockchain network

4.2.1 Framework and programming language

For the blockchain network, Hyperledger Fabric is used. It is an open-source permissioned distributed ledger technology platform, with over 35 organizations and nearly 200 developers in its development community [13]. Hyperledger Fabric is written in Go language, but it also provided Node.js SDK for chaincode [14] and application (client) [15]. We choose javascript as our main programming language, since javascript is flexible and easy to debug, while it is also one of the most well-known language for web development. Using the same programming language across the whole development cycle reduces hassles and provide the ease of integration for different parts of the system.

For database, we would use CouchDB [16].

4.2.2 System architecture

Here is a diagram of the blockchain network system architecture (figure 20):

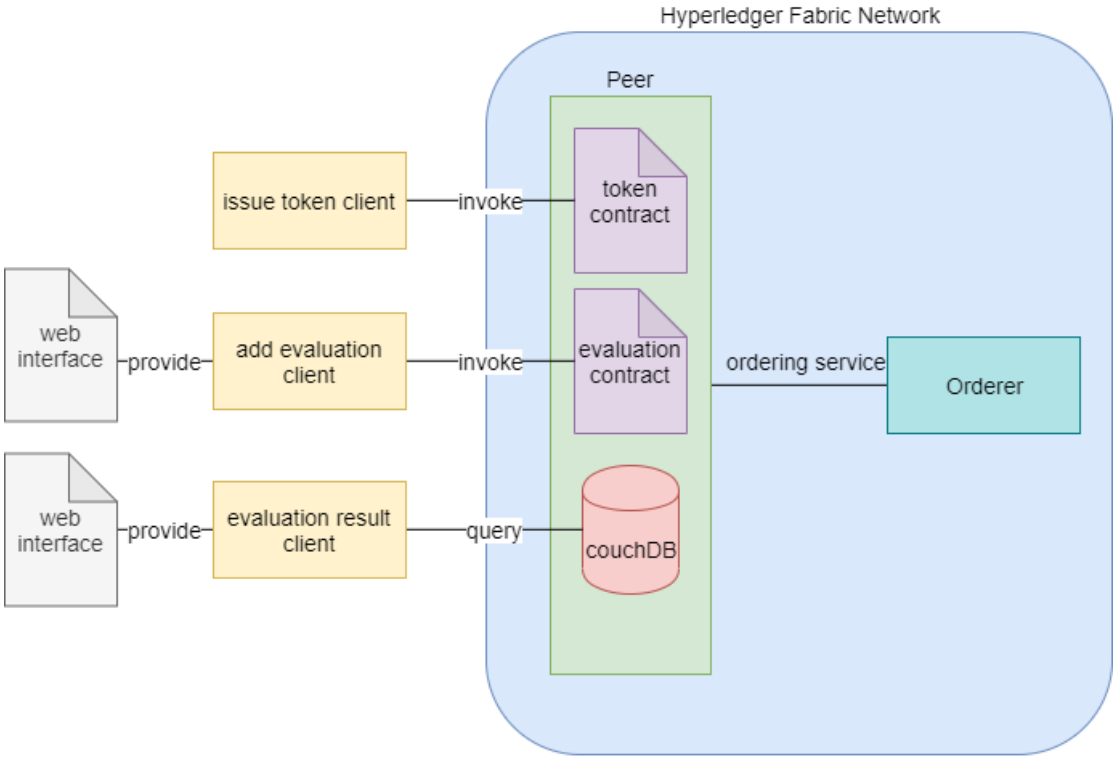


Figure 20: System architecture diagram of blockchain network

Peer

The peer in our current Hyperledger Fabric network handles token generation, evaluation verification and data storage.

When we send a request from the issue token client, the peer invokes the token contract. After the updating information is received by orderer, new blocks which contains updating information will be added in the ledger.

If someone try to submit an evaluation form through the add evaluation client, the peer invokes the evaluation contract and verifies the request by checking the provided token. Orderer will receive the updating information and add new blocks

To view the evaluation results, we can make a request through the evaluation result client and the peer will provide the results stored in the database.

Orderer

The orderer in our current Hyper Fabric network is responsible for the block generation and ordering. Once the clients send the information generated from the response of peer, the orderer will generate the block which stores the provided information.

Chaincode (smart contract)

Token contract is the chaincode which is responsible for querying and updating token information in the ledger.

Evaluation contract is the chaincode which is responsible for querying and updating evaluation information in the ledger.

The chaincode for managing key pairs is not yet implemented in this term.

Database

CouchDB is used for developing our system state database. As we choose to use javascript as our programming language for this system, the chaincode values are modeled as JSON data. CouchDB has richer query for such chaincode data [16].

Client

In current implementation, issue token client, add evaluation client and evaluation result client are implemented.

In issue token client, we can make requests of issuing new tokens for particular course evaluation. By giving the information of issuer, course ID and token expiration date, the client will request the system to generate a new token with the given information. Once the request is accepted, the we can view the token in the client side. Currently there is no GUI for the issue token client and all operations have to be done in command line. The GUI will be provided in next term.

We can submit our evaluation form through add evaluation client. In web UI, an evaluation form is provided. Once we finish evaluation, we can submit the form and the client will make a request of adding evaluation.

The results of course evaluation can be viewed in evaluation result client. In the web UI, we can submit the course ID to view the target course evaluation results.

4.3 Web interface

4.3.1 Framework and programming language

We use React-Bootstrap as our web interface framework. React-Bootstrap is a library which is a re-implementation of Bootstrap with React [17]. Bootstrap is a popular front-end development framework [18] while React is a popular javascript library for creating UI components [19]. We choose React-Bootstrap because we have use it in another project and we are more familiar with it.

4.3.2 User Interface

4.3.2.1 Evaluation

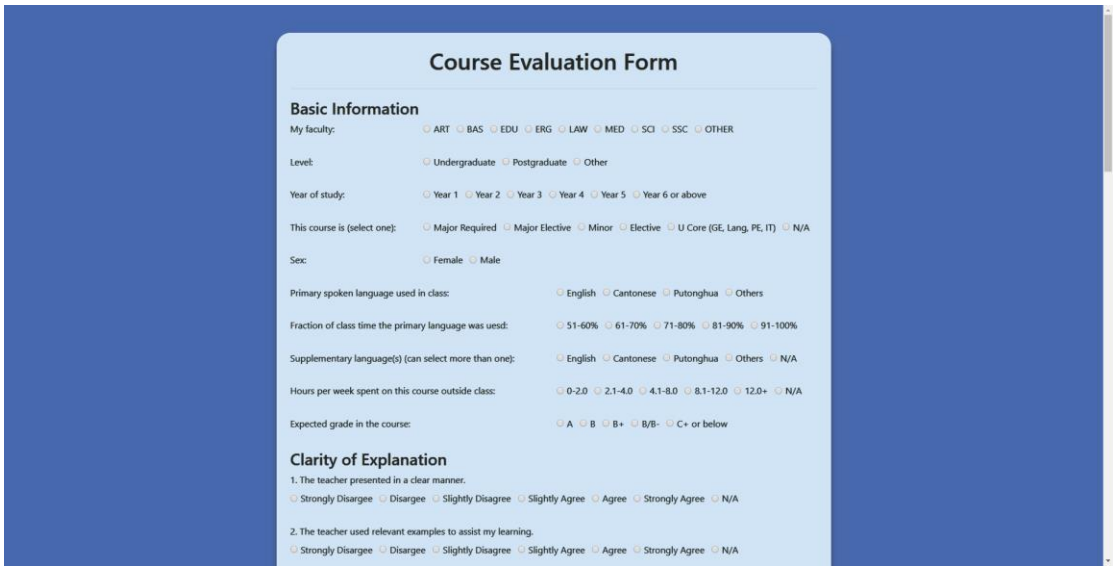


Figure 21: User interface of evaluation form part 1

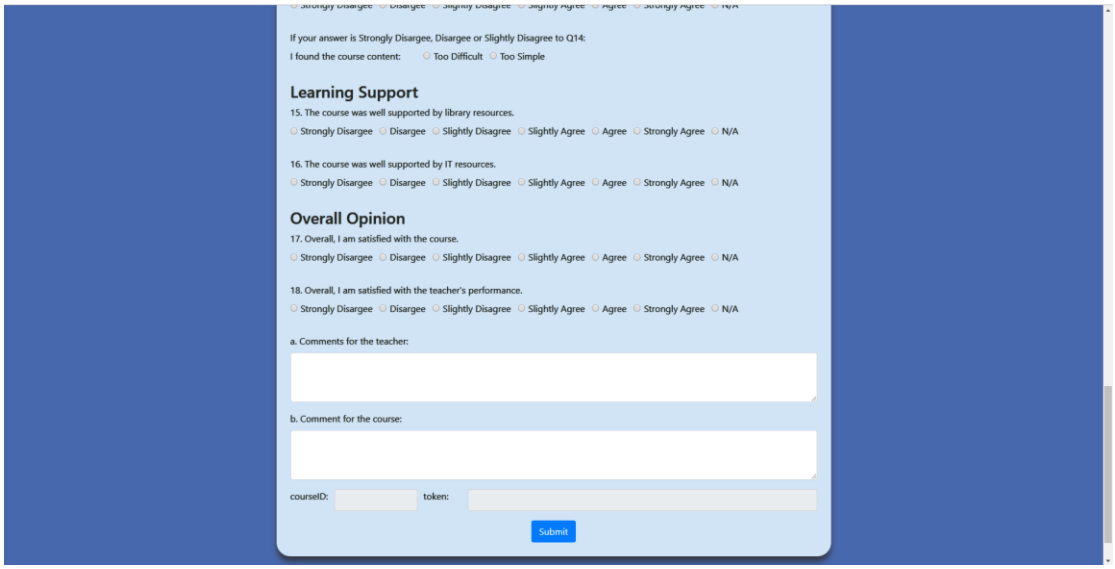


Figure 22: User interface of evaluation form part 2

The evaluation form interface consists of all the multiple-choice questions and the open-ended questions of a normal paper-based evaluation form. Users can easily pick their answers and fill in their comments. The course ID and token section is disabled for manual input and currently empty, but those value would be automatically filled in when opening the URL provided to the user in the actual evaluation process. It would be more convenient and prevent manual input error.

4.3.2.2 Summary

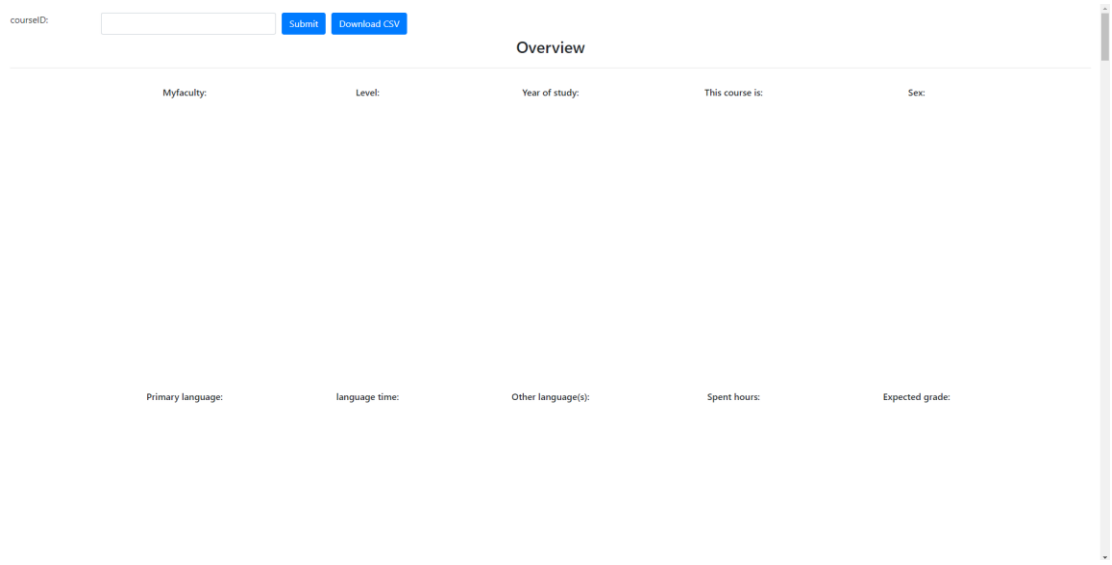


Figure 23: User interface of evaluation result (default)

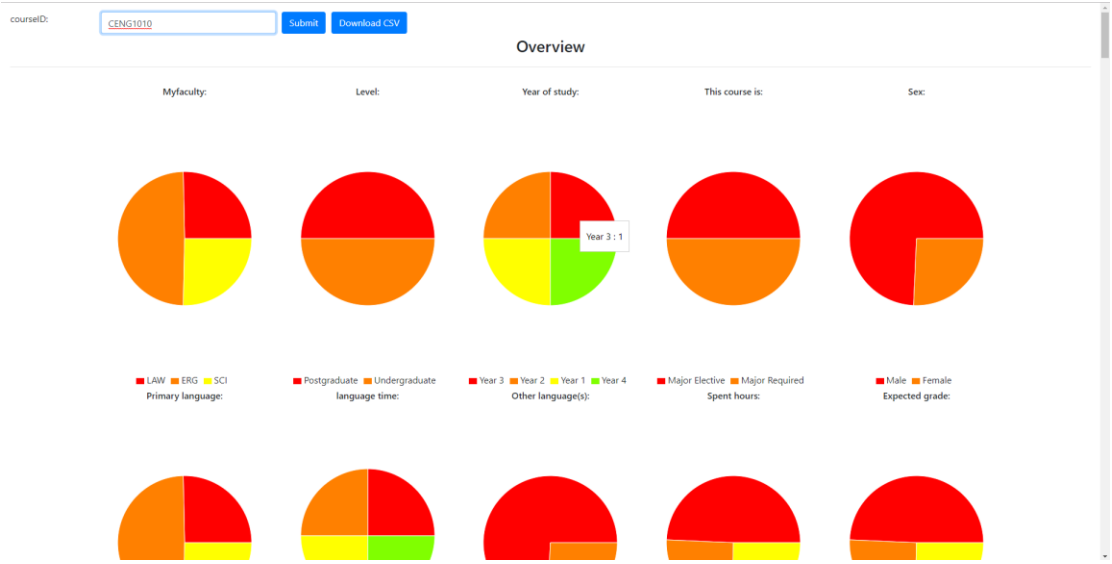


Figure 24: User interface of evaluation result (with course ID entered)

Figure 25: User interface of evaluation result (open-ended questions)

The summary page provides a visualization of the evaluation results. It consists of pie charts for each multiple-choice question and a list of comments for the open-ended questions. Also, it provides the option to download the result in CSV format for other uses.

4.4 Demonstration

4.4.1 Create token

A token is required for each user in each course to submit their evaluation. In order to generate a token, a node.js command-line program is written so that the administrator uses it.

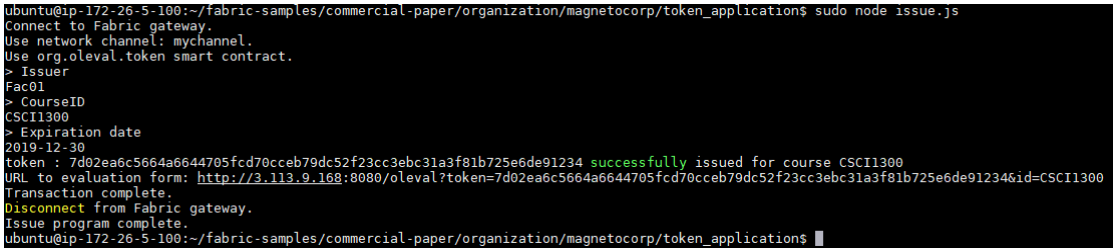


Figure 26: Command line interface of token generation program

Input the necessary information, and the program would invoke the token contract's issue method, which would add a new token into the ledger. The program would also provide a URL to the corresponding evaluation form, which needs to be delivered to the user in a reliable way such as sending it to the student's email.

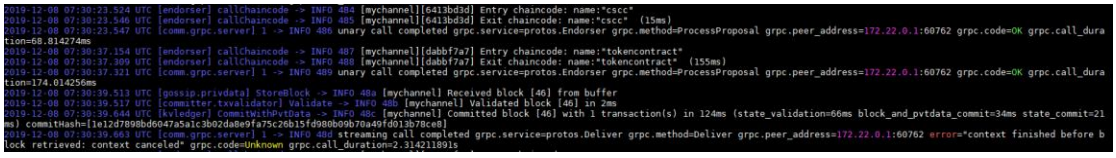
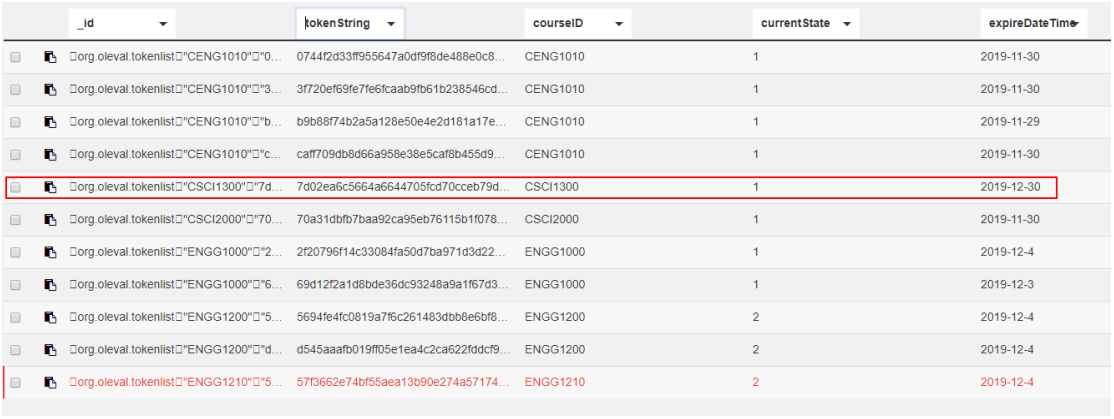


Figure 27: Peer logs of invoking contract and add block



_id	tokenString	courseID	currentState	expireDateTime
□org.oleval.tokenlist□"CENG1010"□"0…	0744f2d33ff955647a0df9f8de488e0c8…	CENG1010	1	2019-11-30
□org.oleval.tokenlist□"CENG1010"□"3…	3f720ef69fe7fe6fcaab9fb61b238546cd…	CENG1010	1	2019-11-30
□org.oleval.tokenlist□"CENG1010"□"b…	b9b88f74b2a5a128e50e4e2d181a17e…	CENG1010	1	2019-11-29
□org.oleval.tokenlist□"CENG1010"□"c…	caff709db8d66a958e38e5caf8b455d9…	CENG1010	1	2019-11-30
□org.oleval.tokenlist□"CSCI1300"□"7d…	7d02ea6c5664a6644705fcd70cceb79d…	CSCI1300	1	2019-12-30
□org.oleval.tokenlist□"CSCI2000"□"70…	70a31dbfb7baa92ca95eb76115b1f078…	CSCI2000	1	2019-11-30
□org.oleval.tokenlist□"ENGG1000"□"2…	2f20796f14c33084fa50d7ba971d3d22…	ENGG1000	1	2019-12-4
□org.oleval.tokenlist□"ENGG1000"□"6…	69d12f2a1d8bde36dc93248a9a1f67d3…	ENGG1000	1	2019-12-3
□org.oleval.tokenlist□"ENGG1200"□"5…	5694fe4fc0819a7f6c261483dbb8e6bf8…	ENGG1200	2	2019-12-4
□org.oleval.tokenlist□"ENGG1200"□"d…	d545aaafb019ff05e1ea4c2ca622fddcf9…	ENGG1200	2	2019-12-4
□org.oleval.tokenlist□"ENGG1210"□"5…	57f3662e74bf55aea13b90e274a57174…	ENGG1210	2	2019-12-4

Figure 28: Database records of the added token

At the blockchain network side, token contract got invoked and then go through endorsement process. Once the invocation finished, new ledger record is generated and pack into a new block. The peer will then validate this new block then commit it, while also update the database record of the ledger. Thus new token is generated and put into the ledger.

4.4.2 Submit evaluation

(create fake email)

http://3.113.9.168:8080/oleval/?token=7d02ea6c5664a6644705fcd70cceb79dc52f23cc3ebc31a3f81b725e6de91234&id=CSCI1300

Once the student received the email, they can click into the attached URL to access the course evaluation form.

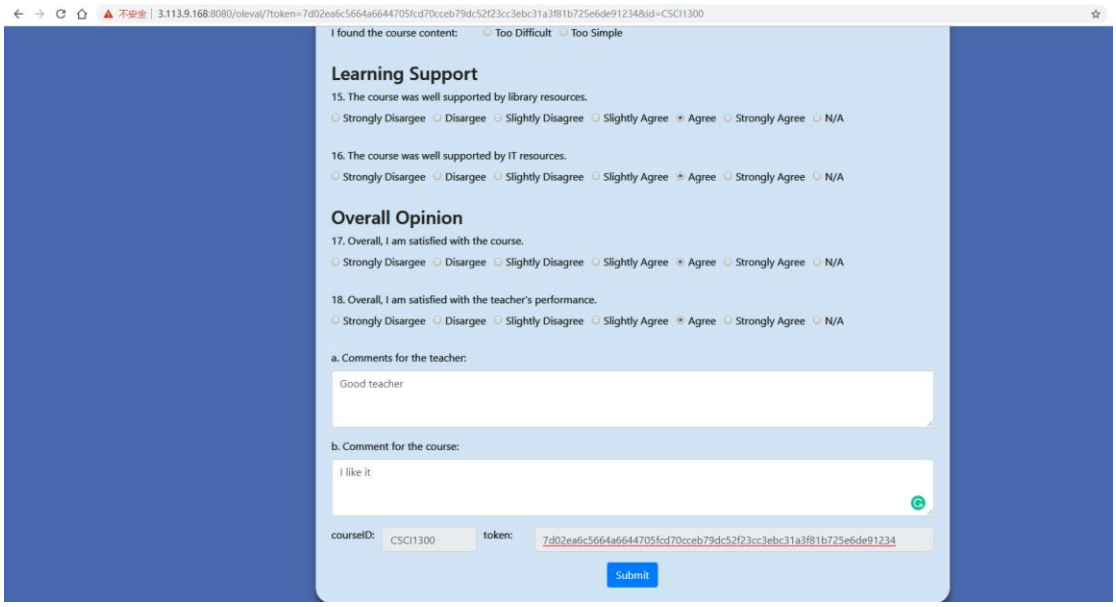


Figure 29: A filled evaluation form

Then, in the evaluation, course ID and token section will be automatically filled in by the system. The students choose their answers, fill in their comments and click submit button to upload their evaluation form.

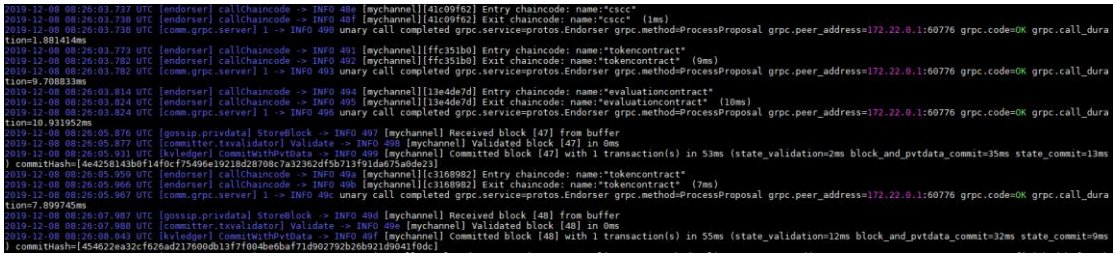


Figure 30: Peer logs of invoking contract and add block

_id	tokenString	courseID	currentState	expireDateTime
□org.oleval.tokenlist□"CENG1010"□"0...	0744f2d33ff955647a0df9f8de488e0c8...	CENG1010	1	2019-11-30
□org.oleval.tokenlist□"CENG1010"□"3...	3f720ef69fe7fe6fcaab9fb61b238546cd...	CENG1010	1	2019-11-30
□org.oleval.tokenlist□"CENG1010"□"b...	b9b88f74b2a5a128e50e4e2d181a17e...	CENG1010	1	2019-11-29
□org.oleval.tokenlist□"CENG1010"□"c...	caff709db8d66a958e38e5caf8b455d9...	CENG1010	1	2019-11-30
□org.oleval.tokenlist□"CSCI1300"□"7d...	7d02ea6c5664a6644705fcd70cceb79d...	CSCI1300	2	2019-12-30
□org.oleval.tokenlist□"CSCI2000"□"70...	70a31dbfb7baa92ca95eb76115b1f078...	CSCI2000	1	2019-11-30
□org.oleval.tokenlist□"ENGG1000"□"2...	2f20796f14c33084fa50d7ba971d3d22...	ENGG1000	1	2019-12-4
□org.oleval.tokenlist□"ENGG1000"□"6...	69d12f2a1d8bde36dc93248a9a1f67d3...	ENGG1000	1	2019-12-3
□org.oleval.tokenlist□"ENGG1200"□"5...	5694fe4fc0819a7f6c261483dbb8e6bf8...	ENGG1200	2	2019-12-4
□org.oleval.tokenlist□"ENGG1200"□"d...	d545aaafb019ff05e1ea4c2ca622fddcf9...	ENGG1200	2	2019-12-4
□org.oleval.tokenlist□"ENGG1210"□"5...	57f3662e74bf55aea13b90e274a57174...	ENGG1210	2	2019-12-4

Figure 31: Updated token record

Similar to adding a new token, 2 contracts are invoked and 2 new records of the ledger are packed into new blocks. Going through similar process, new evaluation record would be added and the token's state would be updated to mark it as consumed.

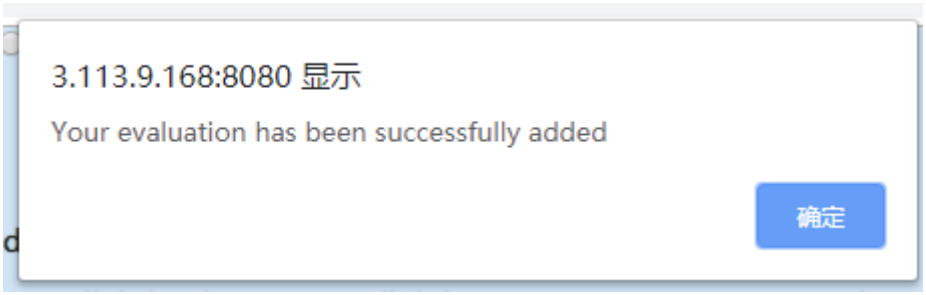


Figure 32: Success message

If the previous process finishes successfully, the server would send a successful message to the user, and a message would pop up to inform the user.

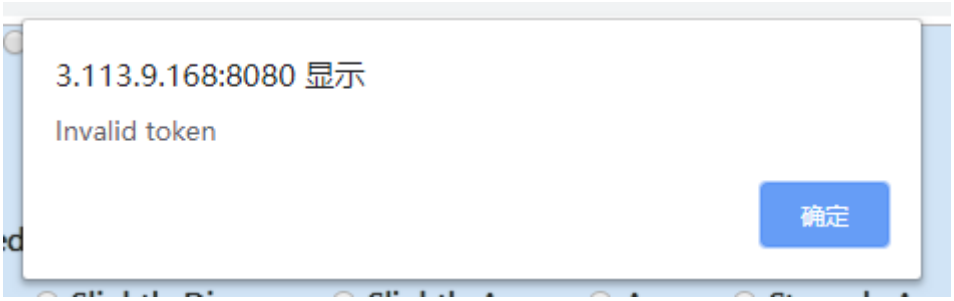


Figure 33: Failure message

But if the token is invalid (in case such as repeat submission), a failure message would be shown instead, and the evaluation results will not be added into the ledger.

4.4.3 View result

To view the evaluation result, the user can access the summary web interface, input the course ID they want to query, and click the submit button. The website would then fetch the data from the peer's database and generate the charts.

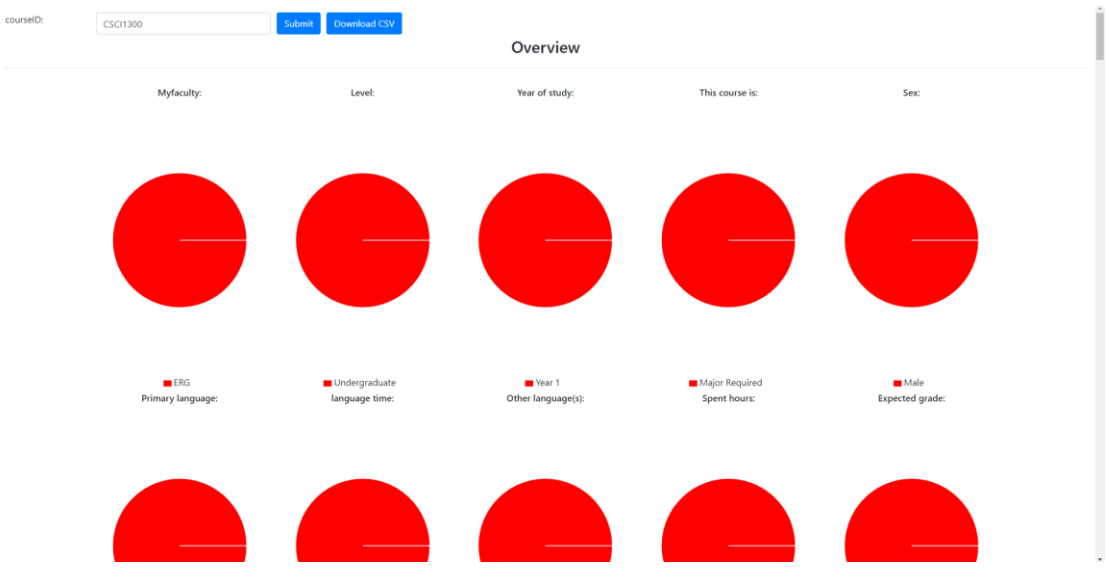


Figure 34: Multiple-choice questions result

Figure 35: Open-ended questions result

In this case, only one evaluation has been submitted, so all the charts consist of only one result. And then the comments would be listed out in the last section.

5 Conclusion

5.1 1st Term Summary

In Hong Kong, online course evaluation is not popular based on our experience. From secondary school to university, we never have a course evaluation which is not paper-based. This is very strange as online course evaluation has more advantages. We decided to make our own anonymous online course evaluation system, even though we will not have the chance to use it.

In this term, we have studied the blockchain technologies and some current e-voting systems. Although we may not fully understand all the related topics, we have designed our own course evaluation system and a simple course evaluation system is implemented with our limited understanding.

Our simple course evaluation system is built on a Hyperledger fabric network and we can use tokens to verify the participant identities and run an evaluation demonstration. However, the current implementation still misses some key components to achieve our expectations on the proposed design. In the next term, we will focus on completing the design and testing our system.

5.2 2nd Term Objectives

In the second term, we have the following objectives:

Adding key pair contract into peer

In our proposed design, there should be a chaincode which is responsible for updating valid public key information. Without this chaincode, we cannot use the key pair to verify the submitted evaluation form. Implementing the chaincode will be our first objective.

Adding multiple peers

Currently, we only have one peer in our network and this is not enough for forming a reliable distributed network. Expanding the network will be another objective.

Data encryption

The collected form data is not encrypted in current implementation. It can provide further data protection if the data is encrypted in the browser. In next semester, we will attempt to do the encryption. If it is possible, we will try to use homomorphic encryption on our evaluation mc answer.

UI design

The current UI design is not appealing and there is even no GUI for the issue token client. The UI of our application will be redesigned in next term.

References

[1] J. Benaloh, R. Rivest, P. Y. A. Ryan, P. Stark, V. Teague and P. Vora, "End-to-end verifiability," arXiv preprint arXiv:1504.03778, 2014.

[2] S. Zhang, J. Lee, "Analysis of the main consensus protocols of blockchain," ICT Express, 12 8 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S240595951930164X [Accessed: 2 12 2019].

[3] S. Cocco, G. Singh, "Top 6 technical advantages of Hyperledger Fabric for blockchain networks," developer.ibm.com, 18 3 2018. [Online]. Available: https://developer.ibm.com/articles/top-technical-advantages-of-hyperledger-fabric-for-blockchain-networks/. [Accessed 2 12 2019].

[4] Hyperledger, "Peers," in Hyperledger Fabric Doc, [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-1.4/peers/peers.html. [Accessed 2 12 2019].

[5] "Online Course Evaluations | Registrar's Office," Stanford University, [Online]. Available: https://registrar.stanford.edu/students/online-course-evaluations. [Accessed 2 12 2019].

[6] "Helios Voting," Helios, [Online]. Available: https://heliosvoting.org/. [Accessed 2 12 2019].

[7] "Privacy," Helios, [Online]. Available: https://heliosvoting.org/privacy. [Accessed 2 12 2019].

[8] "Follow My Vote: The Online Voting Platform of The Future," Follow My Vote, [Online]. Available: https://followmyvote.com/. [Accessed 2 12 2019].

[9] "Elliptic Curve Cryptography & Online Voting," Follow My Vote, [Online]. Available: https://followmyvote.com/online-voting-technology/elliptic-curve-cryptography/. [Accessed 2 12 2019].

[10] "Polys — Online Voting System," Polys, [Online]. Available: https://polys.me/. [Accessed 2 12 2019].

[11] M. Riveiro, "How does Polys achieve anonymity and validate voters?," Polys, [Online]. Available: https://docs.polys.me/en/articles/1740042-how-does-polys-achieve-anonymity-and-validate-voters. [Accessed 2 12 2019].

[12] Uriel Feige, Amos Fiat, Adi Shamir, "Zero-knowledge proofs of identity," *Journal of Cryptology,* vol. 1, no. 2, p. 77, 1988.

[13] "Introduction," Hyperledger Fabric Doc, [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/latest/whatis.html#. [Accessed 2 12 2019].

[14] Hyperledger, "fabric-chaincode-node," GitHub repository, 2019. [Online]. Available: https://github.com/hyperledger/fabric-chaincode-node.

[15] Hyperledger, "fabric-sdk-node," GitHub repository, 2019. [Online]. Available: https://github.com/hyperledger/fabric-sdk-node.

[16] "CouchDB as the State Database," Hyperledger Fabric Doc, [Online]. Available:

https://hyperledger-fabric.readthedocs.io/en/release-1.4/couchdb_as_state_database.html. [Accessed 2 12 2019].

[17] "React Bootstrap," [Online]. Available: https://react-bootstrap.github.io. [Accessed 2 12 2019].

[18] "Bootstrap," [Online]. Available: https://getbootstrap.com/. [Accessed 2 12 2019].

[19] "React – A JavaScript library for building user interfaces," [Online]. Available: https://reactjs.org/. [Accessed 2 12 2019].