Abstract

Nowadays, professional files have many functions in life. And the traditional paper version of the professional profile has been gradually replaced by the electronic version of the professional profile. But the problem with the electronic version of the professional profile is that it does not guarantee the authenticity of the content. In this project, a Blockchain-enabled professional profile management system will be designed and built to solve this problem.

Table of Contents

1 Introduction 6

1.1 Objectives 7

1.2 Risk Assessment ..... **Error! Bookmark not defined.**

1.3 Summary 7

2 Background and Related Work 8

2.1 LinkedIn 8

2.2 Coursera 8

2.3 Blockchain 9

2.4 Ethereum 9

2.5 Smart contract 10

2.6 Solidity 10

2.7 Decentralized application 10

3 Completed Work 11

3.1 Architecture Design 11

3.1.1 Front-end side 11

3.1.2 Back-end side 12

3.2 Identify the major stakeholders 12

3.2.1 User 12

3.2.2 Employer 12

3.2.3 University 12

3.3 “Person” contract 13

3.3.1 “getter” function and “setter” function of basic information 13

3.3.2 Education, Certificates, and the functions of adding them. 13

3.4 “Ownable” contract.....15

3.4.1 “isOwner” function 15

3.4.2 “onlyOwner” modifier 15

3.5 “PersonFactory” contract 16

3.5.1 “createPersonContract” function 16

3.5.2 “userRegister” function.....16

3.5.3 “getInfo” function 17

3.6 “FundMe” contract.....18

3.6.1 “PriceConverter” library 18

3.6.2 “fund” function 18

3.6.3 “withdraw” function.....19

4 On-going and Future Work.....**Error! Bookmark not defined.**

4.1 On-going objective.....**Error! Bookmark not defined.**

4.2 What to do in next semester.....**Error! Bookmark not defined.**

4.2.1 Design a professional profile management system**Error! Bookmark not defined.**

4.2.2 Implement and evaluate a prototype of the system**Error! Bookmark not defined.**

5 Conclusion .....**Error! Bookmark not defined.**

6 Reference 22

Table of Figures

Figure 1: Probability impact matrix before proposed solution **Error! Bookmark not defined.**

Figure 2: Architecture design.....11

Figure 3: Logic design of the Adding education .....**Error! Bookmark not defined.**

Figure 4: Gantt chart of semester 2.....**Error! Bookmark not defined.**

List of Tables

Table 1: Table of prioritized risk .....**Error! Bookmark not defined.**

1 Introduction

In our lives, Professional Profiles have many functions. It usually includes personal contact information, education, working experience, professional qualification, skills, academic publication, and CPD (continuous professional development). Students need to use the profile to apply to the school which is their preferred. People need to use profiles to apply for the jobs they want. Professional profiles play a very important role in people's lives. Therefore, the content of the professional profiles is crucial. The profile's content is the most direct way to let other people know you immediately. It should not contain any fraudulent information. For the paper version of the professional profiles, the authenticity of the content of the profile can be confirmed by presenting the corresponding paper proof, such as proof of graduation, proof of certification, etc.

Normally, people are more familiar with the paper version of professional profiles. But the use of the electronic version of the professional profile has gradually become mainstream. Compared to the paper version of the professional profile, the electronic version is more convenient. This is due to the fact that individuals may simply need to transmit a link or URL when submitting their professional profile to HR personnel. However, there is a problem with the electronic version compared to the paper version. Both the electronic and paper versions of the professional profile are written by the individual himself. The paper version of the professional profile can be confirmed by providing supporting documents to verify the authenticity of the content, but the electronic version has not yet solved this problem well. Therefore, ensuring that the content of the electronic version of the profile doesn't contain any false information is a big concern right now. What people are requesting right now is essential to ensuring the accuracy of the profile contents as well as enhancing the profile's usefulness. In this project, feasible solutions will be discussed and proposed.

LinkedIn, one of the most famous social networks in the world, will be the reference for this project. After registration, it automatically generates and brings in e-business cards, specifically for business people. An E-business card is just like a profile. It contains the basic information about the user, and also includes the education or certification. Employers can visit the personal homepage to check the information about that person. Not only can job seekers update their profiles on the platform, but companies and employers can also post jobs.

The project aims to design and build a decentralized professional profile management system. In this system, the profile's material must be created by a number of official organizations in addition to the users themselves. Users can edit some basic personal information such as name, age, height, birthday, etc. In another part, the education and certifications of the user will be written by the university or third-party official organizations. After the profile has been accomplished, users will be able to display only part of the profile by encrypting the relevant content. In addition, the system will be created using the Ethereum DApp development platform and distributed architecture. For this reason, it will be decentralized, robust, and non-repudiation. A data storage system with blockchain technology can offer higher security and integrity. Compared to centralized data storage, which may only have a few redundancy points, decentralized data storage makes it more challenging to hack into and erase all the data on the network.

1.1 Objectives

The main goal of this project is to build a decentralized professional profile management system. To achieve this goal, it will be separated into several objectives which are:

- Identify the major stakeholders of professional profile management, and the operations they are allowed on the profile.
- Investigate how blockchain technology like Ethereum can enforce the ownership of the profile information and ensure the authenticity of entries about educational and professional qualifications.
- Design a professional profile management system with a distributed architecture, which consists of both on-chain and off-chain components.
- Implement and evaluate a prototype of the professional profile management system as a distributed app (DApp) on Blockchain.

1.2 Summary

The following report is organized as follows: Chapter 2 introduces the background and related work of our work. Chapter 3 presents the work that has been done during this semester. Chapter 4 shows the work that is ongoing or will be completed in the future.

2 Background and Related Work

This section introduces the background of the professional profile management system and compares the differences between different websites and this system. And it describes the basic concepts of blockchain technology and the advantages of Dapp.

2.1 LinkedIn

As mentioned above, LinkedIn is one of the most famous social networking sites in the world. It provides a platform for countless job seekers, and users can record their profiles on their homepage. This provides a very convenient way for recruiters to know about users. After registration, it automatically generates and brings in e-business cards. E-business cards likes an electronic business card is like an electronic version of a professional profile. When a user is interested in a position offered by a company, he or she can contact the HR staff of that company through the chat function on the website. Or, if users want to apply for a job at a company, users can attach the URL of the homepage to the application. When the HR staffs are interested in the user, they will also check the e-card to learn more about the user's background information.

2.2 Coursera

Coursera, Inc. is a U.S. provider of large-scale open online courses, founded in 2012 by Stanford University computer science professors Andrew Wu and Daphne Kohler. Coursera partners with universities and other organizations to offer online courses, certificates, and degrees in a variety of disciplines. In 2021, it is estimated that approximately 150 universities offered through Coursera over 4, 000 courses [1].

After the user has completed the course on the website and passed the test, an electronic version of the qualification certificate will be generated. The certificate contains the personal information of the recipient and a URL link to the certificate. By clicking on the link, the page will confirm the authenticity of your Certificate and show more about the course’s syllabus.

2.3 Blockchain

A blockchain is a type of distributed ledger technology DLT that consists of growing list of records, called blocks, which are securely linked together using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data [2]. In this way, multiple blocks can be effectively linked into a chain-like structure. So blockchain transactions are irreversible, and once a transaction is completed, no block can be changed without altering the blocks that follow.

The key features of blockchain are: (1) Immutable (2) Distributed (3) Decentralized (4) secure [3]. Each modification of data or addition of data is a transaction. Each node first verifies a transaction's legality before adding it to the network if most of the nodes agree that it is valid. This implies that no transaction block can be added to the ledger without the consent of most nodes. If a malicious person wants to tamper with the data, then they must modify the data of most of the nodes in the chain, which is extremely difficult. Thus, the data stored in the blockchain by the professional profile management system can be very securely protected. Each block on the blockchain that contains information must generate a hash value using a hashing algorithm. When the content of the information has been changed, the hash value will be different from the original hash value. By comparing the hash values, it is possible to confirm whether the data has been modified. This also ensures the security of the information.

2.4 Ethereum

Ethereum is a decentralized, open source blockchain with smart contract functionality [4]. It is also a technology for building apps and organizations, holding assets, transacting, and communicating without being controlled by a central authority [5]. Ethernet enables the creation and deployment of smart contracts and decentralized applications (dApps) without downtime, fraud, control or interference from third parties. To achieve the above benefits, Ethereum has its own programming language which is called “Solidity”. Ethereum also has its own cryptocurrency, Ether, which is used to pay for certain activities on the Ethereum network.

2.5 Smart contract

Smart contracts are the fundamental building blocks of Ethereum applications. They are computer programs stored on the blockchain that allow converting traditional contracts into digital parallels [6]. By converting an agreement's provisions into computer code that runs automatically when the conditions of the contract are met, smart contracts digitalize contracts. They are able to run automatically. Users can keep track of their transactions, forecast their behaviour, and even utilize pseudonyms.

2.6 Solidity

Solidity is an object-oriented, high-level language for implementing smart contracts. Smart contracts are programs which govern the behaviour of accounts within the Ethereum state [7]. Solidity is heavily influenced by python and JavaScript. It is designed for the Ethereum virtual machine. So solidity is used to develop some programs based on the Ethereum platform. Contracts in Solidity are similar to classes in object-oriented languages. Each contract can contain declarations of State Variables, Functions, Function Modifiers, Events, Errors, Struct Types and Enum Types. Furthermore, contracts can inherit from other contracts.

2.7 Decentralized application

A decentralized application (DApp) is an application that can run autonomously, usually through the use of smart contracts, on a decentralized computing, blockchain or other distributed ledger system. [8]As an application, it can accomplish a number of user needs for the user. But what makes it different from traditional apps is that it runs without human intervention. DApps are built on top of the Ethereum blockchain or other blockchains. So far DApps have been used in many fields, such as games, gambling, and health and so on.

3 Completed Work

In this section, the parts that have been completed will be described here. In the first part will describe the major stakeholder of the professional profile management system. After that, the architecture design will be shown. Then in the third part will briefly introduce the “Person” contract. Following part will talk about the “Ownable” contract. In the last part, “FundMe” contract will be introduced.

3.1 Architecture Design

The core part of the system consists of two parts which are the front-end side and the back-end side. On the front-end side, the user interface will be designed used by JavaScript. For the back-end side, Blockchain technology will be the main part. Figure 2 shows the architecture design of the system.

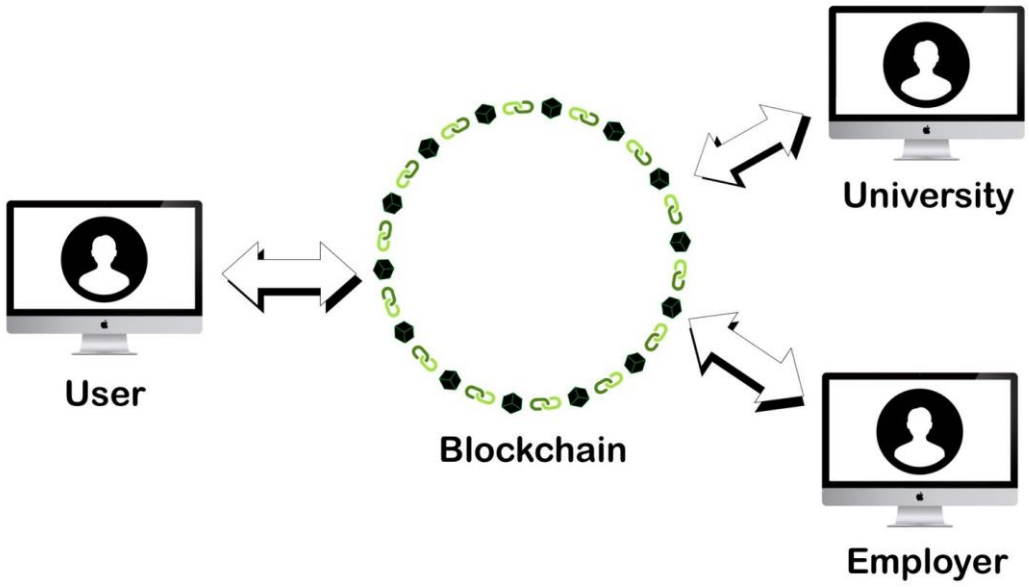

Figure 1: Architecture design

3.1.1 Front-end side

In the front-end side, the system provides a web page for users to use. This web page is designed and completed in JavaScript. Its main function is to facilitate the user to edit the

information with a visual page and to transfer the information to the backend. Nowadays, Web3 technology is available to transfer data directly to the blockchain through the browser.

3.1.2 Back-end side

In the back-end side, blockchain will be used to save the data of the system. Due to the nature of blockchain, this can be a good guarantee that the data will not be tampered with in the blockchain. And the nodes in the blockchain will all save the incoming information or execute the incoming commands, so it is more able to ensure the correctness of the output data.

3.2 Identify the major stakeholders

This section will introduce the major stakeholders of the professional profile management system which are the user, employer and university.

3.2.1 User

For the user, professional profile system helps users to quickly generate an electronic version professional profile. And users can select the parts they want to show in the professional profile.

3.2.2 Employer

For the employer, they need to be distinguished into two different types, “Producers” and “Consumers”. As the “Producer”, they can add the working experience of the user into the professional profile. As the “Consumers”, they can view the content of the resume that the user wants to display. And professional profile management system could make sure the authentication of content of the professional profile. This has been very helpful because the employer can select the employees with confidence when recruiting employees.

3.2.3 University

For universities, they can add relevant academic certificates to the resume to authenticate the user's education degree. As with the recruitment of employees, the system will guarantee the authenticity of the content of the professional profile. When a student applies to a university, the university can directly view the student's profile, which can simplify a lot of tedious steps.

3.3 “Person” contract

Each personal professional profile must include basic information about the individual such as name, age, height, and birthday. In addition, each user may have more than one education or more than one certificate.

3.3.1 “getter” function and “setter” function of basic information

For the basic information, each of them has a get function and a set function. This is very convenient for users who only want to return or modify an attribute. The user does not need to enter all the information to modify one of the attributes.

All setter functions are public and view functions. View functions allow the use of the function without consuming gas, which reduces the cost to the user when using the system.

3.3.2 Education, Certificates, and the functions of adding them.

The paper version of the certificate of degree (education) should have written school, degree, and major this basic information. So, in the person contract, there will be an Education structure, and each Education will have three attributes: school, degree, and major.

Each user may have more than one education, so in the person contract, there will be an array of Education, called "education". To add the education to the array, there is a function called “addEducation”. When using the “addEducation” function, the user needs to enter the attributes of school, degree, and major.

Certificate need contains name of the certificate, grade, and the getting date. It also has an array “certificate” to store it. Same as the education array, there is a function which is use to add the certificate to the array.

```
pragma solidity ^0.8.0;

import "./ownable.sol";
import "@chainlink/contracts/src/v0.8/interfaces/AggregatorV3Interface.sol";

contract Person is Ownable{

    string _name;
    uint _age;
    uint _height;
```

```
    string _birthday;

    struct Education {
        string school;
        string degree;
        string major;
    }
    struct Certificate {
        string nameOfcertificate;
        string grade;
        string date;
    }

    Education[] public education;

    function addEducation(string memory _school, string memory _degree, string
memory _major) onlyOwner public {
        education.push(Education( _school, _degree, _major));
    }

    Certificate[] public certificate;

    function addCertificate(string memory _nameOfCertificate, string memory
_grade, string memory _date) onlyOwner public {
        certificate.push(Certificate( _nameOfCertificate, _grade, _date));
    }

    function getName() public view returns(string memory){
        return _name;
    }
    function getAge() public view returns(uint){
        return _age;
    }
    function getHeight() public view returns(uint){
        return _height;
    }
    function getBirthday() public view returns(string memory){
        return _birthday;
    }
    function setName(string memory name) onlyOwner public{
        _name = name;
    }
    function setAge(uint age) onlyOwner public{
        _age = age;
    }
    function setHeight(uint height) onlyOwner public {
        _height = height;
    }
```

```
    function setBirthday(string memory birthday) onlyOwner public{
        _birthday = birthday;
    }
}
```

3.4 “Ownable” contract

The professional profile contains a lot of personal information. It is quite private. And most of the functions in the system require an “ownable” function to enforce the ownership of the profile information.

Each “Ownable” contract has a private address attribute called “owner”. The value of “Owner” will be set as the address of message sender.

3.4.1 “isOwner” function

This function is used to check if the user's address is the same as the registered address. The “isOwner” function is the core part of the “Ownable” contract.

3.4.2 “onlyOwner” modifier

Modifier can change the behaviour of a function. In fact, modifiers are most often used for behaviours checking, which makes it possible to reduce the reuse of checking code and make the code look easier to understand.

The “onlyOwner” modifier requires that the “isOwner” function be run at first. If the return value is true, then the rest of the function is executed. If the return value is false, then the subsequent functions will not be executed. "onlyOwner" is also used in the setter function of the "Person" contract to make those setter function only used by the owner of the address.

```
pragma solidity ^0.8.8;

contract Ownable{

  address private _owner;

  constructor(){
    _owner = msg.sender;
  }

  function owner() onlyOwner public view returns(address) {
```

```
    return _owner;
  }

  modifier onlyOwner() {
    require(isOwner(),
    "Function accessible only by the owner !!");
    _;
  }

  function isOwner() public view returns(bool){
    return msg.sender == _owner;
  }
}
```

3.5 “PersonFactory” contract

By the time the professional profile management system is completed, there will be more than one user. In the "Person" contract, we treat each user as a contract, and the "PersonFactory" contract is used to connect these contracts. This is like the shallow meaning of blockchain. The role of this contract is to connect each block together to form a chain structure. And in this contract, there is a function to add a new "Person" and a function to get information about "Person". In the "PersonFactory" contract, there is an array called "personArray" to store these "Person" contracts.

3.5.1 “createPersonContract” function

This function is to create a new “Person” contract. After creating a new contract, it will be automatically adding to last position of the person array.

3.5.2 “userRegister” function

When the "person" contract is first created, it does not contain any value. So the "userRegister" function is used to allow the user to fill in the basic information that they need to have. This process is very similar to user registration. So this function is called "userRegister". When using this function, users will be asked to input information such as the “person” index, name, age, height, and birthday. Then the system will automatically using the setter function of the “Person” contract to set those values. Education and certificate are not the necessary to the “Person” contract. So if the user wants to add the education or the

certificate. They could use “addEducation” function or “addCertificate” function after creating contract.

3.5.3 “getInfo” function

This function is preparing to the situation which is user want to show their professional profile to the employer or university. User just input the corresponding “personIndex”, all the information of that “Person” contract will be returned. Because of the “personIndex” is like a primary key of the “Person” array.

```
pragma solidity ^0.8.0;

import "./Person.sol";

contract PersonFactory{
    Person[]  public personArray;

    function  createPersonContract() public returns(uint256){
        Person person = new Person();
        personArray.push(person);
        uint256 personIndex = personArray.length;
        return personIndex;
    }

    function userRegister(uint256 _personIndex, string memory _name, uint _age,
uint _height, string memory _birthday) public{
        personArray[_personIndex].setName(_name);
        personArray[_personIndex].setAge(_age);
        personArray[_personIndex].setHeight(_height);
        personArray[_personIndex].setBirthday(_birthday);
    }
    function getInfo(uint256 _personIndex) public returns (string memory name,
uint age, uint height, string memory birthday)
        {
         string memory name = personArray[_personIndex].getName();
         uint age = personArray[_personIndex].getAge();
         uint height = personArray[_personIndex].getHeight();
         string memory  birthday = personArray[_personIndex].getBirthday();
         }
}
```

3.6 “FundMe” contract

Whenever a user creates a new "personal" contract or uses most of the functions, this is making a transaction. The transaction will consume Wei or Ether. So, there is a balance of each contract. In the contract, each address will map to one amount. The variable “addressToAmountFunded” is just like a balance variable. And there is an address array that store the funders’ address.

3.6.1 “PriceConverter” library

Libraries are similar to contracts, but their purpose is that they are deployed only once at a specific address. As a library is an isolated piece of source code, it can only access state variables of the calling contract if they are explicitly supplied (it would have no way to name them, otherwise) [9].

This contract has been connected to a web interface that updates conversion rates in real-time before it starts. Users can view the conversion rates between currencies on that webpage in real-time. Also, this web page is decentralized. The system can get the correct conversion rate from this web interface.

“The "PriceConverter" library has an internal function "getPrice". Search the website for the address of the currency you want to convert. If you want to get the latest round of data, it will be returned together with other data. The purpose of this “getPrice” function is to get the price of Ether. There is another internal function called “getConversionRate” which is to find the.

3.6.2 “fund” function

The fund function is payable. It is the core part of the “FundMe” contract. The function will first check if the value of the fund is greater than the minimum value of the limitation. If it is lower than the minimum value, then the system will return "You need to spend more ETH!". If it is greater than the minimum value, then the address of the message sender's mapped “addressToAmountFunded” will be increased and the address will be added to the last position of the funder array.

3.6.3 "withdraw" function

The system provides the "withdraw" function to the owner. Only the owner could use this function for the withdrawal of remain of balance.

```
pragma solidity ^0.8.8;

import "@chainlink/contracts/src/v0.8/interfaces/AggregatorV3Interface.sol";

library PriceConverter {
    // We could make this public, but then we'd have to deploy it
    function getPrice() internal view returns (uint256) {
        AggregatorV3Interface priceFeed =
AggregatorV3Interface(0xD4a33860578De61DBAbDc8BFdb98FD742fA7028e);
        (, int256 answer, , , ) = priceFeed.latestRoundData();
        return uint256(answer * 10000000000);
    }

    function getConversionRate(uint256 ethAmount) internal view returns
(uint256)
    {
        uint256 ethPrice = getPrice();
        uint256 ethAmountInUsd = (ethPrice * ethAmount) / 1000000000000000000;
        return ethAmountInUsd;
    }
}
    error NotOwner();

contract FundMe {
    using PriceConverter for uint256;
    mapping(address => uint256) public addressToAmountFunded;
    address[] public funders;
    address public _owner;
    uint256 public constant MINIMUM_USD = 20 * 10 ** 18;

    constructor() {
        _owner = msg.sender;
    }

    function fund() public payable {
        require(msg.value.getConversionRate() >= MINIMUM_USD, "You need to
spend more ETH!");
        addressToAmountFunded[msg.sender] += msg.value;
        funders.push(msg.sender);
    }

    function getVersion() public view returns (uint256){
```

```
        AggregatorV3Interface priceFeed =
AggregatorV3Interface(0xD4a33860578De61DBAbDc8BFdb98FD742fA7028e);
        return priceFeed.version();
    }

    modifier onlyOwner {
        if (msg.sender != _owner) revert NotOwner();
        _;
    }

    function withdraw() public onlyOwner {
        for (uint256 funderIndex=0; funderIndex < funders.length;
funderIndex++){
            address funder = funders[funderIndex];
            addressToAmountFunded[funder] = 0;
        }
        funders = new address[](0);
        (bool callSuccess, ) = payable(msg.sender).call{value:
address(this).balance}("");
        require(callSuccess, "Call failed");
    }

    fallback() external payable {
        fund();
    }

    receive() external payable {
        fund();
    }

}
```

4 Reference

[1] "Coursera," [Online]. Available: https://zh.wikipedia.org/zh-tw/Coursera.

[2] "Blockchain," [Online]. Available: https://en.wikipedia.org/wiki/Blockchain.

[3] "Features of Blockchain," [Online]. Available: https://www.geeksforgeeks.org/features-of-blockchain/.

[4] "Ethereum," [Online]. Available: https://en.wikipedia.org/wiki/Ethereum.

[5] "Eherreum," [Online]. Available: https://ethereum.org/en/what-is-ethereum/.

[6] "smart-contracts," [Online]. Available: https://ethereum.org/en/smart-contracts/.

[7] "Solidity," [Online]. Available: https://docs.soliditylang.org/en/v0.8.17/.

[8] "DApps," [Online]. Available: https://en.wikipedia.org/wiki/Decentralized_application.

[9] "library," [Online]. Available: https://docs.soliditylang.org/en/v0.8.14/contracts.html?highlight=library#libraries.

[10] "Linkedin," [Online]. Available: https://www.linkedin.com/company/linkedin/?originalSubdomain=cn. [Accessed 7 9 2022].