



COMP5521 DISTRIBUTED LEDGER TECHNOLOGY, CRYPTOCURRENCY AND E-PAYMENT

2024-Fall Project: Student Attendance Application Based on NaiveCoin

Presenter: MA Zhiyuan, QIN Cailing

November 25, 2024



Contents

- 1** Introduction
- 2** Analysis & Design
- 3** Implementation
- 4** Demonstration



□ Project Background

❖ Problem Statement:

- To build a blockchain-powered student attendance system based on the **NaiveCoin** project.

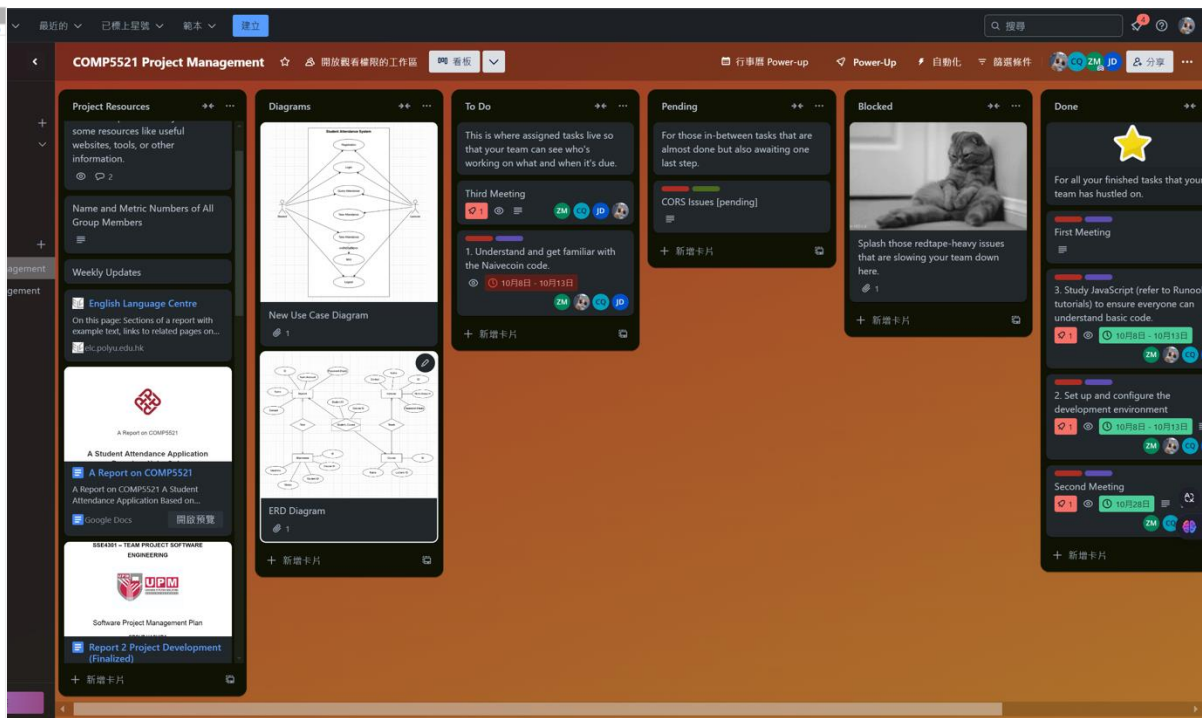
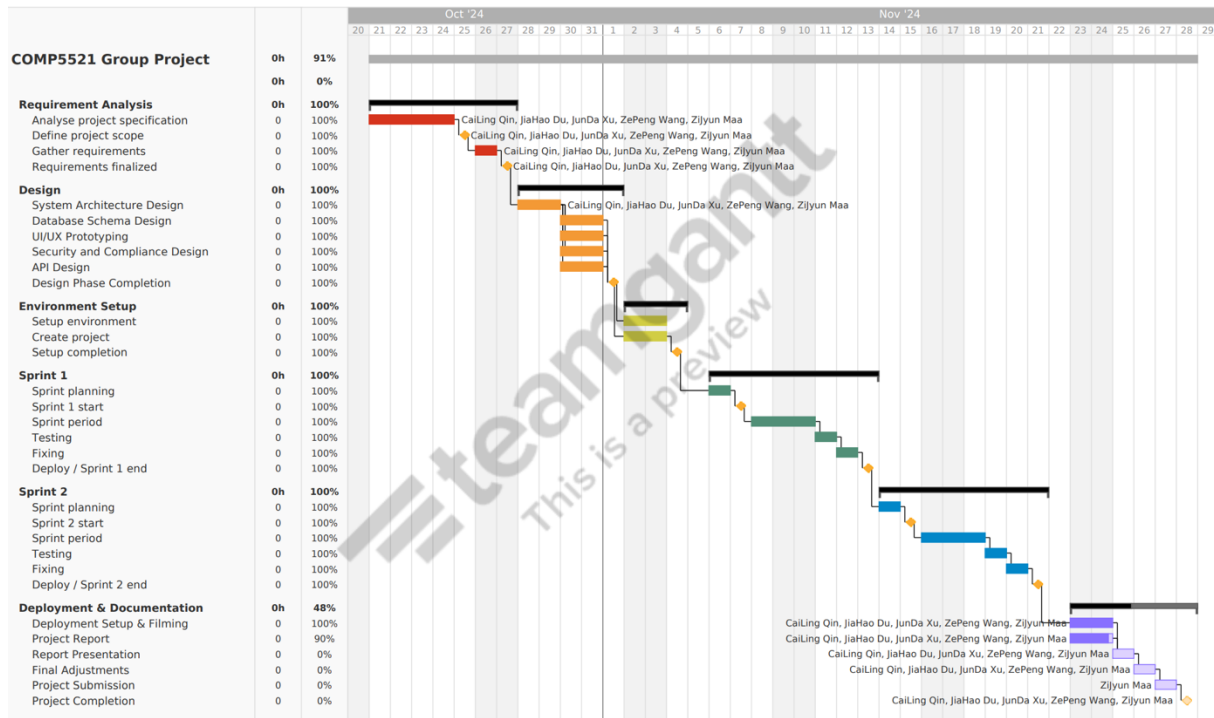
❖ Objective:

- Design and implement a blockchain-based attendance management system based on the **NaiveCoin**, supporting core functions such as student sign-in, attendance record and query.

❖ Methodology:

- We used **RAD** as our methodology of project development. Because of **SHORT deadlines** and **FREQUENT changes**.

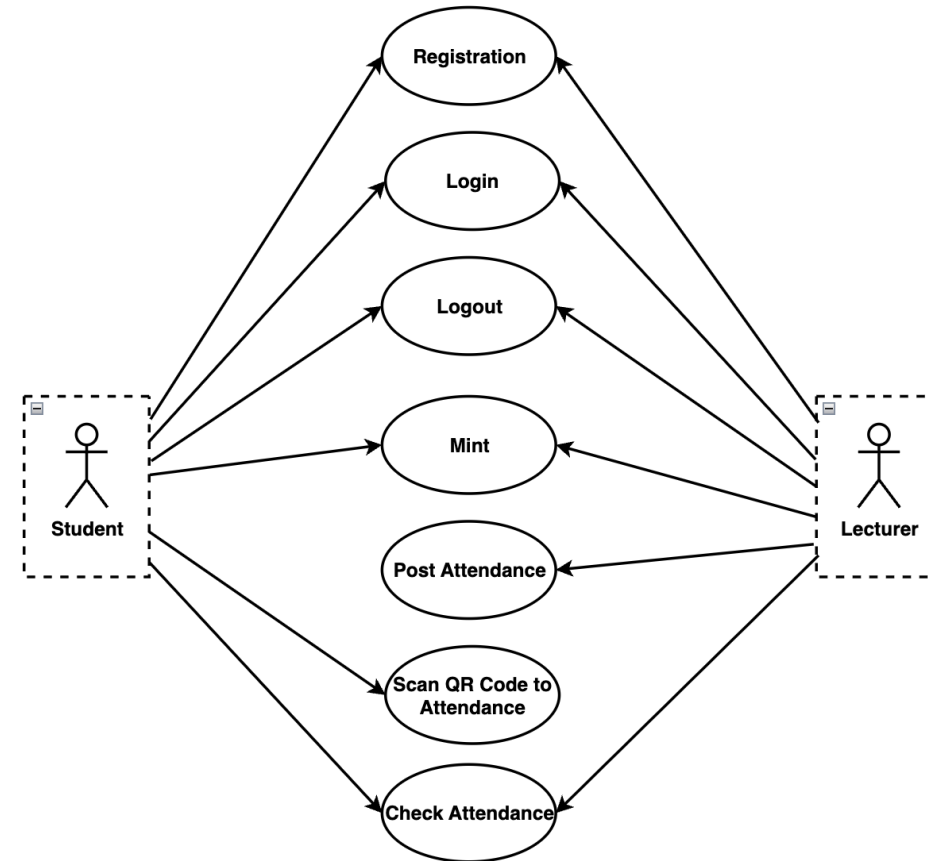
□ We use **Gantt charts** to plan tasks and allocate work, and **Trello** to share information and track progress in real time.



Requirement Analysis

Use Case Diagram

- ❖ Identify Actors and Interactions
- ❖ Define Functional Requirements
- ❖ Scope Validation



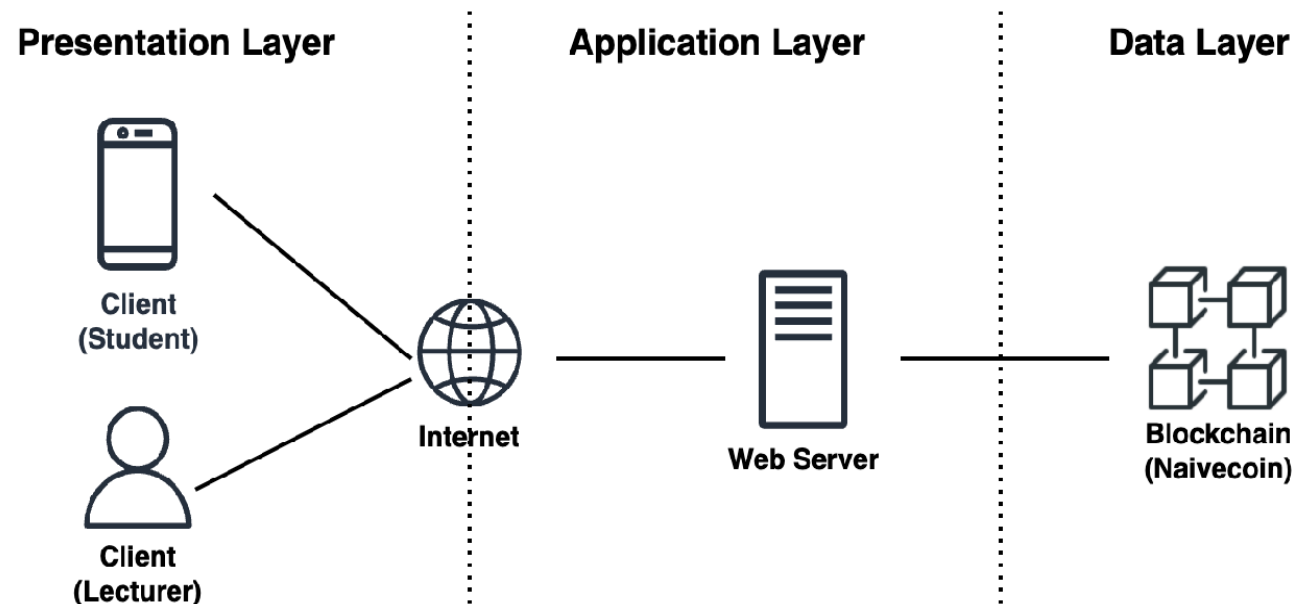


Source	Type	Priority	Requirement Description	Status
Stakeholder	Non-Func.	High	<i>*Dynamic difficulty</i>	Finished
Stakeholder	Non-Func.	High	<i>*Fork resolution mechanism</i>	Finished
User	Func.	Medium	Registration	Finished
User	Func.	Medium	Login	Finished
User	Func.	Medium	Logout	Finished
User	Func.	High	New Attendance/ Event	Finished
User	Func.	High	Query Attendance	Finished
User	Func.	High	*Take Attendance/ Sign Certificate	Finished
User	Func.	High	*Mint (Mining) with Certificate Verification	Finished
User	Non-Func.	Medium	<i>User-friendly Operating Interfaces</i>	Finished



❏ Presentation Layer

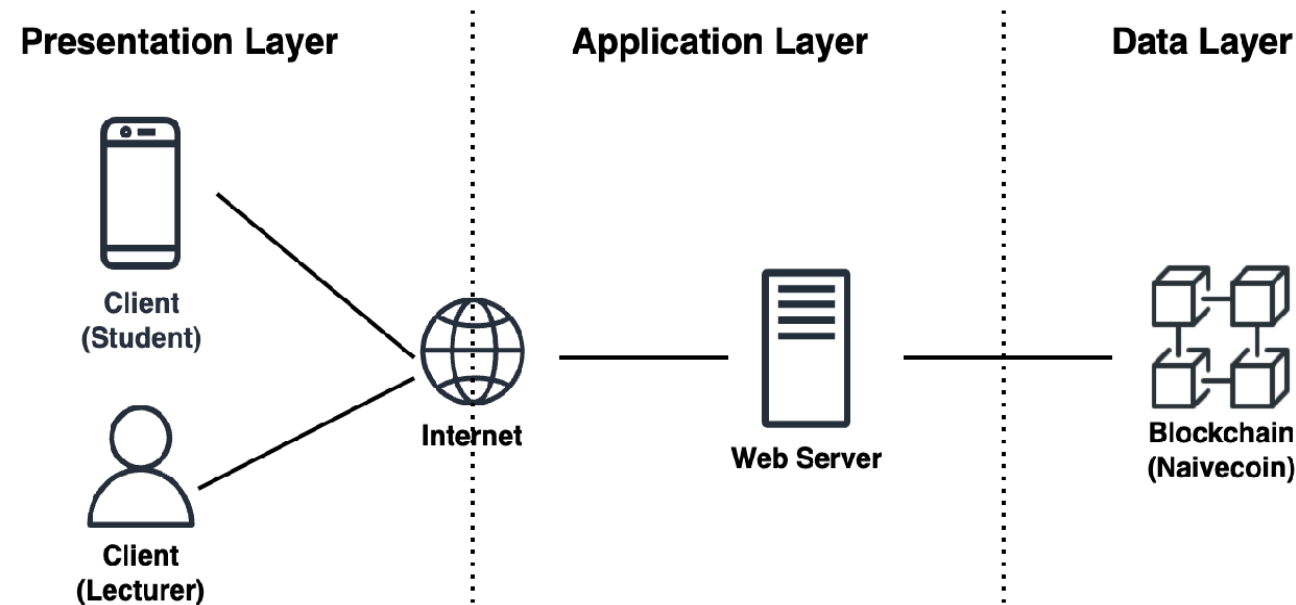
- ❖ Frontend: **Next.js**
- ❖ Handles user interaction, provides UI, and forwards requests to the backend.





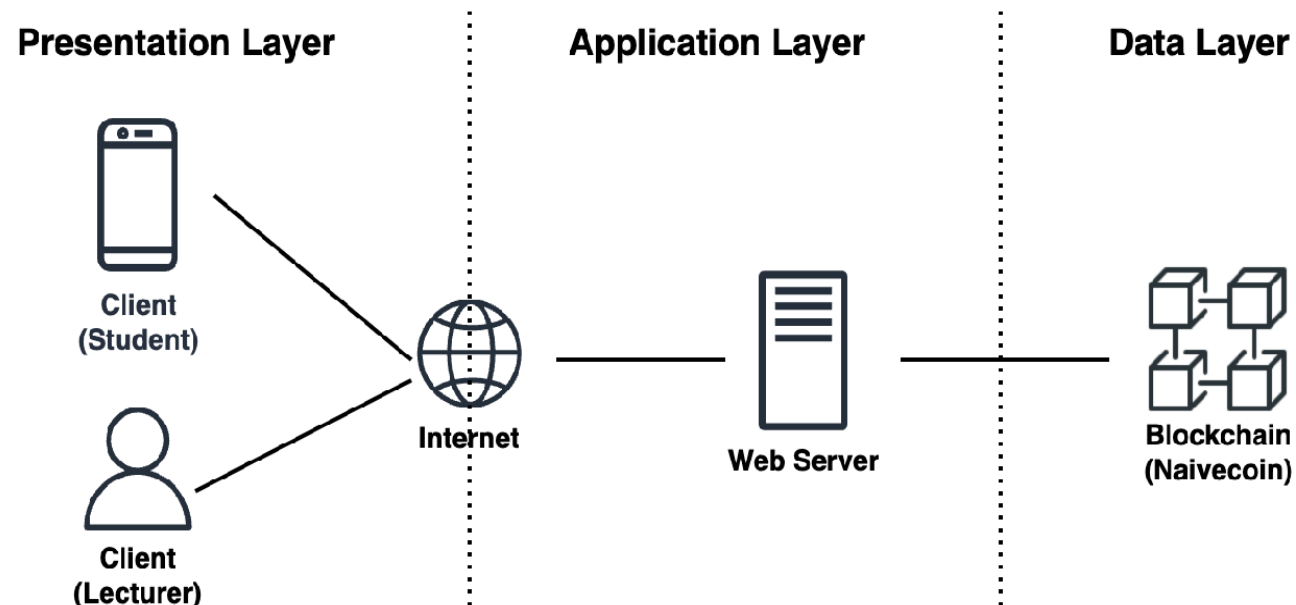
Business Logic Layer

- ❖ Backend: **Node.js**
- ❖ Responsible for implementing the core blockchain logic and processing requests from the frontend.



❏ Data Layer

- ❖ Ledgers (**NaiveCoin**)
- ❖ Where data about blocks, transactions, and blockchain history is stored



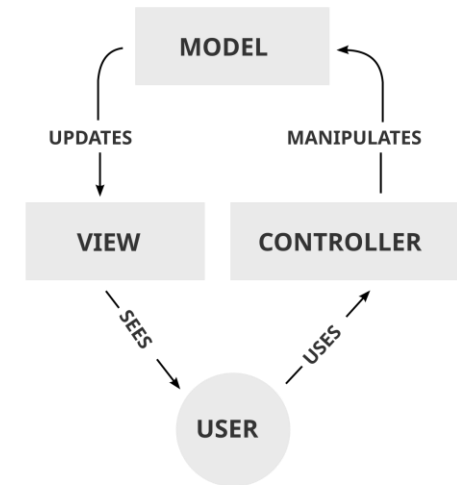
□ Front-End Development

❖ Development Framework:

- Next.js - *Fast routing, High performance, Easy to use*
- MVC Pattern – *Model, View, Controller*

❖ CSS Style:

- Tailwind





□ Authentication

❖ Functions:

- **Registration**
- **Login**
- **Logout**

❖ Use **LocalStorage** to store user (lecturer & student) information, you can only use this application if the user information exists in **LocalStorage**.

```
const setAttribute = async (
  name: string,
  value: string
): Promise<boolean> => {
  try {
    localStorage.setItem(name, value);
    console.log(`Set ${name}: ${value} in localStorage`);
    return true;
  } catch (error) {
    console.error(`Failed to set ${name} in localStorage:`, error);
    return false;
  }
};
```

```
useEffect(() => {
  const user = LocalStorage().getAttribute('user');
  if (!user) {
    router.push('/login'); // 用户未登录时跳转到登录页
  }
}, [router]);
```



□ Attendance – Event (Lecturer & Student):

❖ Lecturer:

- *Lecturers* records the Information of the created Event on the blockchain by **sending transaction to themselves**.

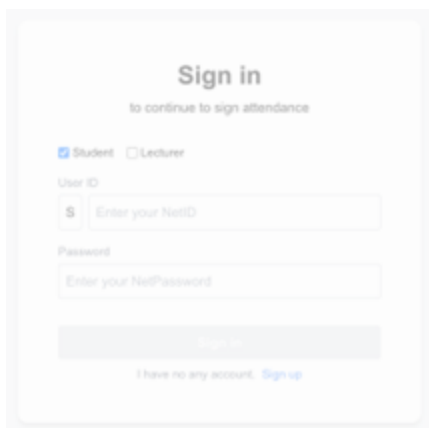
❖ Student:

- *Students* sign in existing events by **sending transactions to** the event owners (**Lecturers**).

Tips: Transactions created by *Lectures* are different from those created by *Students*.
“<eventId>-created” & “stuld” is void.

□ Data Structure

- ❖ Class Diagram
- ❖ To show what kind of attributes we used for contain the Event and User information to communicate between Frontend and Backend.



Sign in
to continue to sign attendance

☒ Student ☐ Lecturer

User ID
\$ Enter your NetID

Password
Enter your NetPassword

Log in

I have no any account. [Sign up](#)



UserClass
walletId: string
userId: string
address: string
constructor(walletId: string, userID: string, address: string): UserClass
toJSON(): object
fromJSON(json: any): UserClass

EventClass
fromAddress: string
toAddress: string;
amount: number;
changeAddress: string;
stuld: string;
teacherId: string
deadline: string
remark: string
timestamp?: number
constructor(fromAddress: string, toAddress: string, amount: number, changeAddress: string, stuld: string, teacherId: string, eventId: string, deadline: string, remark: string, timestamp?: number): EventClass
toJSON(): object
fromJSON(json: any): EventClass



```
{"fromAddress":"050174cbfb1b5670ad7e5ee3de22cd9bf544ddd5cf8b76ac2053283889189f12","toAddress":"","amount":0,"changeAddress":"050174cbfb1b5670ad7e5ee3de22cd9bf544ddd5cf8b76ac2053283889189f12","stuld":"","teacherId":"L200801","eventId":"COMP1234-create","deadline":"2024-11-24T11:50","remark":"Lecture1"}
```

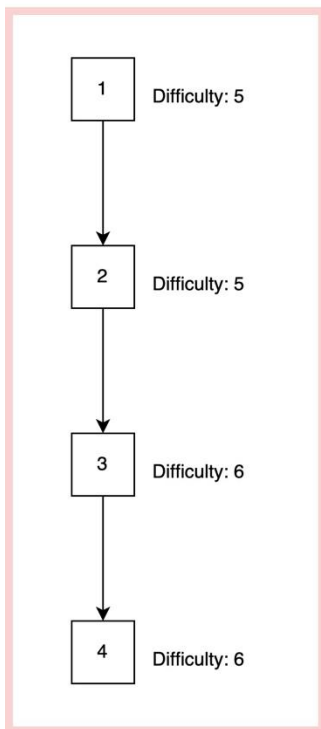




Basic Fork Resolution

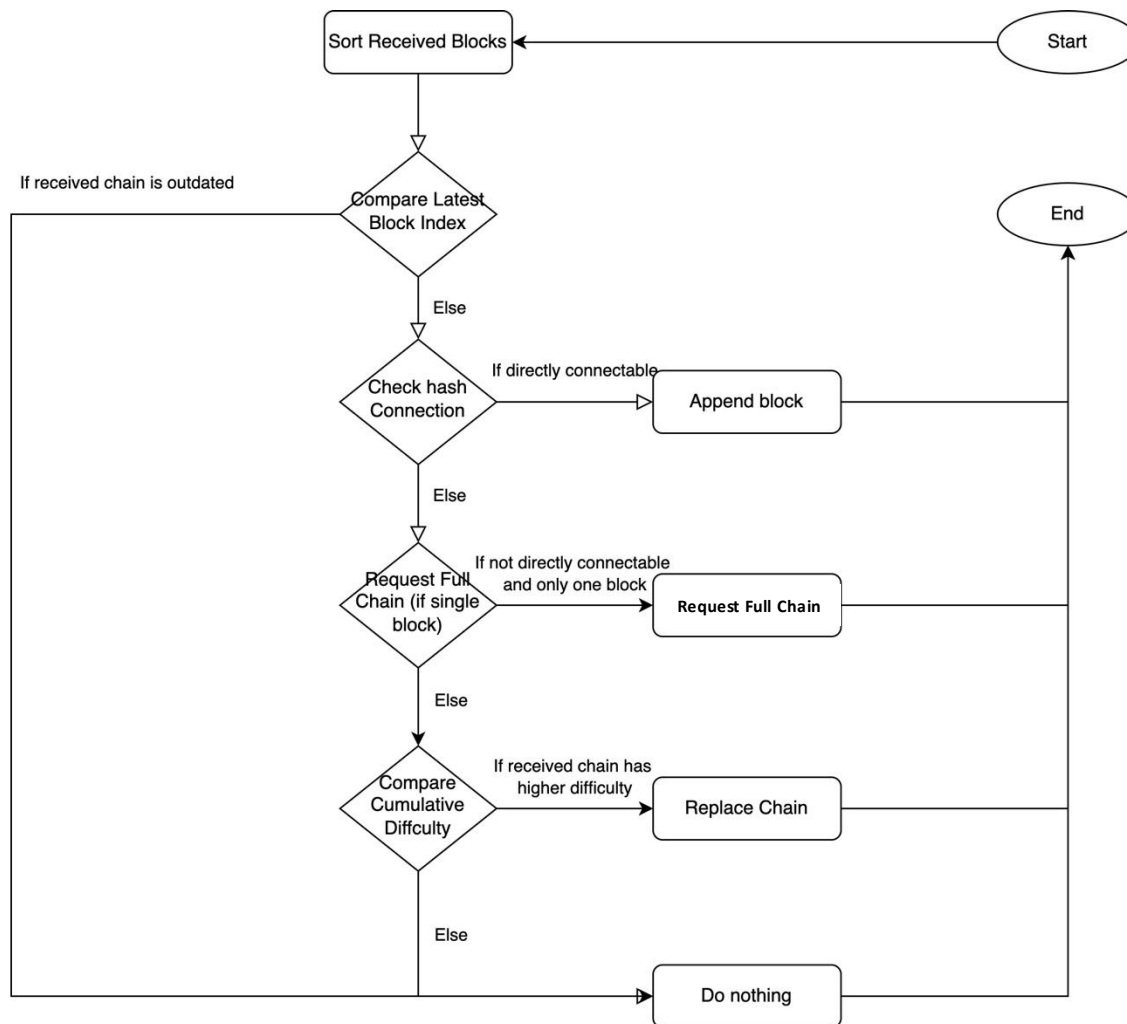
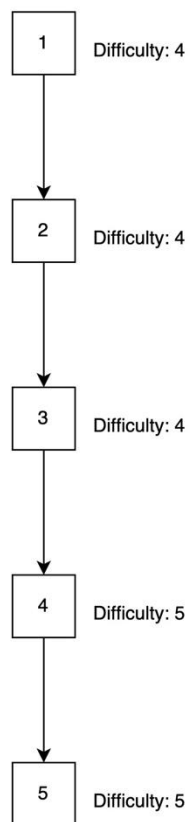
Chain A

$$2^5 + 2^5 + 2^6 + 2^6 = 192$$



Chain B

$$2^4 + 2^4 + 2^4 + 2^5 + 2^5 = 112$$





❏ Dynamic Difficulty

❖ Purpose:

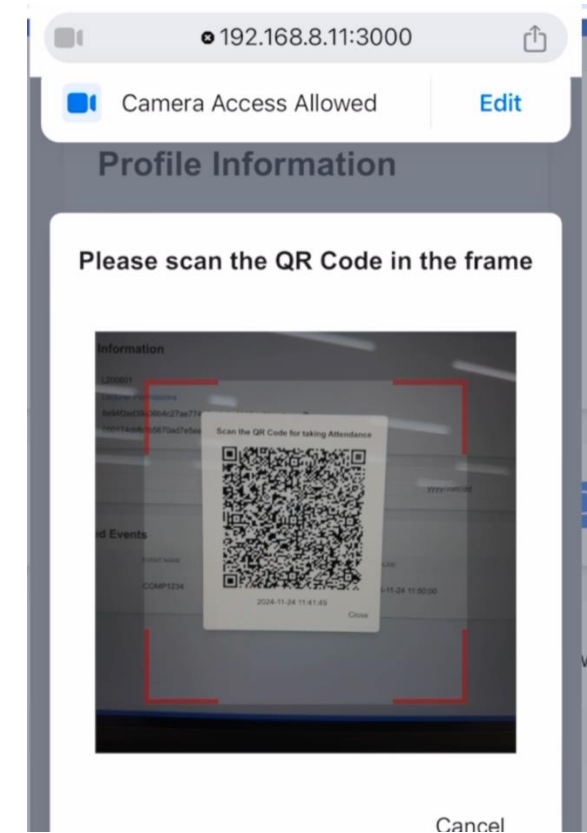
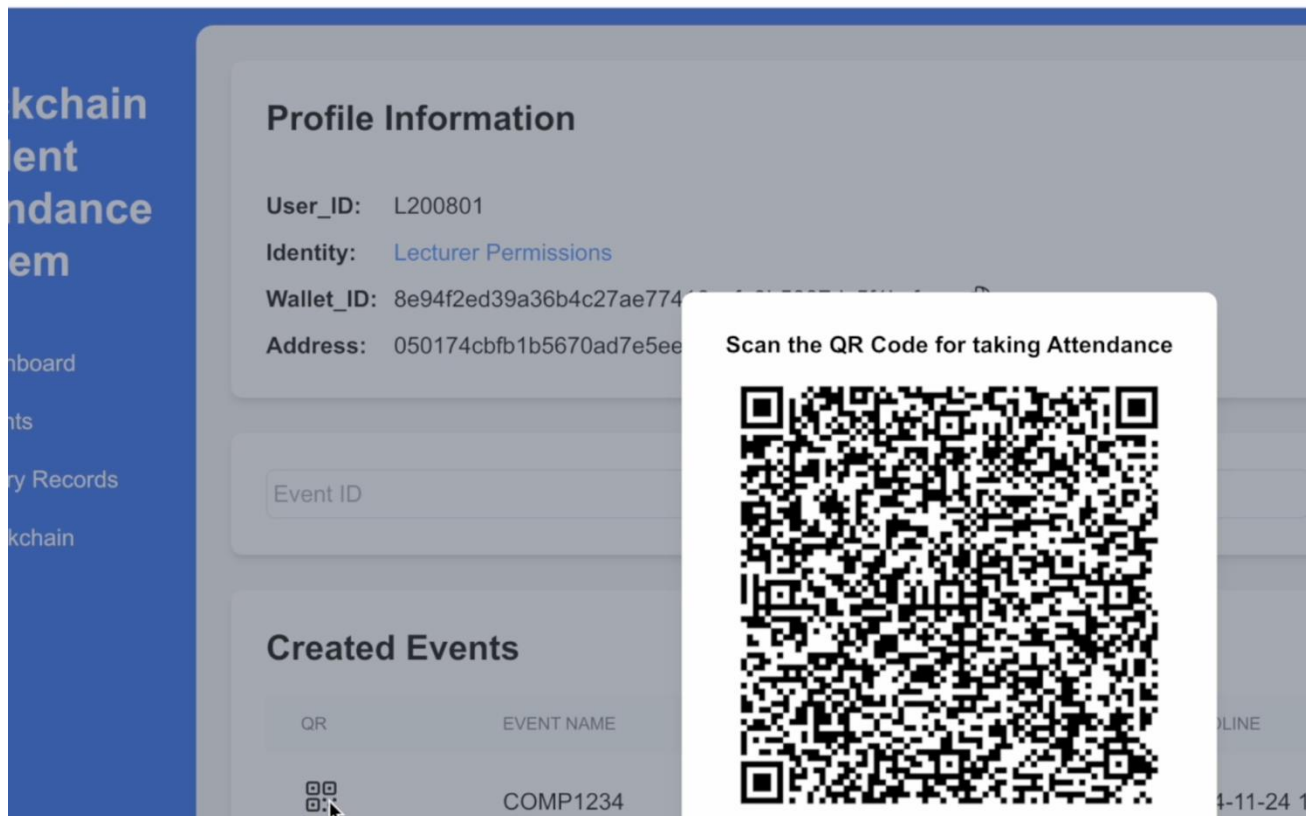
- **Maintain** the time **stability** of block generation.
- **Avoid** excessive **consumption** of network resources.

❖ Design logic :

- Compare the **actual generation time** of the block with the **expected time**:
 - If the actual time is **less than** the expected time (block generation is too fast), **increase** the mining difficulty.
 - If the actual time is **greater than** the expected time (block generation is too slow), **reduce** the mining difficulty.

Design Principles

- User interface is simple and clean
- Convenient check-in (Support students to sign in by scanning **QR code**)




```
JS index.js .../node JS naivecoin.js M JS index.js .../httpServer M X
lib > httpServer > JS index.js > HttpServer > constructor > app.get('/operator/:addressId/balance')
21 ; HttpServer
22 constructor
245 this.app.post('/operator/wallets/:walletId/addresses', (req, res) =
251
252   try {
253     if (!operator.checkWalletPassword(walletId, passwordHash))
254
255     let newAddress = operator.generateAddressForWallet(walletId
256     res.status(201).send({ address: newAddress });
257   } catch (ex) {
258     if (ex instanceof ArgumentError) throw new HTTPError(400, e
259     else throw ex;
260   }
261 };
262
263 this.app.get('/operator/:addressId/balance', (req, res) => {
264   let addressId = req.params.addressId;
265
266   try {
267     let balance = operator.getBalanceForAddress(addressId);
268     res.status(200).send({ balance: balance });
269   } catch (ex) {
270     if (ex instanceof ArgumentError) throw new HTTPError(404, e
271     else throw ex;
272   }
273 });
```

demoStRatiON

```
api.tsx M page.tsx M X Profile.tsx M ProductWidget.tsx U
src > app > dashboard > page.tsx > page
10 }RCodeGenerator
11 DateTimeFormatter from '../_components/DateTimeFormatter';
12 EventList from '../_components/EventList';
13 StudentEventList from '../_components/StudentEventList';
14 StudentEventHistoryList from '../_components/StudentEventHistoryList';
15 ProductWidget from '../_components/ProductWidget';
16
17 page = () => {
18   [user, setUser] = useState<UserClass>();
19   [loading, setLoading] = useState<boolean>(true);
20   [isStudent, setIsStudent] = useState<boolean>(true);
21
22   fetch(() => {
23     let data = localStorage().getAttribute("user");
24     let userString = JSON.stringify(data);
25     let userInstance = UserClass.fromStorage(userString);
26     if (userInstance) {
27       setUser(userInstance);
28     } else {
29       setLoading(false);
30     }
31   });
32
33   fetch(() => {
34     test
35     (user) {
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ...

```
> node + v □ □ □ □
2024-11-24T10:09:00.944Z - log - 1: be1a15bf9c4122af2961f2d035dd1f06a17f958800727667a71785
6439dd3fb6 and 1 1 1 1
2024-11-24T11:41:06.760Z - info - 1: Selected 1 candidate transactions with 0 being reject
ed.
2024-11-24T11:41:06.774Z - info - 1: Mining a new block with 3 (fee: 1, regular: 1, reward
: 1) transactions
2024-11-24T11:41:07.012Z - info - mine-worker: Block found: time '0 sec' dif '4' hash '13e
0e178dd90b2db72b3dbecd91f0bd126663ab015d92beff1c3f462e2f8b422' nonce '1'
2024-11-24T11:41:07.022Z - info - 1: Block added: 13e0e178dd90b2db72b3dbecd91f0bd126663ab0
15d92beff1c3f462e2f8b422
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL ...

```
> node + v □ □ □ □
GET /dashboard 200 in 15ms
GET /dashboard 200 in 28ms
GET /favicon.ico 200 in 11ms
GET /favicon.ico 200 in 11ms
✓ Compiled in 187ms (774 modules)
GET /dashboard 200 in 44ms
GET /dashboard 200 in 25ms
GET /favicon.ico 200 in 11ms
GET /favicon.ico 200 in 71ms
```

ThAnk yOU vEry mUCh

