

2024년 전기 졸업과제 중간보고서

허가형 블록체인을 이용한 데이터 수집 시스템 개발



팀명 : 윤죽정

201924484 박형주

201924427 김민종

201924566 전원균

지도교수 : 염근혁 (인)

목차

1. 과제 선정 배경	3
2. 과제 목표	3
3. 요구사항 변경점 및 제약사항 대응	4
1) 요구사항 변경점	4
2) 제약사항 대응	10
4. 설계 상세화	11
1) 전체 시스템 구조	11
2) 기능별 설계 상세화	13
5. 갱신된 과제 추진 계획	25
6. 구성원별 진척도	26
7. 과제 수행 내용 및 중간 결과	27
1) 스마트 컨트랙트 템플릿	27
2) 스마트 컨트랙트 풀	
3) 하이퍼레저 패브릭 네트워크	
4) 하이퍼레저 패브릭 네트워크에 연동되는 사용자 인증 기술	
8. 향후 계획	38

1. 과제 선정 배경

다양한 분야에 IT 기술을 접목하고자 하는 시도가 증대됨에 따라 기술 융합에 필요한 데이터 수집과 보관의 중요성도 함께 부각되고 있다. 효과적인 데이터 수집을 위해서는 수집된 데이터의 신뢰성, 민감 정보가 포함되는 데이터의 보안성, 데이터 변조를 방지하는 무결성이 필요하며 위와 같은 데이터의 특성을 보장할 수 있는 기술로 블록체인 기술을 활용할 수 있다. 예를 들어, 선거관리위원회는 블록체인 기술을 적용한 온라인 투표 시스템을 통해 온라인 투표를 수행하고 있으며 블록체인 기술의 특성을 통해 투표 결과의 신뢰성을 보장하고 있다.

블록체인 기술은 플랫폼의 참여자 구분에 따라 비허가형(퍼블릭), 허가형(프라이빗) 블록체인으로 구분할 수 있다. 현재, 데이터 수집 시스템과 같이 정보를 수집하기 위해 활용되는 플랫폼 형태는 불특정 다수의 참여자의 접근성을 고려하여 퍼블릭 블록체인에 기반하여 구성된다. 그러나, 퍼블릭 블록체인 기반의 정보 수집 기술은 신뢰성을 보장하기 위해 제한적인 범위의 데이터를 수집하고 있다는 문제가 있다. 또한, 퍼블릭 블록체인 기반의 데이터 수집은 불특정 다수 사용자의 데이터 생성으로 인해, 수집된 데이터가 활용 가능한지를 추가 검증하는 등 데이터 신뢰성의 문제를 해소할 필요가 있다.

예를 들어, Golem은 분산형 컴퓨팅 플랫폼으로, 이더리움 블록체인을 기반으로 한 시스템이다. 사용자의 유휴 컴퓨팅 자원을 네트워크에 공유하고, 다른 사용자가 금액을 지불하고 필요한 컴퓨팅 파워를 얻는 구조이다. 퍼블릭 블록체인 네트워크를 사용하기 때문에 사용자가 쉽게 접근할 수 있어 악의를 가진 사용자가 허위 정보를 네트워크에 공유하고, 이에 대한 부당한 이익을 취할 수 있다는 문제점이 발생한다.

데이터 신뢰도를 높이기 위해서는 블록체인 플랫폼의 참여자를 제한하여 인가받은 사용자를 참여시켜 무분별한 데이터 생성 및 활용을 방지하는 방법을 활용할 수 있다. 따라서, 본 과제에서는 허가형 블록체인을 활용한 데이터 수집 시스템을 제안한다. 제안하는 시스템은 1) 데이터 수집 기능 구현을 위한 하이퍼레저 패브릭 기반 스마트 컨트랙트 구성, 2) 하이퍼레저 패브릭 데이터 확인을 위한 대시보드 구축으로 구성된다.

2. 과제 목표

- i. 스마트 컨트랙트 기반 하이퍼레저 패브릭 플랫폼 인프라 구축
- ii. 스마트 컨트랙트를 활용한 데이터 수집 시스템 구축
- iii. 수집 결과에 기반한 데이터 모니터링 구현

3. 요구사항 변경점 및 제약사항 대응

1) 요구사항 변경점

표 1. 유스케이스 변동사항

유스케이스명	변동사항	설명
사용자 인증	변경	CA 서버를 별도 구축, 사용자의 인증 리소스 관리
스마트 컨트랙트 실행	삭제	스마트 컨트랙트 실행이 수집 이벤트에 따라 발생한다 판단하여 삭제
데이터 수집 항목 설정	변경	수집자가 설정할 수집 항목을 투표, 수치 데이터, 줄글 데이터로 제한하고 템플릿으로 미리 구현
수집 데이터 입력	변경	참여자가 수집 데이터를 입력하는 과정에 스마트 컨트랙트 실행을 포함

표 1은 도출한 유스케이스의 변동사항과 그 설명을 나타낸 표이다.

i. 사용자 인증

사용자 인증 기능은 CA(Certificate Authority)를 활용하여 사용자가 갖는 역할과 권한을 확인하는 기능이다. 기존 사용자 인증 기능은 회원가입 시 사용자가 제공한 아이디와 비밀번호로 CA를 구성하고 하이퍼레저 패브릭의 MSP(Membership Service Provider)를 통해 CA의 역할과 권한을 할당하는 방식으로 설계하였다. 이는 하이퍼레저 패브릭의 보안설계에 위반할 수 있는 상황으로 확인되어, 별도의 CA 서버를 구축하여 사용자의 인증 리소스를 관리할 수 있도록 설계를 변경하였다.

• 기존의 사용자 인증 유스케이스 명세

유스케이스명	사용자 인증
개요	사용자는 네트워크에 참여하기 위해 사용자 인증을 받아야 한다.
관련 액터	MSP
선행 조건	없음
이벤트 흐름	<p>기본 흐름:</p> <ol style="list-style-type: none"> 1. 시스템은 CA(Certificate Authority)를 호출하여 새로운 사용자에게 인증서를 발급 요청한다. 2. CA는 사용자에게 인증서를 발급하고 이를 시스템에 반환한다. 3. 시스템 관리자는 등록된 사용자에게 하이퍼레저 패브릭 네트워크 접근 권한을 설정한다. 4. 시스템은 MSP를 호출하여 사용자의 역할과 권한을 설정한다. 5. MSP는 사용자의 역할과 권한을 설정하고 시스템에 반환한다.
후행 조건	없음

• 변경된 사용자 인증 유스케이스 명세

유스케이스명	사용자 인증
개요	사용자는 네트워크에 참여하기 위해 사용자 인증을 받아야 한다.
관련 액터	MSP, DBMS
선행 조건	없음
이벤트 흐름	<p>기본 흐름:</p> <ol style="list-style-type: none"> 1. 사용자는 시스템에서 회원가입에 필요한 정보를 입력해 회원가입을 시도한다. 2. 시스템은 입력한 정보의 유효성을 검사한다. 3. 시스템은 Fabric CA Client를 통해 사용자의 id의 고유한 CA key를 생성한다. 4. 시스템은 사용자가 입력한 회원정보와 CA key를 회원정보를 저장하는 데이터베이스에 저장한다. 5. 시스템은 사용자에게 회원가입이 완료되었음을 알린다. <p>대안흐름:</p> <p>1.1 이미 존재하는 id인 경우</p> <ol style="list-style-type: none"> 1. 시스템은 사용자의 회원정보가 저장되어 있는 데이터베이스에 id를 조회해 이미 존재하는 id인지 확인한다. 2. 동일한 id가 존재하는 경우, 사용자에게 이를 알린다. <p>2.1 사용자가 잘못된 정보를 입력한 경우</p> <ol style="list-style-type: none"> 1. 시스템은 정보가 유효하지 않음을 사용자에게 알린다. 2. 사용자는 피드백을 받아 올바른 입력을 수행한다.
후행 조건	없음

ii. 스마트 컨트랙트 실행

기능 설계 당시 사용자가 스마트 컨트랙트를 수동적으로 실행하도록 설계하였다. 실험 실습을 통해 도출한 사용자의 데이터 수집 흐름은 스마트 컨트랙트의 실행이 데이터 수집 이벤트에 따라 이루어 질 수 있음을 확인하였다. 따라서, 스마트 컨트랙트 실행 기능은 도출 유스케이스에서 제외하였다.

iii. 데이터 수집 항목 설정

기존에 계획했던 데이터 수집 항목 설정 기능은, 데이터 수집 항목 설정이 완료되면 시스템이 스마트 컨트랙트 템플릿을 불러와 에셋 설정을 적용한 후 하이퍼레저 패브릭에 스마트 컨트랙트를 배포하는 기능이었다. 그러나 데이터 수집을 진행할 수집자가 수집하고자 하는 데이터의 범위가 방대하다는 문제가 있었다. 따라서 그 범위를 좁힐 필요가 있었고, 그 과정에서 고려한 사항은 다음과 같다.

투표 데이터는 이산적으로 명확한 선택지를 제공하여 수집된 데이터를 빠르게 분석하고 통계적으로 처리하는데 유리하다. 숫자 데이터는 정량적으로 측정 가능하며, 평균, 분산, 표준편차 등 다양한 통계적 지표를 산출하기 용이하다. 줄글 데이터는 비교적 유연한 형식으로 참여자의 주관적 의견이나 설명을 수집할 수 있다.

위와 같은 이유로 수집자가 설정할 수 있는 범위를 투표, 숫자 데이터, 줄글 데이터로 한정하여 각 주제별로 수집자가 설정할 수 있는 항목을 스마트 컨트랙트 템플릿으로 구현하였다.

● 기존의 데이터 수집 항목 설정 유스케이스 명세

유스케이스명	데이터 수집 항목 설정
개요	수집자는 수집할 데이터의 항목을 설정할 수 있다.
관련 액터	수집자, 스마트 컨트랙트 풀
선행 조건	수집자는 인증된 사용자여야 한다.
이벤트 흐름	<p>기본 흐름:</p> <ol style="list-style-type: none"> 1. 시스템은 스마트 컨트랙트 풀에 저장되어 있는 스마트 컨트랙트 템플릿을 불러온다. 2. 수집자가 수집하고자 하는 데이터의 주제와 항목들을 사용자 인터페이스에 입력한다. 3. 시스템은 불러온 스마트 컨트랙트 템플릿에 입력된 에셋 설정을 적용해 스마트 컨트랙트를 생성한다. 4. 시스템은 생성된 스마트 컨트랙트를 하이퍼레저 패브릭에 배포한다. <p>대안흐름:</p> <p>2.1 수집자가 잘못된 데이터를 입력한 경우</p> <ol style="list-style-type: none"> 1. 시스템은 입력 데이터의 유효성을 검사한다. 2. 오류가 발견되면 수집자에게 이를 알린다. 3. 수집자는 피드백을 받아 올바른 입력을 수행한다. <p>3.1 템플릿을 불러오는 데 실패한 경우</p> <ol style="list-style-type: none"> 1. 시스템은 로그를 남기고 스마트 컨트랙트 템플릿을 불러오는 데 실패했다는 문구를 출력한다. 2. 수집자는 네트워크 문제 등 문제를 해결한 후 다시 시도한다. <p>4.1 배포에 실패한 경우</p> <ol style="list-style-type: none"> 1. 하이퍼레저 패브릭은 배포에 실패했다는 로그를 남긴다. 2. 수집자는 오류가 수정된 후 다시 배포를 시도한다.
후행 조건	없음

● 변경된 데이터 수집 항목 설정 유스케이스 명세

유스케이스명	데이터 수집 항목 설정
개요	수집자는 수집할 데이터의 항목을 설정할 수 있다.
관련 액터	수집자, 하이퍼레저 패브릭, DBMS
선행 조건	수집자는 인증된 사용자여야 한다.
이벤트 흐름	<p>기본 흐름:</p> <ol style="list-style-type: none"> 1. 시스템은 수집자에게 데이터 수집을 진행할 주제를 선택할 수 있는 데이터 수집 주제 목록을 보여준다. 2. 수집자는 선택한 주제에 필요한 수집 항목과 수집 기한을 설정한다.

	<p>3. 시스템은 수집자가 입력한 수집 항목을 하이퍼레저 패브릭에 저장하는 스마트 컨트랙트를 실행한다.</p> <p>4. 데이터 수집 기한을 저장하는 데이터베이스에 수집 기한을 저장하고 사용자에게 완료되었음을 알린다.</p> <p>대안흐름:</p> <p>2.1 수집자가 잘못된 데이터를 입력한 경우</p> <ol style="list-style-type: none"> 1. 시스템은 입력 데이터의 유효성을 검사한다. 2. 오류가 발견되면 수집자에게 이를 알린다. 3. 수집자는 피드백을 받아 올바른 입력을 수행한다. <p>3.1 스마트 컨트랙트 실행에 실패한 경우</p> <ol style="list-style-type: none"> 1. 시스템은 로그를 스마트 컨트랙트 실행에 실패했다는 문구를 출력한다. 2. 수집자는 오류가 수정된 후 다시 시도한다.
후행 조건	없음

iv. 수집 데이터 입력

앞서 스마트 컨트랙트 실행이 수집 이벤트에 따라 수행될 수 있음을 확인하여 스마트 컨트랙트 실행 기능을 삭제하였다. 따라서 수집자가 하이퍼레저 패브릭에 저장한 수집 항목을 불러오는 과정과 참여자가 수집 데이터를 입력하는 과정에 스마트 컨트랙트 실행을 포함시켰다.

- 기존의 수집 데이터 입력 유스케이스 명세

유스케이스명	수집 데이터 입력
개요	참여자는 데이터를 입력할 수 있음
관련 액터	참여자, 하이퍼레저 패브릭
선행 조건	참여자는 인증된 사용자여야 한다.
이벤트 흐름	<p>기본 흐름:</p> <ol style="list-style-type: none"> 1. 참여자는 데이터 입력 메뉴를 선택한다. 2. 시스템은 데이터 입력 항목을 사용자에게 출력한다. 3. 참여자는 각 항목에 해당하는 데이터를 입력한다. 4. 참여자는 제출 버튼을 클릭하여 데이터를 제출한다. 5. 시스템은 제출된 데이터를 하이퍼레저 패브릭에 저장하도록 요청한다. 6. 제출된 데이터는 하이퍼레저 패브릭에 저장된다. <p>대안 흐름:</p> <p>3.1 참여자가 잘못된 데이터를 입력한 경우</p> <ol style="list-style-type: none"> 1. 시스템은 참여자에게 입력받은 데이터가 잘못되었다는 메시지를 출력한다. 2. 참여자는 올바른 데이터를 입력한다. <p>4.1 참여자가 항목을 누락한 경우:</p> <ol style="list-style-type: none"> 1. 시스템은 참여자에게 입력 항목을 모두 입력해 달라는 메시지를 출력한다. 2. 참여자는 누락된 항목을 입력한 뒤 다시 제출한다.
후행 조건	없음

● 변경된 수집 데이터 입력 유스케이스 명세

유스케이스명	수집 데이터 입력
개요	참여자는 데이터를 입력할 수 있음
관련 액터	참여자, 하이퍼레저 패브릭
선행 조건	참여자는 인증된 사용자여야 한다.
이벤트 흐름	<p>기본 흐름:</p> <ol style="list-style-type: none"> 1. 참여자는 데이터를 제출할 데이터 수집 세션을 선택한다. 2. 시스템은 해당 데이터 수집 세션의 수집 항목을 스마트 컨트랙트를 실행해 인터페이스에 출력한다. 3. 참여자는 수집 주제에 맞는 데이터를 입력한다. 4. 시스템은 참여자가 입력한 데이터의 유효성을 검사한다. 5. 시스템은 참여자가 입력한 데이터를 스마트 컨트랙트를 실행해 하이퍼레저 패브릭에 저장한다, <p>대안 흐름:</p> <p>3.1 참여자가 항목을 누락한 경우:</p> <ol style="list-style-type: none"> 1. 시스템은 참여자에게 입력 항목을 모두 입력해 달라는 메시지를 출력한다. 2. 참여자는 누락된 항목을 입력한 뒤 다시 제출한다. <p>4.1 참여자가 잘못된 데이터를 입력한 경우</p> <ol style="list-style-type: none"> 1. 시스템은 참여자에게 입력받은 데이터가 잘못되었다는 메시지를 출력한다. 2. 참여자는 올바른 데이터를 입력한다.
후행 조건	없음

2) 제약사항 대응

기존 제약사항으로는 실 사례에 기반하여 데이터를 쌓는 과정이 한정적이고, 실 세계의 모든 카테고리에 대해 데이터 수집을 진행하기 어려울 것으로 보았으며 수집된 데이터를 시각화할 방법이 방대하다는 것이 있었다.

데이터의 형식이 다양해질수록 데이터의 시각화 방법이 다양해진다. 데이터의 형식을 제한하면 그에 맞는 시각화 방식도 제한되어 시각화의 복잡성을 줄일 수 있다. 먼저 투표 데이터의 경우, 선택지의 득표 비율을 원형 차트로 시각화한다. 숫자 데이터의 경우, 숫자 데이터의 분포를 히스토그램의 형태로 시각화한다. 줄글 데이터의 경우, 설문의 응답에서 자주 등장하는 단어를 워드 클라우드로 시각화한다.

4. 설계 상세화

1) 전체 시스템 구조

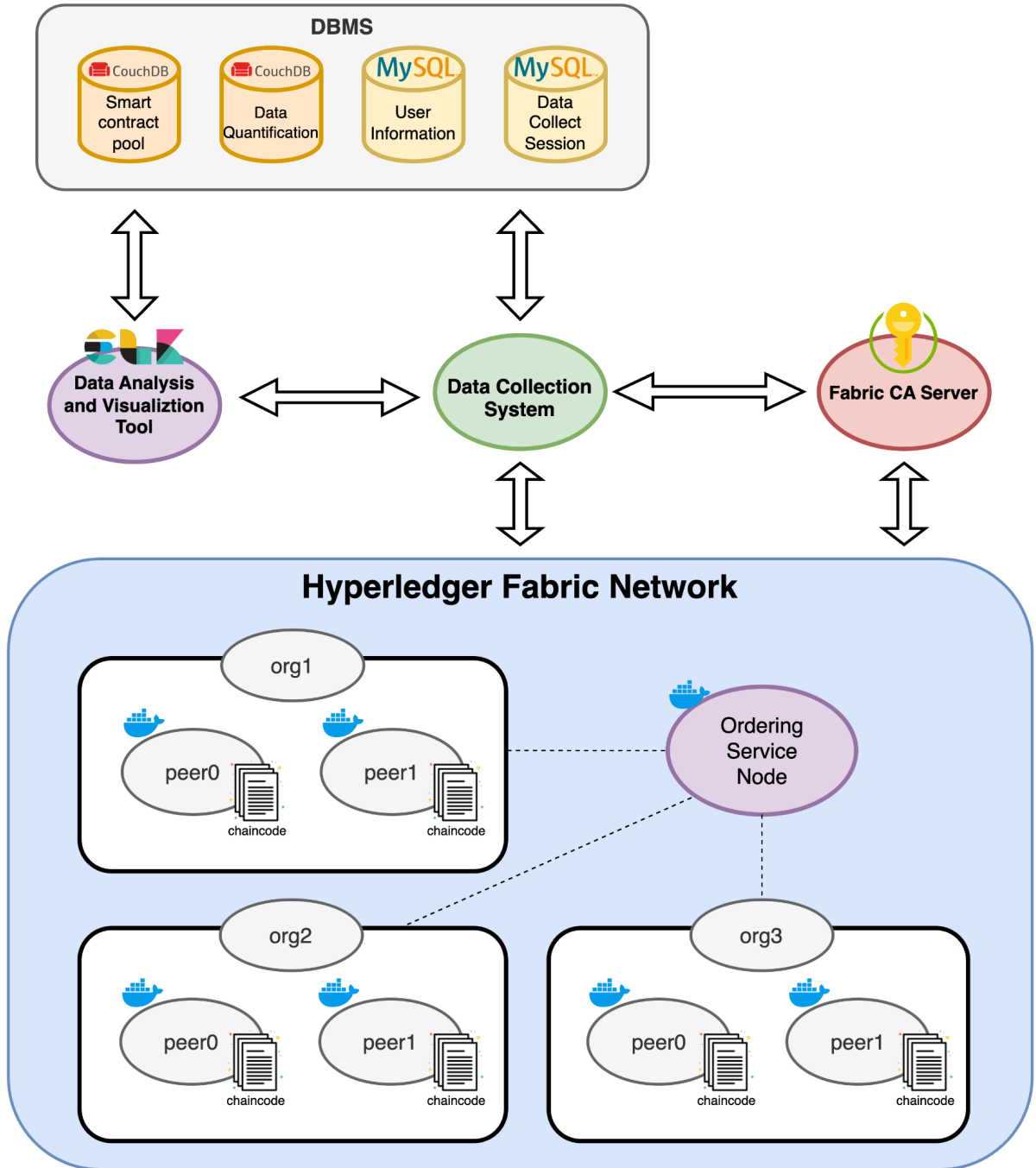


그림 1. 전체 시스템 구성도

그림 1은 데이터 수집 시스템의 전체 시스템을 나타낸 구성도이다. 전체 시스템 구조는 크게 데이터 수집 시스템, DBMS(Database control System), 데이터 분석 및 시각화 도구(), Fabric CA 서버, 하이퍼레저 패브릭 네트워크로 나뉜다.

먼저, 데이터 수집 시스템은 시스템의 중심 허브로서 기능하며, 시스템에 등록된 사용자 정보, 수집된 데이터의 흐름을 관리한다.

데이터 수집 시스템은 DBMS와 연결되어, 스마트 컨트랙트 풀, 데이터 정량화, 사용자 정보, 데이터 수집 세션 정보를 포함하는 데이터베이스와 상호작용한다. CouchDB와 MySQL이 사용되며, CouchDB에는 스마트 컨트랙트 템플릿과 정량화된 데이터를 저장한다. MySQL에는 시스템에 등록된 사용자 정보와 데이터 수집 세션 데이터를 저장한다. 시스템과 연결된 DBMS를 통해 관리자는 스마트 컨트랙트의 템플릿을 관리하고 등록된 사용자 정보를 관리할 수 있다. 또한 정량화된 데이터를 저장하여 데이터 시각화에 사용한다.

또한, 데이터 수집 시스템은 데이터 분석 및 시각화 도구와도 연결되어 있다. 수집된 데이터를 실시간으로 분석하고, ELK(Elasticsearch, Logstash, Kibana) Stack을 통해 데이터를 시각화하여 사용자에게 제공하는 역할을 한다. 이를 통해, 관리자는 트랜잭션 로그와 같은 데이터를 기반으로 유의미한 정보를 도출해낼 수 있다.

Fabric CA 서버는 시스템의 인증 및 보안 관리를 담당한다. 해당 서버는 하이퍼레저 패브릭 네트워크에 참여하는 모든 노드와 사용자에게 인증서를 발급하며, 이러한 인증서를 통해 트랜잭션이 안전하게 처리된다. 데이터 수집 시스템은 Fabric CA 서버를 통해 네트워크 내에서 발생하는 모든 활동이 적절하게 검증되도록 한다.

마지막으로, 데이터 수집 시스템은 하이퍼레저 패브릭 네트워크와 연결되어 있다. 하이퍼레저 패브릭 네트워크는 3개의 조직으로 구성되어 있으며, 각 조직은 2개씩의 피어 노드를 갖는다. 하이퍼레저 패브릭 네트워크는 배포된 스마트 컨트랙트를 실행하고 트랜잭션을 검증한다. 데이터 수집 시스템은 이 네트워크와 상호작용하여 블록체인 상의 데이터를 수집하고 이를 불러와 시각화 및 분석 도구로 전송하는 등 데이터의 흐름을 관리한다.

2) 기능별 설계 상세화

i. 사용자 인증

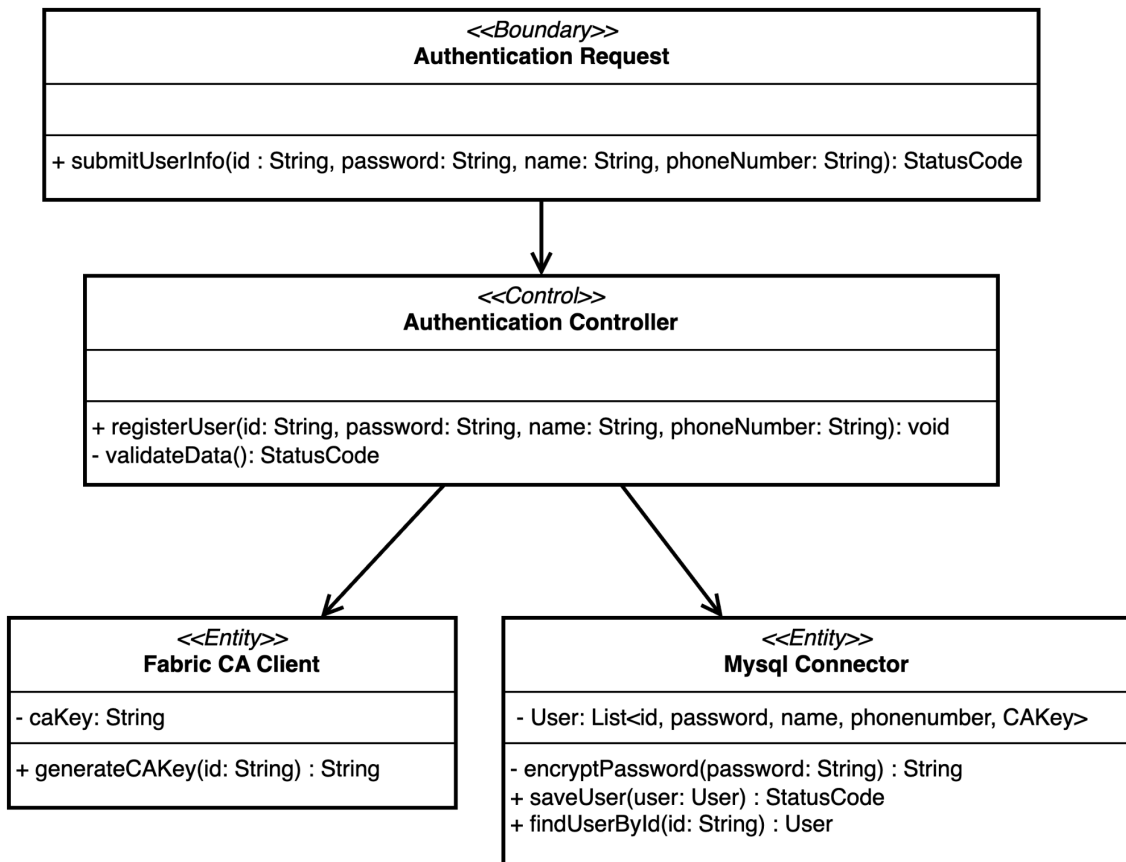


그림 2. 사용자 인증 클래스 다이어그램

그림 2는 사용자 인증 기능을 ECB 패턴을 적용하여 도출한 클래스 다이어그램이다. Authentication Request는 사용자가 시스템을 사용하기 위해 필요한 정보를 처리하는 Boundary class이다. submitUserInfo 메서드는 사용자가 입력한 아이디, 비밀번호, 이름, 전화번호 정보를 받아서 Authentication Controller로 전달하고 그 결과로 상태 코드를 반환한다.

Authentication Controller는 사용자의 인증 요청을 처리하고, 필요한 데이터를 검증하거나 저장하는 역할을 하는 Control class이다. validateData 메서드를 사용해 데이터의 유효성을 검증하고, registerUser 메서드로 사용자의 정보를 받아서 등록 처리를 한다.

Fabric CA Client는 Hyperledger Fabric에 사용되는 인증서를 발급하는 Entity class이다. 사용자가 등록될 때, 인증서를 생성하여 이를 Mysql Connector에 저장한다. generateCAKey 메서드는 주어진 사용자 ID를 기반으로 CA Key를 생성한다.

Mysql Connector는 사용자 데이터를 Mysql 데이터베이스에 저장하고, 필요한 경우 데이터를 조회하는 Entity class이다. User attribution은 사용자 정보를 저장하는 테이블을 나타낸다. findUserById 메서드를 사용해 주어진 사용자 ID로 사용자를 조회할 수 있으며, saveUser 메서드로 사용자의 정보를 데이터베이스에 저장하고, 그 결과로 상태 코드를 반환한다. 이때 사용자의 정보를 저장하는 과정에서 encryptPassword 메서드가 실행되어 사용자의 비밀번호를 암호화한다.

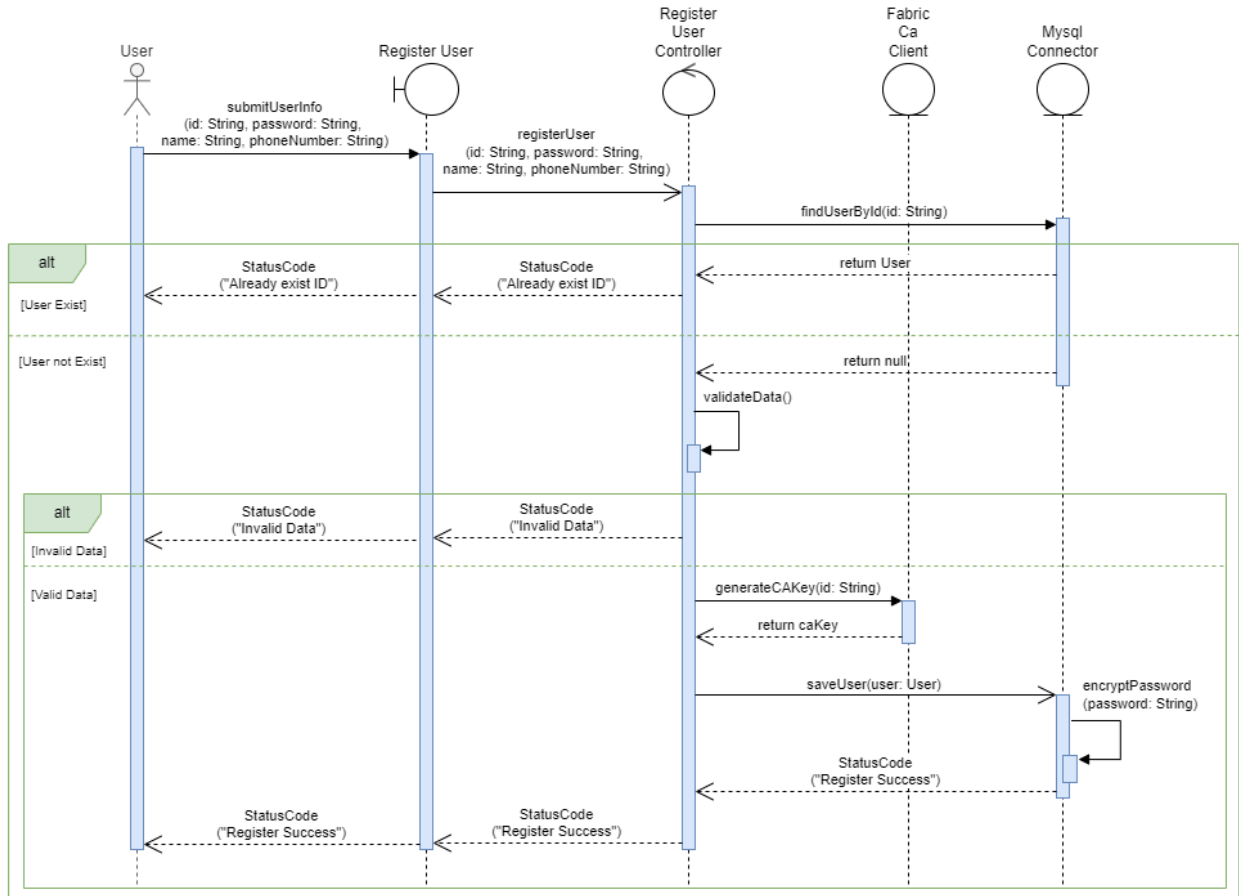


그림 3. 사용자 인증 시퀀스 다이어그램

그림 3은 사용자 인증 기능의 과정을 표현한 시퀀스 다이어그램이다. 사용자가 시스템에 자신의 회원정보를 입력하여 제출하면, 시스템은 사용자가 이미 존재하는 사용자인지 확인한다. 사용자가 존재하는 경우 사용자에게 이미 존재하는 사용자임을 알리는 상태 코드를 반환한다. 사용자가 존재하지 않으면, 입력된 데이터의 유효성을 검증한다. 데이터가 유효하지 않을 경우 사용자에게 데이터가 유효하지 않음을 알리는 상태 코드를 반환한다. 데이터가 유효한 경우, 시스템은 Fabric CA Client를 통해 CA 키를 생성하고, 비밀번호를 암호화하여 사용자 정보를 데이터베이스에 저장한다. 모든 과정이 성공적으로 완료되면, 시스템은 사용자 등록이 성공했음을 알리는 상태 코드를 반환한다.

ii. 데이터 수집 항목 설정

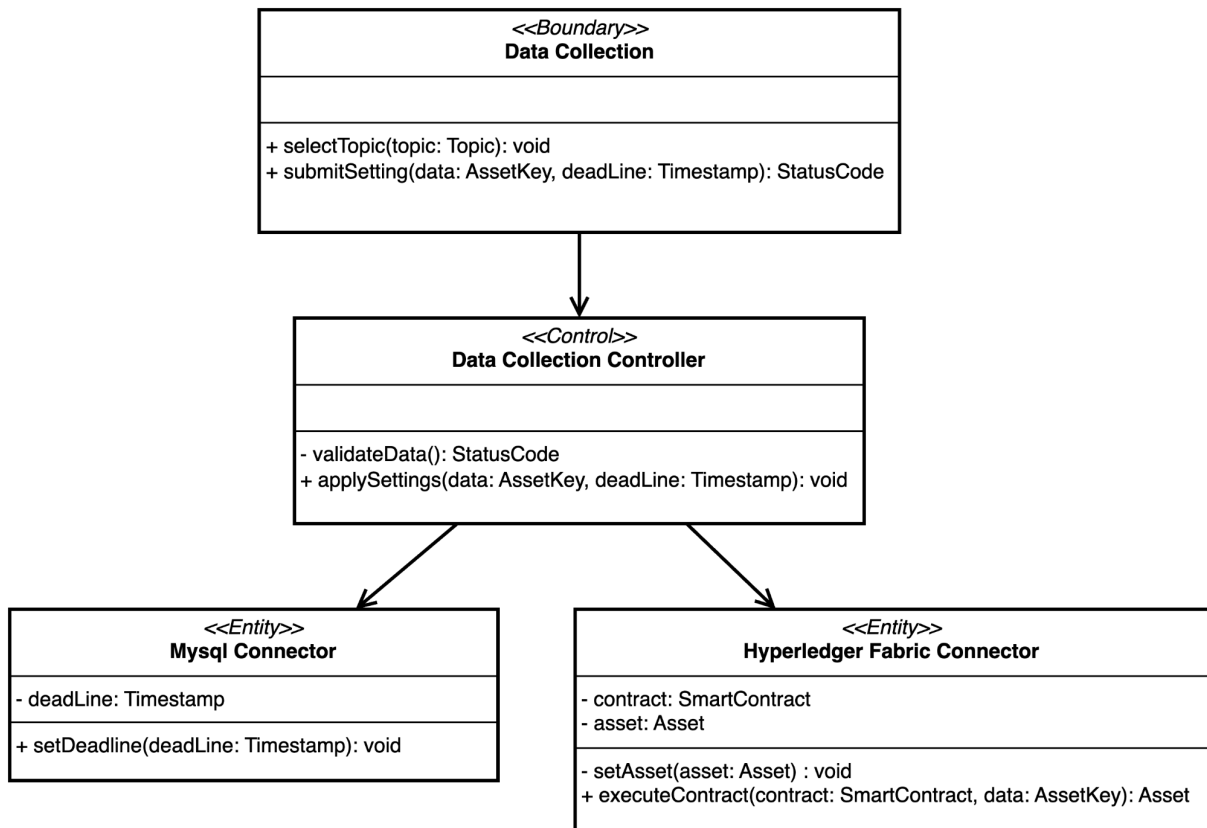


그림 4. 데이터 수집 항목 설정 클래스 다이어그램

그림 4는 데이터 수집 항목 설정 기능을 ECB 패턴을 적용하여 도출한 클래스 다이어그램이다. Data Collection은 수집자가 데이터 수집 항목을 설정하는 Boundary class이다. 수집자는 selectTopic 메서드로 사용자가 수집할 주제를 선택한다. 수집자가 선택한 주제의 설정 항목과 수집 기한을 submitSetting 메서드로 수집 항목(AssetKey)과 마감일(deadLine)을 입력하여 제출한다. 이 메서드는 결과를 상태 코드로 반환한다.

Data Collection Controller는 수집자가 설정한 데이터를 검증하고, 적용하는 역할의 Control class이다. validateData 메서드를 사용해 데이터의 유효성을 검증하고, applySettings 메서드로 검증된 데이터를 사용해 데이터 수집 항목 설정을 적용한다.

Mysql Connector는 데이터 수집 항목의 마감일을 저장하거나 관리하는 Entity class이다, setDeadline 메서드를 사용해 데이터베이스에 마감일을 설정한다.

Hyperledger Fabric Connector는 수집자가 입력한 데이터를 스마트 컨트랙트를 통해 asset으로 저장하는 Entity class이다. executeContract 메서드로 주어진 주제에 맞는 스마트 컨트랙트와 수집자가 수집 항목을 이용해 스마트 컨트랙트를 실행하고, 그 결과로 에셋을 반환한다. 이후 반환한 에셋을 setAsset 메서드를 사용해 저장한다.

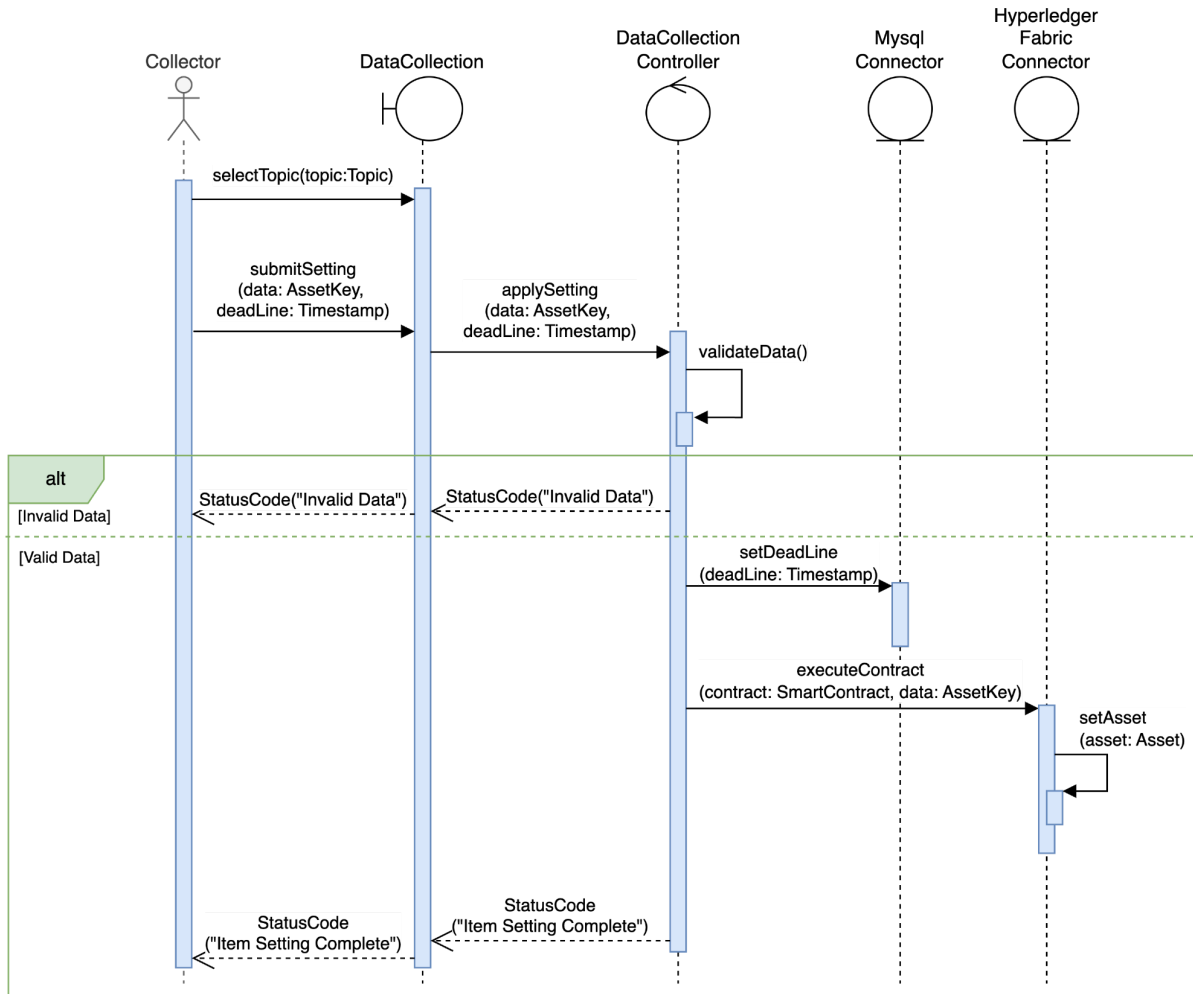


그림 5. 데이터 수집 항목 설정 시퀀스 다이어그램

그림 5는 데이터 수집 항목 설정 기능의 시퀀스 다이어그램이다. 수집자가 수집할 데이터의 주제를 선택하고 수집 항목과 수집 기한을 시스템에 제출하면, 시스템은 수집자가 제출한 데이터의 유효성을 검사한다. 데이터가 유효하지 않을 경우, 수집자에게 유효하지 않은 데이터임을 알리는 상태 코드를 반환한다. 데이터가 유효한 경우 수집자가 입력한 수집 기한을 저장하고, 선택된 주제의 스마트 컨트랙트를 실행해 에셋을 저장한다. 이후, 데이터 수집 항목 설정이 완료되었음을 수집자에게 알리는 상태 코드를 반환한다.

iii. 수집 데이터 입력

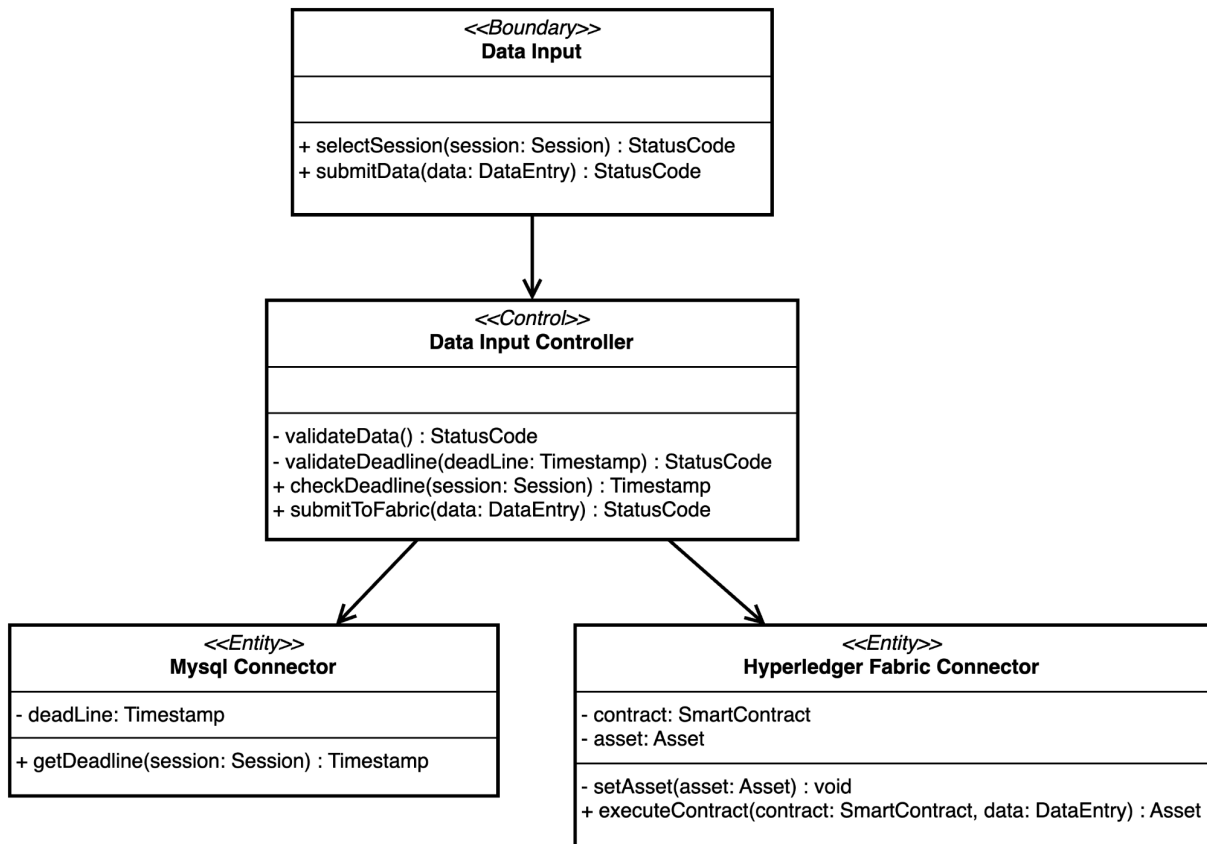


그림 6. 수집 데이터 입력 클래스 다이어그램

그림 6은 수집 데이터 입력 기능을 ECB 패턴을 적용하여 도출한 클래스 다이어그램이다. Data Input은 데이터 수집에 참여할 참여자가 데이터를 입력할 주제를 선택하고, 데이터를 입력하는 Boundary class이다. selectSession 메서드로 수집에 참여할 주제를 선택하고, submitData를 통해 데이터를 제출한다. 두 메서드는 실행 결과로 상태 코드를 반환한다.

Data Input Controller는 참여자가 제출한 데이터를 검증하고, 데이터 수집 기한이 완료되었는지를 검증한다. 또한, Hyperledger Fabric Connector에 받은 데이터를 전달하는 역할의 Control class이다. validateData를 통해 참여자가 제출한 데이터의 무결성을 검증하고 validateDeadline, checkDeadline 메서드들을 통해 참여자가 참여하고자 하는 데이터 수집 주제의 기한을 확인 및 검증한다. 참여자가 제출한 데이터를 submitToFabric 메서드를 통해 Hyperledger Fabric Connector로 전달한다.

Mysql Connector는 데이터 수집의 기한을 관리하는 Entity class이다. 마감 기한 deadLine을 어트리뷰트로 가지며, 이를 반환하는 메서드 getDeadline을 갖는다.

Hyperledger Fabric Connector는 스마트 컨트랙트와 에셋을 관리하는 Entity class이다. 스마트 컨트랙트 contract와 에셋 asset을 어트리뷰트로 가지며 에셋을 설정하는 메서드 setAsset, 스마트 컨트랙트를 실행하는 메서드 executeContract를 가진다.

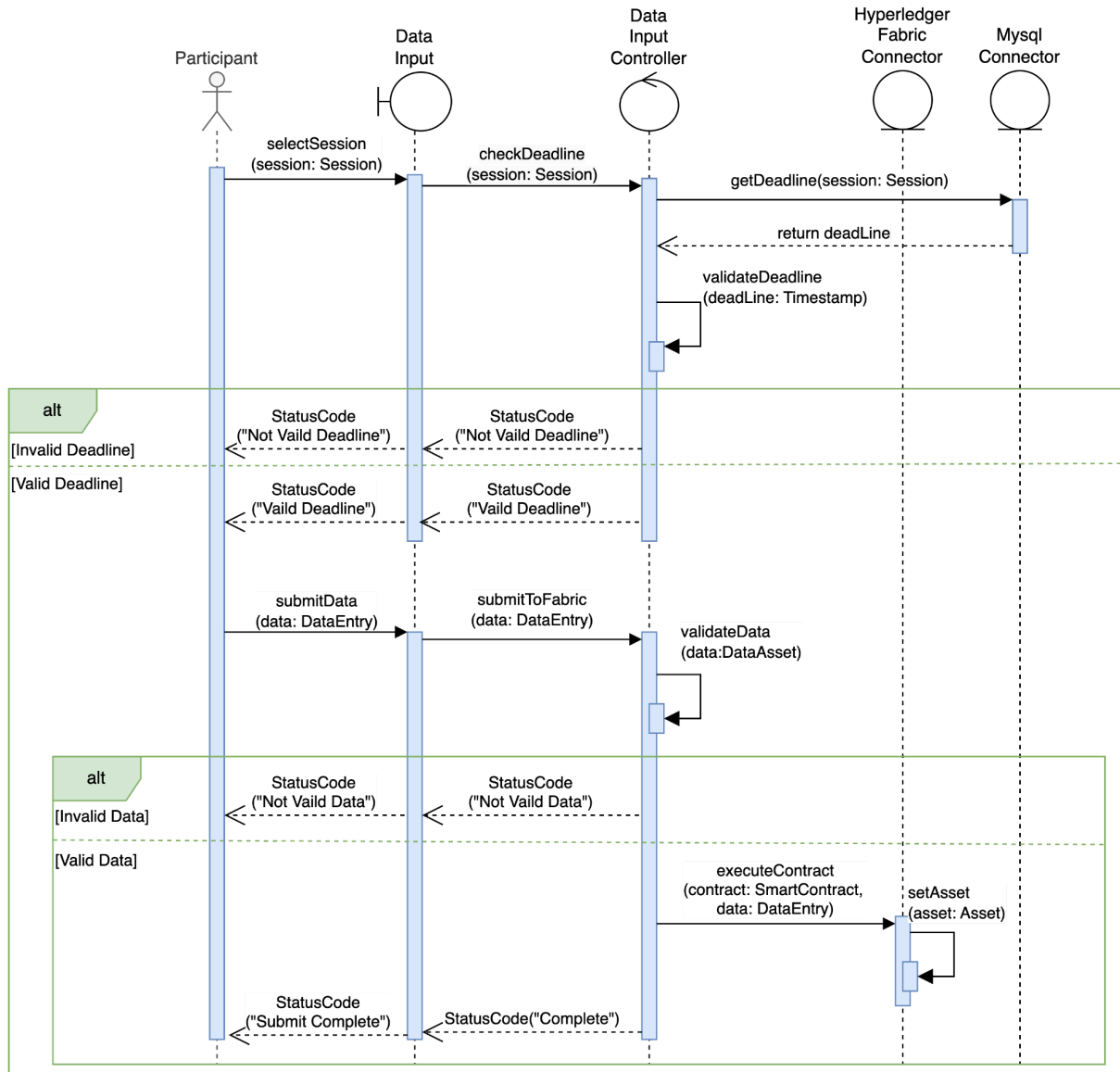


그림 7. 수집 데이터 입력 시퀀스 다이어그램

그림 7은 수집 데이터 입력 기능의 시퀀스 다이어그램이다. 참여자가 수집할 데이터를 입력하기 위해 세션을 선택하면, 시스템은 해당 세션의 마감 기한을 확인한다. 시스템은 Mysql에 저장되어 있는 마감 기한을 조회한다. 마감 기한이 지났을 경우, 시스템은 참여자에게 기한이 유효하지 않음을 알리는 상태 코드를 반환한다. 기한이 유효한 경우 제출 유효 기간이라는 상태 코드를 반환하며, 참여자는 데이터를 제출할 수 있다.

참여자가 데이터를 제출하면 제출된 데이터의 유효성을 검사한다. 데이터가 유효하지 않을 경우, 시스템은 유효하지 않은 데이터임을 알리는 상태 코드를 반환하여, 참여자가 데이터를 다시 제출하도록 요청한다. 데이터가 유효한 경우 시스템은 해당 데이터를 포함한 스마트 컨트랙트를 실행하고 Hyperledger Fabric에 데이터를 저장한다. 저장이 완료되면 제출이 완료되었음을 알리는 상태 코드가 반환된다.

iv. 데이터 모니터링

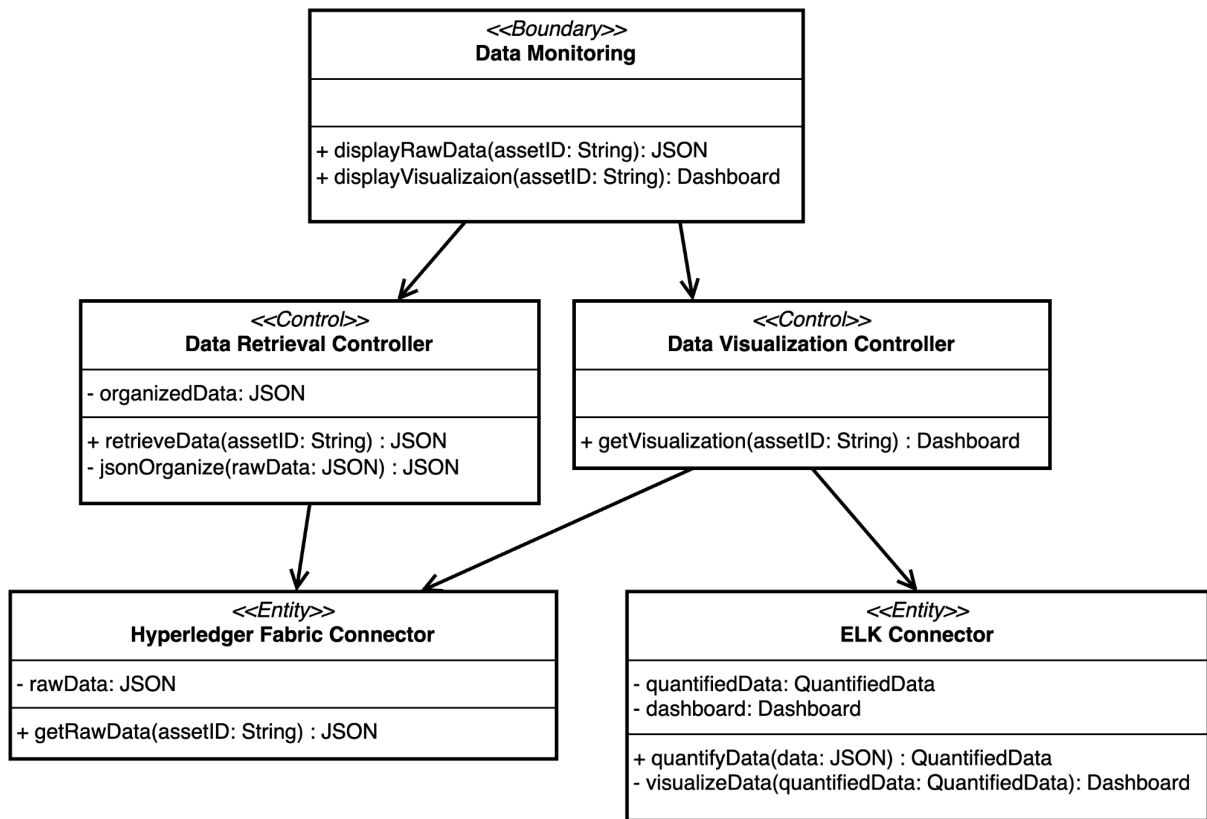


그림 8. 데이터 모니터링 클래스 다이어그램

그림 8은 데이터 모니터링 기능을 ECB 패턴을 적용하여 도출한 클래스 다이어그램이다. Data Monitoring은 수집된 데이터를 모니터링하고 시각화하는 Boundary class이다. 사용자는 `displayRawData` 메서드를 사용하여 원시 데이터를 확인할 수 있다, 사용자는 `displayVisualization` 메서드를 통해 시각화된 대시보드를 확인할 수 있다. 이 메서드는 각각 JSON과 Dashboard 형식의 결과를 반환한다.

Data Retrieval Controller는 수집된 데이터를 조회하고, 그 데이터를 정리하는 역할의 Control class이다. `retrieveData` 메서드를 통해 `assetID`에 해당하는 원시 데이터를 Hyperledger Fabric Connector에서 가져온다, 가져온 원시 데이터를 `jsonOrganize` 메서드를 통해 정리하여 반환한다.

Data Visualization Controller는 데이터를 시각화하는 역할의 Control class이다. `getVisualization` 메서드를 통해 `assetID`에 해당하는 데이터를 기반으로 대시보드를 생성하고 ELK Connector를 통해 시각화된 결과를 얻는다.

Hyperledger Fabric Connector는 수집된 원시 데이터를 Hyperledger Fabric에서 가져오는 Entity class이다. `getRawData` 메서드를 통해 주어진 `assetID`에 맞는 원시 데이터를 조회하고, 그 결과를 JSON 형식으로 반환한다.

ELK Connector는 데이터를 정량화하여 시각화를 수행하는 Entity class이다. `quantifyData` 메서드는 JSON 형식의 데이터를 받아 정량화된 데이터(QuantifiedData)로 변환하며, `visualizeData` 메서드는 정량화된 데이터를 기반으로 대시보드를 생성하여 시각화한 결과를 반환한다.

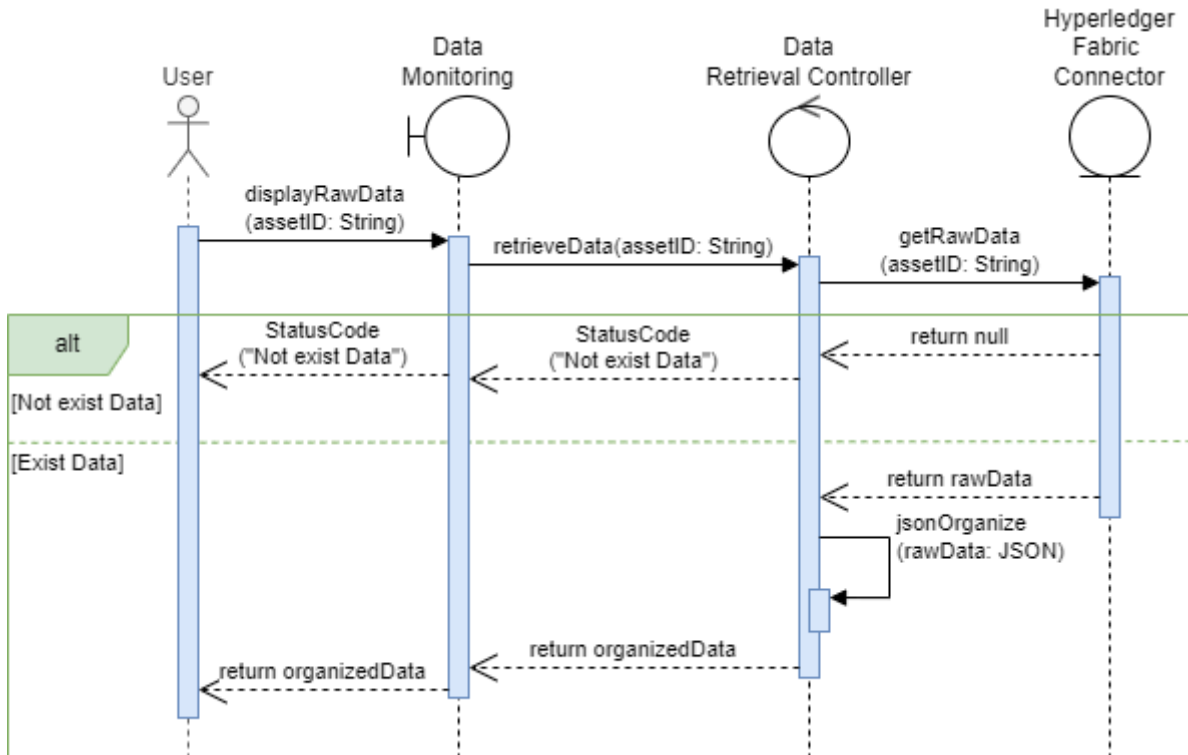


그림 9. 원시 데이터 조회 시퀀스 다이어그램

그림 9는 원시 데이터 조회 기능의 시퀀스 다이어그램이다. 사용자가 조회하고자 하는 원시 데이터의 `assetID`를 입력하고 시스템에 제출하면, 시스템은 해당 `assetID`에 대한 원시 데이터를 조회한다. 이 과정에서 사용자가 입력한 `assetID`의 원시 데이터가 존재하지 않는 경우, 시스템은 데이터가 존재하지 않는다는 상태 코드를 반환한다. 데이터가 존재하는 경우, 시스템은 Hyperledger Fabric에서 해당 원시 데이터를 가져와 데이터를 정리한다. 정리된 데이터는 사용자에게 전달되어, 사용자가 조회된 데이터를 확인할 수 있다.

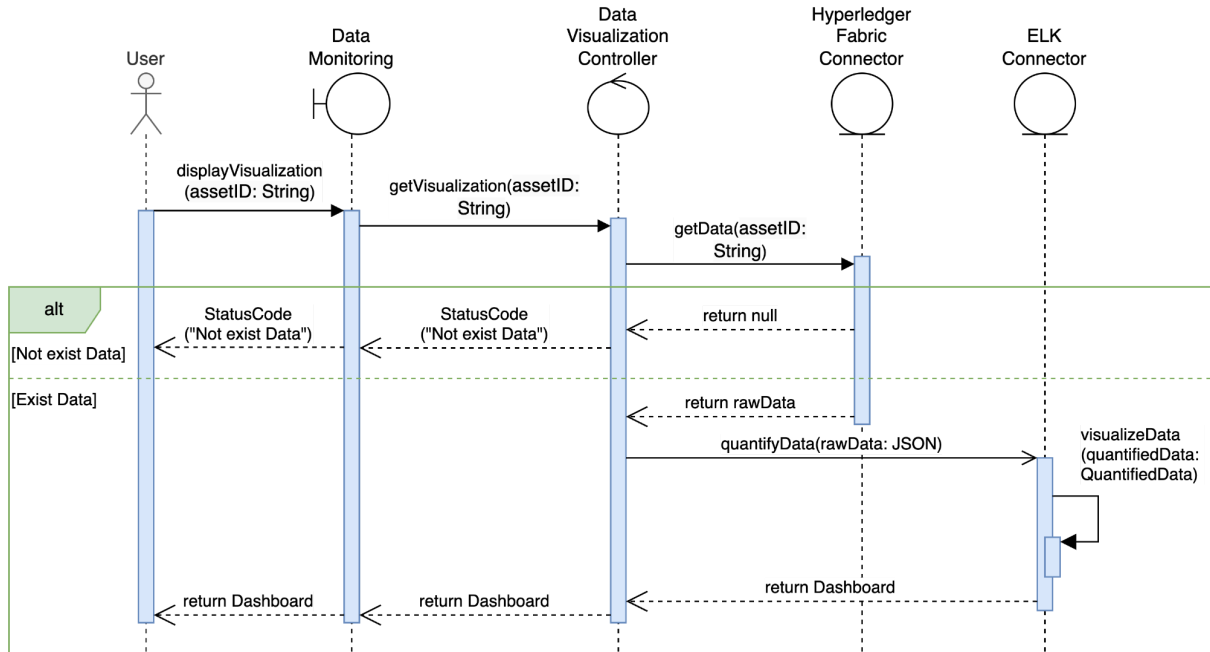


그림 10. 데이터 시각화 시퀀스 다이어그램

그림 10은 데이터 시각화 과정의 시퀀스 다이어그램이다. 사용자가 시각화할 데이터의 시각화를 시스템에 요청하면, 시스템은 해당 assetID에 대한 데이터를 조회한다. 데이터가 존재하지 않을 경우, 시스템은 데이터가 존재하지 않는다는 상태 코드를 반환하여 사용자에게 데이터가 존재하지 않음을 알린다. 데이터가 존재할 경우, 시스템은 Hyperledger Fabric에서 해당 원시 데이터를 가져온다. 가져온 원시 데이터는 ELK로 전달되고, 정량화된다. 이후, 정량화된 데이터를 시각화한 대시보드를 생성해 사용자에게 반환한다.

v. 스마트 컨트랙트 관리

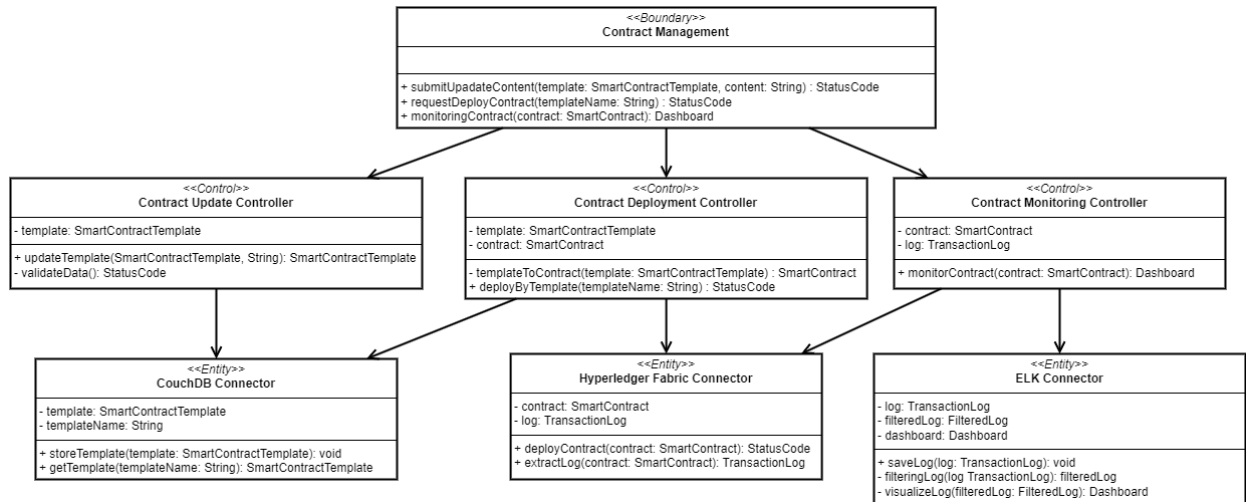


그림 11. 스마트 컨트랙트 관리 클래스 다이어그램

그림 11은 스마트 컨트랙트 관리 기능을 ECB 패턴을 적용하여 도출한 클래스 다이어그램이다. Contract Management는 스마트 컨트랙트를 관리하는 Boundary class이다. submitUpdateContent 메서드를 통해 사용자가 수정된 스마트 컨트랙트를 제출한다, monitoringContract 메서드를 통해 스마트 컨트랙트를 모니터링할 수 있으며 모니터링 결과는 대시보드 형식으로 반환된다.

Contract Update Controller는 스마트 컨트랙트 템플릿을 업데이트하는 역할을 하는 Control class이다. updateTemplate 메서드를 사용해 기존 스마트 컨트랙트 템플릿을 업데이트하고, validateData 메서드를 통해 입력된 데이터의 유효성을 검증한 후 CouchDB Connector에 저장을 요청한다.

CouchDB Connector는 스마트 컨트랙트 템플릿을 저장하고 관리하는 Entity class이다. storeTemplate 메서드로 스마트 컨트랙트 템플릿을 저장하고, getTemplate 메서드로 저장된 템플릿을 조회한다.

Contract Deployment Controller는 스마트 컨트랙트 템플릿을 하이퍼레저 패브릭 네트워크에 배포하는 역할을 수행하는 Control class이다. templateToContract 메서드를 통해 템플릿을 스마트 컨트랙트로 변환한 뒤, 이를 Hyperledger Fabric Connector를 통해 배포한다.

Hyperledger Fabric Connector는 스마트 컨트랙트를 하이퍼레저 패브릭 네트워크에 배포하고, 트랜잭션 로그를 관리하는 Entity class이다. deployContract 메서드를 통해 스마트 컨트랙트를 배포하며, extractLog 메서드를 통해 트랜잭션 로그를 추출한다.

Contract Monitoring Controller는 배포된 스마트 컨트랙트를 모니터링하는 역할을 하는 Control class이다. monitorContract 메서드를 통해 스마트 컨트랙트를 모니터링하며, 이 과정에서 트랜잭션 로그를 관리한다. 모니터링 결과는 ELK Connector를 통해 대시보드로 시각화한다.

ELK Connector는 트랜잭션 로그를 저장하고 필터링하며, 필터링된 로그를 시각화하는 Entity class이다. saveLog 메서드는 트랜잭션 로그를 저장하며, filteringLog 메서드를 통해 전달받은 로그를 필터링한다. visualizeLog 메서드를 통해 필터링된 로그를 대시보드 형식으로 시각화하여 제공한다.

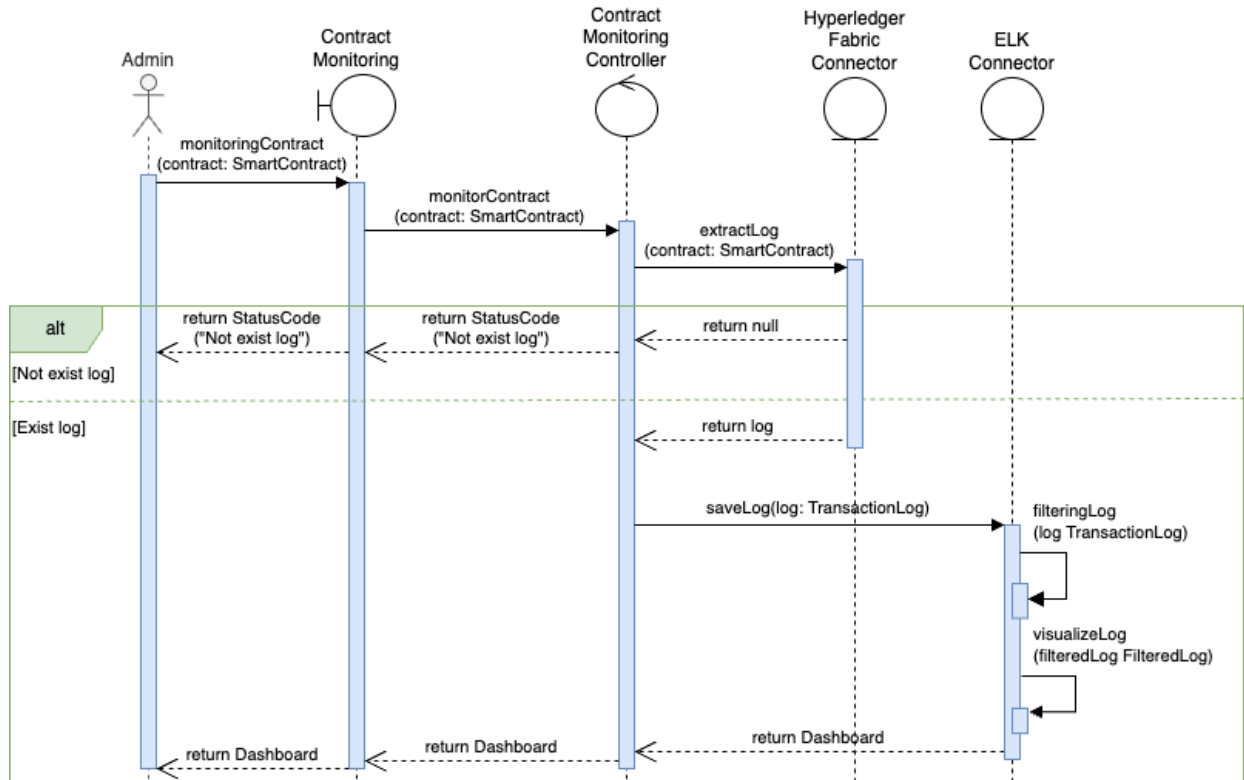


그림 12. 스마트 컨트랙트 모니터링 시퀀스 다이어그램

그림 12는 스마트 모니터링 과정의 시퀀스 다이어그램이다. 사용자는 모니터링할 스마트 컨트랙트를 지정해 시스템에 모니터링 요청을 보낸다. 시스템은 하이퍼레저 패브릭에 지정한 스마트 컨트랙트의 로그의 추출을 요청한다. 추출할 로그가 없는 경우, 로그가 존재하지 않음을 알리는 상태 코드를 사용자에게 반환한다. ELK Connector는 요청받은 트랜잭션 로그를 필터링 후 시각화하여 대시보드 형태로 반환한다. 대시보드는 사용자가 확인할 수 있도록 Data Monitoring Page로 전달된다.

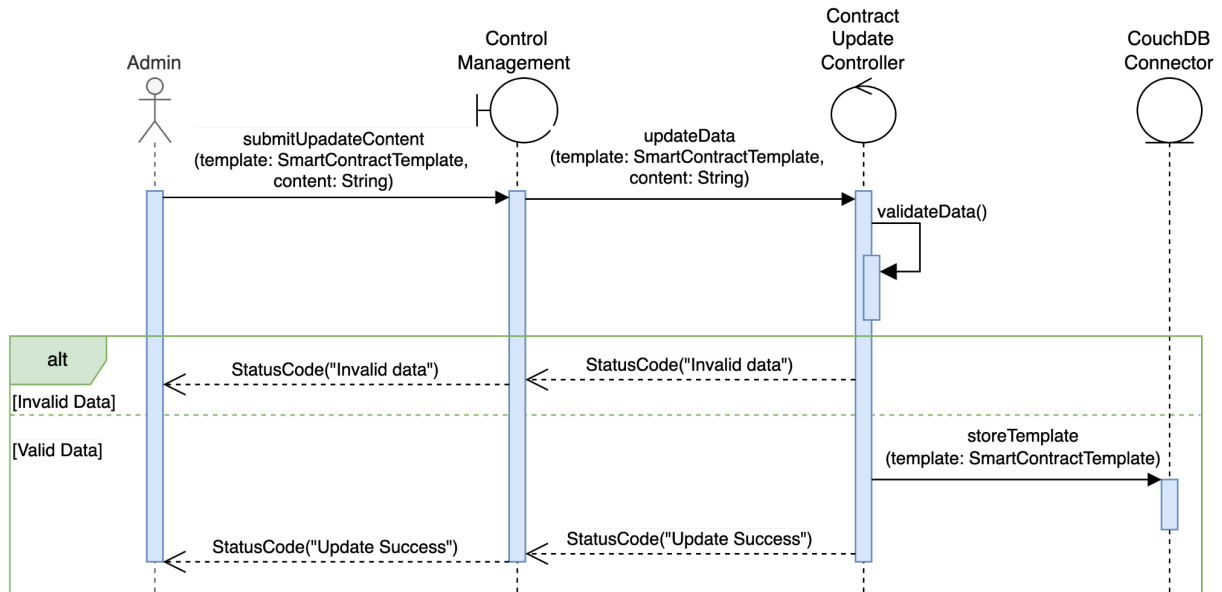


그림 13. 스마트 컨트랙트 업데이트 시퀀스 다이어그램

그림 13은 스마트 컨트랙트 업데이트 과정을 표현한 시퀀스 다이어그램이다. 관리자가 스마트 컨트랙트를 업데이트하기 위해 새로운 템플릿과 내용을 시스템에 제출한다. 이 과정에서 submitUpdateContent 메서드를 호출하며, Control Management는 제출된 데이터를 받아 Contract Update Controller로 전달한다.

Contract Update Controller는 받은 데이터를 validateData 메서드를 통해 검증한다. 검증 과정에서 데이터가 유효하지 않은 경우, 데이터가 유효하지 않음을 알리는 상태 코드가 반환되고 업데이트 과정이 종료된다.

데이터가 유효한 경우 성공을 알리는 상태 코드가 반환된다. Contract Update Controller는 검증된 스마트 컨트랙트 템플릿을 CouchDB Connector에 저장한다. 이 저장 작업은 storeTemplate 메서드를 통해 이루어지며, 저장이 완료된 후, 관리자에게 업데이트가 성공적으로 완료되었음을 알리는 상태 코드가 반환된다.

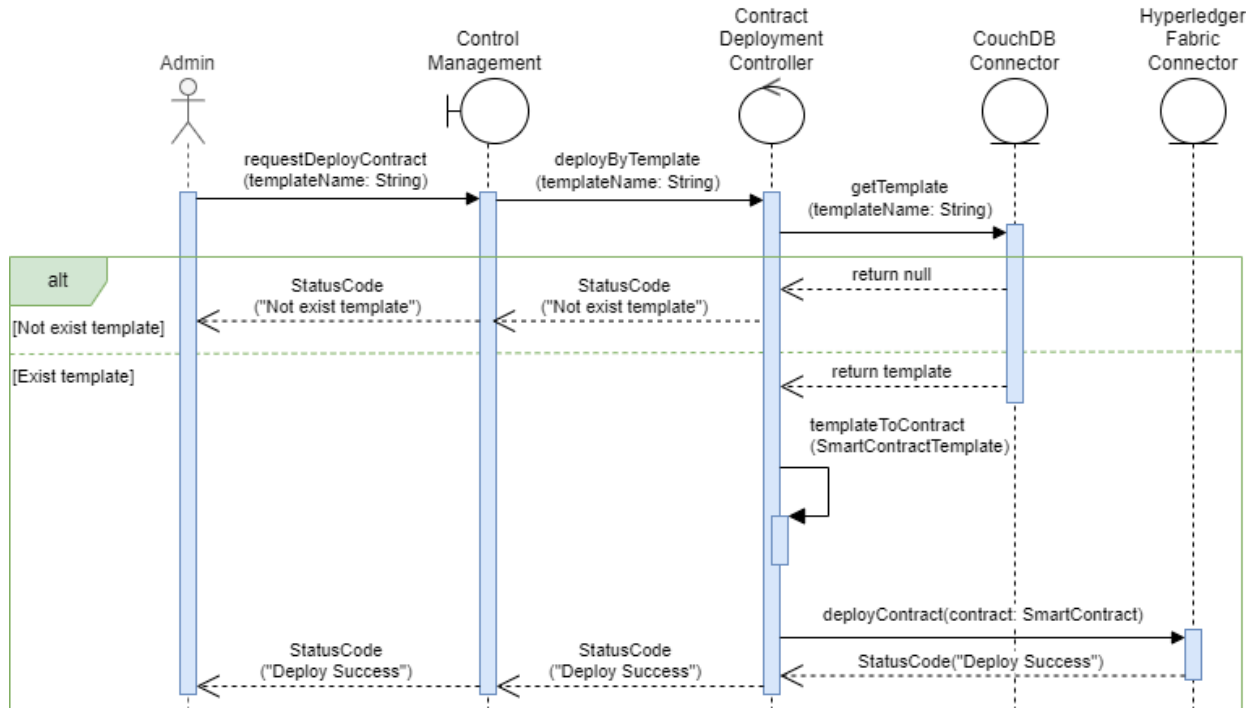


그림 14. 스마트 컨트랙트 배포 시퀀스 다이어그램

그림 14는 스마트 컨트랙트 배포 과정을 표현한 시퀀스 다이어그램이다. 관리자는 배포하고자 하는 스마트 컨트랙트 템플릿의 이름을 시스템에 제출한다. 시스템은 받은 템플릿의 이름에 해당하는 템플릿을 CouchDB 스마트 컨트랙트 풀에서 반환받는다. 해당하는 템플릿이 존재하지 않는 경우, 시스템은 관리자에게 템플릿이 존재하지 않음을 알리는 상태 코드를 반환한다. 해당하는 템플릿을 반환받은 경우 해당 템플릿은 하이퍼레저 패브릭 네트워크상에 배포할 수 있는 스마트 컨트랙트로 변환된 후 배포된다. 배포가 완료되면 시스템은 관리자에게 성공적으로 배포가 완료되었음을 알리는 상태 코드를 반환한다.

5. 갱신된 과제 추진 계획

표 2. 갱신된 개발 일정

수행내용	기간	7월			8월				9월				10월	
		3	4	5	1	2	3	4	1	2	3	4	1	2
1. 스마트 컨트랙트 템플릿 구현														
2. 스마트 컨트랙트 풀 저장소 구현														
3. 스마트 컨트랙트 네트워크 배포 자동화 구현														
4. 스마트 컨트랙트 모니터링 구현														
5. 스마트 컨트랙트 업데이트 구현														
6. 패브릭 CA이용 사용자 인증 구현														
7. 수집 데이터 분석 및 시각화 구현														
8. 애플리케이션 구현(UI, 웹 서버)														
9. 중간보고서 작성														
10. 사용자 인증 연동 구현														
11. 샘플 데이터 생성 및 테스트														
12. 서비스 테스트 및 보완														
13. 최종보고서 작성														

표 2는 실험을 진행하며, 수행내용을 구체화하여 개발 일정을 주차별로 정리한 것이다.

6. 구성원별 진척도

표 3. 구성원별 진척도

이름	구현 내역
김민중	<ul style="list-style-type: none"> Go 언어를 사용하여 데이터 수집 이벤트에 사용되는 스마트 컨트랙트 구현 서버와 CouchDB의 연동을 통해 UI에서 템플릿을 수정할 수 있는 기능인 스마트 컨트랙트 업데이트 구현 HTML을 사용해 구현한 기능들의 동작을 확인하기 위한 데이터 수집 플랫폼 프로토타입 구현 데이터 수집, 저장, 조회 등 전반적인 백엔드 로직을 관리하는 서버 구현
박형주	<ul style="list-style-type: none"> 스마트 컨트랙트 템플릿에서 반복적으로 사용되는 공통 에셋들의 재사용성을 높일 수 있는 스마트 컨트랙트 공통 에셋 템플릿화 CouchDB를 사용해 스마트 컨트랙트 템플릿을 관리하는 스마트 컨트랙트 템플릿 풀 구현 사용자의 회원가입 시 기입 정보 및 CA에서 발급받은 인증서를 저장하는 사용자 정보 데이터베이스 구현 HTML을 사용해 구현한 기능들의 동작을 확인하기 위한 데이터 수집 플랫폼 프로토타입 구현
전원균	<ul style="list-style-type: none"> 하이퍼레저 패브릭 네트워크의 조직과 각 조직별 피어 배치 등 네트워크의 전반적인 구조를 설계하여 하이퍼레저 패브릭 네트워크 구현 하이퍼레저 패브릭 네트워크의 초기 설정과 구성 과정을 네트워크 배포 스크립트로 작성하여 네트워크 배포 자동화 구현 스마트 컨트랙트의 설치, 승인, 커밋 과정을 자동으로 진행하는 기능인 스마트 컨트랙트 배포 자동화 구현

표 3은 해당 과제의 구성원별 진척도를 나타낸 표이다.

7. 과제 수행 내용 및 중간 결과

1) 스마트 컨트랙트 템플릿 구현

수집하고자 하는 데이터의 형식에 따라 템플릿을 분류하였다. 수집 데이터의 형태는 수치 데이터, 득표수 데이터, 줄글 데이터로 분류하고 주제별로 사용자 설정 package를 생성하여 수집자가 템플릿을 import하여 필요에 맞게 템플릿을 수정할 수 있다.

```

5 // CommonAttributes 구조체는 여러 템플릿에서 공통으로 사용
6 type CommonAttributes struct {
7     ID          string `json:"id"`          // 사용자가 로그인 시 사용되는 고유한 ID
8     Name        string `json:"name"`        // 사용자의 이름
9     Age         int    `json:"age"`         // 사용자의 나이
10    Region       string `json:"region"`       // 데이터를 수집할 때 필요한 지역 정보
11    Gender       int    `json:"gender"`       // 사용자의 성별 (남성: 0, 여성: 1)
12    VoteCount    int    `json:"voteCount"`    // 1씩 증가시킬 수 있는 득표수 변수
13 }
```

그림 15. 공통적으로 사용되는 참여자 에셋 구성 코드

그림 15는 템플릿에서 공통적으로 들어가는 참여자 에셋 구조체이다. 각 템플릿별로 사용되는 에셋의 공통적인 부분을 묶어 공통 템플릿을 추가로 생성하였다. 공통 템플릿은 참여자의 아이디(id), 이름(name), 나이(age) 성별(gender), 지역(region)과 같은 참여자에 관련된 데이터를 에셋으로 구성하였다. 이를 통해 수집 템플릿에서 공통 템플릿을 import하여 재사용할 수 있도록 하였다.

```

candidate := Candidate{
    CandidateNumber: candidateNumber,    // 투표 후보 기호
    CandidateName:   candidateName,      // 투표 후보 이름
    CommonAttributes: common.CommonAttributes{
        VoteCount: 0,                    // 득표 : 0
    },
}
```

그림 16. 투표 형태 데이터의 수집자 생성 에셋 구성 코드

그림 16은 데이터 수집 항목 설정 기능에서 수집자가 투표 형태의 데이터를 수집하게 될 경우 설정할 항목의 에셋 구성이다. 수집자는 투표 후보 기호(CandidateNumber), 투표 후보 이름(CandidateName)을 설정하면 해당 후보의 득표(VoteCount)는 자동으로 0으로 초기화된다.

```

voter := Voter{
    // common의 에셋을 재사용
    CommonAttributes: common.CommonAttributes{
        ID:      id,
        Name:    name,
        Age:     age,
        Gender:  gender,
        Region:  region,
    },
    // candidate의 에셋을 재사용
    Candidate: Candidate{
        CandidateNumber: candidateNumber,
    },
}

```

그림 17. 투표 형태 데이터의 참여자 생성 에셋 구성 코드

그림 17은 참여자가 수집 데이터 입력 기능에서 투표 형태의 데이터를 입력할 때 사용되는 에셋의 구성이다. 에셋은 참여자와 관련된 공통 데이터들과 투표 번호(CandidateNumber) 데이터로 이루어져 있다. 참여자가 수집 데이터 입력을 마친 경우 해당 에셋의 형태로 하이퍼레저 패브릭 네트워크에 저장된다. 저장되는 과정에서 참여자가 입력한 투표 후보 기호에 해당하는 candidate 에셋을 검색해 득표 변수를 1 증가시키는 로직을 추가하였다.

```

temperature := Temperature{
    CommonAttributes: common.CommonAttributes{ // 데이터 수집에 사용되는 공통 항목들
        ID:      id,
        Name:    name,
        Region:  region, // 온도 데이터에 해당하는 지역
    },
    NumericData: numericData, // 숫자 데이터(예시: 온도 데이터)
    Timestamp:   timestamp,   // 시간 데이터(예시: 온도 데이터에 대응하는 시간)
}

```

그림 18. 숫자 형태 데이터의 참여자 생성 에셋 구성 코드

그림 18은 숫자 형태의 데이터를 수집할 때 사용되는 에셋의 구성이다. 숫자 형태의 데이터를 온도 데이터로 구체화하여 템플릿을 구현하였다. 온도 데이터를 선택한 이유는 시간 데이터(Timestamp)를 추가해 연속적으로 변하는 숫자 데이터의 흐름을 히스토그램으로 시각화하여 표현할 수 있기 때문이다.

온도 데이터 에셋의 구성은 참여자와 관련된 공통 데이터들과 숫자 데이터(NumericData), 시간 데이터(Timestamp)로 구성되어 있다.

```

surveyItem := SurveyItems{
  SurveyQuestionNumber: surveyQuestionNumber, // 설문 질문의 고유한 ID
  SurveyQuestionContent: surveyQuestionContent, // 설문 질문 내용
  SurveyAnswer:        "", // 질문을 생성하는 과정이기 때문에 답변 항목은 빈칸으로 저장
}

```

그림 19. 줄글 형태 데이터의 수집자 생성 예셋 구성 코드

그림 19는 수집자가 줄글 형태의 데이터의 수집 시 설정하는 예셋의 구성이다. 줄글 형태의 데이터를 설문 데이터로 구체화하여 템플릿을 구현하였다. 설문 데이터를 선택한 이유는 참여자가 제출할 줄글 형태의 데이터의 범위를, 수집자가 설정하는 설문의 질문 내용을 통해 대략적으로 좁힐 수 있기 때문이다. 수집자는 데이터 수집 항목 설정 기능을 통해 설문 질문의 고유 ID(SurveyQuestionNumber), 설문 질문 내용(SurveyQuestionContent)을 설정해 예셋을 생성한다. 예셋 생성 시 설문 답변(SurveyAnswer)은 빈 문자열로 자동 초기화된다.

```

surveyItem := SurveyItems{
  SurveyQuestionNumber: surveyQuestionNumber, // 설문 질문의 고유한 ID
  SurveyAnswer:        surveyAnswer,         // 설문에 대한 응답을 저장
}

participant := SurveyParticipant{
  CommonAttributes: common.CommonAttributes{ // 공통으로 사용되는 예셋 재사용
    ID:    id,
    Name:  name,
    Age:   age,
    Region: region,
    Gender: gender,
  },
  SurveyItems: surveyItem,
}

```

그림 20. 줄글 형태 데이터의 참여자 생성 예셋 구성 코드

그림 20은 줄글 형태의 데이터를 제출하는 참여자가 생성하는 예셋의 구성을 나타낸 그림이다. 설문 데이터 예셋의 구성은 참여자와 관련된 공통 데이터들과 설문 질문의 고유 ID(SurveyQuestionNumber), 설문 응답(SurveyAnswer)이 있다.

2) 스마트 컨트랙트 풀 구현

스마트 컨트랙트 템플릿을 저장하기 위해 CouchDB를 사용하여 스마트 컨트랙트 풀을 구현하였다.








	_id ▼	content ▼	file_path ▼	type ▼
<input type="checkbox"/>	 common	package common import "githu...	common/common.go	asset configure
<input type="checkbox"/>	 survey	package survey import ("smart...	survey/survey.go	asset configure
<input type="checkbox"/>	 survey_contract	package survey import ("enco...	survey/survey_contract.go	contract
<input type="checkbox"/>	 temperature	package temperature import "s...	temperature/temperature.go	asset configure
<input type="checkbox"/>	 temperature_contract	package temperature import ("...	temperature/temperature_contr...	contract
<input type="checkbox"/>	 vote	package vote import "smartcon...	vote/vote.go	asset configure
<input type="checkbox"/>	 vote_contract	package vote import ("encodin...	vote/vote_contract.go	contract

그림 21. 스마트 컨트랙트 풀에 저장되는 템플릿 테이블

그림 21은 구현한 스마트 컨트랙트 풀의 테이블 구조를 CouchDB Fauxton을 통해 나타낸 그림이다. _id 행은 템플릿들의 고유한 이름을 나타내는 행이다. content 행은 템플릿들의 코드 내용들이 저장된 행이다. file_path 행은 중복된 데이터는 재사용할 수 있도록 템플릿을 구현하였기 때문에 import 할 수 있는 경로를 저장하는 행이다. type 행은 템플릿이 에셋 정의 템플릿, 컨트랙트 템플릿으로 구분되어 있어 각 템플릿의 타입을 나타내는 행이다. 스마트 컨트랙트 풀에는 데이터 수집 주제 별 에셋 정의 문서와 데이터 수집 플랫폼에서 호출할 수 있는 함수들이 작성되어 json 형식으로 저장되어 있다.

2024 전기 졸업과제

시스템 관리자는 웹을 통해 스마트 컨트랙트 풀에 존재하는 템플릿들을 확인하고 수정하여 업데이트할 수 있도록 구현하였다. 스마트 컨트랙트 풀에서 불러온 템플릿을 화면에 출력하고 관리자가 템플릿 코드를 수정하면 변동 내용이 스마트 컨트랙트 풀에 저장된다.



그림 22. 스마트 컨트랙트 풀에 존재하는 템플릿을 웹 브라우저에서 조회한 화면 출력

그림 24는 관리자 기능 중 하나인 스마트 컨트랙트 템플릿 업데이트 화면이다. 스마트 컨트랙트 템플릿 코드에서 func 문자열을 찾아 코드를 함수별로 구분해서 출력한다. 웹 상에서 바로 수정이 가능하며 수정된 템플릿 코드가 스마트 컨트랙트 풀에 저장된다.

3) 스마트 컨트랙트 네트워크 배포 자동화

```

peer0.org1에 체인코드 설치 중
2024-08-22 22:52:39.703 UTC 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200 payload:"\nJtestLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c\022\ttestLabel" >
2024-08-22 22:52:39.704 UTC 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: testLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c
peer1.org1에 체인코드 설치 중
2024-08-22 22:52:39.885 UTC 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200 payload:"\nJtestLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c\022\ttestLabel" >
2024-08-22 22:52:39.885 UTC 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: testLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c
peer0.org2에 체인코드 설치 중
2024-08-22 22:52:40.083 UTC 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200 payload:"\nJtestLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c\022\ttestLabel" >
2024-08-22 22:52:40.083 UTC 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: testLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c
peer1.org2에 체인코드 설치 중
2024-08-22 22:52:40.258 UTC 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200 payload:"\nJtestLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c\022\ttestLabel" >
2024-08-22 22:52:40.259 UTC 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: testLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c
peer0.org3에 체인코드 설치 중
2024-08-22 22:52:40.440 UTC 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200 payload:"\nJtestLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c\022\ttestLabel" >
2024-08-22 22:52:40.440 UTC 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: testLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c
peer1.org3에 체인코드 설치 중
2024-08-22 22:52:40.612 UTC 0001 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Installed remotely: response:<status:200 payload:"\nJtestLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c\022\ttestLabel" >
2024-08-22 22:52:40.613 UTC 0002 INFO [cli.lifecycle.chaincode] submitInstallProposal -> Chaincode code package identifier: testLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c
Package ID: testLabel:454aaa6d71f49b9214aef8ef8cb9185788a6e2c96c72d4970805189ead5cb95c

```

그림 21. 스마트 컨트랙트 설치 자동화 실행 출력화면

그림 21은 각 조직과 해당 조직에 속한 피어들에 체인코드가 자동으로 설치되는 과정을 보여주는 화면이다. 이 과정은 하이퍼레저 패브릭 네트워크에서 체인코드를 배포하는 첫 번째 단계로, 각 조직 내 피어들이 체인코드를 실행할 수 있도록 하기 위한 필수 절차이다. 구체적으로, 그림 21에서는 3개의 조직(org1, org2, org3) 각각에 속한 피어(peer0, peer1)들에 체인코드를 설치하는 과정이 자동화된 스크립트를 통해 실행되는 모습을 확인할 수 있다. 이 과정에서는 체인코드의 패키징 작업도 함께 수행된다. 설치 과정이 성공적으로 완료되면 설치된 체인코드의 식별자와 함께 상태 코드가 출력되어 각 피어에 체인코드가 제대로 설치되었음을 확인할 수 있다.

```
Org1 체인코드 승인 중
2024-08-22 22:52:42.904 UTC 0001 INFO [chaincodeCmd] ClientWait -> txid [8ff253e75d5376e79706d28e1388c14c1b84d16
aa6dc59581bf5c256c034252c] committed with status (VALID) at peer0.org1.data-collector.com:7051
Org2 체인코드 승인 중
2024-08-22 22:52:45.026 UTC 0001 INFO [chaincodeCmd] ClientWait -> txid [61fc1713fae494bb4a197fef1772d4a2bcf805a
4c65b31bf1c79474a8736cdea] committed with status (VALID) at peer0.org2.data-collector.com:9051
Org3 체인코드 승인 중
2024-08-22 22:52:47.190 UTC 0001 INFO [chaincodeCmd] ClientWait -> txid [f0a448a24161c55e425448bf5222c92359143ff
611b5528103c928736242f218] committed with status (VALID) at peer0.org3.data-collector.com:11051
```

그림 22. 스마트 컨트랙트 승인 자동화 실행 출력 화면

그림 22는 앞서 설치된 체인코드가 각 조직에서 자동으로 승인되는 과정을 보여주는 화면이다. 각 조직의 승인 없이는 체인코드가 네트워크에서 사용될 수 없으므로 체인코드는 승인 과정을 거쳐야 한다. 승인 과정에서는 각 조직이 체인코드를 검토하고, 이를 승인함으로써 체인코드를 사용할 수 있도록 허가한다. 그림에서는 각 조직(org1, org2, org3)에서 체인코드가 어떻게 승인되는지, 그리고 그 결과가 상태 코드 형태로 출력되는 모습을 볼 수 있다. 승인 과정에서 트랜잭션 ID와 함께 승인 여부가 기록되며, 성공적으로 승인된 경우 해당 조직의 모든 피어가 체인코드를 사용할 수 있게 된다.

```
커밋 중
2024-08-22 22:52:49.365 UTC 0001 INFO [chaincodeCmd] ClientWait -> txid [030bcc9d1ec715fcc4fec303c99d1727c8ae200
44f32d5eccc770ba8ad9e4e76] committed with status (VALID) at peer0.org2.data-collector.com:9051
2024-08-22 22:52:49.365 UTC 0002 INFO [chaincodeCmd] ClientWait -> txid [030bcc9d1ec715fcc4fec303c99d1727c8ae200
44f32d5eccc770ba8ad9e4e76] committed with status (VALID) at peer0.org1.data-collector.com:7051
2024-08-22 22:52:49.367 UTC 0003 INFO [chaincodeCmd] ClientWait -> txid [030bcc9d1ec715fcc4fec303c99d1727c8ae200
44f32d5eccc770ba8ad9e4e76] committed with status (VALID) at peer0.org3.data-collector.com:11051
커밋 결과:
Committed chaincode definition for chaincode 'test' on channel 'mychannel':
Version: 1.0, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [Org1MSP: true, Org2MSP
: true, Org3MSP: true]
```

그림 23. 스마트 컨트랙트 커밋 및 커밋 결과 출력 화면

그림 23은 체인코드가 각 조직의 앵커 피어에 커밋되는 과정을 보여주는 화면이다. 커밋 과정은 체인코드가 최종적으로 네트워크에 적용되는 단계로, 이를 통해 체인코드가 네트워크 상에서 실행 가능한 상태가 된다. 그림 23에서는 체인코드가 각 조직(org1, org2, org3)의 앵커 피어(각 조직의 peer 0)에 자동으로 커밋되는 과정이 나타나 있다. 커밋이 완료되면, 배포된 체인코드의 목록과 세부 정보가 출력되며, 각 조직에서 체인코드가 승인되었는지 여부도 함께 표시된다. 이 정보를 통해 체인코드가 모든 조직에서 올바르게 승인되고 커밋되었음을 확인할 수 있으며, 이후에는 네트워크 상에서 실제로 체인코드를 실행하고 트랜잭션을 처리할 수 있게 된다.

8. 향후 계획

1) 데이터 시각화 구현

ELK를 활용하여 데이터 시각화를 진행할 것이다. 투표 데이터는 원형 차트, 숫자 데이터는 히스토그램, 줄글 데이터는 워드 클라우드 대시보드를 활용할 예정이다.

2) 수집 항목 피드백 구현

수집 항목 피드백 부분에서는 데이터 수집 참여자들이 수집된 항목에 대한 피드백을 제공할 수 있는 기능을 구현할 것이다. 피드백의 내용은 점수와 줄글 형식 데이터로 구성할 것이다. 제출된 피드백은 데이터베이스에 저장되며 관리자가 조회하여 스마트 kontrak트의 업데이트나 시스템 개선에 사용할 수 있다.

3) 실제 데이터 수집 및 처리 과정 검증

IoT(Internet of Things) 디바이스나 가상 머신을 활용하여 시뮬레이션 데이터를 생성 및 측정하고 측정한 데이터들을 시스템을 통해 수집하는 테스트를 진행할 것이다. IoT 디바이스에서 온도, 습도 등의 환경 데이터를 수집하여 네트워크에 전송하거나, 가상 머신에서 시뮬레이션된 데이터를 사용하여 네트워크의 성능을 테스트할 수 있다. 이 과정에서 데이터가 손실 없이 정확하게 전송되고, 네트워크 내에서 적시에 처리되며, 수집된 데이터의 진위성을 보장하는지 여부를 검증할 것이다.

4) 사용자 인터페이스 개선

HTML 및 CSS를 사용하여 구현한 데이터 수집 시스템 프로토타입의 사용자 인터페이스를 리액트를 사용하여 새롭게 구현할 것이다.