**Blockchain-JDP**

**architecture**

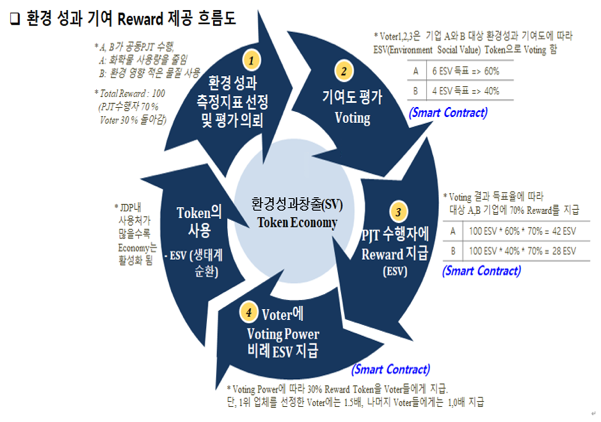
Theme : SK Hynix JDP 환경성과 기여에 따른 Reward 제공

플랫폼2그룹 이종석 수석

Digital Billing팀 박현진 선임

금융/전략 Digital 추진그룹 조재혁 선임

1. **과제 개요**



\* SK Hynix JDP(Joint Development Platform) 환경성과 기여에 따른 Reward 제공

S1. 환경 성과 측정지표 선정 및 평가 의뢰

S2. 블록체인 기반 Voting으로 환경 성과 평가 Smart Contract

S3. Reward의 70%를 Voting 결과 대상 업체들에게 Token으로 보상 Smart Contract

S4. Reward의 30%는 Voter들에게 분배(Top Voter에게 1.5배 지정) Smart Contract

1. **기술 아키텍쳐**

Server : Node.Js

Blockchain : OpenZeppelin, Remix

Android : Kotlin

Co-Work : Git

UI : Figma

1. **업무 프로세스**
2. ESV Token 활용 Voting Event 생성 (미구현)
3. 로그인
4. 기업 선택 후 Voting (Smart Contract)
5. 사용자와 기업에게 Voting 결과에 대한 보상 제공 (Smart Contract)

* On-chain Data
  + 꼭 필요한 정보만 저장
  + EvnetID, Evnet State
  + 후보자 address 및 받은 Token
  + 승리한 후보자
  + 투표자 address 및 투표한 Token
  + 참여자별 정산받을 Token
* Off-chain Data
  + 참여자 ID/PW/Blockchain account
  + Evnet 이름 총 토큰수
  + 후보 기업 이름 및 공적
  + 투표자 투표 히스토리

Reward Smart Contract

Vote Smart Contract

Server

Android Application

Event 생성 (관리자)

On-Chain

Off-Chain

1. **개발 및 운영 환경 정보** 
   * 개발 환경 정보
     1. git repository : <https://github.com/ro0opf/blockchain-jdp>
        1. /BlockchainApp : Android App
        2. /contract\_test\_hj : app과 연동하는 Backend
        3. /jdp-backend : 운영자용 Backend + swagger
        4. /jdp-contract : 배포용 trupple , smart contract
     2. REMIX
     3. Ganache
   * 운영 환경 정보
     1. Mainnet Contract : 0x394BfB86641Bf8cA8A757a318499A580Cc1f26C6
     2. Truffle 에JWT token 추가하여 배포
     3. 운영자 backend 서버 (AWS)
        1. <http://ec2-3-34-159-82.ap-northeast-2.compute.amazonaws.com:8000>
     4. app용 backend 서버 (AWS + RDB)
        1. http://ec2-3-34-159-82.ap-northeast-2.compute.amazonaws.com:9999
2. **Truffle 기반 Solidity Testcase**

./test/sampleCallSmartContract.js

Addr로 callTransfer 테스트

1. **시나리오 구현 (준비사항)**
2. 환경 성과 측정 대상 기업 및 투표자 계정 생성
   * 환경 성과 측정 대상 기업 :
     1. SK ENERGY, 0xd95428Ce6513b81aEd992dD4Bda32F1AA65cD6A1
     2. SK PLANET 0xE98ee998648430c6629f5997A7932117b52E116A

-> SK ENERGY : SK ENERGY IS GOING TO INVEST STH FOR ENVIRONMENT

-> SK PLANET : SK PLANET IS GOING TO INVEST STH FOR ENVIRONMENT

* 투표자 :
  + hyunjin, 0xF674D74A10aa18a2e30ad45Fa0721987e9ecF1d9
  + jaehyeok, 0x32727AaDE31b7d98cBeE3acf4881CEA393f6B6f3
  + jongseok 0x6eCb35484005854400d13a5a4fb8e53B0AfaFf08

2. ERC20 Token 발행 (Reward Pot 100 ESV 포함)

- 200 ESV (Environment Social Value) Token 발행

3. Voter에게 투표용 기본 Token 할당

- hyunjin : 10ESV

- jaehyeok : 10ESV

- jongseok : 10ESV

**5-1. 시나리오 구현**

S1. 환경 성과 측정지표 선정 및 평가 의뢰 -> 관리자 생성

* 기업 SK ENERGY, SK PLANET가 공동 PJT 수행 가정

[환경 평가 결과]

-> SK ENERGY : SK ENERGY IS GOING TO INVEST STH FOR ENVIRONMENT

-> SK PLANET : SK PLANET IS GOING TO INVEST STH FOR ENVIRONMENT

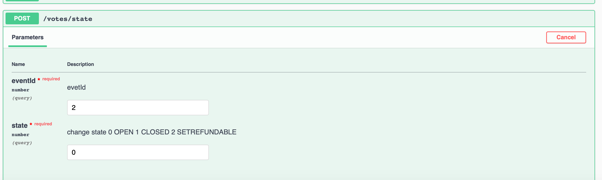
1. 투표 Event 생성



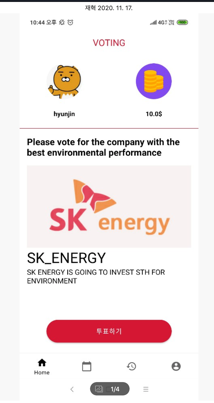
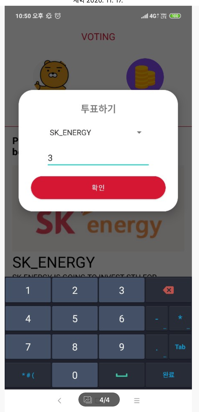
1. 100 ESV Funding



1. 투표시작



**S2. 블록체인 기반 Voting으로 환경 성과 평가 Smart Contract**

1. Voter 들의 Voting
   1. Hyunjin은 SK ENERGY 에 3 ESV vote => 잔액 10 -> 7 ESV
   2. jaehyeok은 SK ENERGY 에 3 ESV vote => 잔액 10 -> 7 ESV
   3. jongseok은 SK PLANET에 4 ESV vote => 잔액 10 -> 6 ESV
   4.  
2. 투표종료 (운영자)



1. 정산 (운영자)



1. 환경 성과 평가 대상 기업 Voting 결과 (기업 득표수가 동일한 경우는 제외)
   1. SK ENERGY = 6 ESV, SK PLANET = 4 ESV
   2. SK ENERGY = 6 / (6 + 4) = 60%
   3. SK PLANET = 4 / (6 + 4) = 40%

**S3. Reward 의 70%를 Voting 결과 대상 업체들에게 Token으로 보상 Smart Contract**

1. Voting 대상 기업들에게 Voting 득표율에 따른 Token 보상 자동화(배당률 : 전체 Reward Pot의 70%)

• SK ENERGY Reward = Reward 용 전체 Token X SK ENERGY 득표율 X 배당률 = 110ESV X 60% X 70% = 46.2ESV

• SK PLANET Reward = Reward 용 전체 Token X SK PLANET 득표율 X 배당률 = 110ESV X 40% X 70% = 30.8ESV

**S4. Reward 의 30%를 Voter들에게 분배(1위 Voter에게는 1.5배 보상) Smart Contract**

1. Voter들에게 Reward 보상 (배당률 : 전체 Reward의 30%)

* 1위 업체에게 Voting 한 Voter들에게 1.5배, 나머지 Voter들에게는 1배
* Voting 모수 = 1위 기업에 Voting한 총 ESV X 1.5 + 나머지 기업에 Voting한 Total ESV X 1.0 = (6 \* 1.5) + (4 \* 1) = 13ESV

1. Voter들에게 30% Reward 보상 (Reward 계산 시 소수점 둘째자리 이하 절사)

* 1위 기업에게 Voting한 Voter의 Reward
  + Voting ESV X Reward의 배당률 30% X 1.5
* 나머지 기업에 Voting한 Voter의 Reward\
  + Voting ESV X Reward의 배당률 30% X 1.0

- Hyunjin 의 Reward (SK ENERGY 기업에 3ESV Vote)

- 3 ESV X 33 ESV X 1.5 / 13ESV = 11.42 ESV -> Hyunjin + 투표하고 남은 7 ESV

- Jaehyeok 의 Reward (SK ENERGY 기업에 3ESV Vote)

- 3 ESV X 33 ESV X 1.5 / 13ESV = 11.42 ESV -> Jaehyeok + 투표하고 남은 7ESV

- JongSeok 의 Reward (SK ENERGY 기업에 4ESV Vote)

- 4 ESV X 33 ESV X 1 / 13ESV = 10.15 ESV -> JongSeok + 투표하고 남은 6 ESV

* 30% Reward (33) – Reward (32.99) = 0.01 ESV (낙전 처리 방안 제시)

각자 확정된 Reward 를 회수



대상 기업 부터 Voter 들의 최종 Token 값 조회



**5-2. Smart Contract 설계**

1. Open zeppelin code import
   1. ERC20.sol
   2. IERC20.sol
   3. Context.sol
   4. SafeMath.sol
2. ERC20 Token Contract Interface–

ERC20 Interface가 표준이라 수정하지 않고 사용 예정

사용 Interface)

function balanceOf(address \_who) public view returns (uint256);

function transfer(address \_to, uint256 \_value) public returns (bool);

1. Voting Contract Interface

voting event 생성, reward 펀딩, voting 수행, voting 종료 처리 후 reward 분배, 낙전 처리, 보안관련 처리 필요(연산에 보안적으로 검증된 Library 사용)

사용 Interface)

function createEvent(uint16 eventId, address[] memory candidateAddress)

function fundReward(uint16 eventId, uint256 token)

function startVoting(uint16 eventId)

function voting(uint16 eventId, address candidate, uint256 token)

function stopVoting(uint16 eventId)

function setRefundable(uint16 eventId)

function withdraw(uint16 eventId)

security audit 결과



**5-3. 앱 내 화면**

