

블록체인기반 핀테크 및 응용 SW 개발자 과정

블록체인을 이용한 프리랜서 경력 보장 웹 서비스

[시스템 구조 설계서]

2022. 08. 18

박광범

목 차

1. 개요

2. 개발 환경

3. 요구사항

4. Work Bench Schedule

5. 레이어 단위 기능

6. 인터페이스

1. 개요

프리랜서들의 경력은 일반 직장인들에 비해 인증과 보장이 쉽지 않습니다. 또한 다양한 경력이 있을 경우 이를 일목요연하게 정리하여 증명하는 것은 또 다른 문제입니다. 저희는 이러한 문제들을 해결하고자 신뢰성을 보장할 수 있도록 블록체인 기술을 도입하여 프리랜서들의 경력을 보장할 수 있는 서비스를 계획했습니다.

2. 개발 환경

개발 환경	Windows, Mac, SpringBoot, VSCode, JDK 11, Apache Tomcat 9.0, Thymeleaf
사용 언어	Java, HTML5, JavaScript, CSS, Golang,
데이터베이스	MySQL
API & Libraries	Spring-JPA, Hibernate, Json, Spring-Security, Github.com/gorilla/mux, Github.com/bigpicturelabsinc/consensusPBFT, Github.com/gorilla/mux, Github.com/btsuite/btcutil

3. 요구사항

요구사항 번호	요구사항 명	요구사항 내용
RQ-1	월렛 생성	입력값 => 월렛생성 signal 출력값 => PrivateKey , PublicKey, Address
RQ-3	경력 등록 요청	입력 값 Address != nil & TxID == nil & PrivateKey 출력값 => 경력 등록된 TxID
RQ-4	경력 조회	비회원도 조회 가능 입력값 => Address 출력값 => TXs
RQ-5	상세 경력 조회	입력값 => TxID (받아와서 가공해 Block ID 추출) 출력값 => Block, TX

3. WBS

		월	화	수	목	금	토	일	월	화	수	목	금	토	일	월	화	수	목	금	토	일	월	화	발표 일
업무	상세작업	2 5	2 6	2 7	2 8	2 9	3 0	3 1	1	2	3	4	5	6	7	8	9	1 0	1 1	1 2	1 3	1 4	1 5	1 6	
미들웨어	Ta(블럭생성요청)																								
	Tb(블럭조회)																								
	Tc(상세조회)																								
	Td(전체조회)																								
프론트웨 어	회원가입(security)																								
	월렛페이지생성구 현																								
	블럭 요청 페이지 구현																								
	공지사항																								
코어	전자서명																								
	월렛 생성																								
	월렛 유효검사																								
	마지막 요청 데이터 저장																								
	노드 추가/제거 테스트																								
	10000 번 합의 테스트																								
	MSP 구현																								
	리더노드 업데이트																								
pBFT	코드 분석																								

4. 레이어 단위 기능

경력 인증 플랫폼 : 서비스 Layer1

1. 회원

1.1 회원 가입

1.2 회원 전자지갑(wallet) ————— A (create)

1.2.1 회원 암호 키페어 (공개키, 개인키) ————— A-1 (create)

- 주소
- 인증 내역

1.3 tx 발생 이력 (ref) ————— B (ref)

2. 경력 인증

2.1 경력 등록(tx/block)

- tx 생성 ————— C - 1 (create)
- block 생성 ————— C - 2 (create)

2.2 경력 조회(tx/block)

2.2.1 전체 리스트 조회(tx) ————— D (ref)

- 경력
- 등록 일자
- 회사명

2.2.2 상세 조회(tx/block) ————— E (ref)

- 경력
- 회사명
- 직무
- 등록일자
- 경력증명서 URL
- 블록체인 메타(header) 정보 ————— F (ref)
- tx 메타 정보(TxID)

Layer2

RoutePath	FunctionName	Layer2	Method
/MakeWallet	ConnectWallet	A, A-1 Layer3(1,2,3,4)	POST
/refTx	FindTxByAddr	B Layer3(6)	GET
/Apply/Career	ApplyCareer	C-1 Layer3(5)	POST

/newBlk	CreateNewBlock	C-2 Layer3(8)	POST
/refTx“, “tx/ref	FindTxByAddr	D Layer3(6)	GET
/searchBlk	SearchBlock	E Layer3(9)	GET
/detailTx	DetailTx	F Layer3(7)	GET

Layer3

<RPC>

1. 키 생성
2. 주소 생성
3. 주소 검증
4. 지갑 생성

<HTTP>

5. 거래 생성
6. 거래 전체 조회 (By Address)
7. 거래 상세 조회 (By TxID)
8. 블록 생성
9. 블록 조회 (by TxID)

4. 인터페이스

<RPC>

-지갑 생성 요청(키 생성, 주소 생성)

- URL: /MakeWallet
- Parameter: {w http.ResponseWriter, re *http.Request}
- Request(receive): Alias (지갑 이름)
- Response(send): 지갑 주소, PublicKey , PrivateKey

- Return: -

```
func (r *Request) ConnectWallet(w http.ResponseWriter, re *http.Request)
```

-지갑 주소 유효성 검사 요청 (RPC Connection) (By Address)

- URL: /CheckAddress
- Parameter : (w http.ResponseWriter, re *http.Request)
- Request(receive): "Address" : 지갑주소
- Response(send): Result
- Return: Boolean

```
func (r *Request) CheckAddress(w http.ResponseWriter, re *http.Request)
```

-지갑 주소 유효성 검사

- TCP: 9000
- Parameter: {Address string, reply *Reply}
- Request(receive): Address : 지갑주소
- Response(send): Result
- Return: Bool

```
func (wRPC *RpcServer) CheckAddress(Address string, reply *Reply) error
```

-지갑 정보 가져오기 요청

- URL: /GetWallet
- Parameter: {(w http.ResponseWriter, re *http.Request)}
- Request(receive): "Address ": 지갑 주소
- Return: w *Wallet

```
func (r *Request) GetWallet(w http.ResponseWriter, re *http.Request)
```

-지갑 정보 가져오기

- TCP: 9000
- Parameter: {(Address string, reply *Reply)}

- Request(receive): “Address”: 지갑
- Response(send): w Wallet
- Return: bool

```
func (wRPC *RpcServer) GetWallet(Address string, reply *Reply) error
```

<HTTP>

-트랜잭션 생성

- URL: /Apply/Career
- Parameter: {http.ResponseWriter w, *http.Request req}
- Request(receive): “address”, “data”, “applier”, “company”, “career”, “payment”, “job”, “proof” json
- Response(send): “address” string, “txid” [32]byte
- Return: -

```
func ApplyCareer(w http.ResponseWriter, req *http.Request)
```

-블록 생성

- URL: /newBlk
- Parameter: {http.ResponseWriter w, *http.Request req}
- Request(receive): {“data”: 기타 메시지(copyright 등), “txid”: TxID}
- Response(send): Block ID [32]byte
- Return: -

```
func (wRPC *RpcServer) GetWallet(Address string, reply *Reply) error
```

```
func FindAllbyAddr(w http.ResponseWriter, req *http.Request)
```

-트랜잭션 전체 조회(By Address)

- URL: /refTx
- Parameter: {http.ResponseWriter w, *http.Request req}
- Request(receive): “address” : 지갑 주소
- Response(send): “txID”(배열): 트랜잭션 해시, “career”(배열): 경력, “company”(배열): 회사
- Return: -

-상세조회

- URL: /detailTx
- Parameter: {w http.ResponseWriter, re *http.Request}
- Request(receive): TxID
- Response(send): Hash(블록 해시), Data(블록 데이터), Timestamp(블록 타임스탬프), Txid(트랜잭션 해시), Applier(신청자), Company(경력회사), Career(경력기간), Job(직종, 업무), Proof(경력증명서 pdf)
- Return: -

```
func DetailTx(w http.ResponseWriter, req *http.Request)
```

-블록조회

- URL: /searchBlk
- Parameter: {w http.ResponseWriter, re *http.Request}
- Request(receive): TxID
- Response(send): Hash(블록 해시), Data(블록 데이터), Timestamp(블록 타임스탬프)
- Return: -

```
func SearchBlock(w http.ResponseWriter, req *http.Request)
```