



네트워크/체인코드-스마트계약 /dApp Front 설계, 구현



프로젝트 설계주간 일정

09월 18일	수	오후	네트워크/체인코드-스마트계약 /dApp Front 설계	환경설정, 복습, 주제선정, 일정작성	1. 환경설정 메뉴얼 2. 팀주제 개요 및 기능문서 3. 주요일정 테이블
19	목	오전	네트워크/체인코드-스마트계약 /dApp Front 설계	네트워크 설계 예제(티메이트) 1. 아이덴티티 설계, 네트워크 구조도 작성 2. 기관, 피어, 채널설계 3. 컨테이너 설계 4. 구동, 운용설계	네트워크 개발 설계서 1. 아이덴티티 구조도 2. 설정정보 (제네시스, 채널, 앵커) 3. 실행 스크립트 슈도코드 4. 실행 컨테이너 설정정보 5. 네트워크 구조도
19	목	오후	네트워크/체인코드-스마트계약 /dApp Front 설계	(팀프로젝트) 네트워크 설계	
23	월	오전	네트워크/체인코드-스마트계약 /dApp Front 설계	체인코드 설계 예제(티메이트) 1. 체인코드 기능리스트 작성 2. 체인코드 데이터 정의 3. 체인코드 인터페이스 설계 4. 체인코드 전체 구조도 작성	체인코드 설계서 1. 체인코드 기능리스트 2. 블록체인 데이터 정의 3. 체인코드 인터페이스 설계서 4. 체인코드 전체 구조도
23	월	오후	네트워크/체인코드-스마트계약 /dApp Front 설계	(팀프로젝트) 체인코드 설계	
24	화	오전	네트워크/체인코드-스마트계약 /dApp Front 설계	프론트앤드 설계 예제(티메이트) 1. 프론트앤드 기능리스트작성 2. 웹서버 EndPoint설계 3. UI 설계 4. 연동플로우챠트 작성	프론트앤드 설계서 1. 프론트앤드 기능리스트 2. 연동 플로우챠트 3. 웹서버 REST EndPoint 4. UI 프로토타입 디자인
24	화	오후	네트워크/체인코드-스마트계약 /dApp Front 설계	(팀프로젝트) 프론트앤드 설계	



프로젝트 구현주간 일정

25	수	오후	네트워크 개발/체인코드-스마트 계약 구현	설계문서수정 및 구현 계획, 일정작성, 전체 예제 구현 시나리오분석	1. 설계문서(네트워크, 체인코드, 프론트앤드) 2. 구현일정 3. 구현 디렉토리 및 개발 라이브러리 설치된 PC
26	목	오전	네트워크 개발/체인코드-스마트 계약 구현	네트워크 구현 예제(티메이트) 1. crypto-config.yaml, configtx.yaml 수정 generate.sh 작성 및 수행 2. docker-compose.yml 작성 start.sh, teardown.sh 작성 및 수행 3. 네트워크 실행 메뉴얼 및 테스트 보고서 작성 4. 네트워크 보고서 정리	네트워크 개발 구현서 1. 아이덴티티 디렉토리 구조 2. 설정파일 문서 (crypto-config, configtx) 3. 실행 컨테이터 설정(docker-compose) 4. 실행 스크립트 (generate, start, teardown) 5. 네트워크 실행 테스트 결과 보고서
26	목	오후	네트워크 개발/체인코드-스마트 계약 구현	(팀프로젝트) 네트워크 구현	
30	월	오전	네트워크 개발/체인코드-스마트 계약 구현	체인코드 구현 예제(티메이트) 1. 체인코드 구현 2. 체인코드 설치 배포 테스트 3. 체인코드 구현서 작성 4. 체인코드 설치배포 메뉴얼 및 테스트 보고서 작성	체인코드 개발 구현서 1. 체인코드 기능구현목록 2. 블록체인 데이터 구현서 3. 체인코드 Contract구현 데이터, 함수목록및 설명 4. 체인코드 설치 배포 스크립트 5. 체인코드 테스트 결과 보고서
30	월	오후	네트워크 개발/체인코드-스마트 계약 구현	(팀프로젝트) 체인코드 구현	
10월 01일	화	오전	네트워크 개발/체인코드-스마트 계약 구현	프론트앤드 구현 예제(티메이트) 1. 웹서버 REST endpoint 라우팅 작성 2. 웹 UI 프로토타입 작성 3. 체인코드 연동부 작성 4. 테스트 및 보고서 작성	프론트앤드 구현서 1. 프론트앤드 기능구현목록 2. 웹서버-체인코드 연동 구현서 3. 웹서버 REST EndPoint 목록 및 설명 4. 웹서버 구동 및 사용 메뉴얼 4. 클라이언트 구조설명 5. 클라이언트 구현 스크린샷 및 테스트 보고서
1	화	오후	네트워크 개발/체인코드-스마트 계약 구현	(팀프로젝트) 프론트앤드 프로토타입 구현, 연동테스트	



1일차 실습

- dAPP 설계 예제(티메이트)

환경설정, 복습, 주제선정, 일정작성

주요 결과물

1. 팀주제 개요 및 기능문서
2. 환경설정 메뉴얼
3. 주요일정 테이블

프로젝트 개요



프로젝트명

teamate (팀메이트)

요약

팀프로젝트 멤버를 구하기 위한 페이지

기능

팀프로젝트 리스트

팀프로젝트 등록

사용자 등록

사용자 리스트

사용자 팀프로젝트 참여

팀프로젝트 참여사용자평가



설계 일정



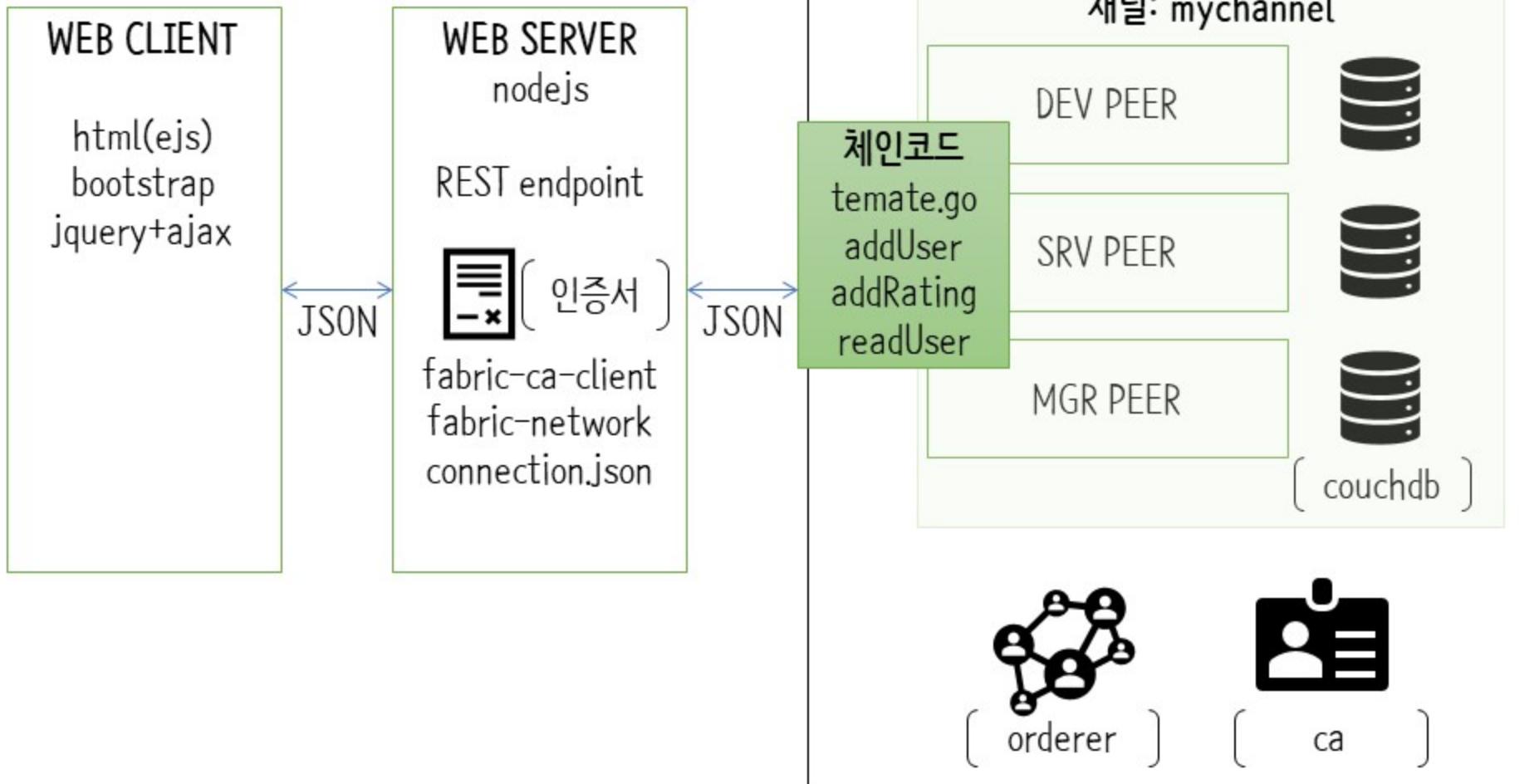
일정상세



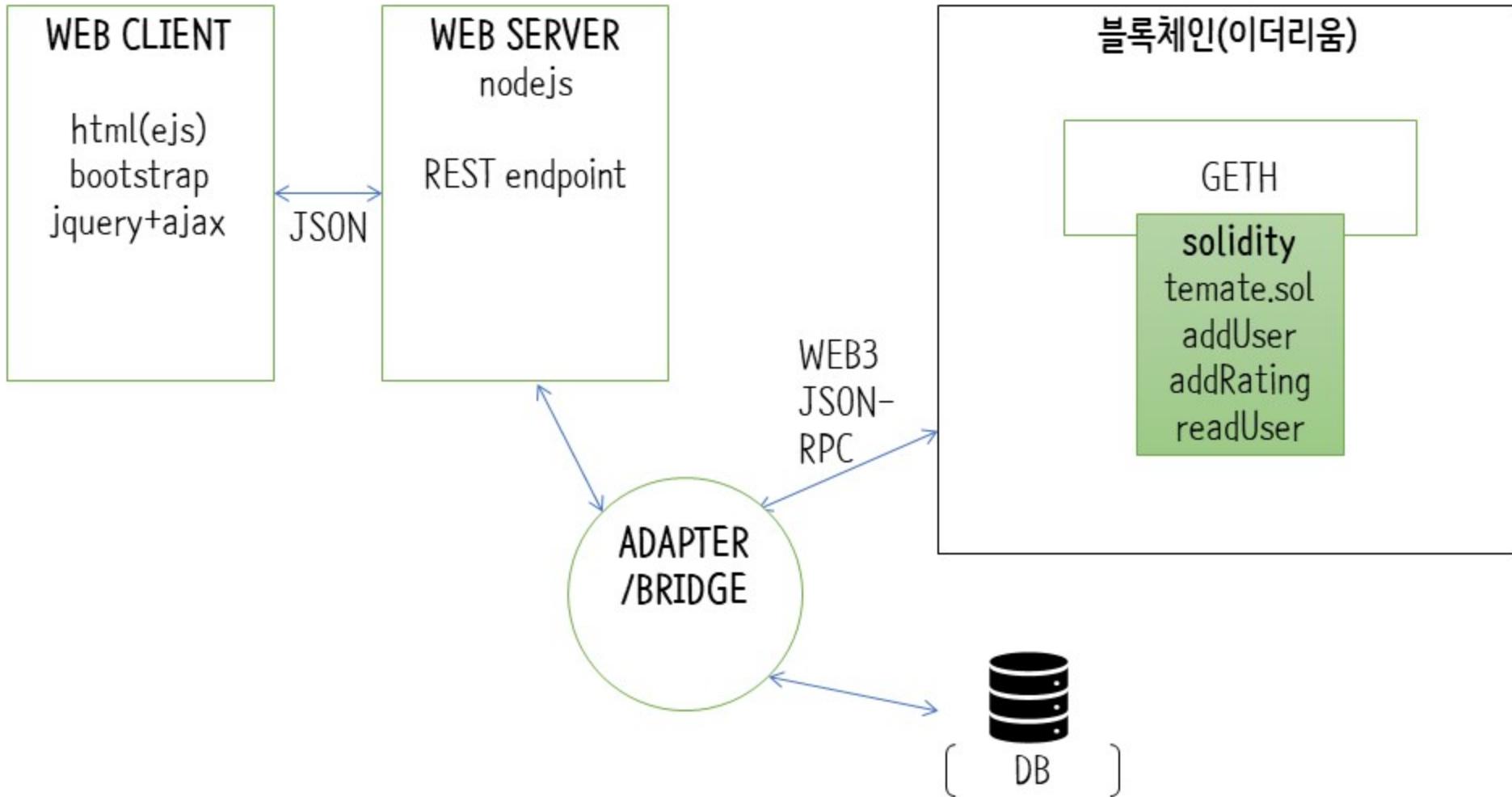
- 블록체인 네트워크 응용프로그램 선택
 - 가장 쉬운 활용 예를 선택
 - 개요 문서 작성
- 디렉토리 구조 작성
 - network, contract, application
- 네트워크 작성
 - crypto-config.yaml, configtx.yaml, generate.sh
 - docker-compose.yml, start.sh, teardown.sh
- 체인코드 작성
- 프론트엔드 작성
 - nodejs (web server)
 - html (web client)
- 테스트 연동



시스템 구조도-티메이트



시스템 구조도-티메이트



프론트엔드 프로토타입 1



프로토타입

이메일로 메이트를 조회합니다

user1

조회하기

average 3.7500000024999998

project title p1

project title p2

project title p4

project title p3

추가에 성공했습니다!

새 메이트를 추가합니다

newUser

추가하기

프론트엔드 프로토타입 2



프로토타입

조회에 성공했습니다!

이메일로 메이트를 조회합니다

user1

조회하기

average	3.7500000024999998
project title	p1
project title	p2
project title	p4
project title	p3

새 메이트를 추가합니다

새로 추가할 메이트의 email을 입력하세요

추가하기



프로젝트 제안서 포함내용

- 팀주제 개요 및 기능리스트
 - 제안서
 - 요구사항분석서
 - 기술분석서
 - 전체 구조도
 - 예상결과물
 - 주요일정 테이블
 - 설계일정 간트차트
 - 상세일정
 - 프로토타입 계획서
 - 네트워크 프로토타입
 - 체인코드 프로토타입
- 웹서버 프로토타입
웹클라이언트프로토타입

예상 결과물 리스트



네트워크 개발 설계서

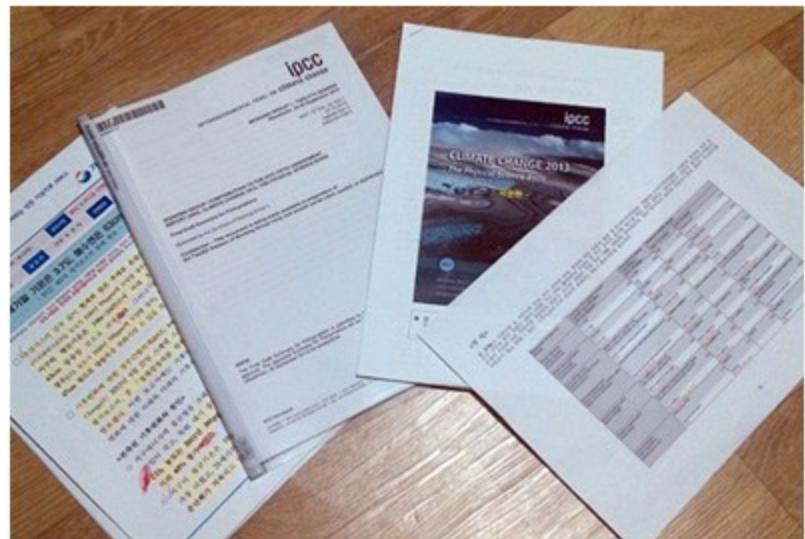
1. 아이덴티티 구조도
2. 설정정보 (제네시스, 채널, 앵커)
3. 실행 스크립트 슈도코드
4. 실행 컨테이터 설정정보
5. 네트워크 구조도

체인코드 설계서

1. 체인코드 기능리스트
2. 블록체인 데이터 정의
3. 체인코드 인터페이스 설계서
4. 체인코드 전체 구조도

프론트앤드 설계서

1. 프론트앤드 기능리스트
2. 연동 플로우챠트
3. 웹서버 REST EndPoint
4. UI 프로토타입 디자인

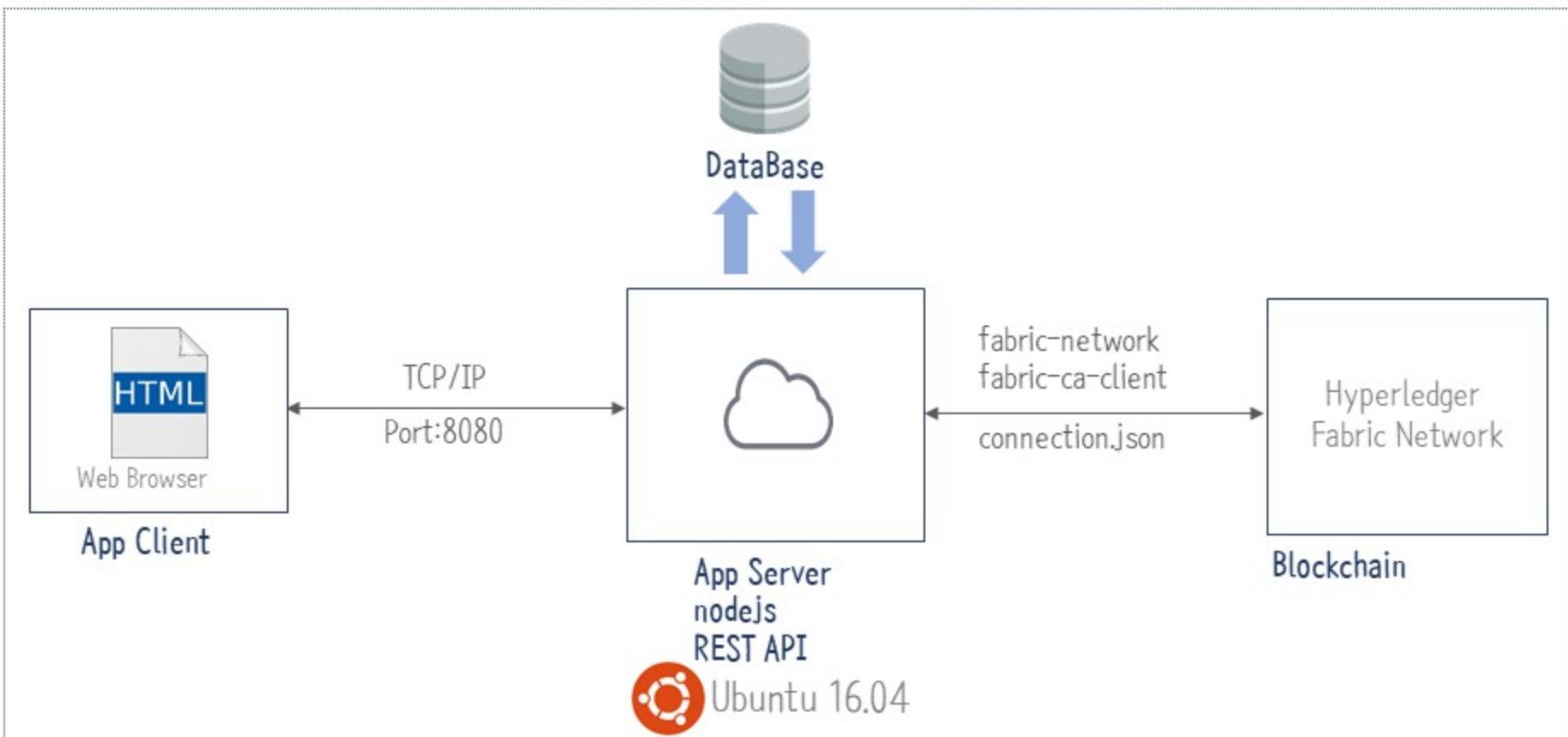




2일차 실습

- 네트워크 설계 예제(티메이트)
 1. 아이덴티티 설계, 네트워크 구조도 작성
 2. 기관, 피어, 채널설계
 3. 컨테이너 설계
 4. 구동, 운용설계
- 네트워크 설계 자유주제

네트워크 관계도





ca.example.com
7054
id:admin
password:adminpw



orderer.example.com
7054
Name/ID
OrdererOrg/OrdererMSP
type : solo



cli
CORE_PEER_ADDRESS=peer0.org1.example.com:7051



docker

Ubuntu 16.04

기관 피어 채널 설계

crypto-config 설계

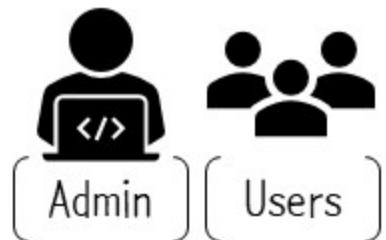
- orderer
- organization
 - devorg, mgrorg, svcorg
 - template
 - user

1
1

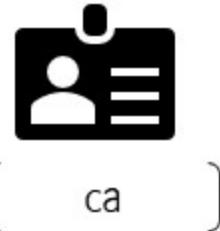
configtx 설계

- organization
 - orderer
 - organization
 - profile
 - genesis
 - channel
- solo
- devorg, mgrorg, svcorg
- devorg, mgrorg, svcorg
- devorg, mgrorg, svcorg

인증서 관리



TCP/IP
Port:7054



```
35 },
36   "orderers": {
37     "orderer.example.com": {
38       "url": "grpc://localhost:7050"
39     }
40   },
41   "peers": {
42     "peer0.org1.example.com": {
43       "url": "grpc://localhost:7051"
44     }
45   },
46   "certificateAuthorities": {
47     "ca.example.com": [
48       "url": "http://localhost:7054",
49       "caName": "ca.example.com"
50     ]
51   }
--
```



컨테이너 설계

- docker compose 설계
 - orderer (TLS/KAFKA/RAFT)
 - mspid/mspdir
 - genesisfile
 - ports
 - volume
 - TLS?
 - ca (TLS/ADMIN-PASSWORD)
 - ca key/cert
 - ports
 - volume

```
6   version: '2'  
7  
8   networks:  
9     | basic:  
10    | services:  
11    > ca.example.com: ...  
27  
28    > orderer.example.com: ...  
48  
49    > peer0.org1.example.com: ...  
87  
88    > peer0.org2.example.com: ...  
126  
127    > peer0.org3.example.com: ...  
165  
166    > couchdb1: ...  
178    > couchdb2: ...  
190    > couchdb3: ...  
202  
203    > cli: ...
```

컨테이너 설계

- docker compose 설계
 - peer (TLS)
 - mspid/mspconfigpath
 - core_peer_id/core_peer_address
 - ports
 - volumn
 - db info
 - TLS?
 - couchdb (leveldb)
 - couchdb id/password
 - ports
 - cli (application identity)
 - gopath, core_peer_address, mspconfigpath
 - volumn

```

6   version: '2'
7
8   networks:
9     | basic:
10
11  services:
12  > ca.example.com: ...
27
28  > orderer.example.com: ...
48
49  > peer0.org1.example.com: ...
87
88  > peer0.org2.example.com: ...
126
127 > peer0.org3.example.com: ...
165
166 > couchdb1: ...
178 > couchdb2: ...
190 > couchdb3: ...
202
203 > cli: ...

```



컨테이터 목록

- ▶ dev-peer0.org1.example.com-sacc-1.0-ad1e599bb9b4ca746846b524003
- ▶ hyperledger/fabric-peer (peer0.org1.example.com - Up 27 hours)
- ▶ hyperledger/fabric-peer (peer0.org2.example.com - Up 27 hours)
- ▶ hyperledger/fabric-peer (peer0.org3.example.com - Up 27 hours)
- ▶ hyperledger/fabric-ca (ca.example.com - Up 27 hours)
- ▶ hyperledger/fabric-couchdb (couchdb1 - Up 27 hours)
- ▶ hyperledger/fabric-couchdb (couchdb2 - Up 27 hours)
- ▶ hyperledger/fabric-couchdb (couchdb3 - Up 27 hours)
- ▶ hyperledger/Fabric-orderer (orderer.example.com - Up 27 hours)
- ▶ hyperledger/fabric-tools (cli - Up 27 hours)



쉘스크립트 슈도코드

- generate.sh
 - PATH환경변수 확인 (~/.fabric-samples/bin 디렉토리)
 - cryptogen
 - configtxgen
- start.sh
 - ca key replacement
 - config, crypto-directory 선 확인절차
 - docker-compose를 이용한 컨테이너 실행
 - 채널 생성
 - 채널 조인



쉘스크립트 슈도코드

- chaincode.sh
 - 체인코드 설치
 - 체인코드 배포
 - 체인코드 테스트
 - (invoke/query)
 - 주요 function and Arguments 설계
- teardown.sh
 - 컨테이터 삭제
 - 이미지 삭제
 - 네트워크 다운



프로토 타입 작성

- 네트워크
 - basic-network 를 활용
- 체인코드
 - sacc를 활용
- 어플리케이션
 - fabcar의 javascript 디렉토리와 express 서버 활용

프로토 타입 데모



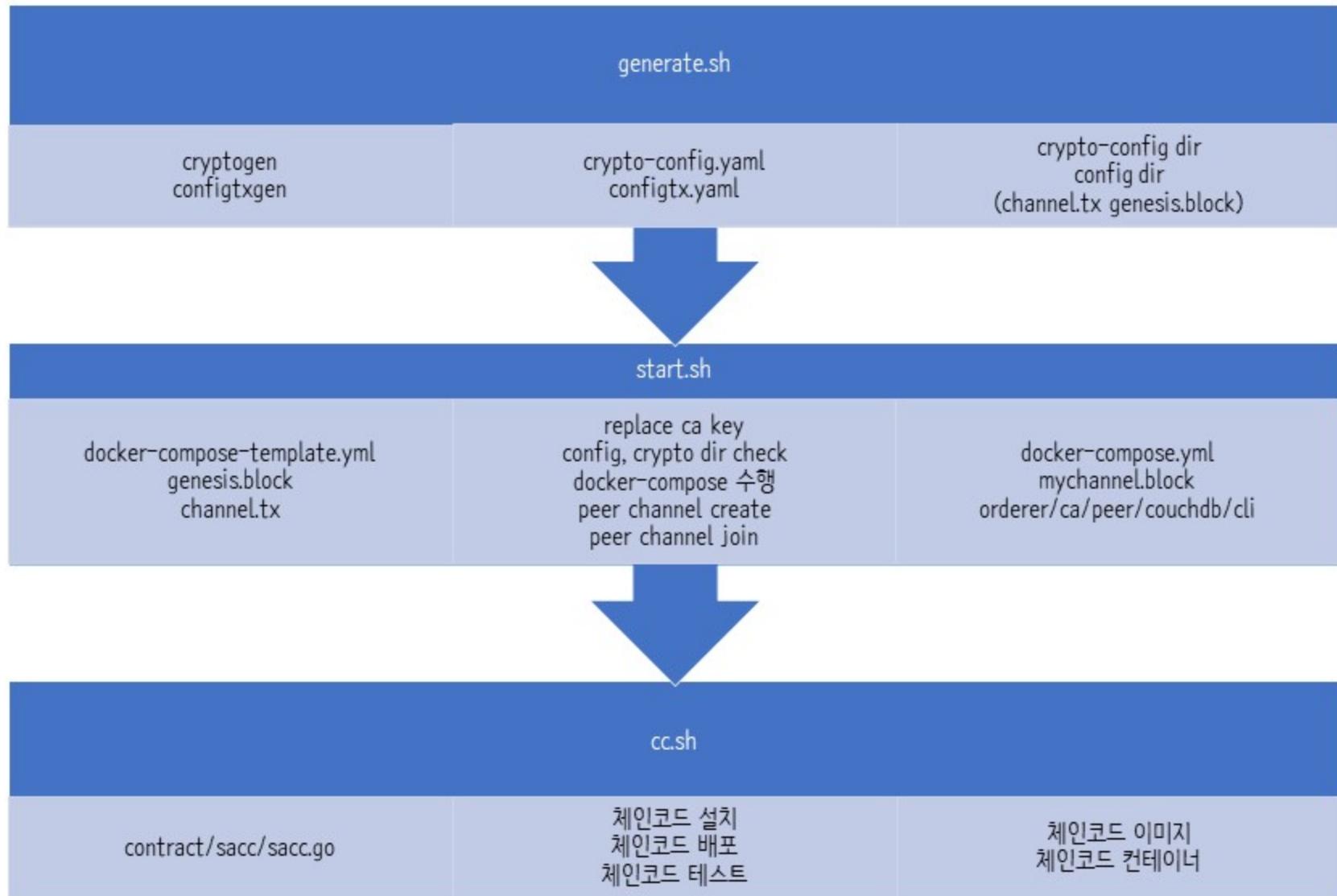
- 네트워크 실행
 - network/start.sh
- 체인코드 실행
 - network/cc.sh
- 실행 환경(명령)과 실행 플로우, 스크린샷 첨부

```
bstudent@bstudent-VirtualBox:~/dev/fabricbook/network$ ./start.sh
function replacePrivateKey() {
    echo "ca key file exchange"
    cp docker-compose-template.yml docker-compose.yml
    PRIV_KEY=$(ls crypto-config/peerOrganizations/org1.example.com/ca/ | grep _sk)
    sed -i "s/CA_PRIVATE_KEY/${PRIV_KEY}/g" docker-compose.yml
}

function checkPrereqs() {
    # check config dir
    if [ ! -d "crypto-config" ]; then
        echo "crypto-config dir missing"
        exit 1
    fi
    # check crypto-config dir
    if [ ! -d "config" ]; then
        echo "config dir missing"
        exit 1
    fi
}
checkPrereqs
replacePrivateKey
ca key file exchange
ls crypto-config/peerOrganizations/org1.example.com/ca/ | grep _sk
docker-compose -f docker-compose.yml down
Removing network net_basic
replacePrivateKey
ca key file exchange
ls crypto-config/peerOrganizations/org1.example.com/ca/ | grep _sk
docker-compose -f docker-compose.yml up -d ca.example.com orderer.example.com couchdb1 couchdb2 couchdb3
Creating network "net_basic" with the default driver
Creating orderer.example.com
Creating ca.example.com
Creating couchdb1
Creating couchdb2
Creating peer0.org2.example.com
Creating peer0.org1.example.com
Creating peer0.org1.example.com
```

```
./
  application
    |- createkey.html
    |- enrollAdmin.js
    |- index.html
    |- node_modules
    |- package-lock.json
    |- package.json
    |- querykey.html
    |- registerUser.js
    |- server.js
    |- wallet
  contract
    |- mymarbles
    |- newfabcar
    |- sacc
  fabric-front-end
    |- connection.json
    |- node_modules
    |- package-lock.json
    |- package.json
    |- public
    |- server.js
    |- views
    |- wallet
  javascript
    |- enrollAdmin.js
    |- invoke.js
    |- node_modules
    |- package-lock.json
    |- package.json
    |- query.js
    |- registerUser.js
    |- server.js
    |- views
    |- wallet
  network
    |- cc.sh
    |- cc_priv.sh
    |- config
    |- configtx.yaml
    |- connection.json
    |- connection.yaml
    |- crypto-config
    |- crypto-config.yaml
    |- docker-compose-template.yml
    |- docker-compose.yml
    |- generate.sh
    |- start.sh
    |- teardown.sh
```

쉘스크립트 구성 및 플로우





네트워크 실행

```
bstudent@bstudent-VirtualBox:~/dev/fabricbook/network
a6a24b080e79    hyperledger/fabric-peer      "peer node start"      5 seconds ago      Up 1 second
0.0.0.0:9051->7051/tcp, 0.0.0.0:9053->7053/tcp  peer0.org3.example.com
ed99bd4fe4aa    hyperledger/fabric-peer      "peer node start"      6 seconds ago      Up Less than
a second        0.0.0.0:7051->7051/tcp, 0.0.0.0:7053->7053/tcp  peer0.org1.example.com
89e53ee5145c    hyperledger/fabric-peer      "peer node start"      7 seconds ago      Up 2 seconds
0.0.0.0:8051->7051/tcp, 0.0.0.0:8053->7053/tcp  peer0.org2.example.com
ff6976a74bce    hyperledger/fabric-couchdb  "tini -- /docker-ent_"  11 seconds ago     Up 8 seconds
4369/tcp, 9100/tcp, 0.0.0.0:6984->5984/tcp       couchdb2
51f1954f632b    hyperledger/fabric-couchdb  "tini -- /docker-ent_"  11 seconds ago     Up 5 seconds
4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp       couchdb1
d9749e4ea2fe    hyperledger/fabric-ca       "sh -c 'fabric-ca-se_"
0.0.0.0:7054->7054/tcp, 0.0.0.0:7984->5984/tcp  ca.example.com
ca.example.com
b6d776bd8610    hyperledger/fabric-tools   "/bin/bash"
b6d776bd8610    hyperledger/fabric-tools   "/bin/bash"      cli
cli
11 seconds ago   Up 6 seconds
39f2486fdadd    hyperledger/fabric-orderer  "orderer"          11 seconds ago     Up 7 seconds
0.0.0.0:7050->7050/tcp, 0.0.0.0:7984->5984/tcp  orderer.example.com
889dc765e17f    hyperledger/fabric-couchdb  "tini -- /docker-ent_"  11 seconds ago     Up 4 seconds
4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp       couchdb3

# wait for Hyperledger Fabric to start
# incase of errors when running later commands, issue export FABRIC_START_TIMEOUT=<larger number>
export FABRIC_START_TIMEOUT=10
#echo ${FABRIC_START_TIMEOUT}
sleep ${FABRIC_START_TIMEOUT}

# Create the channel
docker exec cli peer channel create -o orderer.example.com:7050 -c mychannel -f /etc/hyperledger/configtx/
channel.tx
2019-09-18 11:46:13.874 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-09-18 11:46:13.924 UTC [cli.common] readBlock -> INFO 002 Received block: 0
# Join peer0.org1.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp" peer0.org1.example.com peer channel join -b /etc/hyperledger/configtx/mychannel.block
2019-09-18 11:46:14.279 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-09-18 11:46:14.476 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
sleep 5
# Join peer0.org2.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org2MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org2.example.com/msp" peer0.org2.example.com peer channel join -b /etc/hyperledger/configtx/mychannel.block
2019-09-18 11:46:19.864 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-09-18 11:46:20.030 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
sleep 5
# Join peer0.org3.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org3MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org3.example.com/msp" peer0.org3.example.com peer channel join -b /etc/hyperledger/configtx/mychannel.block
2019-09-18 11:46:25.457 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-09-18 11:46:25.688 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
sleep 5
bstudent@bstudent-VirtualBox:~/dev/fabricbook/network$
```



체인코드 설치 배포 테스트

- 실행조건
 - 네트워크 컨테이너들 실행(orderer/ca/peer/couchdb/cli)
 - contract/sacc/sacc.go 작성
 - cli contract 볼륨설정
- 명령
 - cc.sh

```
bstudent@bstudent-VirtualBox:~/dev/fabricbook/network$ ./cc.sh
2019-09-18 11:57:30.479 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc
2019-09-18 11:57:30.480 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc
2019-09-18 11:57:30.984 UTC [chaincodeCmd] install -> INFO 003 Installed remotely response:<status:200 payload:"OK" >
2019-09-18 11:57:31.370 UTC [chaincodeCmd] InitCmdFactory -> INFO 001 Retrieved channel (mychannel) orderer endpoint: orderer.example.com:7050
2019-09-18 11:57:31.370 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default escc
2019-09-18 11:57:31.370 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 Using default vscc
100
2019-09-18 11:57:50.899 UTC [chaincodeCmd] InitCmdFactory -> INFO 001 Retrieved channel (mychannel) orderer endpoint: orderer.example.com:7050
2019-09-18 11:57:50.908 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 002 Chaincode invoke successful. result: status:200 payload:"200"
200
-----END-----
```



인증서 발급

- 선행조건
 - ca.example.com 컨테이너 동작
 - network/connection.json 존재
 - enrollAdmin, registerUser 소스파일 내용 확인(connection.json 경로 확인)
- javascript
 - npm install
 - enrollAdmin.js
 - registerUser.js

```
bstudent@bstudent-VirtualBox:~/dev/fabricbook/javascript$ tree wallet/
wallet/
└── admin
    ├── 91538b803c486e7b6e7b8efe8f71290380914b0c210d4c8402dfe833180b707f-priv
    └── admin
        ├── ae6703c0dd4b1a3b662062fefc0360ac5d1ce94cd6a6e8df5e940c5d08fb701-priv
        └── ae6703c0dd4b1a3b662062fefc0360ac5d1ce94cd6a6e8df5e940c5d08fb701-pub
└── user1
    ├── 91538b803c486e7b6e7b8efe8f71290380914b0c210d4c8402dfe833180b707f-priv
    └── 91538b803c486e7b6e7b8efe8f71290380914b0c210d4c8402dfe833180b707f-pub
        └── user1
```



웹서버 프로토타입 작성

- 간단한 기능의 REST server endpoint 작성
 - / : index.html
 - /create : create.html
 - /query : query.html
 - /modify: modify.html
- /mate GET 개발자 조회
- /mate POST 개발자 생성
- /mate PUT 개발자 정보 업데이트 및 수정
- /mate DELETE 개발자 정보 삭제



서버수행 메뉴얼

- 선행조건
 - 네트워크 실행
 - 체인코드 배포
 - 인증서 소유
- 실행
 - node server.js

```
bstudent@bstudent-VirtualBox:~/dev/fabricbook/application$ node server.js
Running on http://0.0.0.0:8080
```



웹 클라이언트 프로토타입 작성

- / : index.html
- /create : create.html
- /query : query.html
- /modify: modify.html

- jquery, css(bootstrap), ejs등을 사용한 웹서버와 연동하는 간단한 기능 페이지 구현



프론트엔드 프로토타입 1

- 수행 및 기능 테스트
- 선행조건
 -
- 스크린샷 및 기능설명

프로토타입

이메일로 메이트를 조회합니다

조회하기

average	3.7500000024999998
project title	p1
project title	p2
project title	p4
project title	p3

추가에 성공했습니다!

새 메이트를 추가합니다

추가하기

프론트엔드 프로토타입 2



프로토타입

조회에 성공했습니다!

이메일로 메이트를 조회합니다

user1

조회하기

average	3.7500000024999998
project title	p1
project title	p2
project title	p4
project title	p3

새 메이트를 추가합니다

새로 추가할 메이트의 email을 입력하세요

추가하기



3일차 실습

- 체인코드 설계 예제(티메이트)

1. 체인코드 기능리스트 작성
2. 체인코드 데이터 정의
3. 체인코드 인터페이스 설계
4. 체인코드 전체 구조도 작성

- 체인코드 설계 자유주제



체인코드 설계서

1. 체인코드 기능리스트
2. 블록체인 데이터 정의
 - Application(Client)와의 데이터교환정의
 - World State, Private Data, Index데이터 정의
 - WS, PD와 체인코드와의 데이터교환 정의
3. 체인코드 인터페이스 설계서
 - 체인코드 함수목록, Invoke, Init 함수목록 및 기능정의
 - 각 함수별, 입력, 출력형식 정의
 - 로그정의
 - 에러정의
4. 설치배포 계획서
 - 체인코드 설치, 배포, 업그레이드 계획
 - 체인코드이름, 버전관리, Endorsing Policy
 - 테스트 계획서
5. 체인코드 전체 구조도
 - Application-Chaincode-STATE DB 간 구조



체인코드 기능리스트

- 체인코드 이름
 - teammate
- 체인코드 기능
 - 사용자등록
 - 최초 개발자가입 시에 개발자에 빈 데이터생성
 - 프로젝트 점수추가
 - 프로젝트 종료시점에 프로젝트리더에 의하여 개발자 프로젝트 점수추가
 - 사용자 정보 조회
 - 개발자 프로젝트 점수이력에 대한 정보 반환



블록체인 데이터 정의

- Application(Client)와의 데이터교환정의
- World State, Private Data, Index데이터 정의
- WS, PD와 체인코드와의 데이터교환 정의

체인코드 데이터



email address

Key



ARRAY
[project name :
score]

Project List



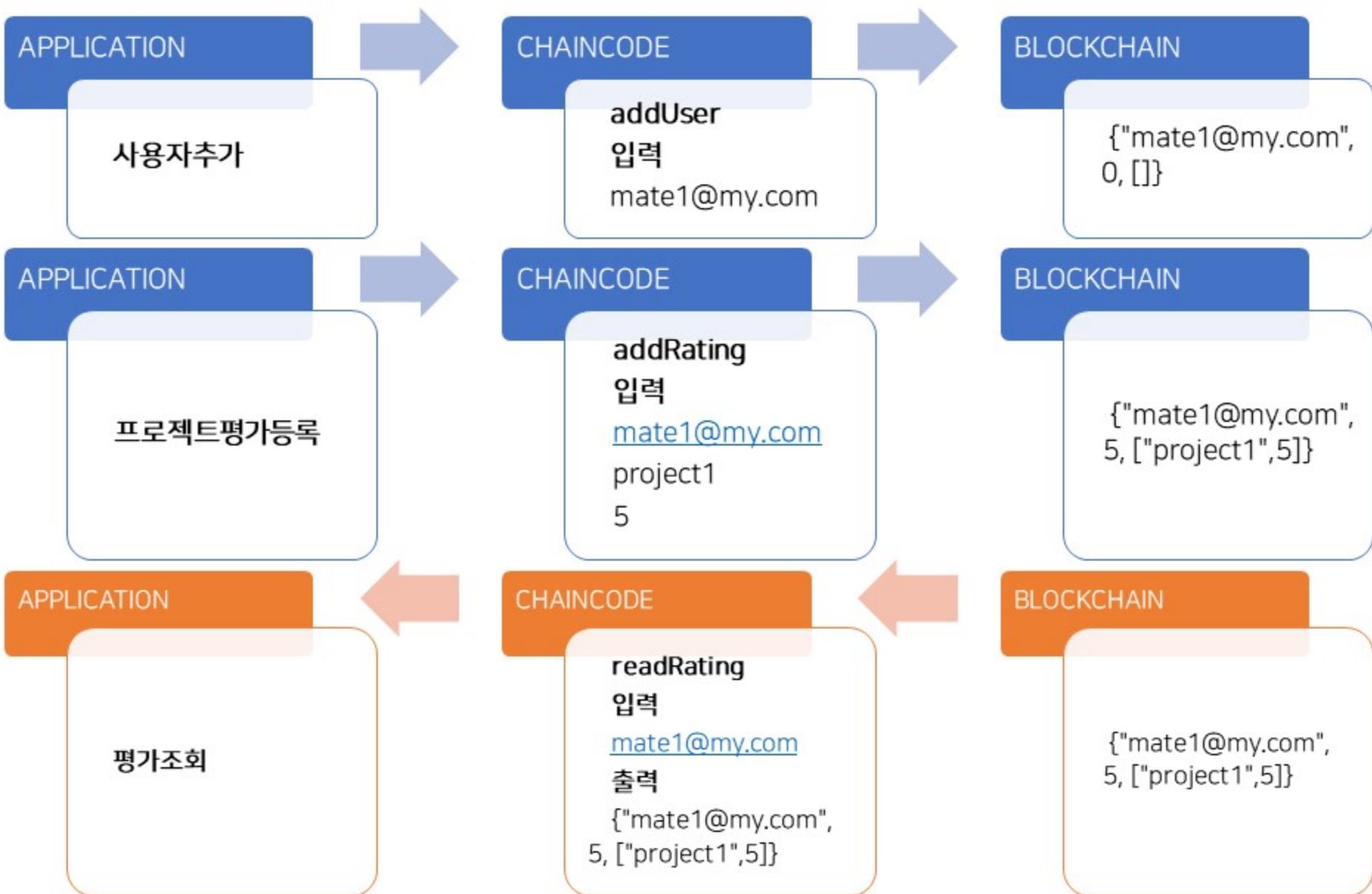
float64

Average
Rate Score



```
23 type UserRating struct{  
24     User string `json:"user"  
25     Average float64 `json:"average"  
26     Rates []Rate `json:"rates"  
27 }  
28 type Rate struct{  
29     ProjectTitle string `json:"projecttitle"  
30     Score float64 `json:"score"  
31 }
```

기능 플로우





체인코드 인터페이스 설계서

- 체인코드 함수목록, Invoke, Init 함수목록 및 기능정의
- 각 함수별, 입력, 출력형식 정의
- 기능별 앱클라이언트 반환값 정의
- 로그, 에러 정의
 - 배포시
 - Invoke, Query시
 - 기능별
 - 아규먼트 (형식과 인자 수 체크)
 - 데이터 접속 시 (함수에러, 데이터접근-값없음)



로그, 예) 예)

- Init
 - instantiate argument error
- Invoke
 - not supported error
- addUser
 - argument error
 - already existed user
 - put state error
- addRating
 - argument error
 - non existed user
 - already scored project
 - get state error
 - put state error
- readRating
 - argument error
 - non existed user
 - get state error

error code	value
ERR_CODE=10	instantiate error
20	not supported function
31	argument error
32	already existed user
33	non existed user
34	already scored project
43	put state error
44	get state error

로그 기록 방식

fmt.println을 사용

로그 확인 방식

container에 logs를 확인
예) docker logs dev-...



기능별 앱클라이언트 반환값 정의

모든 경우 JSON형식으로 반환

- Init
 - {result:sucess} or {result:fail, error_code:10, error_value:"instantiate error"}
- Invoke
 - invoke시:
 - {result:sucess} or {result:fail, error_code:20, error_value:"not supported function"}
 - query시
 - {result:sucess,value:{user:cchoi,avg=5,[{projectname:p1,score:5}]}} or {result:fail, error_code:20, error_value:"non existed user"}
- addUser
 - {result:sucess}
 - {result:fail, error_code:33, error_value:"non existed user"}
- addRating
 - {result:sucess}
 - {result:fail, error_code:34, error_value:"already scored project"}
- readRating
 - {result:sucess}
 - {result:fail, error_code:33, error_value:"non existed user"}



teamate 체인코드

- 이름
 - teammate
- 기능
 - 사용자등록
 - 프로젝트 점수추가
 - 사용자 정보 조회

```
func (s *SmartContract) Invoke(APIstub shim.ChaincodeStubInterface) response {  
    function, args := APIstub.GetFunctionAndParameters()  
  
    if function == "addUser" {  
        return s.addUser(APIstub, args)  
    } else if function == "addRating" {  
        return s.addRating(APIstub, args)  
    } else if function == "readRating" {  
        return s.readRating(APIstub, args)  
    }  
  
    return shim.Error("Invalid Smart Contract function name.")  
}
```

addUser

User정보 생성하기
key: email - args[0]
averate: 0
project list: []

addRating

User의 프로젝트점수 추가하기
key: email - args[0]
project name - args[1]
project rate - args[2]

readRating

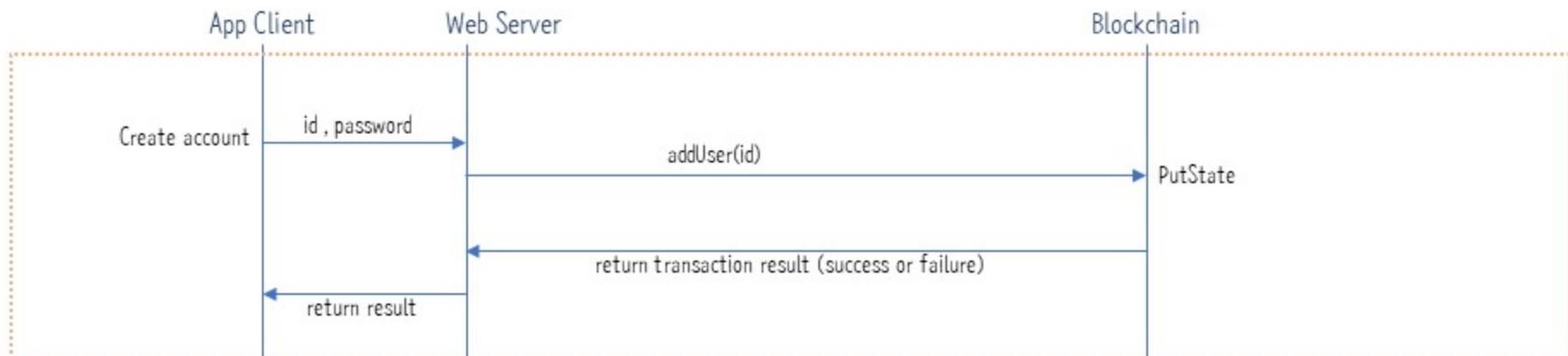
User의 프로젝트점수 읽어오기
key: email -args[0]

인터페이스 설계

- Init
 - arguments
 - output
- Invoke
 - arguments – function name, args...
 - output- invoke시:JSON query시:JSON
- addUser
 - arguments-user name
 - output-JSON(성공여부 및 에러코드)
- addRating
 - arguments-user name, project name, score
 - output- JSON(성공여부 및 에러코드)
- readRating
 - arguments-user name
 - output-JSON(성공여부와 결과 및 에러코드)

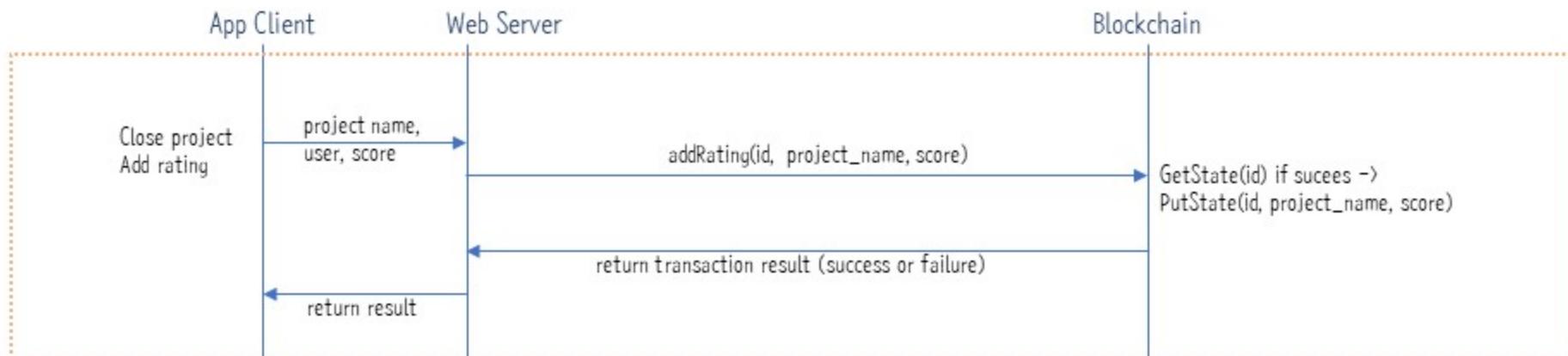
체인코드 플로우챠트

- addUser



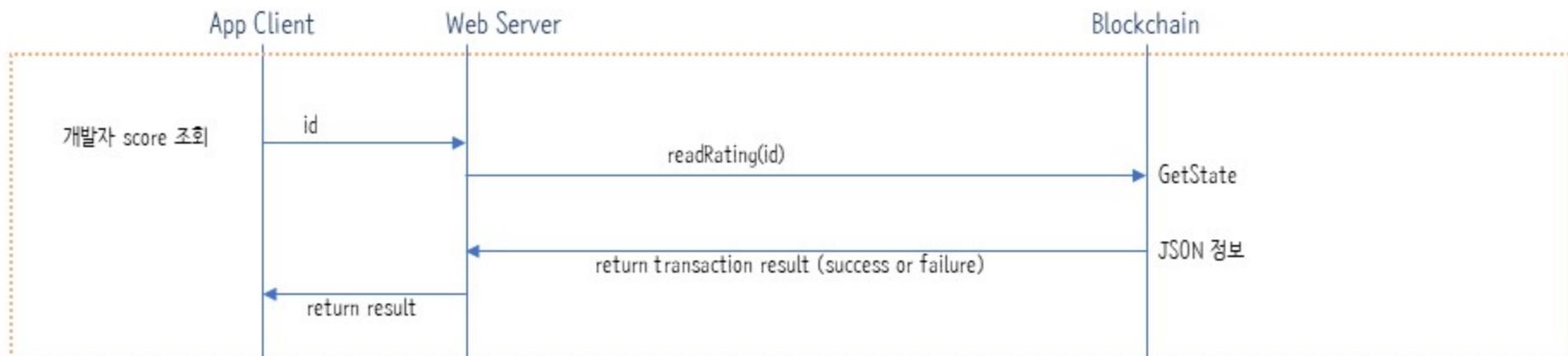
체인코드 플로우차트

- addRating



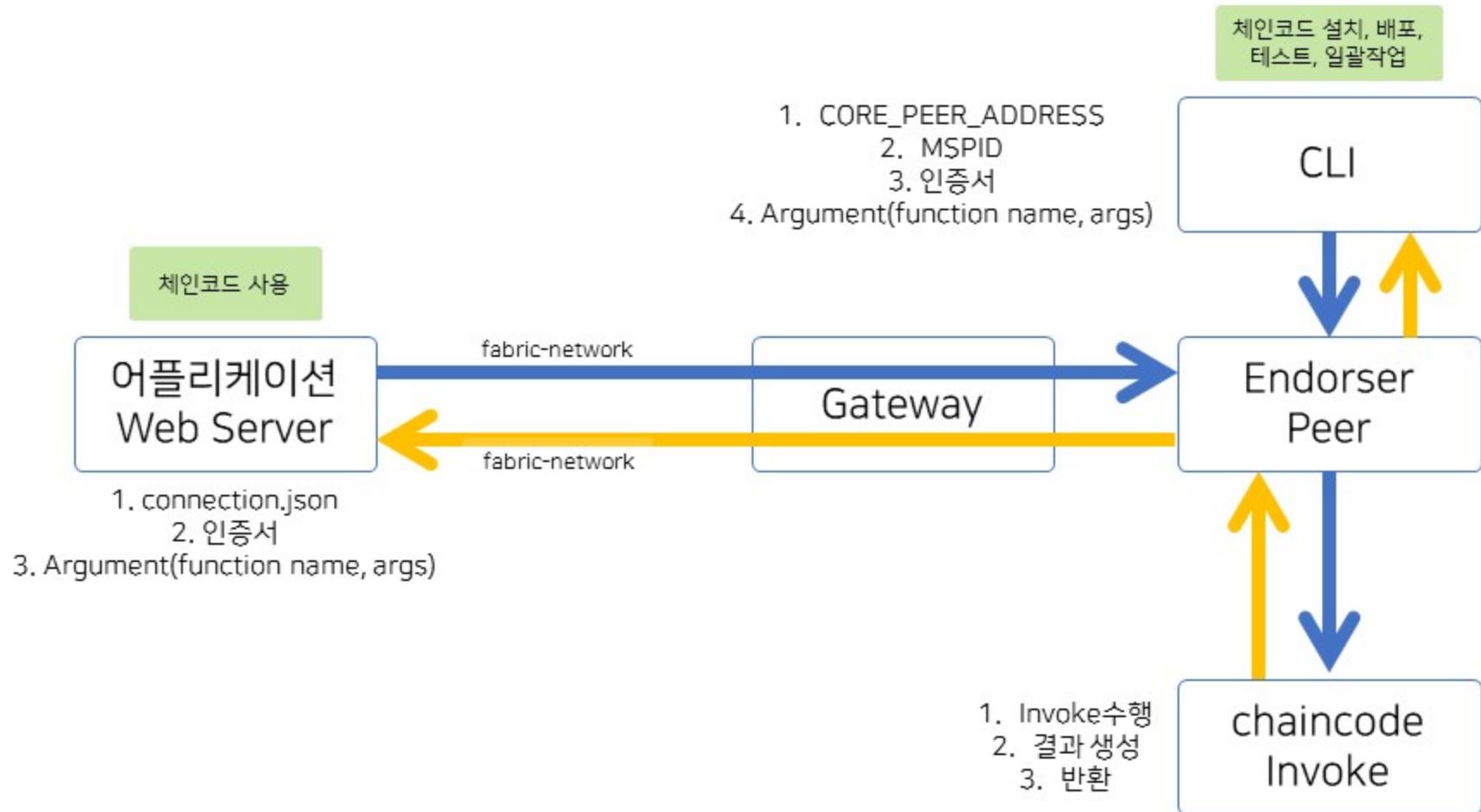
체인코드 플로우챠트

- readRating



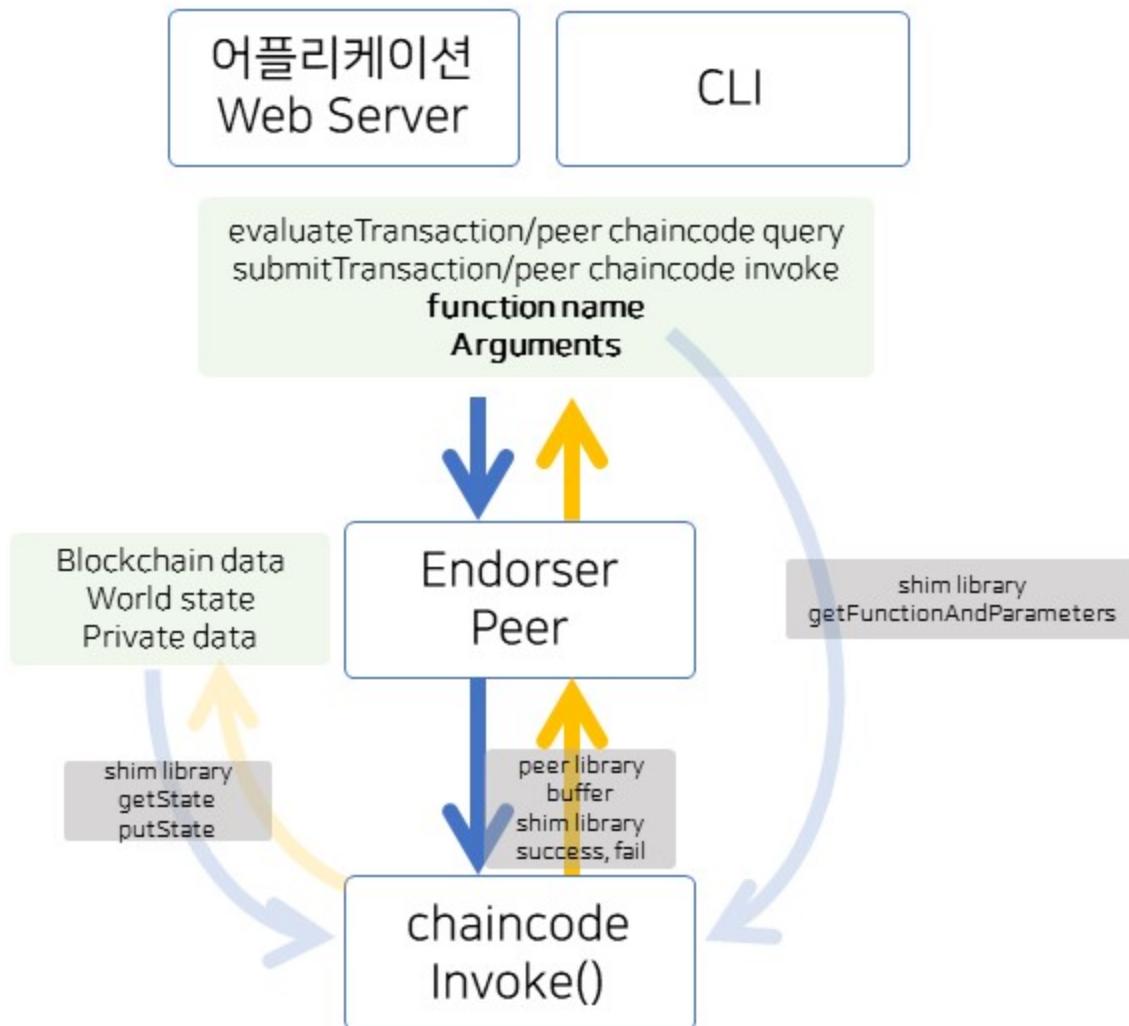
체인코드 전체 구조도

- Application-Chaincode-STATE DB 간 구조



체인코드 관계 구조도

- GO SDK
 - shim
 - contract
 - peer
- Init
- Invoke
- Lodger
 - World State
 - Private Data
 - Transient DB





설치배포 계획서

- 체인코드 설치, 배포, 업그레이드 계획
 - 체인코드 옵션
 - 이름
 - 버전
 - 채널
 - 정책
 - 체인코드 설치 배포 쉘스크립트
- 테스트 계획서



테스트계획서

- 체인코드 설치 배포

선행조건 및 실행환경	예상결과
<ol style="list-style-type: none">네트워크 실행중 orderer, peer, couchdb, cli채널생성 mychannel체인코드 설치, 배포 체인코드 이름: teammate, 버전:1.0체인코드 소스 볼륨연결	
<ol style="list-style-type: none">peer chaincode install -n teammate -v 1.0 -p github.com/teamatepeer chaincode instantiate -n teammate -v 1.0 -C mychannel -c '{"Args":[]}' -P '(OR ("Org1MSP.member", "Org2MSP.member", "Org3MSP.member"))'	블록체인 상태, 결과 스크린샷



테스트계획서

- addUser

선행조건 및 실행환경	예상결과
<ol style="list-style-type: none">네트워크 실행중 orderer, peer, couchdb, cli채널생성 mychannel체인코드 설치, 배포 체인코드 이름: teammate, 버전:1.0	
1. peer chaincode invoke -n teammate -C mychannel -c '{"Args":["addUser","choi"]}'	블록체인 상태, 결과 스크린샷



테스트계획서

- addRating

선행조건 및 실행환경	예상결과
<ol style="list-style-type: none">네트워크 실행중 orderer, peer, couchdb, cli채널생성 mychannel체인코드 설치, 배포 체인코드 이름: teammate, 버전:1.0	
<ol style="list-style-type: none">peer chaincode invoke -n teammate -C mychannel -c '{"Args":["addRating","choi","p1","5"]}'	블록체인 상태, 결과 스크린샷



테스트계획서

- readRating

선행조건 및 실행환경	예상결과
<ol style="list-style-type: none">네트워크 실행중 orderer, peer, couchdb, cli채널생성 mychannel체인코드 설치, 배포 체인코드 이름: teammate, 버전:1.0	
1. peer chaincode query -n teammate -C mychannel -c '{"Args":["readRating","choi"]}'	블록체인 상태, 결과 스크린샷



JSON SDK

- <https://golang.org/pkg/encoding/json/>

```
type UserRating struct{
    User string `json:"user"`
    Average float64 `json:"average"`
    Rates []Rate `json:"rates"`
}

type Rate struct{
    ProjectTitle string `json:"projecttitle"`
    Score float64 `json:"score"`
}
```

The screenshot shows a browser window displaying the official Go documentation for the `json` package. The URL in the address bar is `https://golang.org/pkg/encoding/json/`. The page features a header with the Go logo and the title "Package json". Below the title are links for "Overview", "Index", and "Examples". The "Overview" section contains a brief description of the package's purpose and a link to "JSON and Go". The "Index" section lists several functions, with some highlighted in red, indicating they are part of the package's public API. These highlighted functions include `Compact`, `HTMLEscape`, `Indent`, `Marshal`, `MarshalIndent`, `Unmarshal`, and `Valid`.

json - The Go Programming Language

golang.org/pkg/encoding/json/

=GO

Package json

import "encoding/json"

Overview Index Examples

Overview

Package json implements encoding and decoding of JSON as defined in RFC 7159. The mapping between JSON and Go values is described in the documentation for the Marshal and Unmarshal functions.

See "JSON and Go" for an introduction to this package: https://golang.org/doc/articles/json_and_go.html

- Example (CustomMarshalJSON)
- Example (TextMarshalJSON)

Index

- func Compact(dst *bytes.Buffer, src []byte) error
- func HTMLEscape(dst *bytes.Buffer, src []byte)
- func Indent(dst *bytes.Buffer, src []byte, prefix, indent string) error
- func Marshal(v interface{}) ([]byte, error)
- func MarshalIndent(v interface{}, prefix, indent string) ([]byte, error)
- func Unmarshal(data []byte, v interface{}) error
- func Valid(data []byte) bool
- type Decoder
- func NewDecoder(r io.Reader) *Decoder
- func (dec *Decoder) Buffered() io.Reader
- func (dec *Decoder) Decode(v interface{}) error
- func (dec *Decoder) DisallowUnknownFields()
- func (dec *Decoder) More() bool
- func (dec *Decoder) Token() (Token, error)
- func (dec *Decoder) UseNumber()



```
// getState User
userAsBytes, err := APIstub.GetState(args[0])
if err != nil{
    jsonResp := "\"Error\":\"Failed to get state for "+ args[0]+"\"
    return shim.Error(jsonResp)
} else if userAsBytes == nil{ // no State! error
    jsonResp := "\"Error\":\"User does not exist: "+ args[0]+"\"
    return shim.Error(jsonResp)
}
// state ok
user := UserRating{}
err = json.Unmarshal(userAsBytes, &user)
if err != nil {
    return shim.Error(err.Error())
}
// create rate structure
newRate, _ := strconv.ParseFloat(args[2],64)
var Rate = Rate{ProjectTitle: args[1], Score: newRate}

rateCount := float64(len(user.Rates))

user.Rates=append(user.Rates,Rate)

user.Average = (rateCount*user.Average+newRate)/(rateCount+1)
// update to User World state
userAsBytes, err = json.Marshal(user);
```



SHIM SDK

- <https://godoc.org/github.com/hyperledger/fabric/core/chaincode/shim>

```
type Chaincode

type Chaincode interface {
    // Init is called during Instantiate transaction after the chaincode container
    // has been established for the first time, allowing the chaincode to
    // initialize its internal data
    Init(stub ChaincodeStubInterface) pb.Response

    // Invoke is called to update or query the ledger in a proposal transaction.
    // Updated state variables are not committed to the ledger until the
    // transaction is committed.
    Invoke(stub ChaincodeStubInterface) pb.Response
}
```

A screenshot of a web browser displaying the GoDoc API documentation for the 'shim' package. The URL in the address bar is 'godoc.org/github.com/hyperledger/fabric/core/chaincode/shim'. The page title is 'shim - GoDoc'. The main content area shows the package definition 'package shim' and its import statement 'import "github.com/hyperledger/fabric/core/chaincode/shim"'. Below this, there are two identical paragraphs of text: 'Package shim provides APIs for the chaincode to access its state variables, transaction context and call other chaincodes.' A red box highlights the word 'Chaincode' in the type definition. To the right of the code block, there is a sidebar with links for 'Index', 'Files', and 'Directories'. The bottom half of the page contains a list of methods for the 'Chaincode' type, including 'NewLogger', 'Critical', 'Criticalf', 'Debug', 'Debugf', 'Error', 'Errorf', 'Info', 'Infof', 'IsEnabledForLogLevel', 'Notice', 'Noticef', and 'SetLevel'.

chaincodeStub

- GetState
- PutState
- DelState
- GetStringArgs
- GetFunctionAndParameters

shim - GoDoc

godoc.org/github.com/hyperledger/fabric/core/chaincode/shim

```

type ChaincodeStub
  o func (stub *ChaincodeStub) CreateCompositeKey(objectType string, attributes []string) (string, error)
  o func (stub *ChaincodeStub) DelPrivateData(collection string, key string) error
  o func (stub *ChaincodeStub) DelState(key string) error
  o func (stub *ChaincodeStub) GetArgs() []byte
  o func (stub *ChaincodeStub) GetArgsSlice() ([]byte, error)
  o func (stub *ChaincodeStub) GetBinding() ([]byte, error)
  o func (stub *ChaincodeStub) GetChannelID() string
  o func (stub *ChaincodeStub) GetCreator() ([]byte, error)
  o func (stub *ChaincodeStub) GetDecorations() map[string][]byte
  o func (stub *ChaincodeStub) GetFunctionAndParameters() (function string, params []string)
  o func (stub *ChaincodeStub) GetHistoryForKey(key string) (HistoryQueryIteratorInterface, error)
  o func (stub *ChaincodeStub) GetPrivateData(collection string, key string) ([]byte, error)
  o func (stub *ChaincodeStub) GetPrivateDataByPartialCompositeKey(collection, objectType string, attributes []string) (StateQueryIteratorInterface, error)
  o func (stub *ChaincodeStub) GetPrivateDataByRange(collection, startKey, endKey string) (StateQueryIteratorInterface, error)
  o func (stub *ChaincodeStub) GetPrivateDataHash(collection string, key string) ([]byte, error)
  o func (stub *ChaincodeStub) GetPrivateDataQueryResult(collection, query string)
  o func (stub *ChaincodeStub) GetPrivateDataValidationParameter(collection, key string) ([]byte, error)
  o func (stub *ChaincodeStub) GetQueryResult(query string) (StateQueryIteratorInterface, error)
  o func (stub *ChaincodeStub) GetQueryResultWithPagination(query string, pageSize int32, bookmark string) (StateQueryIteratorInterface, *pb.QueryResponseMetadata, error)
  o func (stub *ChaincodeStub) GetSignedProposal() (*pb.SignedProposal, error)
  o func (stub *ChaincodeStub) GetState(key string) ([]byte, error)
  o func (stub *ChaincodeStub) GetStateByPartialCompositeKey(objectType string, attributes []string) (StateQueryIteratorInterface, error)
  o func (stub *ChaincodeStub) GetStateByPartialCompositeKeyWithPagination(objectType string, keys []string, pageSize int32, bookmark string) (StateQueryIteratorInterface, *pb.QueryResponseMetadata, error)
  o func (stub *ChaincodeStub) GetStateByRange(startKey, endKey string) (StateQueryIteratorInterface, error)
  o func (stub *ChaincodeStub) GetStateByRangeWithPagination(startKey, endKey string, pageSize int32, bookmark string) (StateQueryIteratorInterface, *pb.QueryResponseMetadata, error)
  o func (stub *ChaincodeStub) GetStateValidationParameter(key string) ([]byte, error)
  o func (stub *ChaincodeStub) GetStringArgs() []string
  o func (stub *ChaincodeStub) GetTransient() (map[string][]byte, error)
  o func (stub *ChaincodeStub) GetTxID() string
  o func (stub *ChaincodeStub) GetTxTimestamp() (*timestamp.Timestamp, error)
  o func (stub *ChaincodeStub) InvokeChaincode(chaincodeName string, args []byte, channel string) pb.Response
  o func (stub *ChaincodeStub) PutPrivateData(collection string, key string, value []byte) error
  o func (stub *ChaincodeStub) PutState(key string, value []byte) error
  o func (stub *ChaincodeStub) SetEvent(name string, payload []byte) error
  o func (stub *ChaincodeStub) SetPrivateDataValidationParameter(collection, key string, ep []byte) error
  o func (stub *ChaincodeStub) SetStateValidationParameter(key string, ep []byte) error
  o func (stub *ChaincodeStub) SplitCompositeKey(compositeKey string) (string, []string, error)

```

PEER SDK



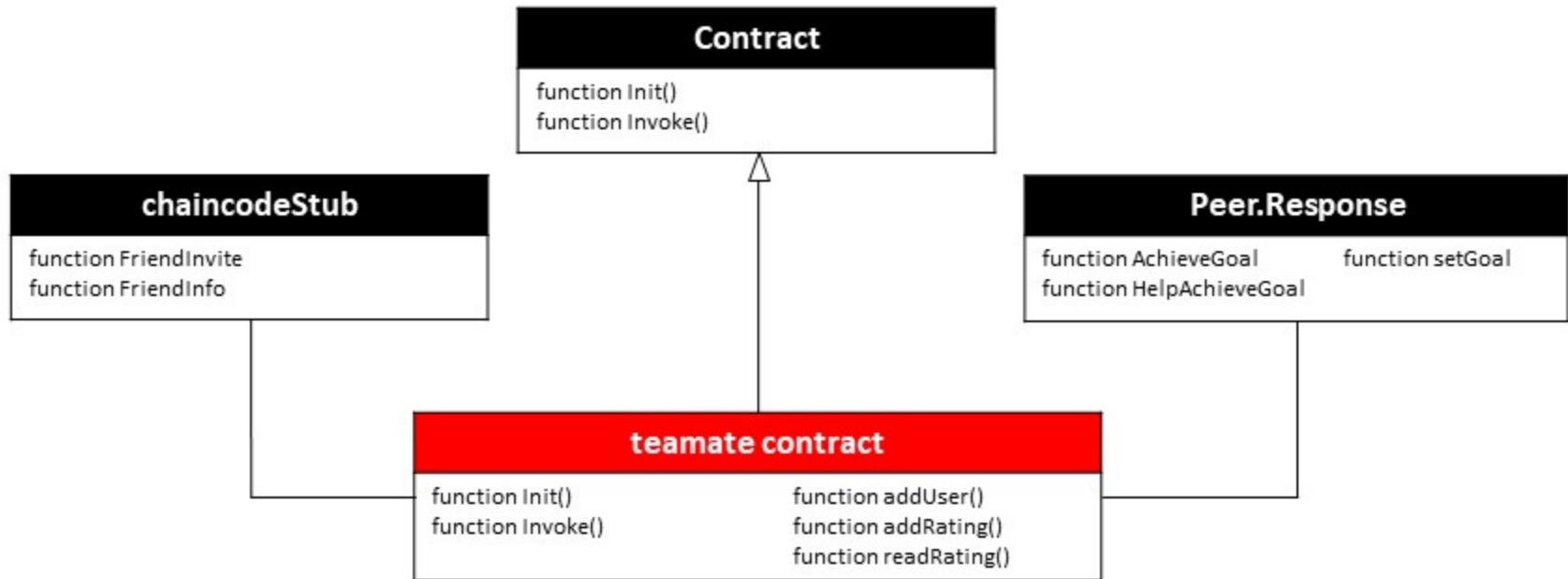
- <https://godoc.org/github.com/hyperledger/fabric/protos/peer#Response>

A screenshot of the godoc.org interface for the Hyperledger Fabric peer package. The URL in the address bar is https://godoc.org/github.com/hyperledger/fabric/protos/peer#Response. The page displays the definition of the Response type. The code is highlighted in a yellow box:

```
type Response struct {
    // A status code that should follow the HTTP status codes.
    Status int32 `protobuf:"varint,1,opt,name=status,proto3" json:"status,omitempty"`
    // A message associated with the response code.
    Message string `protobuf:"bytes,2,opt,name=message,proto3" json:"message,omitempty"`
    // A payload that can be used to include metadata with this response.
    Payload     []byte  `protobuf:"bytes,3,opt,name=payload,proto3" json:"payload,omitempty"`
    XXX_NoUnkeyedLiteral struct{} `json:"-"`
    XXX_unrecognized   []byte  `json:"-"`
    XXX_sizecache     int32   `json:"-"`
}
```

The godoc interface also shows a list of methods for the Response type, such as GetMessage(), GetPayload(), and GetStatus(). The 'type Response' section is highlighted in orange.

체인코드 구조





4일차 실습

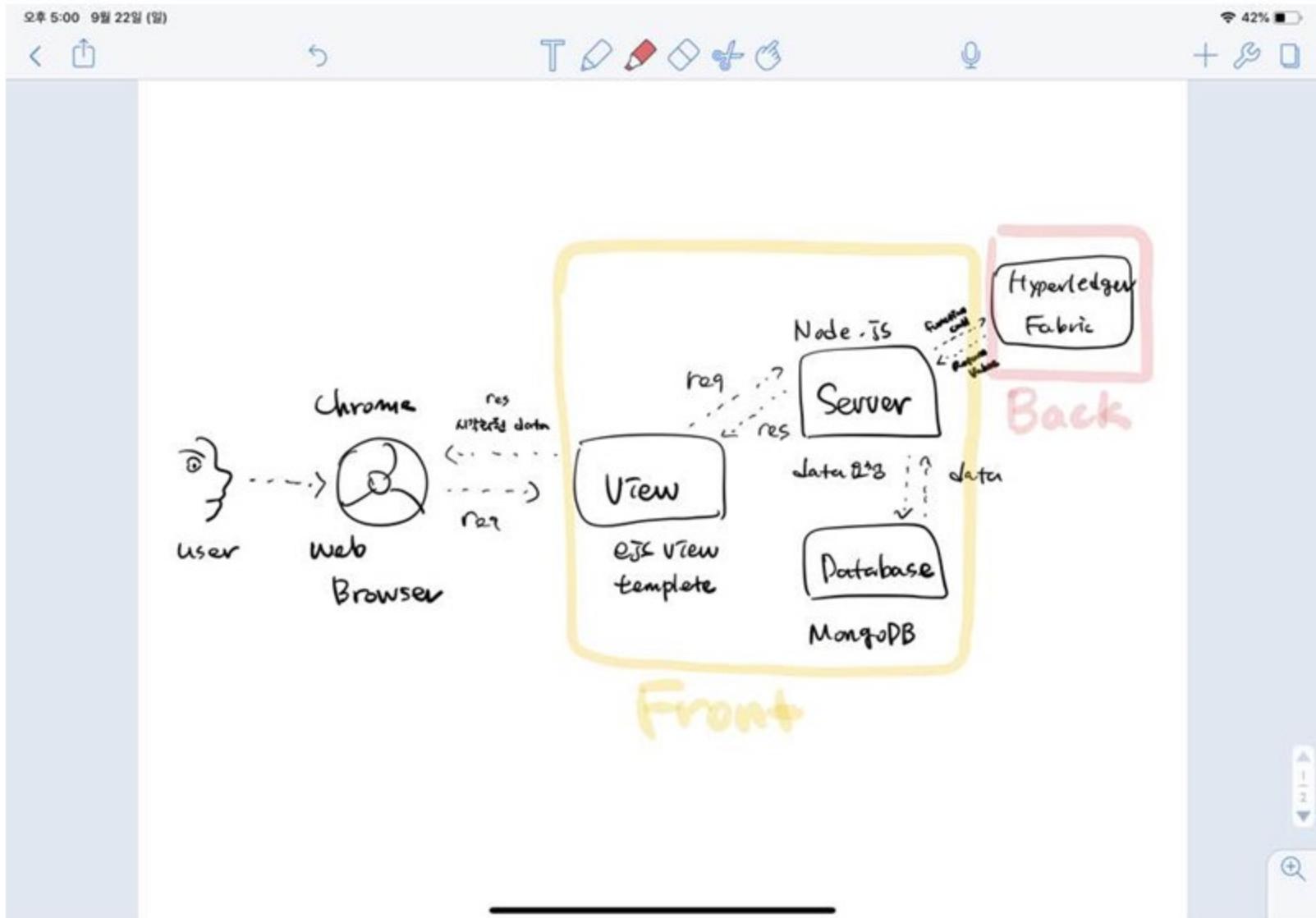
- 프론트앤드 설계 예제(티메이트)
 1. 프론트앤드 기능리스트작성
 2. 웹서버 EndPoint설계
 3. UI 설계
 4. 연동플로우챠트 작성
- 프론트앤드 설계 자유주제



기능리스트

1. 개발자 가입
✓이메일
2. 프로젝트등록
✓저장 항목
3. 개발자 조회
✓블록체인데이터 / 데이터베이스데이터
4. 프로젝트목록 보기
✓필터링
5. 프로젝트 참여 회원 평가
✓블록체인데이터

프론트엔드 구조도





사용툴 의사결정

서버	데이터베이스	백엔드 데이터베이스	뷰
Nodejs express	mongoDB	Hyperledger Fabric	Ejs view engine
프론트엔드와 백엔드를 같은 javascript로 관리할 수 있음	Sql어가 아닌 js문법을 사용하는 DB여서 러닝커브 낮음. noSQL이므로 상대적으로 자유로운 데이터 관리	일반 DB보다 Secure한 데이터베이스 신뢰성, 무결성	클라이언트의 요청에 따라 변화할 수 있는 뷰 데이터를 처리 가능함. Html문법과 거의 유사하여 러닝커브 낮음

디렉토리 구조



```
└─ application
    ├─ .vscode
    ├─ model
    ├─ node_modules
    ├─ public
    ├─ router
    ├─ views
    ├─ wallet
    └─ package-lock.json
    └─ server.js
```

model: 데이터베이스 스키마 파일이 있는 폴더, project.js와 user.js

node_modules: 로컬모드로 설치된 모듈들이 있는 곳

public: 정적 파일들이 있는 폴더. 하위폴더로 /images와 /stylesheets

router: 라우팅 파일들이 있는 폴더

views: 뷰 파일이 있는 폴더. Ejs 뷰 엔진을 사용하므로 ejs파일들이 있다.

하위폴더로 partials/가 있는데, 공용으로 사용하는 헤더나 푸터, 상단바등이 있다

wallet: 인증서 저장 디렉토리

Package-lock.json: package.json이나 node_moduels트리에 변화가 생겼을 때
자동으로 생성되는 파일

server.js: Express 서버를 구동시키는 역할



DB 설계

- Model/project.js

```
2 const projectSchema = new mongoose.Schema({  
3   title: {type: String, required: true},  
4   detail: {type: String, required: true},  
5   content: {type: String, required: true},  
6   createdAt: {type: Date, default: Date.now}  
7 });
```

- Title: 프로젝트 제목
- Detail: 프로젝트 한 줄 설명
- Content: 프로젝트 세부 설명 (textarea)
- createdAt: 프로젝트 생성날짜



DB 설계

- Model/user.js

```
4  const userSchema = new mongoose.Schema({  
5      email: {type: String, unique: true, required: true},  
6      name: {type: String},  
7      job: {type: String},  
8      password: {type: String, required: true}  
9  });
```

- Email: 아이디로 사용할 유저 이메일
- Name: 유저 이름
- Job: 유저 직무 (개발자, 디자이너, PM, 등등)
- Password: 비밀번호

DB 설계



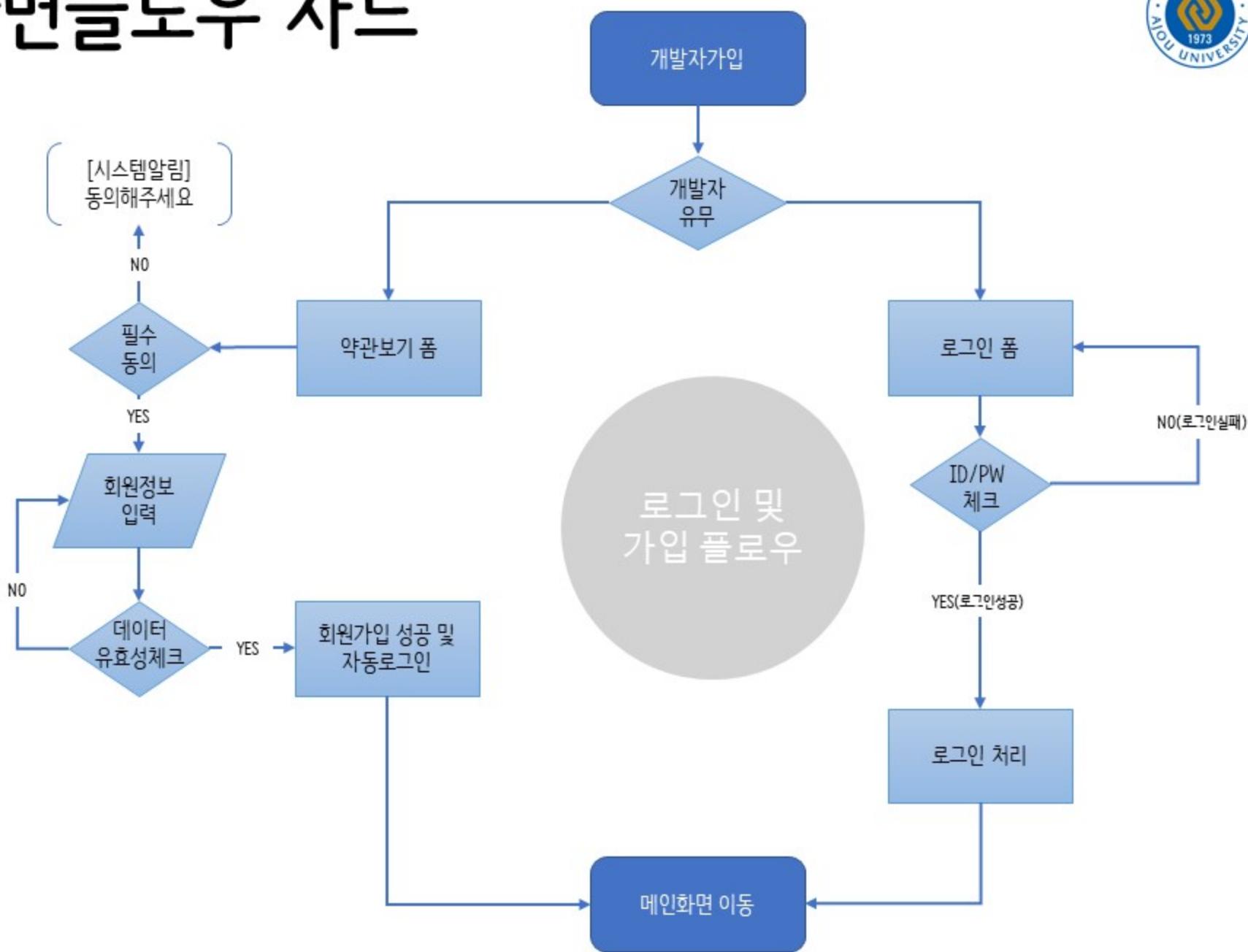
- project 테이블 예시

title	detail	content	createdAt
티메이트	블록체인 프로젝트	블록체인을 이용한 팀 빌딩 플랫폼입니다	2019. 09. 24
면접 스터디	삼성 XX직무 면접 스터디	상반기 XX직무 같이 준비해요	2019. 09. 24

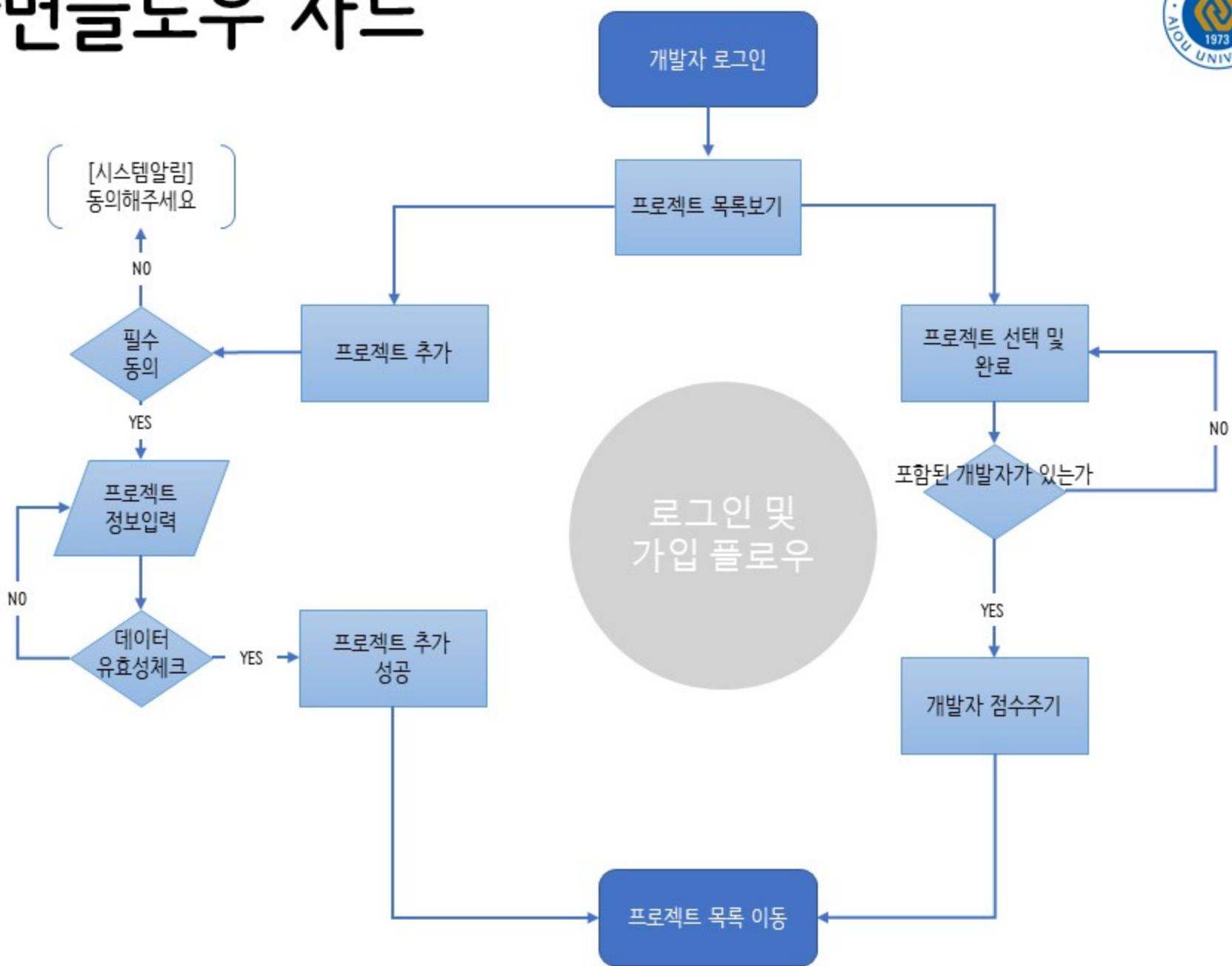
- User 테이블 예시

Email	Name	Job	password
kimgahyun1@ajou.ac.kr	김가현	개발자	kim1234
hong@gmail.com	홍길동	디자이너	hong0000

화면플로우 차트



화면플로우 차트





REST End-Points

- home

Method	URL	Description	Verb
GET	/index	메인 페이지	
GET	/project	모든 프로젝트	
GET	/mate	모든 메이트	Index
POST	/join	회원가입 (새 메이트 추가)	Create
POST	/login	로그인	



REST End-Points

- project

Method	URL	Description	Verb
GET	/project	모든 프로젝트	Index
GET	/project/:id	해당 id의 프로젝트 보기	Retrieve
POST	/project	새 프로젝트 추가	Create
PUT	/project/:id	해당 id의 프로젝트 수정	Update
DELETE	/project/:id	해당 id의 프로젝트 삭제	Destroy

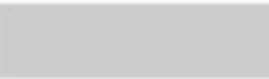
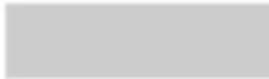
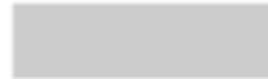


REST End-Points

- mate

Method	URL	Description	Verb
GET	/mate	모든 메이트	Index
GET	/mate/:id	해당 id의 메이트 보기	Retrieve
PUT	/mate/:id	메이트 프로필 수정	Update
DELETE	/mate/:id	메이트 삭제(회원탈퇴)	Destroy

제목	메인페이지 상단	URL	/	Page	1
	이미지			설명	
				전체 DB에서 search(프로젝트, 메이트) 전체 DB에서 search(프로젝트, 메이트)	
	 <p>COWORK NOW!</p> <p>프로젝트를 함께할 메이트를 찾아보세요.</p> <p>더 알아보기</p> <p>클릭하면 티매이트 소개페이지(/about)로 이동</p>			<ul style="list-style-type: none"> - 더 알아보기 버튼 : /about로 이동 - 카드 클릭 : 프로젝트 상세 페이지로 이동 (/project/:id) - Show more 버튼 : 전체 프로젝트 보기 페이지로 이동(/project) 	
	 <p>Project 카드를 클릭하면 해당 프로젝트의 상세페이지(retrieve, project/:id)로 이동</p> <p>클록재인 프로젝트 멤버 모집 by 김기현 제작자 클록재인 기관 및 개인 출판물을 판권자로 지정하는 개방형을 찾습니다. 보기</p> <p>클록재인 프로젝트 멤버 모집 by 김기현 제작자 클록재인 기관 및 개인 출판물을 판권자로 지정하는 개방형을 찾습니다. 보기</p> <p>클록재인 프로젝트 멤버 모집 by 김기현 제작자 클록재인 기관 및 개인 출판물을 판권자로 지정하는 개방형을 찾습니다. 보기</p>				
	 <p>클록재인 프로젝트 멤버 모집 by 김기현 제작자 클록재인 기관 및 개인 출판물을 판권자로 지정하는 개방형을 찾습니다. 보기</p> <p>클록재인 프로젝트 멤버 모집 by 김기현 제작자 클록재인 기관 및 개인 출판물을 판권자로 지정하는 개방형을 찾습니다. 보기</p> <p>클록재인 프로젝트 멤버 모집 by 김기현 제작자 클록재인 기관 및 개인 출판물을 판권자로 지정하는 개방형을 찾습니다. 보기</p>			<p>전체 프로젝트 페이지(/project)로 이동</p> <p>Show more</p>	

제목	메인페이지 하단	URL	/	Page	2
	이미지			설명	
Mate 카드를 클릭하면 해당 메이트의 상세페이지(retrieve, mate/:id)로 이동	 김가현 예전 친구 <small>Node.js Express, Ruby on Rails, Ionic 4.0, Angular.js</small>  김가현 예전 친구 <small>Node.js Express, Ruby on Rails, Ionic 4.0, Angular.js</small>  김가현 예전 친구 <small>Node.js Express, Ruby on Rails, Ionic 4.0, Angular.js</small>  김가현 예전 친구 <small>Node.js Express, Ruby on Rails, Ionic 4.0, Angular.js</small>  김가현 예전 친구 <small>Node.js Express, Ruby on Rails, Ionic 4.0, Angular.js</small>  김가현 예전 친구 <small>Node.js Express, Ruby on Rails, Ionic 4.0, Angular.js</small>			<ul style="list-style-type: none"> - 카드 클릭 : 메이트 상세 페이지로 이동(/mate/:id) - Show more 버튼 : 전체 메이트 보기 페이지로 이동(/mate) 	
	<p style="text-align: center;">전체 메이트 페이지(/mate)로 이동</p> <p style="text-align: center;">Show more</p>			<p style="text-align: center;">Make Something Together</p>	

제목	로그인	URL	/login	Page	3
	이미지			설명	
Project				<ul style="list-style-type: none"> - 로그인 Form - 로그인 성공 시 세션 생성 - 메인 페이지로 돌아감 - 회원가입 시 mate가 생성됨 	

제목

프로젝트 전체 페이지

URL

/project

Page

4

이미지

설명



- 전체 프로젝트 목록 페이지
- 원하는 프로젝트를 필터링하는 필터링&검색 기능
- 새 프로젝트 생성 버튼: project/new로 이동
- 카드 클릭 : 프로젝트 상세페이지로 이동

카드 클릭 시 프로젝트 상세페이지로 이동(/project/1)



제목	새 프로젝트 생성	URL	/project/new	Page	5
이미지				설명	
	 <p>새로운 프로젝트 만들기</p> <p>프로젝트 이름 제작자 이름</p> <p>프로젝트 만들 소개 100자 미만 이 프로젝트의 목적이를 사용자 정보나 경영</p> <p>프로젝트 대체사진 작성일자</p> <p>프로젝트 상세 설명 프로젝트 상세 설명을 작성해주세요.</p> <p>프로젝트 보기</p> <p>새 프로젝트 생성 성공 시 해당 프로젝트의 상세 페이지로 이동</p>			<ul style="list-style-type: none"> - 새 프로젝트 생성 form - 프로젝트 생성 성공 시 상세페이지로 이동 	

제목

프로젝트 상세보기 페이지

URL

/project/:id

Page

6

이미지

설명

TEAMATE Home Project Note Log out

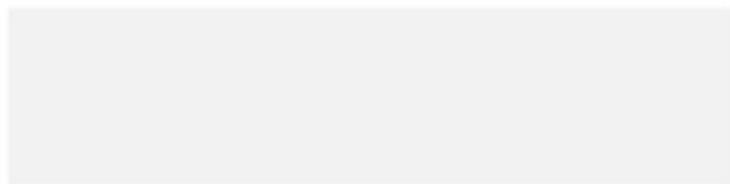
Search

팀메이트

급분야에게만 보이는 수정/삭제 버튼

수정

삭제



단원하세요 팀메이트 팀장 감사합니다.

팀프로젝트를 할 때, 점을 주기는 이 어려울 줄 예상한 적 있으신가요?

아니라면, 원하는 능력을 가진 멤버들을 구하기 힘들진 않으셨나요?

팀메이트는 프로젝트와 단체의 두 가지 플랫폼으로 동시에 제공하는 팀 청탁 플랫폼입니다.

또한 멤버의 평판을 measure한 테이크미에스에 저장 하여 점을 주기 전에 그 사람의 치스토리를 친하게 볼 수 있습니다.

디테일로 팀에서 백번도 개방자를 구하고 있습니다.

현재 참가인원

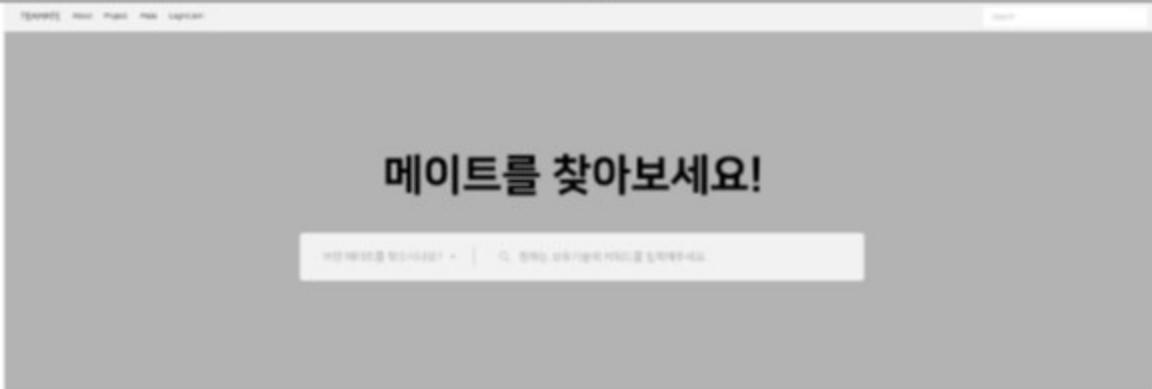
팀원
10명팀장
1명

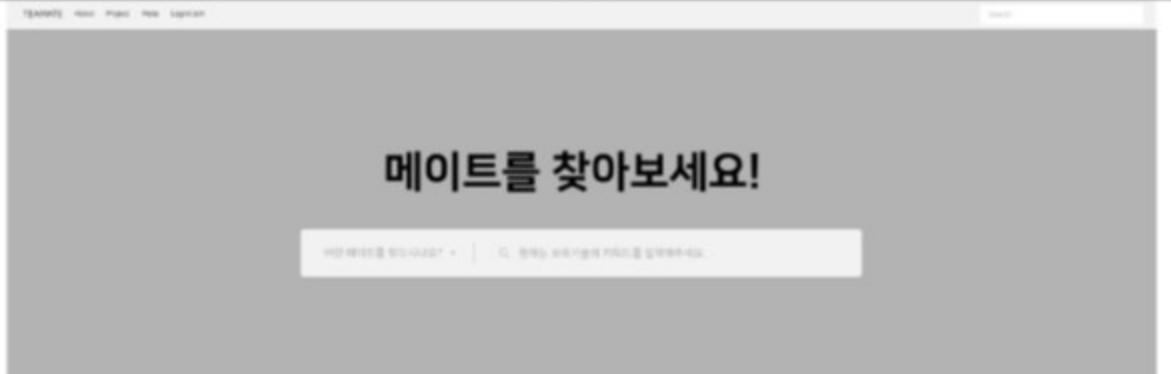
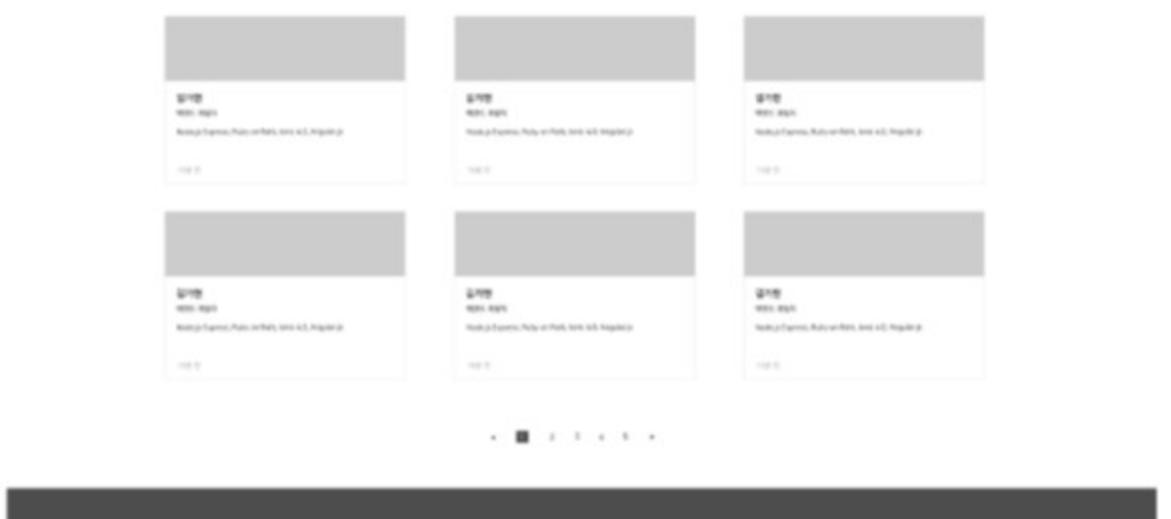
참가신청 보내기 버튼을 클릭 시 해당 프로젝트의 팀장에게 참가신청서가 날아감

참가신청 보내기

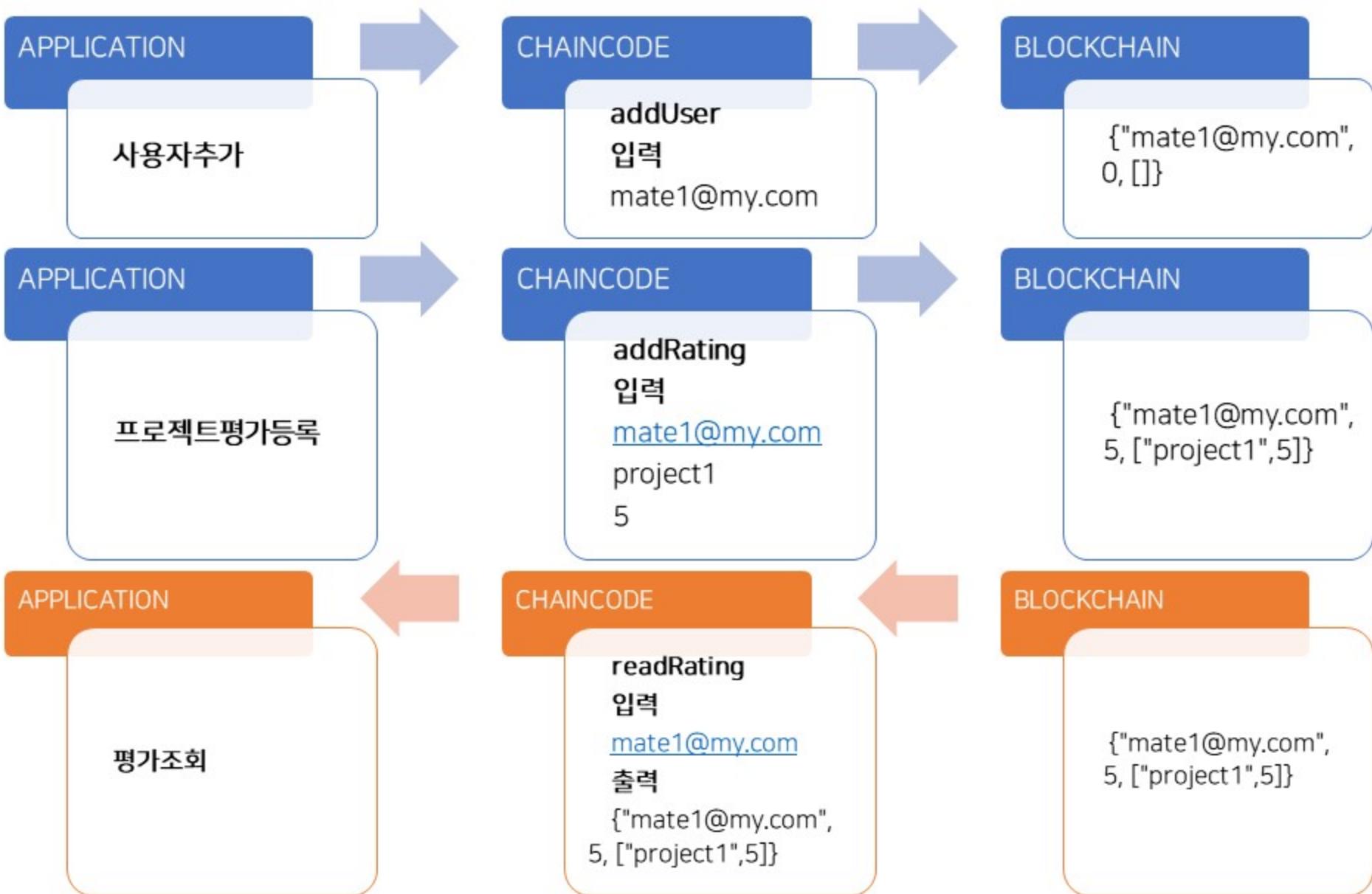
프로필 클릭 시 해당 팀메이트 상세 페이지로 이동

- 단일 프로젝트를 볼 수 있는 페이지
- 참가신청 보내기 버튼 : 클릭 시 해당 프로젝트의 팀장에게 참가신청이 간다
- 본인이 팀장인 프로젝트에서는 버튼이 보이지 않음

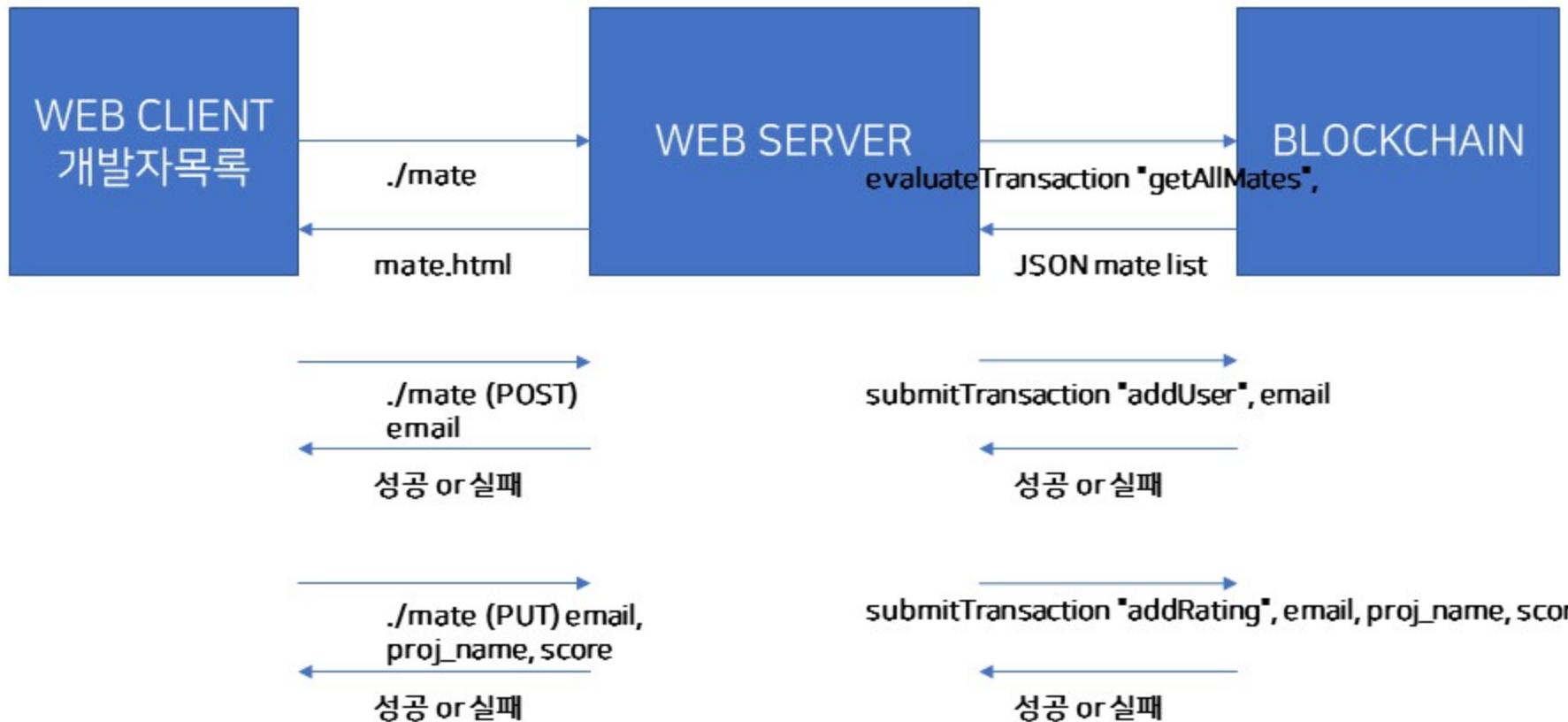
제목	메이트 전체 페이지	URL	/mate	Page	7
이미지	설명				
	<ul style="list-style-type: none"> - 메이트의 조건과 키워드로 검색 및 필터링 가능 - 카드 클릭 시 메이트 상세페이지로 이동 				
					

제목	메이트 전체 페이지	URL	/mate	Page
이미지			설명	
				
				<ul style="list-style-type: none"> - 메이트의 조건과 키워드로 검색 및 필터링 가능 - 카드 클릭 시 메이트 상세페이지로 이동

기능 플로우



기능상세 플로우





프론트 앤드 프로토 타입 계획

- 네트워크 실행
- 체인코드 배포 테스트
- 인증서 발급, 복사



웹서버 프로토타입 작성

- 간단한 기능의 REST server endpoint 작성
 - / : index.html
 - /create : create.html
 - /query : query.html
 - /modify: modify.html
- /mate GET 개발자 조회
- /mate POST 개발자 생성
- /mate PUT 개발자 정보 업데이트 및 수정
- /mate DELETE 개발자 정보 삭제



서버수행 메뉴얼

- 선행조건
 - 네트워크 실행
 - 체인코드 배포
 - 인증서 소유
- 실행
 - node server.js

```
bstudent@bstudent-VirtualBox:~/dev/fabricbook/application$ node server.js
Running on http://0.0.0.0:8080
```



웹 클라이언트 프로토타입 작성

- / : index.html
- /create : create.html
- /query : query.html
- /modify: modify.html

- jquery, css(bootstrap), ejs등을 사용한 웹서버와 연동하는 간단한 기능 페이지 구현



프론트엔드 프로토타입 1

- 수행 및 기능 테스트
- 선행조건
 -
- 스크린샷 및 기능설명

프로토타입

이메일로 메이트를 조회합니다

조회하기

average	3.7500000024999998
project title	p1
project title	p2
project title	p4
project title	p3

추가에 성공했습니다!

새 메이트를 추가합니다

추가하기

프론트엔드 프로토타입 2



프로토타입

조회에 성공했습니다!

이메일로 메이트를 조회합니다

user1

조회하기

average	3.7500000024999998
project title	p1
project title	p2
project title	p4
project title	p3

새 메이트를 추가합니다

새로 추가할 메이트의 email을 입력하세요

추가하기

웹서버 실행 메뉴얼



- **선행조건**
 - 네트워크(BN, fabricbook, FN)
 - 체인코드(teamate)
 - 인증서(admin, user)
 - 포함모듈 (package.json, npm install)
- **설정사항**
 - IP: dmcajou.ac.kr
 - Port:8080
- **실행명령**
 - node server.js
- **확인절차**
 - 웹서버 상태확인 및 로그확인
 - 웹클라이언트 접속
- **웹서버 종료**



TEAMATE
프로토타입

새 메이트를 추가합니다

새로 추가할 메이트의 email을 입력하세요

추가하기

메이트에 새 프로젝트 점수를 추가합니다

프로젝트점수를 추가할 메이트의 email:

프로젝트점수를 추가할 메이트의 프로젝트:

프로젝트점수를 추가할 메이트의 점수를:

추가하기

이메일로 메이트를 조회합니다

조회할 mate의 email을 입력하세요

조회하기

프로토 타입 테스트 결과

- addUser

선행조건 및 실행환경	결과
<ol style="list-style-type: none"> 네트워크 실행중 orderer, peer, couchdb, cli 채널생성 mychannel 체인코드 설치, 배포 체인코드 이름: teammate, 버전:1.0 웹서버 실행 	 <p>The screenshot shows a user interface for adding a new member ('MATE') to a project. A green success message at the top says '추가를 성공했습니다!' (Success). Below it, a button says '새 메이트를 추가합니다' (Add new mate) with the input field 'test_newuser'. A blue '추가하기' (Add) button is visible. At the bottom, there's a note '메이트에 새 프로젝트 점수를 추가합니다' (Add project score to mate) and a placeholder '프로젝트 점수를 추가할 메이트의 email' (Email of mate to add project score).</p>
<ol style="list-style-type: none"> dmc.ajou.ac.kr:8080접속 새로운 메이트 아이디 입력하기 새 메이트 추가하기 버튼클릭 	<p>블록체인 상태 및 이슈</p> <pre> 1 + { 2 "_id": "test_newuser", 3 "_rev": "1-148ddbe260aeafa6279920aaefd3ca545d", 4 "average": 0, 5 "rates": null, 6 "user": "test_newuser", 7 "~version": "\u0000CgMBZQA=" 8 } </pre>

프로토 타입 테스트 결과

- addRating

선행조건 및 실행환경	결과
<ol style="list-style-type: none"> 네트워크 실행중 orderer, peer, couchdb, cli 채널생성 mychannel 체인코드 설치, 배포 체인코드 이름: teammate, 버전:1.0 웹서버 실행 	 <p>새 메이트를 추가합니다 <input type="text" value="test_newuser"/> <input type="button" value="추가하기"/></p> <p>점수추가를 성공했습니다!</p> <p>메이트에 새 프로젝트 점수를 추가합니다 <input type="text" value="test_newuser"/> <input type="text" value="test_project"/> <input type="text" value="3"/> <input type="button" value="추가하기"/></p>
<ol style="list-style-type: none"> dmc.ajou.ac.kr:8080접속 프로젝트 점수주기정보입력 메이트 아이디, 프로젝트이름, 점수 새 프로젝트 추가하기 버튼클릭 	<p>블록체인 상태, 이슈</p> <pre> 1 = [2 "_id": "test_newuser", 3 "_rev": "2-352b962f7da8ee94d2a987030596de56", 4 "average": 3, 5 "rates": [6 { 7 "projecttitle": "test_project", 8 "score": 3 9 } 10], 11 "user": "test_newuser", 12 "version": "\u0000CgIBZgA=" 13] </pre>



프로토 타입 테스트 결과

- readRating

선행조건 및 실행환경	결과						
<ol style="list-style-type: none">네트워크 실행중 orderer, peer, couchdb, cli채널생성 mychannel체인코드 설치, 배포 체인코드 이름: teammate, 버전:1.0웹서버 실행	<p>The screenshot shows a user profile for 'test_newuser'. It includes a '회원가입' (Join) button, a '회원탈퇴' (Logout) button, and a '회원정보수정' (Edit Profile) button. Below this, there is a section for '평균 평점' (Average Rating) with a value of '3'. A table displays project information:</p> <table><thead><tr><th>평균 평점</th><th>3</th></tr></thead><tbody><tr><th>프로젝트 제목</th><td>test_project</td></tr><tr><th>점수</th><td>3</td></tr></tbody></table>	평균 평점	3	프로젝트 제목	test_project	점수	3
평균 평점	3						
프로젝트 제목	test_project						
점수	3						
<ol style="list-style-type: none">dmc.ajou.ac.kr:8080접속조회할 메이트 아이디 입력하기메이트 조회하기 버튼클릭	블록체인 상태, 이슈						