Contract Signing and Employee Portfolio Verification Platform using Blockchain

Supervisor: Dr. Wu Chen Shu

<table>
<thead>
<tr>
<th>Group Member</th>
<th>UID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ling Lok Tin</td>
<td>3035569055</td>
</tr>
<tr>
<td>Sze Ho Yin</td>
<td>3035688758</td>
</tr>
</tbody>
</table>
Abstract

The blockchain industry has been increasingly popular in recent years. In particular, NFT as a product of blockchain has been extremely popular. Since its debut, the NFT industry has exploded. At the moment, NFTs are frequently connected to the ownership of digital art. NFT, however, might be something else, in our opinion. It is because of this quality of NFT—that when one owns an NFT, it gives a certificate of authenticity and serves as proof of ownership—that we chose this particular topic for this project. This project aims to make use of this functionality to simplify the verification process for digital certificates and degrees. Making sure that the issuance of each NFT of a digital certificate is well established, which means that those NFTs can only be granted if certain conditions are met, is the most important aspect of the project. Consequently, for the project to be successful, the NFT's development logic must be explicit. The project is progressing as scheduled. We may now begin the project's coding portion because the necessary research has already been done and the prerequisites have been satisfied. The immediate next step for us is the development of the structure of smart contract for each NFT, followed by the web app development and the implementation of NFT to the web app.

Acknowledgements

I want to start by thanking the computer science department and Dr. Wu Chen Shu, our project supervisor, for all the assistance they provided.

I would really like to express my appreciation to my teammate for his efforts and collaboration.

Finally, I want to thank Miss Mable Choi for all of her assistance with the detail project plan and progress report.
# Table of Content

Abstract............................................................................................................................................... 2  
Acknowledgements.............................................................................................................................. 2  
Table of Content.................................................................................................................................. 3  
List of Figures...................................................................................................................................... 5  
List of Tables....................................................................................................................................... 6  
Abbreviations...................................................................................................................................... 6  

1. Introduction...................................................................................................................................... 7  
   1.1 Overview of Recruitment Process................................................................................................. 7  
   1.2 Blockchain and Non-fungible token (NFT)..................................................................................... 7  
   1.3 Problem Statement......................................................................................................................... 8  
   1.4 Objective....................................................................................................................................... 9  
   1.5 Project Contributions..................................................................................................................... 9  
   1.6 Outline of the report...................................................................................................................... 9  

2. Methodology...................................................................................................................................... 11  
   2.1 Introduction.................................................................................................................................. 11  
   2.2 Workflow...................................................................................................................................... 11  
      2.2.1 Account Creation....................................................................................................................... 11  
      2.2.2 Contract Signing......................................................................................................................... 12  
      2.2.3 Employee Portfolio Verification............................................................................................... 13  
   2.3 Technology Implementation............................................................................................................ 13  
      2.3.1 Blockchain Development........................................................................................................ 13  
      2.3.2 Smart Contract Development.................................................................................................. 14  
      2.3.3 Database Development........................................................................................................... 15  
      2.3.4 Backend Development............................................................................................................. 16  
      2.3.5 Frontend Development............................................................................................................ 16  
   2.4 Summary...................................................................................................................................... 17  

3. Project Development.......................................................................................................................... 18  
   3.1 Project schedule............................................................................................................................. 18  
   3.2 Platform Development.................................................................................................................. 19  
      3.2.1 Smart Contract Development.................................................................................................. 19  
         3.2.1.1 Development Structure...................................................................................................... 19  
         3.2.1.2 Smart Contract Structure.................................................................................................. 20  
         3.2.1.3 Smart Contract Deployment............................................................................................... 20  
         3.2.1.4 Smart Contract Testing....................................................................................................... 21  
      3.2.2 Backend Development............................................................................................................ 22  
         3.2.2.1 Database Design............................................................................................................... 22  

List of Figures

Fig 1. High level overview of account creation process 11
Fig 2. High level overview of Process of Contract Signing 12
Fig 3. High level overview of Employee Portfolio Verification 13
Fig 4. Structure of the Smart Contract folder 19
Fig 5. Overview of fyp_token.sol 20
Fig 6. Script of deploy.js 21
Fig 7. Overview of test_fyp_token.js 21
Fig 8. Structure of the Backend Server folder 22
Fig 9. Structure of the React App folder 25
Fig 10. Login and Sign up page 26
Fig 11. Side Bar in Employer Account 27
Fig 12. Side Bar in Candidate Account 27
Fig 13. Dashboard page 28
Fig 14 & 15. Filter and Sort Button 29
Fig 16. Offer a Contract Page of Employer Account 29
Fig 17. Offer Page of Employer (Request for review) 30
Fig 18. Offer Page of Candidate (Request for review) 31
Fig 19. The employer dashboard showing offer that needs to be modified 31
Fig 20. The offer page of employer for modification 32
Fig 21. Offer page of Candidate (Request for modify) 32
Fig 22. Sign Offer Page 33
Fig 23. Offer Page (Completed Offer) 34
Fig 24. Verification page for input 35
Fig 25. Verification page of result 35
Fig 26. Loading Screen 36
Fig 27. Pop-up Confirmation Screen 36
List of Tables

Table 1. Project schedule 18
Table 2. Account Endpoints 23
Table 3. Offer Endpoints 23
Table 4. Profile Endpoints 24

Abbreviations

API    Application Programming Interface
CID    Content Identifier
DOM    Document Object Model
IPFS   Interplanetary File System
NFT    Non-Fungible Tokens
NPM    Node Package Manager
TPS    Transaction per second
1. Introduction

1.1 Overview of Recruitment Process
Every business relies heavily on recruitment. Recruiting talented and competent personnel for the organization will undoubtedly benefit the company, and vice versa. Recruitment, however, is a labor-intensive process that includes numerous processes, such as marketing for job postings, searching for passive prospects, screening, interviewing, evaluating candidates, making decisions, and even background checks. Every process is crucial since a single mistake could have devastating effects. The business surely does not want to lose its top performers. According to studies, more than 80% of employee resignations are caused by the fact that new and existing employees do not get along well with one another. The organization would want to avoid hiring someone who is incompatible with its present top performers because it could result in their leaving. Furthermore, a poor hiring choice may also lead to zero production because underqualified personnel will find it difficult to work effectively.[1] Meeting with employees to address performance issues might occupy time that could be spent in other beneficial ways. According to a survey, managers must spend 17% of their time managing underwhelming workers. [2] In short, poor recruitment practices could cause a company to lose money and time, as well as have much more serious repercussions. In the labor market, several variables, such as the company's resources, rivalry between competing businesses, and a long list of others, can influence the salary offered for a position. The competition for some well-known, prestigious jobs may be exceedingly fierce. Job seekers would come up with numerous strategies to differentiate themselves to win the race.

1.2 Blockchain and Non-fungible token (NFT)
The distributed ledger technology known as blockchain is used to store, track, and record data. The blockchain and its data are immutable because blocks of data are securely linked to one another using cryptographic hash function. Blockchains are controlled via a peer-to-peer (P2P) computer network where there are no central points of weakness, making it difficult for hackers to attack.[3] A non-fungible token is a one-of-a-kind digital identification used to verify authenticity and ownership. It is a data storage unit that is kept on the blockchain. Any type of data, including text, images, and even formal data, could be included in the data. Each NFT would also have its owner's information. The ownership of an NFT is documented on the blockchain, which is legitimate since it is transparent, traceable, and immutable. Since the use of NFTs can assist artists in resolving the issue of their work being stolen, so securing the ownership of their digital production, the use of NFTs in the field of digital art is currently a frequent application.[4] However, there may be more possibilities for NFTs than only the to authenticate the ownership of artwork. For instance, NFTs may be used as a digital replacement for documents like contracts, diplomas, and certificates that requires strong proof of ownership.
1.3 Problem Statement

Untrustworthy Resume

Due to the competitive labor market, it is difficult to stand out from fellow job applicants. Some people might cheat to stand out. In a 2017 survey, it was found that nearly 85% of employers found candidates had lied on their resumes or applications, typically by exaggerating their experience or skill level. [5] Some would even list phony credentials, or work experience on their resumes. The University Grants Commission (UGC) deemed 24 institutions to be fake in 2021, proving that the fake resume epidemic is widespread.[6] With that said, if recruiters do not verify resumes, the odds of a disastrous hire can increase considerably because the candidates may not perform as expected. However, due to the large number of areas that need to be validated, there is currently no efficient way to swiftly perform the verification processes.

Delayed Job Vacancies Information

Companies frequently publish job advertisements on various agency websites while hiring. The applicants can then submit their applications to the companies via the websites of the agencies, and the companies will get in touch with the applicants upon receiving the applications to arrange an interview, send an aptitude test, etc. However, once the applications have been received by the hiring firms, they are not required to inform the agent of the outcomes or the hiring process. This could lead to inaccurate information being posted on the website because some positions have already been filled and are no longer accepting applications. It would be a waste of the candidates' time and effort.

Unclear Application Process

Candidates frequently need to wait a long period before hearing back from the employer after submitting job applications or even after a job interview. Since they are unable to ascertain the cause, this is perplexing and stressful. Several factors, like the fact that the employer has not yet finished interviewing all other candidates, that they are gathering input from each interviewer, that they are just busy, or that they are still having trouble on decision making, could explain why a candidate is receiving radio silence, according to an HR insider. [7]
1.4 Objective

In this project, we seek to develop a web platform that streamlines the hiring procedure and enables employers to validate the legitimacy of candidates' resumes. The two components that make up the project's scope are the web application and the creation of NFTs through the use of smart contracts.

1.5 Project Contributions

Authentic and Transparent Resume

It is anticipated that users can develop their personal profile or resume on blockchain, taking advantage of its immutability, through the web application. No candidate could unfairly gain an advantage over the others with an honest and open résumé. The hiring firm could swiftly evaluate applications and exactly look for the candidates they require, which may guarantee the caliber of the new hires and, in turn, the performance of the business going forward. At the same time, this platform would hash the identity of candidates, therefore, only companies that are permitted by candidates could view their profiles.

Immediate Update on Job Vacancies

The project aims to give job seekers the latest information on open positions. It would be good to take some of their burdens away during this challenging time of job searching. An instant update on job vacancies on the agency website would be a wonderful feature for job seekers because they would not have to waste time and effort on jobs that are no longer available. This platform's allow employers and employees to sign their employment contract on the blockchain and make a record with it. Through retrieving contract signing records on the blockchain periodically, recruiting firms can simultaneously get a quick update of occupied jobs and provide accurate information for job seekers.

1.6 Outline of the report

Four chapters make up this report. A summary of the recruitment procedure, NFT, and any issues with the existing recruitment process are provided in the first chapter. Additionally, it outlines the development's objective and elaborates on the importance of our effort.

The second chapter examines the project methodology. The process for creating personal profiles, signing contracts, and verifying employee portfolios via the web application will be detailed. It will also be discussed why the technologies used in the project were selected.

The third chapter discusses the project's difficulties and constraints as well as possible future implementations and expansions of the project development. The discussion would provide insight into the project, and help understand some of the unavoidable elements that affect the project.
The forth chapter provides the tentative milestones and project development schedule.
2. Methodology

2.1 Introduction
The workflow of the web application, the technology deployed for the web application, the creation of smart contracts and NFT, and the implementation of NFT on the blockchain are all covered in this chapter. Furthermore, it compares several alternatives and explains the decision-making process.

2.2 Workflow
Our web platform contains three major features, which are Account Creation, Contract Signing, and Employee Portfolio Verification. Here a high-level view of workflow will be shown, details of technology implementation will be mentioned in later chapters.

2.2.1 Account Creation
Employers and employees can create an account through the web interface. They are required to provide their public keys and some extra information for the creation of accounts. (Fig. 1)

Fig 1. High level overview of account creation process

For employer account, details of the company are required. Upon the verification of company credentials, employer account will be created. For employee account, personal information, such as identity card number or email address is required.

Once the account is created, the blockchain address and email address will be saved in our database to provide a smoother user experience for users.
2.2.2 Contract Signing
The web application allows the employer and employee to sign an employment contract that will be tokenized and kept in the worker's blockchain address. (Fig. 2)

Employers can initiate a contract signing request by filling in the target candidate’s email address or blockchain address, employer’s private key (for contract signing), and details of the job, i.e., the role of the job, the commencement date of the job, employment contract documents, etc. through web interface. Then, the candidate will be notified of the event (contract signing request from employers). The candidate can sign the employment contract with their private key. Once the candidate signs, it will trigger the execution of a smart contract. Employment record will be tokenized into NFT and details of the job, the hash of employee’s identity card number, and information of the employer, i.e., employer’s address with employer’s public key, company name, etc. will be saved in metadata of the NFT. NFT representing employment contract would be transferred to the candidate’s blockchain address. Note that the document of employment contract would be hashed before being written to the metadata of NFT to protect the privacy of both parties.

Logic of minting and transferring of NFT will be written in smart contract, which will be deployed beforehand. The smart contract will be triggered when employers initiate the contract signing procedure and target employee signed with his/her private key.
2.2.3 Employee Portfolio Verification

In order to expedite background checks, the web application also offers a feature that enables employers to validate the candidate's personal information. (Fig. 3)

Employer can check a candidate’s portfolio by entering one’s email address (or one’s blockchain address) and candidate’s identity card number through the web interface. If email address is entered, it will search in the platform database and return the matched address. Using the address, backend will call external API(s) for retrieving all NFTs held by that address. Next, backend will match 1) hash of identity card number with that on NFTs 2) decrypted company address by stated company public key with that on NFTs. If any pair gets faults, it will be detected and notice both parties with the issue. Further proof should be provided by the candidates to the company. Instead, details of job experience will be shown on the web interface.

2.3 Technology Implementation

2.3.1 Blockchain Development

Blockchain is essential to the accomplishment of our project. The following factors will be taken into account when deciding which blockchain to use: a) ease of development, b) transaction fees, and c) transaction speed. We choose to start working on Polygon at this point. With a few differences, Polygon operates similarly to Ethereum. What follows is a comparison between the blockchains Ethereum and Polygon in terms of these three criteria.
**Ease of development**

In 2015, Ethereum is launched. Since then, more than 50.5 million smart contracts—programs stored and executed on blockchains[9]—have been created on the Ethereum platform. [8] For developing smart contracts on Ethereum, Solidity is the most widely used programming language. [10] The massive number of smart contracts that have been deployed on Ethereum imply that there are a lot of Solidity libraries, which are reusable pieces of code, available. These libraries are a huge help to Ethereum's smart contract developers and are crucial for ease of development. Additionally, a variety of APIs are supported for interacting with the Ethereum blockchain, including Web3.js for interacting with an Ethereum node, Ehere.js for interacting with an Ethereum wallet, and Alchemy NFT API for retrieving NFT data (for example, custodian service, smart contract deployment, and retrieving NFT data). [12] As a result, thanks to Solidity's massive developer community, developing on Ethereum is simple.

Polygon is a side chain or Layer 2 solution to Ethereum that enables the deployment of Ethereum Smart Contracts to polygon networks. It implies that the benefits of Ethereum mentioned above also apply to Polygon.

**Transaction cost**

Despite being easy to develop on Ethereum, deploying smart contracts on the blockchain is costly. Users are required to pay a gas fee for the energy consumed when minting and transferring NFT. [13] [14] The complexity of the transaction affects the gas fee. As a result, the cost of transactions for NFT is determined by both the complexity of the smart contract and the market price of gas. [13] The average gas fee on Ethereum ranges from 50 to 200 USD,[15] which is a fairly high cost for the development of the project.

On the other hand, Polygon has a far more affordable gas fee—it does not even cost 0.001USD—making it a much more cost-effective option.[16]

**Transaction speed**

In terms of transaction speed, Polygon can process an impressive 65000 transactions per second, compared to Ethereum's impressive 17 transactions per second.[17]

**2.3.2 Smart Contract Development**

For Polygon smart contracts development, there are different token standards, which are different rules amended after a comprehensive review by the Ethereum community that are pertaining to tokens on the blockchain. As of now, ERC-721 and ERC-1155 are the most popular ones. Below is the comparison between the two.

**ERC-721**
ERC-721 was the first standardized interface for creating NFTs. It is the option that a great deal of developers use. ERC-721 is immutable, secure, and transparent. It is hence non-fungible and an ideal NFT. This NFT standard is great for creating and monitoring distinctive NFTs. The most common standard for blockchain gaming and digital art currently would be ERC-721.[18]

ERC-1155

ERC-1155 is a token that likewise supports the development of NFTs. In comparison to ERC-721, ERC-1155 focuses on batch transfers, allowing the inclusion of multiple assets in a single smart contract, enabling the transfer with little network congestion and lowering the gas fee by 90%, making it a more affordable way to mint on the blockchain. The disadvantage of ERC-1155 is that it has specifications on the Ethereum logs that provide less sufficient data, making ownership tracking difficult.[17][19]

One of the project's area of focus is the integrity of users' personal profiles; although having a cheaper gas charge, ERC-1155's difficulty in tracing back the NFT history has led us to adopt ERC-721.

2.3.3 Database Development

IPFS will be used to save metadata of NFT, while PostgreSQL will be used for storing account data.

IPFS

IPFS is a distributed, decentralized system for storing data. [20] With its distributed characteristics, the data will not be tampered or deleted. It can also easily be easily recovered even one of the data points crashed. [20] Data can be uploaded through IPFS API, like INFURA and Pinata. Once the data is uploaded to IPFS, a CID hash will be returned to act as the fingerprint of the content for searching and verifying integrity of the data being stored.

Storing data in blockchain could be expensive. [21] Hence, IPFS can play an important role in storing the metadata of NFTs. [21] Metadata can first upload to IPFS, then API call to the IPFS content with the CID can be saved to the metadata of NFT. [21] As the API call link has a much smaller size comparing to the initial content saved, it would be much cheaper to mint the NFT. [21]

In this project, each Polygon smart contract that mints NFTs holds Metadata that must be uploaded to IPFS. It is simple to connect IPFS to a Polygon smart contract. [22]

PostgreSQL

PostgreSQL is an object-relational database management system, which is supported by cross-platform. [23] Object oriented features of PostgreSQL support object features, including inheritance and
user-defined data type. [24] It can help better organize our database and provide extensibility when we launch out new feature in the future. Besides, relational feature also provides better organization of data for our project as data would be store with pre-designed schema.

2.3.4 Backend Development
Node.js and Express.js will be used.

Node.js
Node.js is an open source and cross-platform JavaScript runtime environment. [25] It is developed in 2009. It is also used by Tech-Giant like Amazon, Netflix, eBay, Reddit, and with more than 30 million websites sit on it. [26] It is also supported with a large, many handy libraries can be found using NPM, the world’s largest software registry that support over 800,000 code packages. [27] Libraries such as web3.js and Alchemy’s NFT API will be able to download in NPM. [28][29]

Express.js
Express.js is a Node.js server-side framework. [30] It provides built in methods and middleware to ease the development of web and mobile application. Libraries and framework would ease our development as we do not have to start everything from scratch.

2.3.5 Frontend Development
As users of the project would come from different backgrounds and may not have a strong knowledge of programming, a user-friendly interactive web interface is needed to deliver our service. React.js will be used.

React.js
React.js is a popular JavaScript framework for building user interfaces. It would be easier for us to work on it given that our backend is written with JavaScript as well. Besides, React.js support building individual components for different part in a web page, which have high reusability and better organization. [31] Besides, React.js support virtual DOM, which allow keeping data state out of DOM and optimize efficiency of the app. [31] Lastly, it supports the integration with blockchain with library available on NPM as it is Node.js based as well. [31] Additionally, the app makes use of Bootstrap and Tailwind CSS for styling and design, resulting in a cohesive and visually pleasing appearance across all pages. The app also makes use of a number of libraries, including React Router for routing, FileUploader for file uploads, and React Form Stepper for delivering offer progress visualization. These technologies enable effective data and file management in addition to an agile and responsive user experience.
2.4 Summary

A thorough explanation of the web application's procedure, from account setup to contract signing and candidate portfolio verification, has been presented in this chapter. The software uses a number of technologies to guarantee quick and secure transactions, including PostgreSQL for database development, Node.js and Express for backend development, Polygon for blockchain, and ERC-721 for smart contract protocol. The project's technology implementations are chosen with a rationale that is explained. The requirements for usability, security, and flexibility guided the choice of these technologies.
3. Project Development

The project's development process, including the project timetable and finished products, are examined in this part.

3.1 Project schedule

Table 1 shows the project schedule. To follow an Agile development approach, the project is divided into different stages, each of which is regarded as an iteration.

Table 1. Project schedule

<table>
<thead>
<tr>
<th>Time Periods</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>September, 2022</td>
<td><strong>Phase 1 Deliverable (Inception):</strong></td>
</tr>
<tr>
<td></td>
<td>- Detailed Project plan</td>
</tr>
<tr>
<td></td>
<td>- Project website</td>
</tr>
<tr>
<td>4th – 31st October, 2022</td>
<td>Research on Blockchain and NFT infrastructure</td>
</tr>
<tr>
<td></td>
<td>- Different blockchain project</td>
</tr>
<tr>
<td></td>
<td>- IPFS integration</td>
</tr>
<tr>
<td></td>
<td>- Tools to interact with Blockchain</td>
</tr>
<tr>
<td></td>
<td>- Smart Contract Development</td>
</tr>
<tr>
<td>1st - 20th Nov, 2022</td>
<td>Smart contract development</td>
</tr>
<tr>
<td>21st Nov - 15th Feb, 2023</td>
<td>Core API endpoint development</td>
</tr>
<tr>
<td></td>
<td>- Internal call of API</td>
</tr>
<tr>
<td></td>
<td>- Integration of blockchain and platform backend</td>
</tr>
<tr>
<td></td>
<td>- With relevant testing</td>
</tr>
<tr>
<td><strong>First Presentation</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Phase 2 Deliverable (Elaboration):</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Preliminary implementation</td>
</tr>
<tr>
<td></td>
<td>- Detailed interim report</td>
</tr>
<tr>
<td>16th Feb – 15th March, 2023</td>
<td>Front end implementation</td>
</tr>
<tr>
<td></td>
<td>- Web interface for employers and employees</td>
</tr>
<tr>
<td></td>
<td>- Build relevant test case</td>
</tr>
<tr>
<td>16th March - 15th April, 2023</td>
<td>Test and review the entire project</td>
</tr>
<tr>
<td><strong>Phase 3 Deliverable (Construction):</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Finalized tested implementation</td>
</tr>
<tr>
<td></td>
<td>- Final report</td>
</tr>
</tbody>
</table>
3.2 Platform Development

The development is broken down into various stages, as stated in the schedule. The platform development is being carried out using the Linux operating system - Ubuntu. It consists of the creation of the web application and smart contracts that will be deployed on the blockchain. Users will be able to perform all the tasks listed in section 2.2. Workflow on the platform. The development of smart contracts, backend development, and frontend development will all be covered in this section.

3.2.1 Smart Contract Development

The smart contract development is a critical part of the project as it provides the backbone for the platform's functionality. The smart contract will be responsible for minting NFTs that represent the candidate's profile and work experience. It will use the Solidity library, ERC721URIStorage.sol of OpenZeppelin to ensure secure coding practices and minimize the risk of vulnerabilities. The smart contract will be deployed on the Mumbai testnet before deployment on the Polygon mainnet.

3.2.1.1 Development Structure

The development structure of the smart contract folder includes multiple directories necessary for the entire lifecycle of the FypToken smart contract. (Fig 4.) The main contract file, fyp_token.sol, is located in the “contracts” folder, while the deploy.js script file for deployment to the network is in the “script” folder. Additionally, there is a “test folder” that contains test_fyp_token.js for testing the functionality of the smart contract. The “artifacts” folder will store the JSON file generated during the deployment process, also the libraries imported from OpenZeppelin. The “cache” folder will store the compiled contract data. The structure of the smart contract project enables easy navigation and separation of concerns, facilitating maintenance and scalability.

![Fig 4. Structure of the Smart Contract folder](image-url)
### 3.2.1.2 Smart Contract Structure

The smart contract development is essential to this project since it defines the functionality of each NFTs. In this project, the smart contract we developed is called “fyp_token”. It is an ERC721 token that mints NFTs representing a candidate's profile and work experience. The contract contains several key functions (Fig. 5), including the ability to create new NFTs using the mint() function, terminate the contract in NFTs using the terminate() function, verify if both candidate and employer has signed a contract using the isSigned() function, get the termination time of a contract using the terminate_time() function, get the employer's address of the contract using the company_address() function, and lock the transfer function using the beforeTokenTransfer() function. These functions enable the platform to execute the actions outlined in section 2.2, Workflow.

```
contract FypToken is ERC721URIStorage, Ownable {
    using Counters for Counters.Counter;
    Counters.Counter private _tokens;
    mapping(uint256 => uint256) private _terminated;
    mapping(uint256 => address) private _company_address;
    mapping(uint256 => bool) private _disable;

    event transfer(uint256,uint256,address); // new
    event return(uint256,address,uint256); //

    constructor() ERC721(“FypToken”, “FypToken”) []

    function isSigned() ...
    function mint(string memory message) ...
    function terminate(uint256 token_id) public view returns (uint256)...
    function company_address(uint256 token_id) public view returns (address)...
    function terminate(uint256 token_id, uint8 v_1, bytes32 r_1, bytes32 s_1) public{...
    function beforeTokenTransfer(address, address, uint256 id, uint256) internal view override{...
```

Fig 5. Overview of fyp_token.sol

### 3.2.1.3 Smart Contract Deployment

The “deploy.js” file is a script used to deploy the FypToken smart contract to the network. It uses the “ethers.js” library to interact with the blockchain. The script exports a main() function, which first gets the contract factory from ethers, deploys the contract to the network, and logs the address of the deployed contract to the console. (Fig. 6)
3.2.1.4 Smart Contract Testing

The smart contract testing is done to ensure that the smart contract functions are working properly. The tests for the FypToken smart contract are written in JavaScript and the framework used is Mocha. (Fig. 7) The tests are written to test various functions like mint, sign, transfer, terminate, etc. The tests are executed by running the JavaScript file, which will deploy the smart contract and run the tests. The test suite consists of five tests. The first test "Test isSigned()" verifies whether the token can only be minted when both parties signs correctly on the same document. The second and third test, “Test mint()” and “Test terminate_time()” verify whether the minted token has the correct information stored in it. The forth test “Test terminate()” verifies whether the terminate function is operating properly in different situation. The last test “Test disable transfer function” verifies that minted tokens could not be transferred. Through these tests, we can be confident that the FYP Token contract is functioning as intended.
3.2.2 Backend Development
As mentioned in the methodology, the backend was developed using Node.js and Express.js. The backend development for the project involved, setting up a project structure, establishing a database schema, and building a RESTful API to handle data storage, processing, and retrieval, thereby providing proper information to the frontend and ensuring the database's integrity.

3.2.2.1 Database Design
The backend database for the project was implemented using PostgreSQL. The database design for the project is based on two tables, Account and Agreement. The Account table contains columns of email (primary key), id number, password, account type, name, and chain_address. The Agreement table contains columns of id (primary key), cid, token_id, candidate_sig, company_sig, job_title, job_description, contract, candidate_email, company_email, and status. The database schema is designed using Sequelize, a promise-based ORM for Node.js, which makes it easier to interact with the database using JavaScript objects.

3.2.2.2 Development Structure
The project structure for the backend was organized into several folders,(Fig. 8) The account, offer, and profile folders were used to handle the endpoints for /account, /offer, and /profile, respectively. The model folder defined the database schema for two tables, Account and Agreement. The service folder provided various services such as email verification, access token verification, and mandatory field input box checking, and also included blockchain-related services for minting NFTs and fetching metadata from token URIs. The test folder contains test files to ensure that the API endpoints and services work as intended. The util folder contains utility functions and modules used throughout the application. The config.js file contains configuration settings for the application, such as the ipfs gateway details. The middleware.js file contains middleware functions used by the application. Lastly the server.js file, which is the entry point for the application, it direct requests to different folder according to the URL.

![Fig 8. Structure of the Backend Server folder](image-url)
3.2.2.3 RESTful API

Our RESTful API provided the interface between the frontend and the backend. We used HTTP methods to handle the requests and responses to our API. The API endpoints were structured around the resources of the system, with each endpoint performing a specific CRUD operation on a resource. Account (Table 2), Offer (Table 3), and Profile (Table 4) are the three key use cases for which distinct endpoints have been built. All of these endpoints have been checked using automated tests in the "test" folder to confirm they are functioning as expected and meeting the system's criteria. Furthermore, access tokens were utilized to authenticate users and provide secure connection between the frontend and backend. Ultimately, the goal of our API architecture was to provide smooth and safe communication between the frontend and backend.

<table>
<thead>
<tr>
<th>Route</th>
<th>HTTP methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/account/:email</td>
<td>GET</td>
<td>Get the blockchain address of a email holder</td>
</tr>
<tr>
<td>/account/create</td>
<td>POST</td>
<td>Sign up for an account</td>
</tr>
<tr>
<td>/account/login</td>
<td>POST</td>
<td>Log in to an account</td>
</tr>
</tbody>
</table>

Table 2. Account Endpoints

<table>
<thead>
<tr>
<th>Route</th>
<th>HTTP methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/offer</td>
<td>POST</td>
<td>Give offer to candidate</td>
</tr>
<tr>
<td>/offer</td>
<td>GET</td>
<td>Get all the offers received for candidate, offers sent for employer</td>
</tr>
<tr>
<td>/offer/specific/:id</td>
<td>GET</td>
<td>Get the detail of a specific offer</td>
</tr>
<tr>
<td>/offer/accept</td>
<td>PUT</td>
<td>Accept an offer as a candidate, update status of the offer from “REQ_REVIEW” to “REQ_SIGN” and upload the offer to ipfs</td>
</tr>
<tr>
<td>/offer/modify</td>
<td>PUT</td>
<td>Request for modification for the offer, change the status of the offer to “REQ_MODIFY”, as an candidate; Update the modified version of the offer, change the status of the offer back to “REQ_REVIEW”, as a employer</td>
</tr>
</tbody>
</table>

Table 3. Offer Endpoints
<table>
<thead>
<tr>
<th>Route</th>
<th>HTTP methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/offer/sign</td>
<td>POST</td>
<td>Create a signature for contract signing using the private key provided by user</td>
</tr>
<tr>
<td>/offer/sign</td>
<td>PUT</td>
<td>Sign the contract using the created signature, if both parties have signed the contract, update the status of the contract to “COMPLETED” and mint the NFT to the candidate blockchain address</td>
</tr>
<tr>
<td>/offer/contract</td>
<td>GET</td>
<td>Get the encrypted contract in pdf format</td>
</tr>
<tr>
<td>/offer/reject</td>
<td>PUT</td>
<td>Reject offer as a candidate</td>
</tr>
<tr>
<td>/offer/remove</td>
<td>PUT</td>
<td>Remove offer as a employer</td>
</tr>
<tr>
<td>/offer/terminate</td>
<td>POST</td>
<td>Create a signature for contract termination using the private key provided by user</td>
</tr>
<tr>
<td>/offer/terminate</td>
<td>PUT</td>
<td>Terminate the contract using the created signature</td>
</tr>
</tbody>
</table>

Table 4. Profile Endpoints

<table>
<thead>
<tr>
<th>Route</th>
<th>HTTP methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/profile/record</td>
<td>GET</td>
<td>Read specific NFT token of the given token id</td>
</tr>
<tr>
<td>/profile</td>
<td>POST</td>
<td>Retrieve all the NFTs held by the given blockchain address</td>
</tr>
</tbody>
</table>

3.2.3 Frontend Development

The frontend of our system was developed using React.js, providing a user-friendly interface for our users to interact with the backend. Our frontend implementation incorporates all of the backend's functionalities. We also concentrated on creating a responsive design that would provide a consistent user experience across multiple devices. Our frontend implementation has undergone extensive testing to ensure usability, functionality, and compatibility with various browsers and devices. As a whole, we intended to design a simple and efficient frontend that improves the user experience while maintaining in sync with the functionality of the backend.
3.2.3.1 Development Structure

Our frontend application is organized into a clear and logical folder structure. (Fig 9) The "/src" directory is the main source directory for our React application and contains all the components and assets of our app. The "/assets" folder contains all the image files used in our app. There are different components in our application, some components are organized into separate folders such as "EntryForm", "FileUploader", "Stepper", these components are used to handle the different functionalities of our system such as handling the forms, file uploads, steps in the process. Besides these components, "candidate_dashboard.jsx", "candidate_offer.jsx", "employer_dashboard.jsx", and "employer_offer.jsx" are the dashboard and offer view for employers and candidates. The “popup_confirmation.jsx” component is a pop up box that overlay the entire app when the app is doing a confirmation with the user before making important decision. The "side_bar.jsx" component handles the navigation bar on the left side of the dashboard. Our main application is defined in "/App.jsx" and our styling is defined in "/index.css". The configuration files like "package-lock.json", "package.json", and "tailwind.config.js" are also included in our directory.

Fig 9. Structure of the React App folder

3.2.3.2 Pages

In this section, different pages will be shown and their functionality will be explained.

- Entry Page (Fig 10)
The login page is the very first page that users encounter when they access the platform. To log into their account, users have to enter their email address and password. If the credentials are correct, the user is taken to their respective dashboard. If the user is new to the system and has yet to create an account, they can go to the sign-up page and establish one. Users must input their name, blockchain address, email, password, and account type (individual or company) on the sign-up page. If it is an individual account, the Id number is necessary.

![Login and Sign up page](image)

**Fig 10. Login and Sign up page**

- **Side Bar**

  The side bar component is a navigation menu that allows the user to quickly access different pages inside the application. The name of the user will be displayed on the side bar. For an employer account, there are four buttons available (Fig 11): "Dashboard", "Offer a Contract", "Verification", and "Logout". When the employer clicks the "Dashboard" button, they are taken to the path "/" which is their dashboard page where they may examine their offers. The "Offer a Contract" button takes companies to "/offer" where they can create a new offer and send it to a candidate. The "Verification" button directs employers to "/profile" and allows employers to verify a candidate's blockchain address to ensure they hold a contract of related job experience. Finally, the "Logout" button exits the employer's account. There are only two buttons available for a candidate account(Fig 12): "Dashboard" and "Logout".
● Dashboard page (Fig 13)

The dashboard page has the path of “/”. It is the primary page of the application where users can view the basic information and status of each offer and perform some action. It is divided into two parts, the side bar and the main section. The main section of the dashboard page contains all the offers sent by the employer, each block is one offer sent. Each block displays the job title, candidate email address and the
progress bar. There is a “View” button for each block, user will be direct to the path “/offer” where user can perform more action and view the detail of the offer. Only some blocks has the “Remove” button, since the remove function is only available for offers that have not been signed yet. On the top of the page, user can filter or sort the offer by clicking the “Filter by” and “Sort by” buttons, for clearer view of the offers.(Fig 14 & 15) The dashboard page is designed to be intuitive and user-friendly, with clear calls to action and easy-to-understand visual elements.

Fig 13. Dashboard page
Offer a Contract page (Fig 16)

The offer a contract page has the path of “/offer”. It is a employer-only component which allows the employer to create a new job offer for a candidate. It features several input fields for the employer to fill out, including the job title and job description (optional), the email address of the candidate, and a password to protect the contract file that will be uploaded to the system. Besides, the employer need to choose the PDF file of the contract they wish to offer to the candidate. Once all of the required fields have been completed, the company can submit the job offer for the candidate to review.
When an employer sends an offer to a candidate, a confirmation page appears to alert the employer that the offer was successfully sent. This page will show the job title and candidate email address to whom the offer was delivered. The confirmation page will also have two buttons from which the employer can select. The "View Contract" button allows the employer to download the PDF contract that was emailed to the candidate for confirmation. The "Go to Dashboard" button takes the employer to the dashboard page.

![Offer Page of Employer](image1.png)

**Fig 17. Offer Page of Employer (Request for review)**

The view offer page of candidate displays the status of the offer they have received. The page displays the job title and the name of the employer who sent the offer, provides the options for them to make a decision regarding the offer they received. There are four buttons on this page that allow the candidate to take different actions. The "View Contract" button allows the candidate to download the contract and review its terms and conditions. The "Request to Modify Offer" button allows the candidate to request changes to the contract, such as a modification of the job title or salary. It turns the status of the offer to "REQ_MODIFY" in the database. The "Reject Offer" button allows the candidate to decline the offer. It turns the status of the offer to "REJECTED" in the database. Lastly, the "Accept Offer" button updates the
status of the offer in the database, and user will be redirected to a page where they can sign the contract. It turns the status of the offer to “REQ_SIGN” in the database.

![A Technical Support contract has been offer by abc@gmail.com](image)

**Fig 18. Offer Page of Candidate(Request for review)**

- Dashboard and Offer page of Employer (Request for modify)

If the candidate requests to modify the offer, the status of the offer will be changed, as previously stated. The block of the offer will turn grey on the employer's dashboard page, suggesting that special action is required. (Fig 19) A "Modify" button will also be displayed, allowing the employer to take further action on this offer. The employer can then open the offer page by clicking the "Modify" option. (Fig 20) The employer will be invited to upload an updated copy of the contract and set a password for contract protection on this page. They can then click the "Submit" button to send the revised offer to the candidate for review. The candidate will then receive a notification informing them that the offer has been modified and will need to review the offer and make a decision.

![The employer dashboard showing offer that needs to be modified](image)

**Fig 19. The employer dashboard showing offer that needs to be modified**
Fig 20. The offer page of employer for modification

- Offer page of Candidate (Request for modify) (Fig 21)

A confirmation message will show on the screen after the applicant presses the "Request to Modify Offer" button during the review portion. This message will notify the candidate that their request for adjustment was successfully submitted to the company.

Fig 21. Offer page of Candidate (Request for modify)
• Sign Offer Page (Fig 22)

The Sign Offer page is where both the employer and candidate will sign the contract using their private key of their blockchain address. This page is only accessible after both parties have agreed to the terms of the contract. The contract is signed securely over the blockchain network, ensuring that it is tamper-proof and legally binding. The status of the offer will be changed from "REQ_SIGN" to "SIGNED" if the first party signs the contract. The second party will then have access to the contract to sign, and the status will be updated to "COMPLETED" once signed. Once both parties have signed the contract, the contract will be minted and stored securely on the blockchain, it can then be accessed by both parties for reference in the future.

![Sign Offer Page](image)

Fig 22. Sign Offer Page

• Offer page (Completed Offer) (Fig 23)

The offer page will be updated to reflect a confirmation notice that the contract has been signed and completed after both the company and the candidate have signed it. The status of the offer is "COMPLETED" at the time. There will also be an input box for both parties to enter their private key if they want to terminate the contract. The contract will be terminated if the private key is entered and submitted.
Verification Page (Fig 24 & 25)

The verification page is an important part of our system because it allows companies to swiftly confirm candidates' experience and qualifications. Employers may provide the candidate's Hong Kong Identity Card (HKID) number and blockchain address on this page. The portal will then provide a dashboard with a list of all FYP tokens held at that address. If the token matches the candidate's HKID, the dashboard will show it as a valid token. Employers can use this function to quickly determine whether a candidate has the relevant experience or qualifications for the job. Employers can be confident that they are hiring the best candidate for the job due to this verification process.
● Loading Screen (Fig 26)

When the program is retrieving data or waiting for a response from the server, the loading screen will appear. This may include sending an offer, signing a contract, or registering for an account. To avoid repeated input from the user, a spinner and the text "loading..." will be displayed on the screen while blocking all action input from the user. This page helps to avoid confusion by informing the user that the application is processing their request and that they should wait for the system to complete. When the application receives a response from the server, the loading screen will automatically disappear, and the user will be able to continue using the application.
Pop-up Confirmation Screen (Fig 27)

The platform's pop-up confirmation screen serves as a protection to avoid unexpected or accidental operations. When users click buttons like terminate, remove, or reject offer, a confirmation page will display over the current screen, indicating the action that is about to be executed. The confirmation box will display a message asking the user whether they are certain they want to proceed with the action, as well as two options - "Cancel" and "Reject/Remove/Terminate" depending on which button the user clicked. When the user hits "Cancel," the pop-up confirmation box disappears and the current screen remains unaffected. If, on the other hand, the user clicks on "Reject/Remove/Terminate," the action will be carried out as intended, and the website will refresh to reflect the result of the action. The pop-up confirmation page ensures that users are fully aware of the actions they are going to perform, minimizing the possibility of unintended consequences.
3.2.4 Codebase, Git, Bitbucket, Jira
This project's codebase is hosted on Bitbucket and versioned using Git. Bitbucket was selected because of its set of capabilities that may help in the development process. Bitbucket's free private repositories for small teams enable us to host our codebase and seamlessly collaborate with our groupmates. The use of Git allows us to track changes in our codebase and work on different features simultaneously, while also easily reverting back to previous versions if necessary.

Furthermore, Bitbucket integrates with Jira, a great tool for agile development. We can simply track and manage tasks, issues, and project progress by connecting our Bitbucket repository to Jira. Jira assists our team in being organized and focused on completing a successful project by offering a central spot for tracking our work, identifying obstacles, and ensuring that everyone remains aware of the project's progress.

4. Discussion
This section discusses the challenges and limitations faced by this project, and lay out the future plan for this project.

4.1 Challenges
This part address the challenges faced during the development of this platform.

4.1.1 Storing PDF Files in Database
We encountered a problem when developing our application: storing PDF files in the database. Although we thought about keeping the file's URL instead, we decided to store the file itself in our database's "Agreements" table. This decision presented some technical difficulties, such as the need for efficient file handling and optimization of SQL queries. To address this issue, we streamlined our SQL queries so that they only returned the necessary data from the table, instead of selecting the file when it was unnecessary. This allowed us to ensure the smooth handling of large files in our application, while maintaining its overall performance and efficiency.

4.1.2 Setting Up and Using PostgreSQL in Linux Environment
Setting up PostgreSQL on Ubuntu is difficult because it involves setting up a new account and configuring various settings. Furthermore, after configuring the database under Linux, the PostgreSQL program, pgAdmin, is unavailable for some reason. This requires opening the terminal each time I want to run queries or perform other operations, which can be time-consuming and bothersome. Despite these obstacles, there are ways to streamline the process, such as using scripts or utilizing command-line tools to facilitate database management.
4.1.3 Responsive Interface
A critical component of the development process is ensuring that the platform's UI is responsive and operates flawlessly across all devices and platforms. Thorough testing is required to accomplish it. This testing involves testing the platform's compatibility with various browsers and mobile devices, as well as confirming that the layout and functionalities perform properly. It took a long time to complete the procedure. However, after extensive testing, we were able to ensure that the platform's UI is both user-friendly and responsive, giving a consistent user experience across all platforms.

4.2 Limitation
The project is subject to some limitations, which are not solvable by us at the moment.

4.2.1 Adoption of the Platform Depends on User Participation
It's crucial to remember that any new platform or technology takes time to catch on. It is true that the current hiring procedures have been in place for a long time, and changing them would require an immense amount of effort on the part of both companies and employees. While this platform has the potential to revolutionize the recruitment process, it may take a bunch of time to gain traction and widespread adoption. As a result, it is critical to set realistic expectations for the platform's adoption rate and potential impact.

4.2.2 KYC and Identity Verification
While the KYC process is necessary for verifying the identity of platform users, it is also a costly procedure. Due to the associated costs, implementing the KYC process may not be possible at this point in time. Furthermore, proof of company identity establishes an additional obstacle in terms of security. The company's public key is stored in the platform's centralized database, which exposes it to the risk of being hacked. While effective security measures can lessen this risk, it remains an issue that must be addressed.

There are some alternatives to upgrading the platform. One possible solution is to require a public key upon company registration, which means that the information will be held on a relatively secure government database. Another possibility is for the firm to formally reveal its public key, removing the need for a centralized database to store the company's verified public key. Furthermore, putting account information on IPFS may decrease our platform's control over the data, making it less vulnerable to hacker attempts. It should be noted, however, that some solutions might demand substantial adjustments to the current system and may not be practicable in the short term.
4.2.3 Performance Scalability for Larger User Base
Only a small number of tokens were produced during the development phase, therefore the database was not very large. As a result, it is difficult to predict how the platform will operate when a big number of users access it and make numerous requests to the system. When there is a great volume of traffic, the platform may have performance issues or significantly slow down. This will need to be continuously monitored, and we may need to enhance the platform’s architecture or implement extra scaling solutions to ensure that it can manage significant volumes of traffic.

4.2.4 Gas Fee of Blockchain
The network was initially intended to operate on the Ethereum blockchain, but its high gas fees made it unsuitable for widespread use. As a result, the development was renamed Polygon, which has substantially cheaper gas fees. However, if the application acquires a large number of users, the cost burden may fall on the developers, making it unsustainable. One potential solution would be to implement a charge structure for users, which could help cover the cost of transactions while also ensuring the platform's long-term stability. To avoid discouraging usage the price structure would need to be thoughtfully planned, but it may potentially alleviate the issue of high gas prices.

4.2.5 Irreversible Human Error on the Blockchain
One risk of blockchain technology is that user errors cannot be easily reversed or corrected. For example, if a user unintentionally terminates a smart contract on the platform, the record of the termination will be kept on the blockchain as a valid record. This means that users must be cautious when interacting with the platform and double-check their actions before making any permanent modifications. We will need to give consumers clear instructions and cautions to help them avoid making mistakes that could result in irreversible acts on the platform.

4.2.6 Adapting to New Technology for Older Companies
Certain older businesses may lack the requisite resources or employees to manage internet and device usage, limiting their capacity to use the platform properly. Companies in conventional industries, such as manufacturing or construction, may, for example, have a more limited technology infrastructure and personnel. Furthermore, due to limited access to technology and resources, smaller enterprises or those operating in less developed locations may experience difficulties in using the platform. This problem can be overcome in part by providing training and support to help businesses navigate the platform, as well as pursuing partnerships or collaborations with organizations that specialize in digital transformation and adoption.
4.3 Future plans

Aside from the current objective and deliverables, the group focuses on expanding the application's reach by solving challenges and improving the platform.

4.3.1 Building User Base

Gaining users is vital for the success of any new platform. As a brand new app, our user acquisition strategy will first involve collaboration with colleges to encourage students to create accounts on our platform. University graduates are commonly job seekers, and by targeting this demographic, we intend to develop a strong foundation of users ready to explore career prospects. Furthermore, having a big number of job seekers on our platform will encourage more businesses to join because they will have access to a larger pool of potential employees. We intend to collaborate with university career centers to promote our platform to students and provide resources for effective use of the app. We may also establish a referral scheme to encourage current users to invite their friends and colleagues to join the platform, hence increasing our user base. As our platform expands and develops popularity, we will continue exploring and applying innovative tactics for attracting new users and retaining existing ones.

4.3.2 Implementation of Identity Verification

As previously stated, due to the high expense of identity verification, it is currently not implemented. However, it may become an essential in the future. As a consequence, the platform can consider adding identity verification using less expensive technologies or as an optional service for consumers.

4.3.3 Implementation of New Features

We plan to launch more features in the future that will significantly improve the overall user experience and attract more users to the platform. Integrating machine learning techniques, for instance, into our platform for the matching process and recommendation system for employers and job seekers. The machine learning system can deliver accurate and customised job recommendations by evaluating user data such as work experience, education, skills, and preferences. To continuously enhance the matching process, the algorithm can also learn from user interactions on the platform, such as the types of jobs they apply for and their feedback on job offers. This will increase the competitiveness of our platform while providing additional value to our users. Furthermore, new minor features like as chat systems and review systems can significantly improve user experience and platform usability. They are also our focus for new system upgrades.

4.3.4 Development of a Mobile Application

The development of a mobile application is an important future plan for improving the platform's accessibility and convenience. It will also broaden the possible user base, as many individuals prefer mobile devices over desktop computers.
4.3.5 Expansion to Other Industries

The platform's current focus is on the recruitment procedure. The platform, however, has the potential to be expanded into other industries in the future. The expansion has the ability to bring in additional customers. Extending the platform to include loan information, for example, would open up an entirely new sector for future growth. Users can use this functionality to apply for loans and receive funds via blockchain smart contracts, increasing transparency and security for both lenders and borrowers. To improve the loan application process even further, the platform may integrate capabilities such as credit scoring, collateral management, and repayment tracking. Significant smart contract development, as well as connection to external data sources, would be required to assure accurate and up-to-date credit scores.
5. Conclusions

The emergence of blockchain and NFT has encouraged us to investigate the seemingly unlimited applications for these technologies.

This report provides an in-depth examination of a blockchain-based recruitment platform that allows businesses to validate the authenticity of candidate resumes while also streamlining the hiring process. This project intends to improve the hiring process's efficiency and dependability by reducing the time and resources required for recruiting while guaranteeing that the best-qualified individuals are chosen for the position. The following topics are covered in the report: An overview of the current employment process, a summary of blockchain technology and NFT, and an explanation of the issues that led to the establishment of this project, its objectives, workflow, technology use, and discussion.

One of the platform's distinguishing features is its implementation of blockchain technology to ensure that all data is securely and openly preserved. This secures the platform against fraud or manipulation and allows companies to immediately verify the authenticity of resumes presented by candidates.

Despite all of the platform's advantages, there are still challenges that must be addressed. For example, the program must acquire its first user base. To address this issue, the corporation may explore partnerships with educational institutions in order to encourage students to create accounts on the site. This would increase the platform's user base and appeal to businesses.

In conclusion, this platform has the potential to alter the recruitment process while also increasing its efficacy and dependability. By using cutting-edge technology such as blockchain and AI, it provides a user-friendly design that makes it simple to use for both companies and candidates. To enhance the interest of users, the platform may consider future developments such as the addition of new features and expansion. In summary, this project represents a significant advancement in streamlining the hiring process and developing a more open and reliable job market.
References


