**Blockchain-JDP**

**architecture**

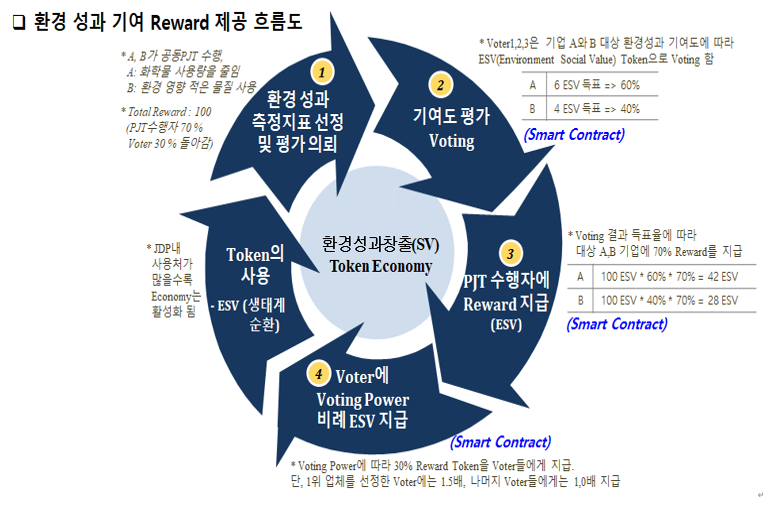
Theme : SK Hynix JDP 환경성과 기여에 따른 Reward 제공

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)

Reward Smart Contract

Vote Smart Contract

Server

Android Application

Event 생성 (관리자)

On-Chain

Off-Chain

1. **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 : SK ENERGY, SK PLANET

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

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

* 투표자 : hyunjin, jaehyeok, jongseok

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

**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
2. 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 = 100ESV X 60% X 70% = 42ESV
   * SK PLANET Reward = Reward 용 전체 Token X SK PLANET 득표율 X 배당률 = 100ESV X 40% X 70% = 28ESV

**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 30 ESV X 1.5 / 13ESV = 10.38 ESV -> Hyunjin
* Jaehyeok 의 Reward (SK ENERGY 기업에 3ESV Vote)
  + 3 ESV X 30 ESV X 1.5 / 13ESV = 10.38 ESV -> Jaehyeok
* JongSeok 의 Reward (SK ENERGY 기업에 4ESV Vote)
  + 3 ESV X 30 ESV X 1.5 / 13ESV = 9.23 ESV -> JongSeok
* 30% Reward (30) – Reward (29.99) = 0.01 ESV (낙전 처리 방안 제시)

**5-2. Smart Contract 설계**

1. 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 받기, voting 수행, voting 종료 처리 후 reward 분배, 낙전 처리, 보안관련 처리 필요(연산에 보안적으로 검증된 Library 사용)

사용 Interface)

fuction createEvent(uint32 eventId, address[] targetAddress) public returns (address)

fuction startVoting(uint32 eventId, uint duration) public

fuction stopVoting(uint32 eventId, uint duration) public

function voting(address target, uint32 token) public

fuction fundReward(uint256 Reward) public (uint256)

fuction withdraw() public returns (uint256)

**5-3. 앱 내 화면**

