2019

Used2Block

DESIGN SPECIFICATION

Student Number	Name
2015314213	이재원 Jae Won Lee
2013313737	김무성 Mu Sung Kim
2015318385	김현우 Hyun Woo Kim
2015310250	김동환 Dong Hwan Kim
2016315146	이상우 Sang Woo Lee

0. Table of Contents

1. Preface

1.1. Objective

The readership of Design Specification is defined. The structure of the entire document is specified.

1.2. Readership

The document is targeting software engineers who develop and maintain Used2Block application, system architecture engineers who design the entire system and engineers who are in charge of customer service

.

1.3. Document Structure

1.3.1. Preface

In this section, the readership is defined as well as the structure of the entire document.

1.3.2. Introduction

In this section, Unified Modeling Language (UML) is used in designing the system. Many diagrams are used.

1.3.3. System Architecture

In this section, rather high-level abstraction of system functionalities is specified for the target system. For visual representation, many diagrams are used.

1.3.4. Existing System

In this section, the analysis and some diagrams to help understand environment and context where our idea will be are placed before the illustrations for the what we will make. This will set the boundary between our system and related outer system which already exist.

1.3.5. Products Management System

In this section, design for the first sub-system named 'Product Management System' is located. This is where our expected users will be serviced with enrolling request function and searching function to find out some used products. For visual representation, many diagrams are used.

1.3.6. User Management System

In this section, the second sub-system, 'User Management System' is illustrated with designed UML diagrams. This will help the readers understand our main functions like sign-in/out, sign-up and change user information. This sub-system will work at the front of users who use Used2Block service.

1.3.7. Transaction Validation System

In this section, the most special and discriminative function in Used2Block system is described. The sub-system 'Transaction Validation' is about how a transaction is validated by the 3rd party of it named 'Validator'. This is illustrated with UML diagrams with detailed statements.

1.3.8. Protocol Design

In this section, it explains a list of protocols that the sub-systems must follow. The communication messages are explained in terms of type, purpose and meaning.

1.3.9. Database Design

In this section, the database is designed based on Requirement Specification. ER Diagram is illustrated. The Relation Schema is demonstrated as well.

1.3.10. Testing Plan

In this section, the plan is built based on the scenarios written in the Requirement Specification to make sure the system is working as planned. The scenarios are from our requirements specification documents like "서비스 가입/접속/접속_종료 시나리오", "물품 등록/관리/검색 시나리오" and "거래 시나리오". The tests are done to prevent errors way beforehand. The policies for testing are clarified and designed. In each testing plan, the policies and test cases are demonstrated.

1.3.11. Development Environment [동환님, 적어 주신 부분과 이 부분 확인 및 수정 부탁드립니다]

In this section, programming environment and policies are specified.

1.3.12. Development Plan

In this section, schedule for the development is illustrated using Gantt Chart.

1.3.13. Index

In this section, indices for the entire diagrams, charts and figures in this document are listed.

1.3.14. References

In this section, all references are listed in order.

2. Introduction

2.1 Objective

In Introduction, Unified Modeling Language (UML) is used in designing the system. Many diagrams are used.

2.2 Applied Diagram

Used2Block uses the following diagrams – Deployment Diagram, Class Diagram, State Diagram, Sequence Diagram and ER Diagram.

2.2.1 Context Diagram

Figure 1. Context Diagram Example for Online Community. (Chris Adams, 2019).

Context diagram can be used to show the relationship between actors and the system from the widest view. The system is on the center and outer systems and actors are placed around it. Arrows can describe data flow or other interaction.

2.2.2 Activity Diagram

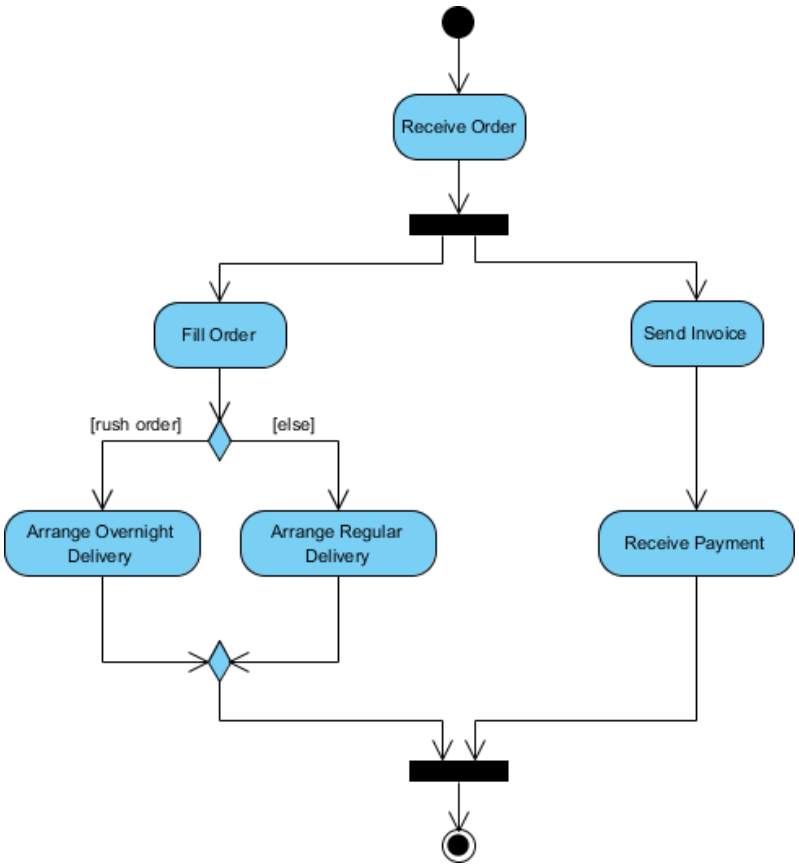


Figure 2. Activity Diagram Example for Student Enrollment. (Visual Paradigm, 2019).

The purpose of this diagram is to show the activities of the system. Here, nodes and communication paths are use to show the flow, and each node represent an activity.

2.2.3 Class Diagram

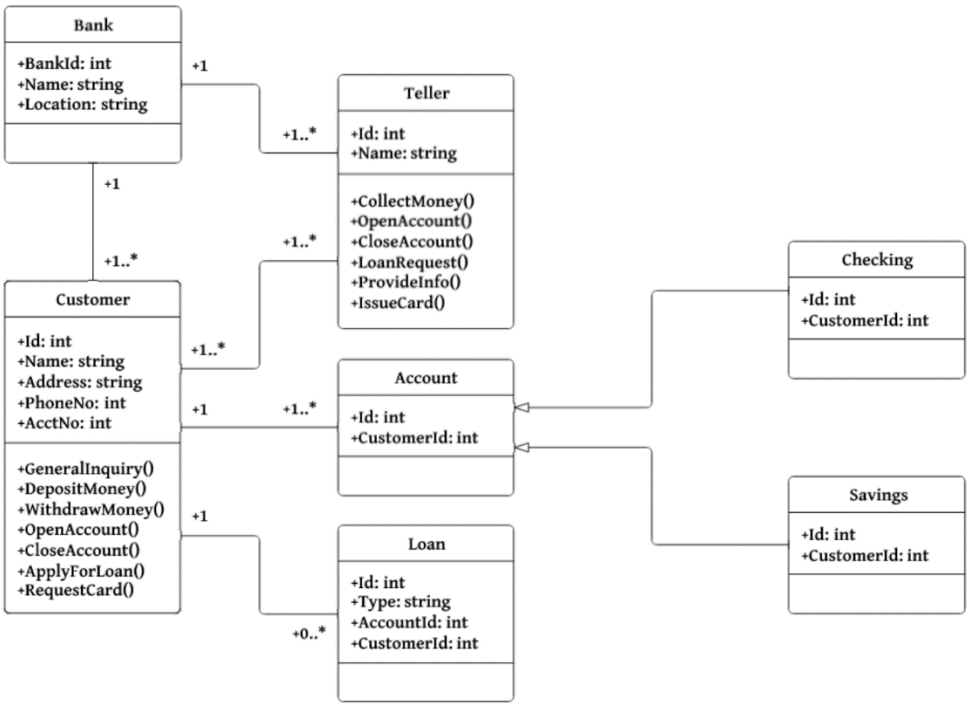


Figure 3. Class Diagram Example for a Banking System. (Salma, 2017).

The purpose of this diagram is to show the object classes in the system and the related associations between the specified classes. Here, a class can be considered as one kind of system object. An association between classes is represented by a link.

2.2.4 State Diagram

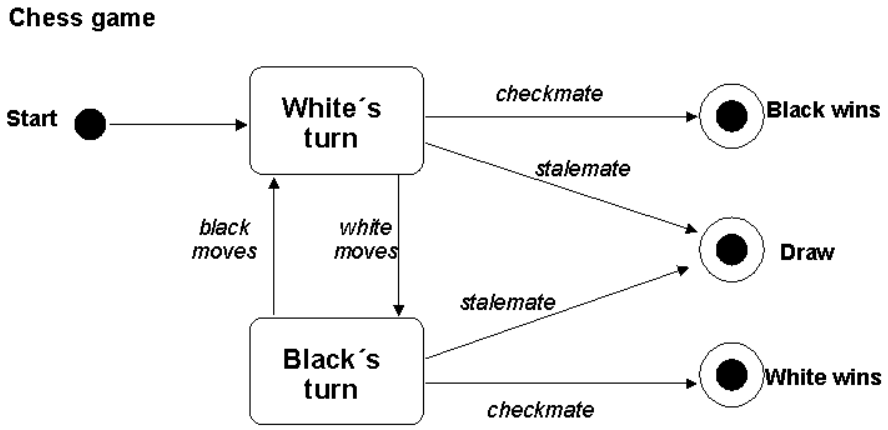


Figure 4. State Diagram Example for Chess Game. (Dalbey, 2017).

The purpose of this diagram is to specify external and internal events and show the behavior of the system in response to the events. A state is represented as a node. An event is represented as an arc between the nodes.

2.2.5 Sequence Diagram

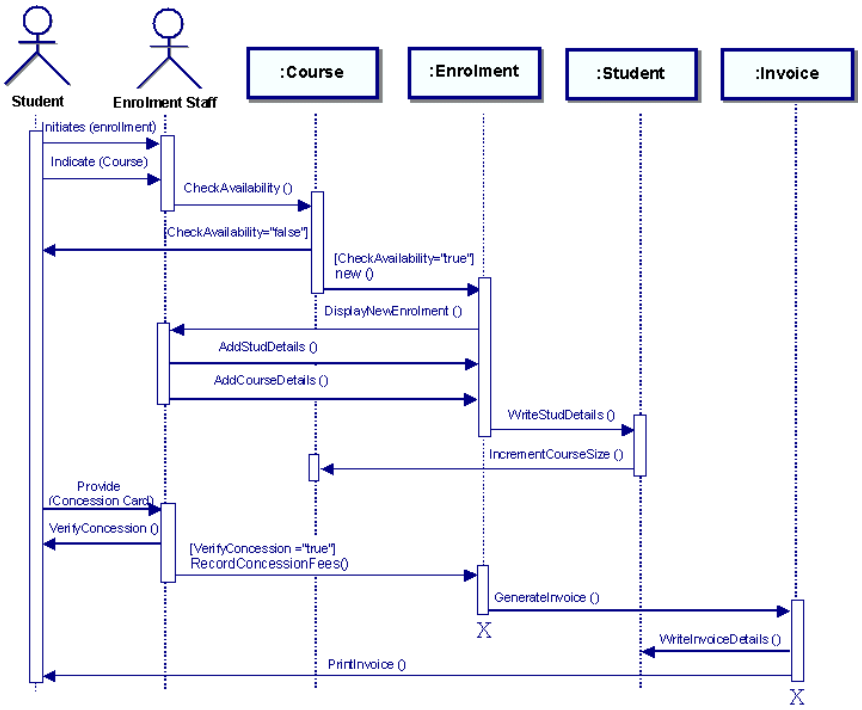


Figure 5. Sequence Diagram Example for Course Enrollment System. (RMIT University, n.d.).

The purpose of this diagram is to specify actors and objects and then show their interactions within the system. A particular use case is chosen to illustrate the sequence of interactions. Here, arrows are annotated between the objects. The actors and objects are listed along the top of the diagram. Dotted lines are drawn vertically down from the actors and objects, so that the interactions can be shown.

2.2.6 ER Diagram

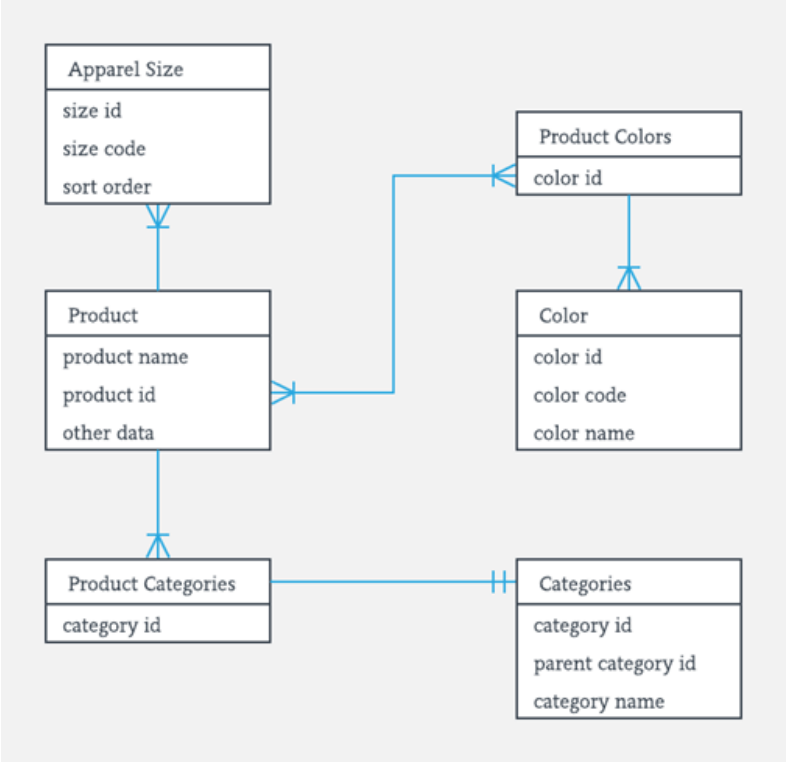


Figure 6. ER Diagram Example in DBMS. (University of Regina, 2017).

ER Diagram stands for Entity Relationship Diagram. The purpose of this diagram is to demonstrate the data in the system according to entity, entity type and entity set.

Here, an entity, also known as an object, can be a person or a conceptual existence such as wallet.

2.3 Applied Tool [모든 팀원 분들께, 아래 내용 추가해야 될 내용있다면 알려주시길 바랍니다.]

2.3.1 Ionic

2.3.2 Django

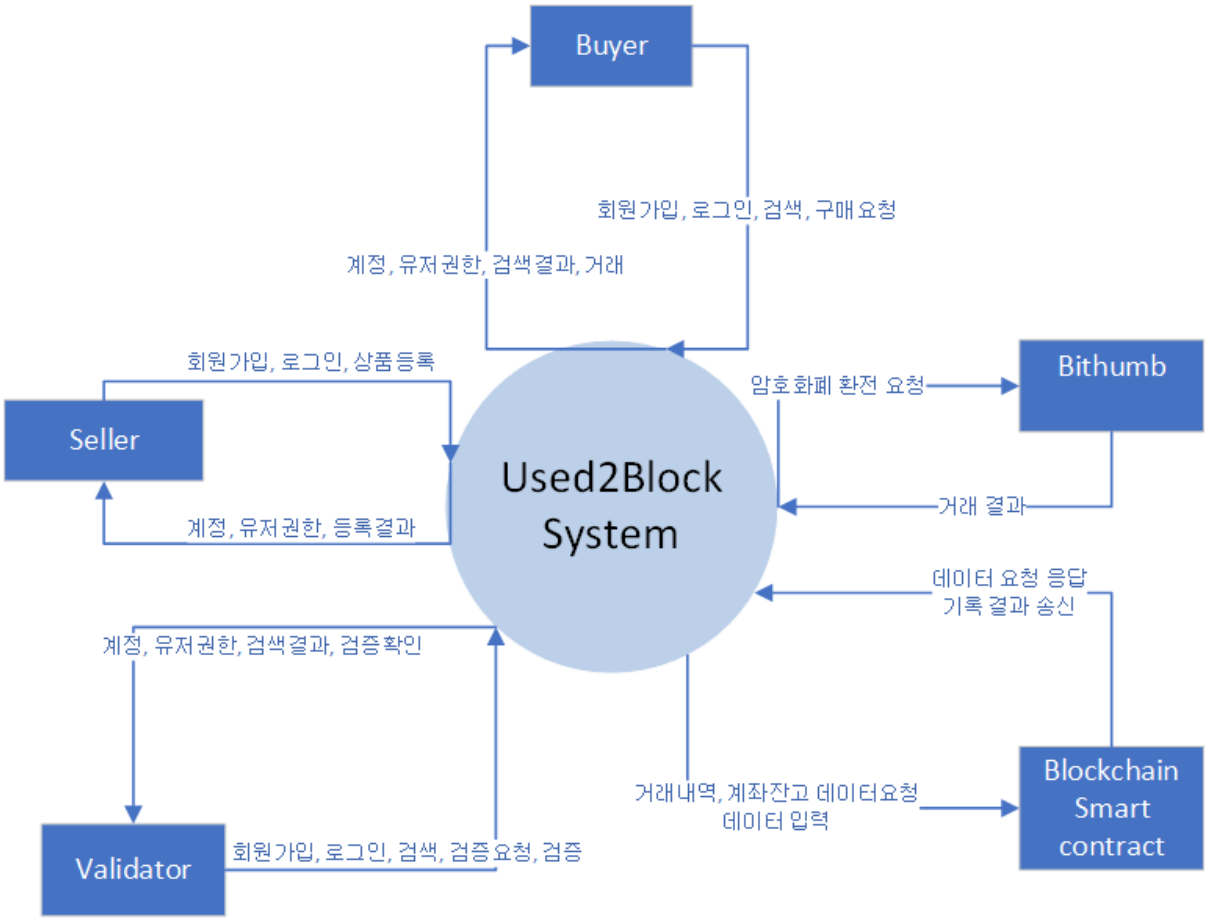
2.3.3 Remix for ETH-net

2.3.4 Mysql

2.3.5 Amazon Web Service

3. System Architecture

3.1 Context Diagram



해당 시스템에 대한 Context diagram 이다. Used2Block 과 Interaction 하는 System 은 총 5 개로 구성된다.

Blockchain Smart Contract - 거래에 대한 내역 및 가상화폐가 관리 되는 블록체인 상의 시스템이다. 해당 시스템을 통해 Validator 는 거래 검증을 진행할 수 있다.

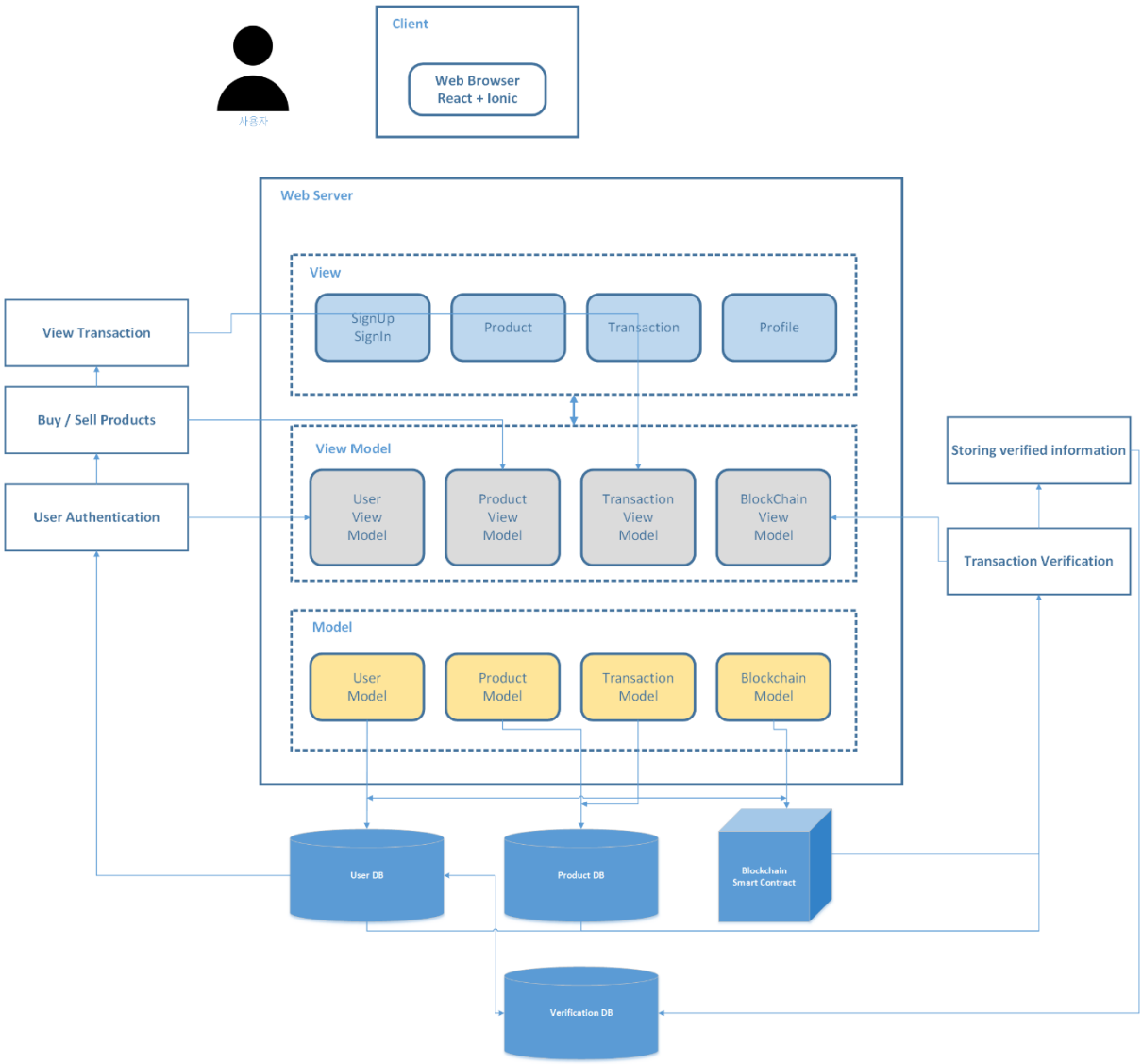
Validator - Buyer 와 Seller 사이의 거래를 검증하는 데 참여하는 Entity 이다.

Buyer - Seller 에 의해 등록된 물품을 구매하는 Entity 이다.

Seller - Used2Block 에 물품을 등록하는 Entity 이다. Validator 에게 검증을 받는 Entity 이기도 하다.

Bithumb - 거래에 사용하는 또는 거래 검증을 통해 Incentive 로 얻은 가상 화폐를 거래 및 환전하는 시스템이다.

3.2 System Overview

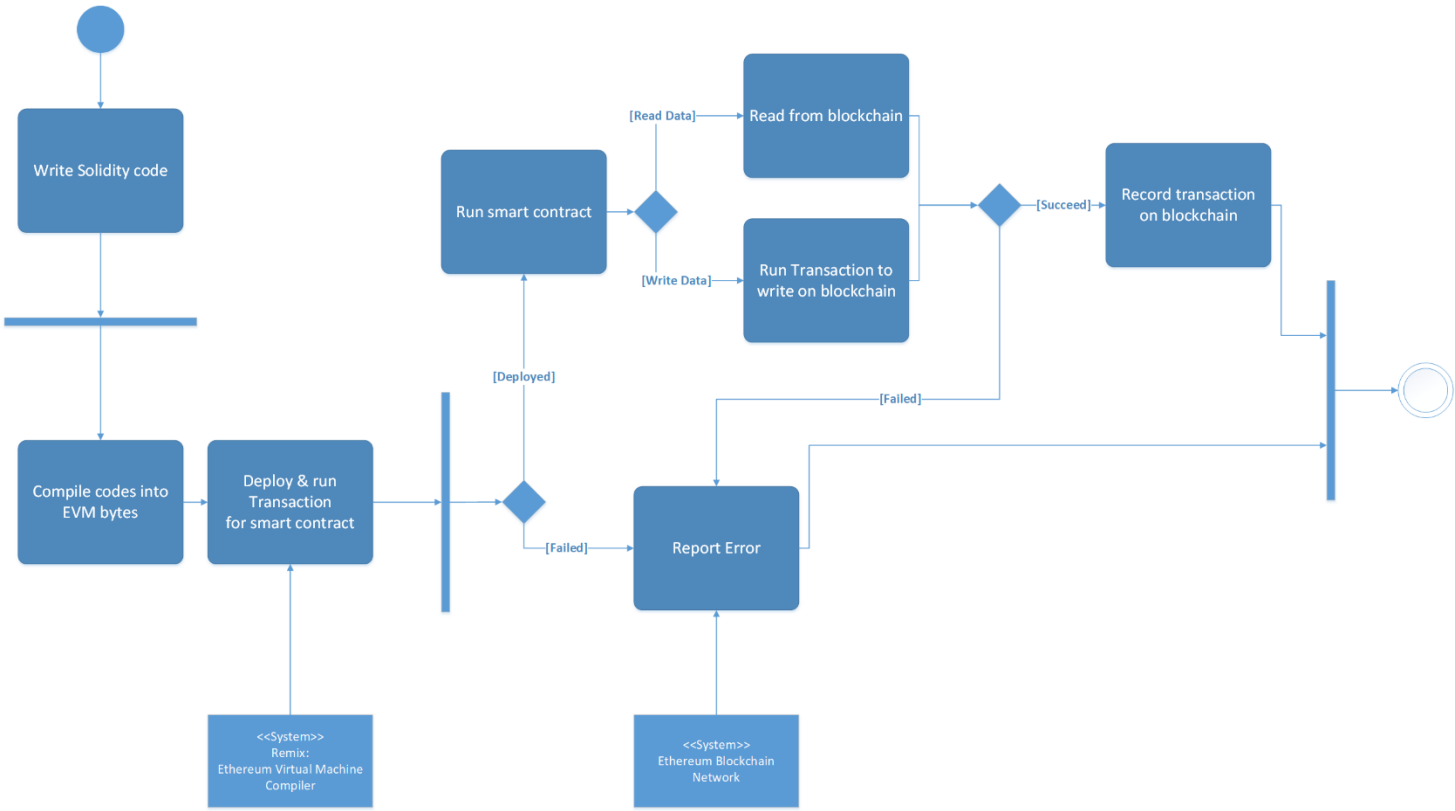


Used2Block 의 전체적인 시스템에 대한 Overview 이다. 해당 시스템은 Web Application 구조로서 Web browser 에 Product 를 보여주고 Seller 와 Buyer 사이의 Transaction 에 대한 Platform 을 구성한다. 일반적인 중고 시장 Web application 의 System 과 달리, Blockchain 을 이용한 Transaction Validation System 이 추가되었다.

4. Existing System

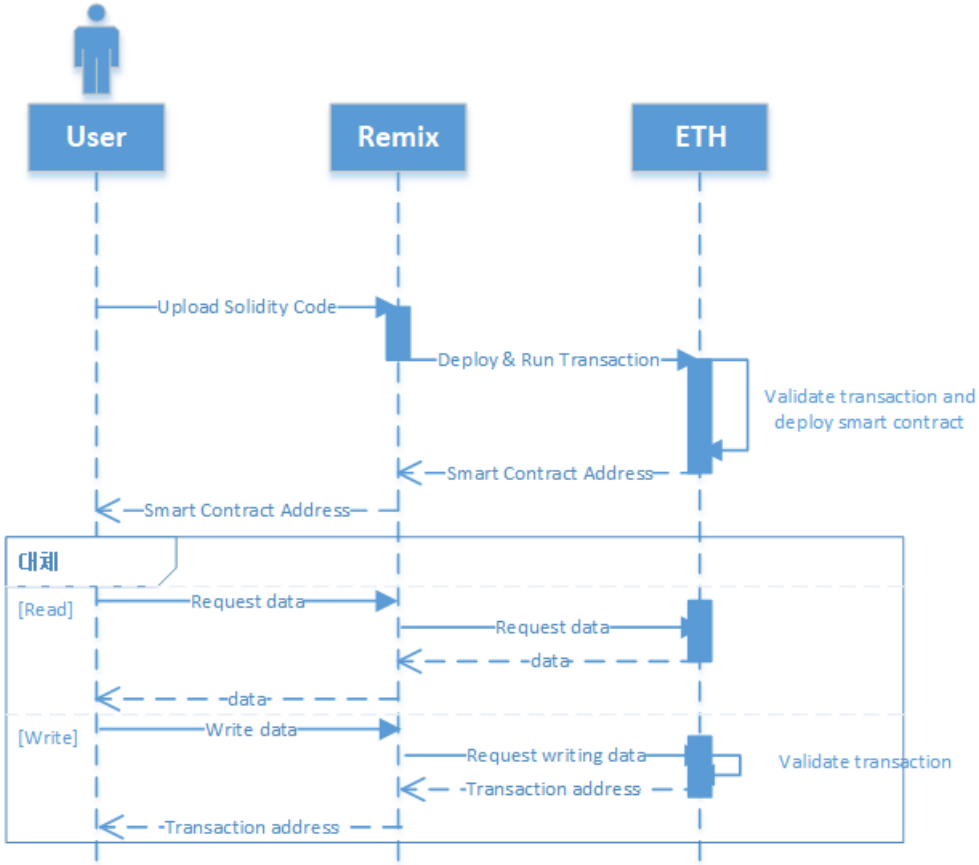
4.1 ETH blockchain: smart contract

4.1.1 Activity diagram



Ethereum Blockchain 에 대한 Activity diagram 이다. Solidity code 를 작성한 뒤, Remix IDE 를 이용해 EVM Byte 코드로 변환한 뒤 Blockchain 에 Smart Contract 를 배포하는 Process 를 보여준다. 중간에 수수료와 같은 문제로 인해 배포가 실패하면 해당 Process 는 바로 종료된다.

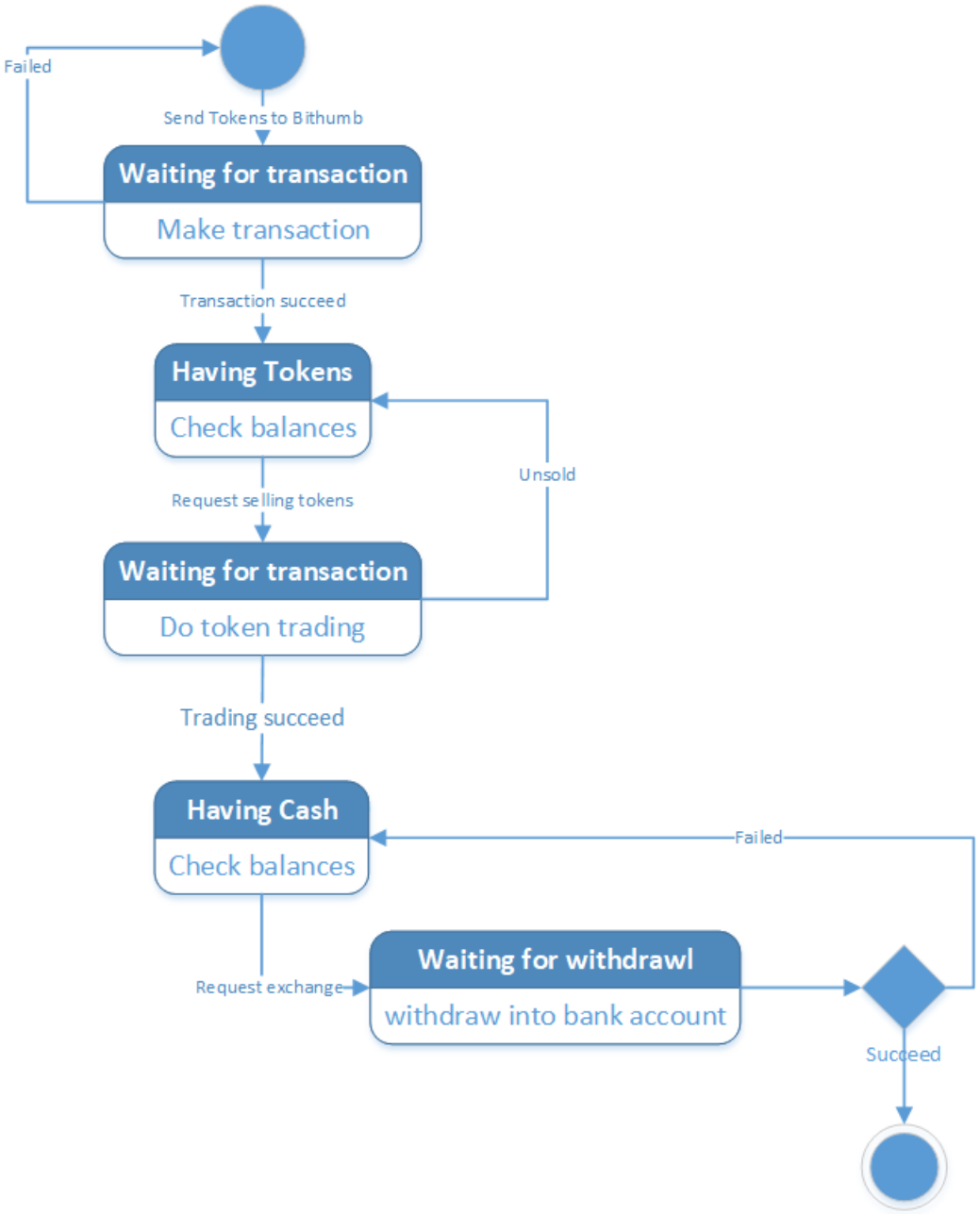
4.1.2 Sequential diagram



Smart Contract Deployment Process 에 대한 Sequential Diagram 이다. 먼저 프로그램을 이용하기 위해서는 Solidity 코드를 Remix IDE 에 전달한 뒤, ETH Network 에 EVM Bytecode 를 전달해 해당 프로그램을 배포하게 된다. 이후 해당 프로그램에 Read Operation 을 위해서는 ETH Network 에 Request 를 전달해 Data 를 갖고 오게 된다. 또는 Write Operation 을 위해서는 Write Operation 에 대한 Transaction 을 만들어 ETH Network 에서는 해당 Transaction 을 증명하고 Transaction address 값을 반환한다.

4.2 Bithumb

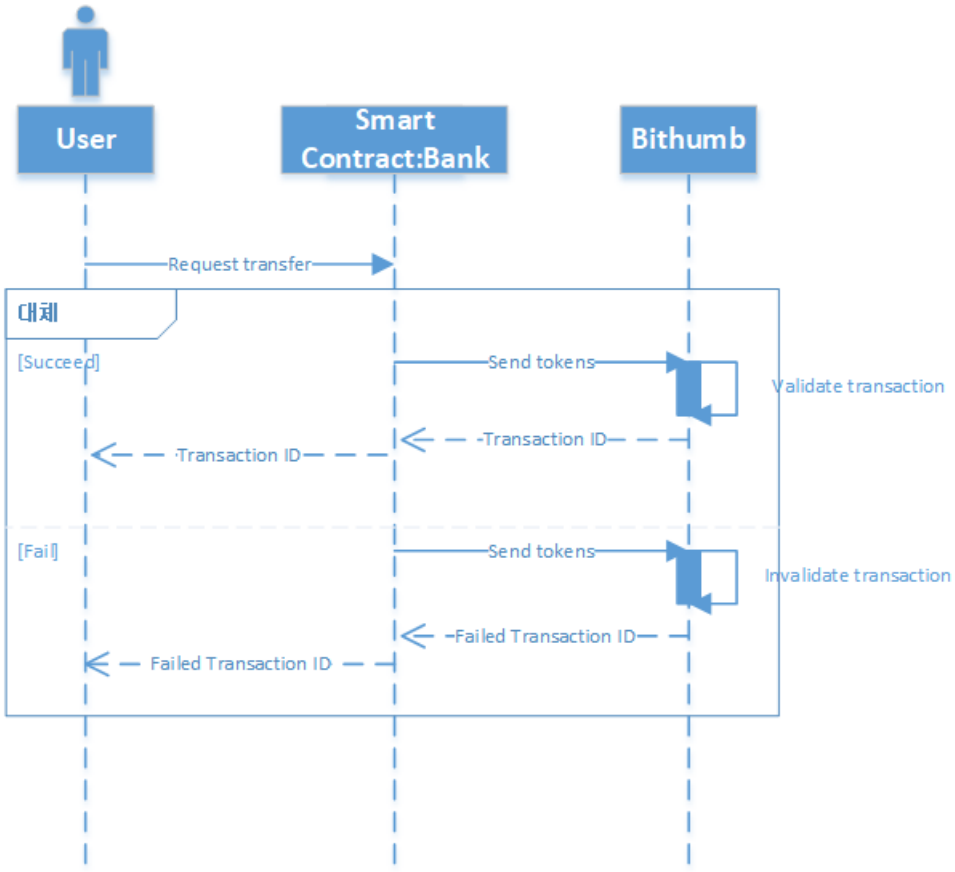
4.2.1 State Diagram for exchange tokens



Bithumb 을 이용해 Used2Block 의 가상화폐를 환전하는 과정을 State diagram 을 통해 나타내었다. Bithumb 의 Used2Block coin 에 대한 address 에 자신의 가상 화폐를 전달하고 Bithumb 의 Trading 서비스를 이용해 가상화폐를 현금으로 환전하는 과정을 보여준다.

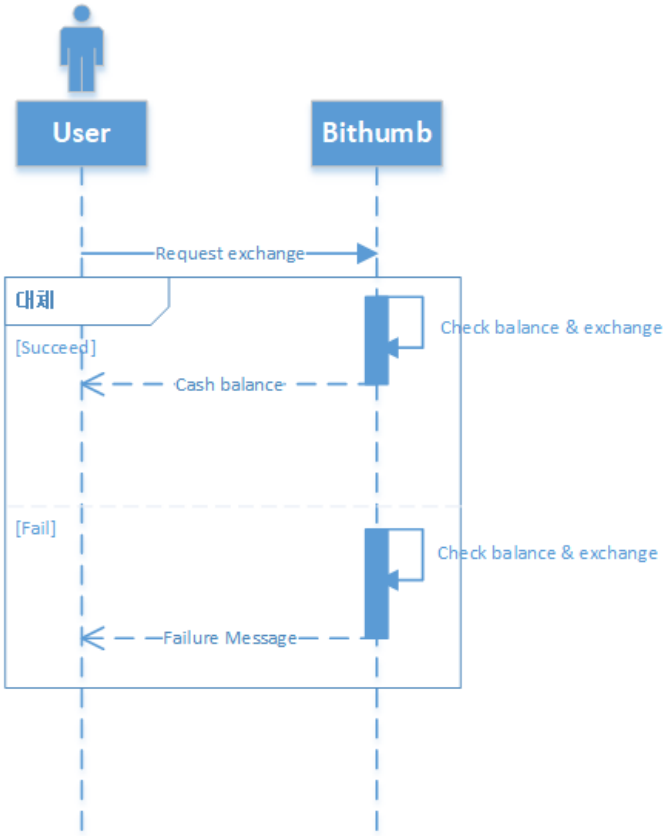
4.2.2 Sequential diagram for exchange tokens

4.2.2.1 Transfer tokens to Bithumb



Bithumb 을 이용해 Used2Block 의 가상화폐를 환전하는 과정의 일부를 Sequential diagram 을 통해 나타내었다. Used2Block coin 을 Bithumb 의 개인 address 에 전송하는 과정을 보여준다. Used2Block 의 서비스를 이용해 Ethereum 의 Smart Contract 에 가상화폐를 전송하는 Request 를 보낸다. 이후 거래에 대한 결과를 확인하여, 성공적으로 전송이 완료되면 Transaction ID 를 반환하고 실패하면 실패에 대한 Transaction ID 를 반환한다.

4.2.2.2 Exchange tokens into Cash



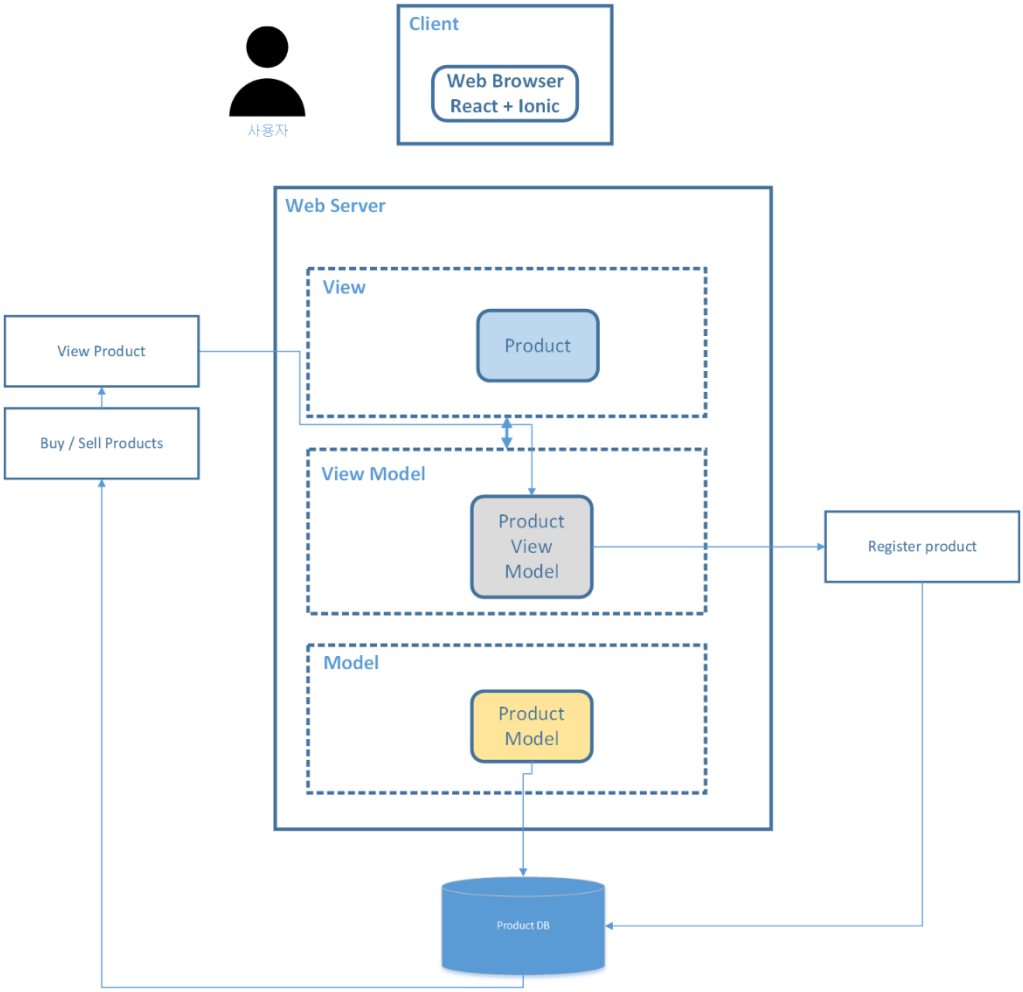
Bithumb 에 가상화폐를 전달한 뒤 현금으로 환전하는 과정이다. 해당 가상화폐를 Trading 이후 현금으로 환전해 현금 잔고를 반환한다. Trading 이 실패하거나 잔고가 부족할 경우 Failure message 를 반환한다.

5. Products Management System

5.1 Objective

Used2Block 의 Products management system 에 대해 설명한다. 해당 시스템은 Product 를 플랫폼에 등록하고 판매 및 구매를 위한 프로세스이다. 시스템의 구조를 설명하기 위해 먼저 System architecture 에 대한 Overall Design 을 보여주고 Class diagram, Sequence diagram 을 이용해 시스템의 프로세스를 설명하고자 한다.

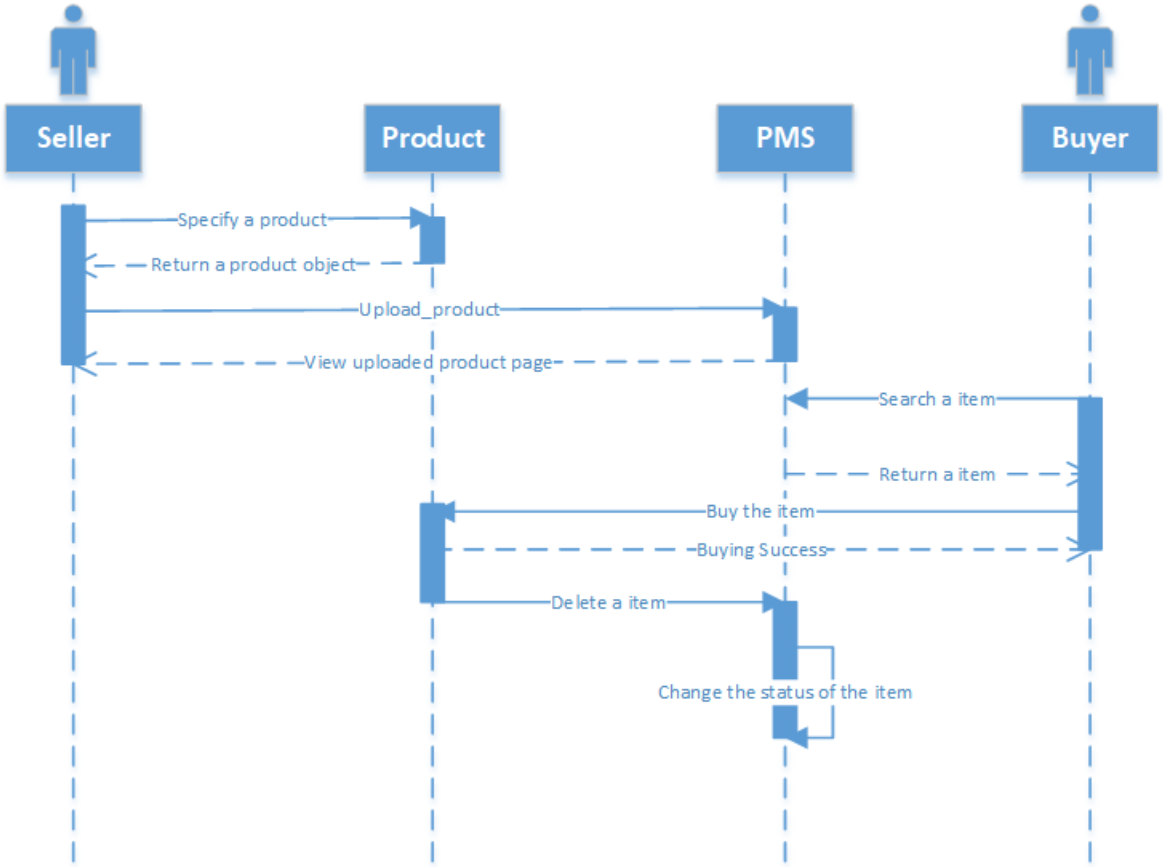
5.2 System Overview



해당 시스템은 판매되기 위해 등록된 물품을 관리하는 시스템이다. Product Model 의 standard 에 따라 판매자가 등록한 Product 는 Product Database 에 저장되게 되며,

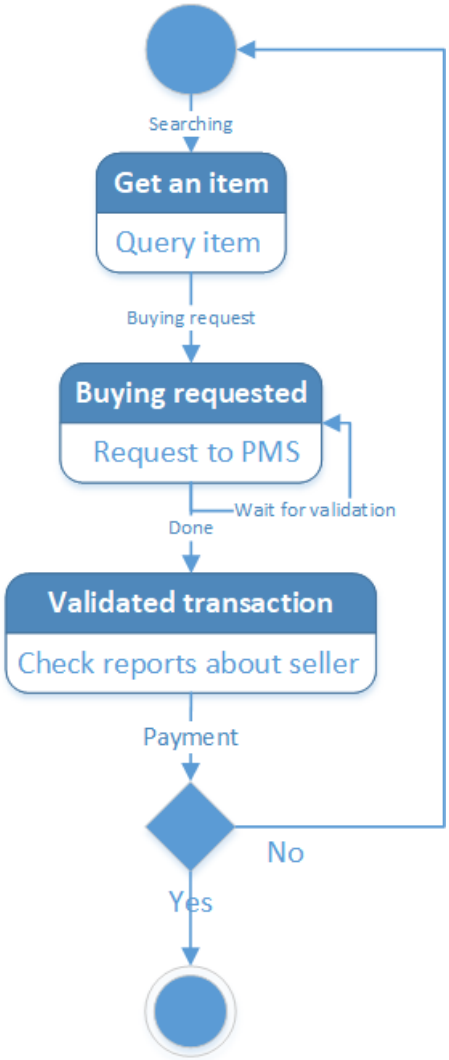
Product View Model 을 통해 Client 단에서 검색을 통해 Product 를 확인할 수 있게 구성되었다.

5.3 Sequence diagram



1) 해당 시스템의 Process 는 Seller 가 물품에 대한 정보를 입력하는 것에서 시작한다. Seller 는 판매하고자 하는 product 에 대한 종류, 이름, 가격 등을 기재한 뒤, Product Management System 에 product 를 등록한다.
2) 이후 Buyer 는 PMS 로부터 원하는 물품을 검색하고 검증 과정을 거쳐 구매를 진행한다.
3) 구매가 성공적으로 이루어지면, PMS 를 통해 해당 product 는 product list 에서 삭제되고 판매되었다는 사실을 기록하기 위해 status 를 "판매 완료"로 변경한다.

5.4 State diagram



Product Management System 에서 주요한 Buyer 의 구매 과정에 대한 State diagram 이다.

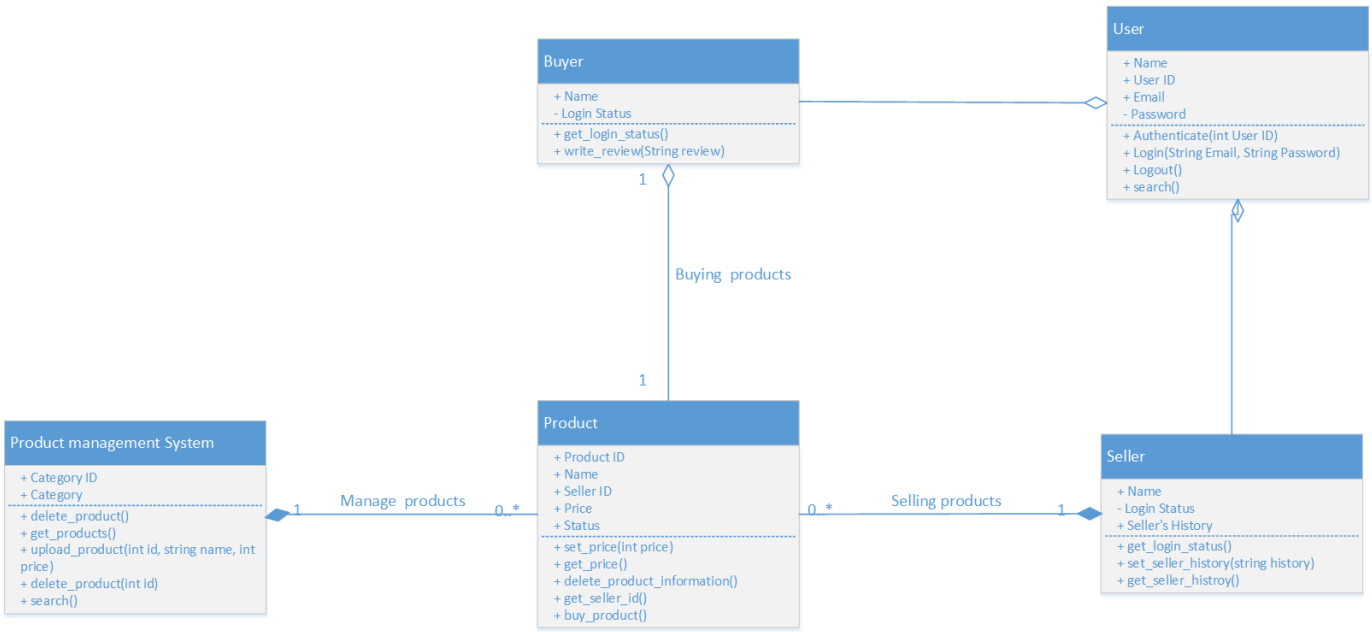
1) Buyer 는 PMS 를 통해 product object 를 가져온다.

2) 해당 물품에 대한 구매 request 를 PMS 에 보낸 뒤 validation 을 기다리게 된다.

3) Wait 이 종료된 뒤, Seller 에 대한 정보를 확인한다.

4) Seller 가 product 를 구매하기 위해 지불을 하면 해당 process 는 종료된다.

5.5 Class diagram



Product Management System(PMS)에 관련된 object 는 총 다섯 개로 이루어진다.

Product - 가장 기본적인 Component 이며 물건에 대한 이름과 가격, 판매자의 ID 를 담고 있다.

Seller - User class 의 Subclass 로서 물건을 Product Management System 에 판매할 product 를 등록할 객체이다.

Buyer - User class 의 Subclass 로서 Product Management System 을 통해 물건을 검색 및 구매할 객체이다.

PMS - Product 의 등록, 검색, 및 삭제의 기능을 수행할 객체이다.

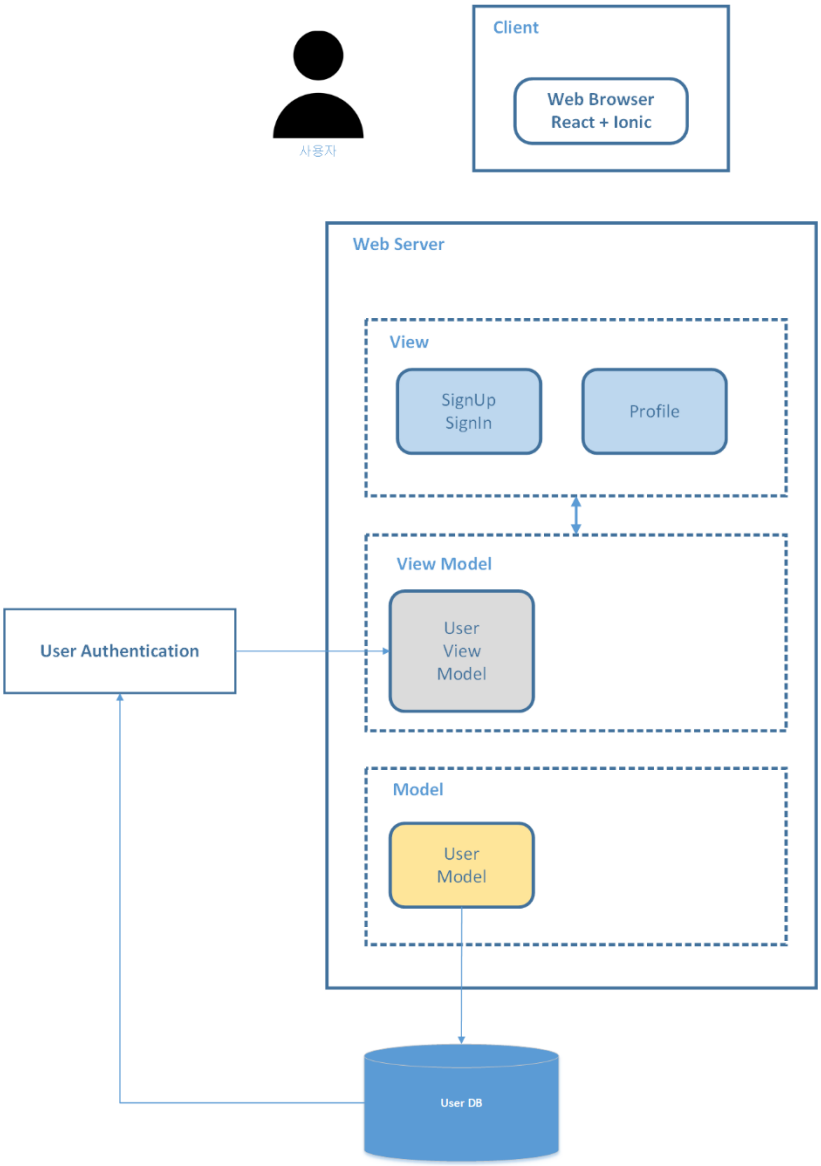
User - Buyer 와 Seller class 의 Super class 로서 타입에 맞는 User 들에게 상속된다.

6. User Management System

6.1 Objective

Used2Block 의 User Management System 에 대해 설명한다. User Management System 디자인은 User 가 사용할 수 있는 각종 기능들을 서술하고 있으며, 세부적으로 component 들 간의 상호작용으로 발생하는 프로세스로 구성되어 있다.

6.2 System Overview



해당 Sub System 은 Used2Block 시스템을 이용하는 User 를 관리할 목적으로 만들어졌다. User 가 가입 절차에 따라 시스템에 등록되면, 인증을 받아 User2Block service 를 이용할 수 있다.

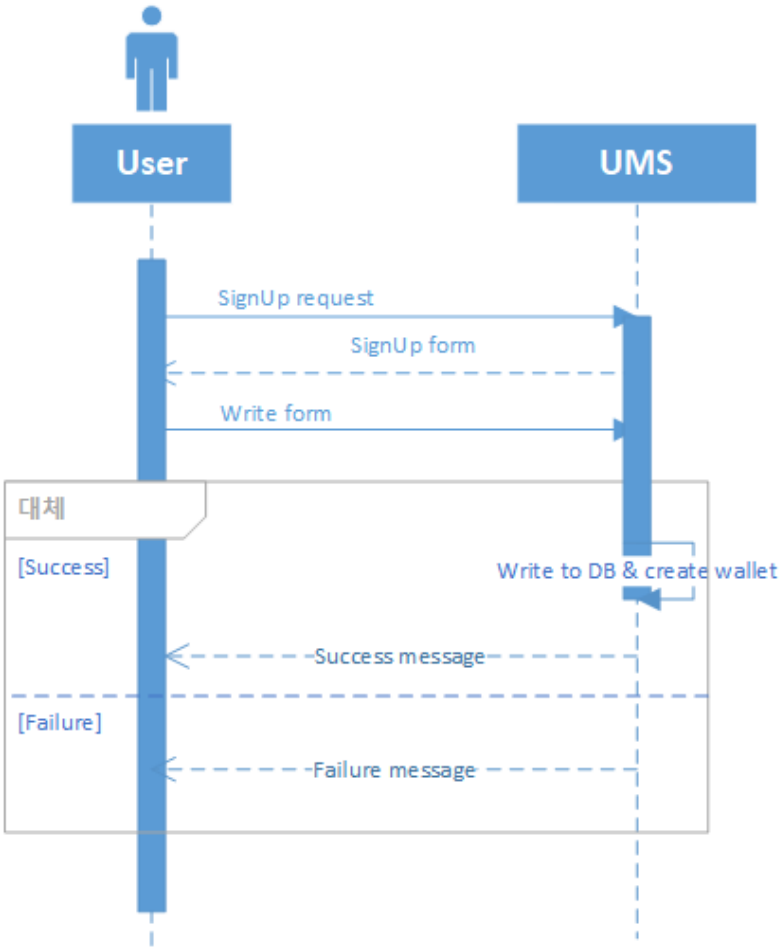
6.3 Sequence Diagram

시스템 내에서 Actor 와 시스템, 그리고 다른 Component 간의 Sequential 한 작업 과정을 보여주는 프로세스이다. User 관리 시스템에서 크게 기능은 5 가지로 이루어진다.

1) Signin process 2) Login process 3) Profile management 4) Account check

5) Exchange cryptocurrency

6.3.1 Sign in process



1) User 는 해당 시스템을 접속한 후, 회원가입 요청을 한다.

2) User Management System 은 회원가입 페이지를 반환한다.

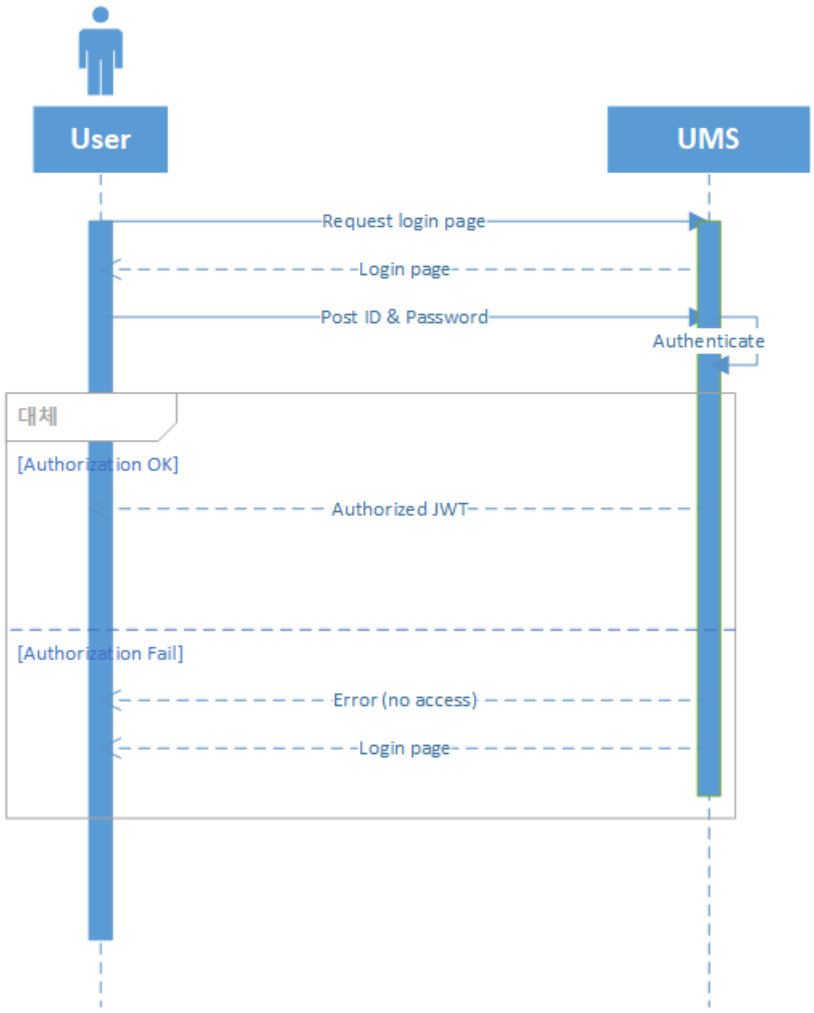
3) User 가 회원가입을 모두 진행하면, UMS 는 해당 내용을 Database 에 기록하고 Blockchain 에 대한 Wallet 을 생성하게 된다.

3-1) 만약 가입이 실패하면 실패 메시지를 보내 Login 과정을 다시 진행하도록 한다.

3-2) 가입이 성공적으로 이루어지면 User 의 가상화폐에 대한 계좌를 만든 뒤, 가입의 최종 확인을 User 에게 전달한다.

*회원 탈퇴의 경우에도 같은 순서를 거쳐서 신호를 주고받되, 생성이 삭제 작업으로 변경된다.

6.3.2 Login process



1) User 는 해당 시스템을 접속한 후, 로그인 요청을 전달한다.

2) UMS 는 User 에게 Login page 를 전달한다.

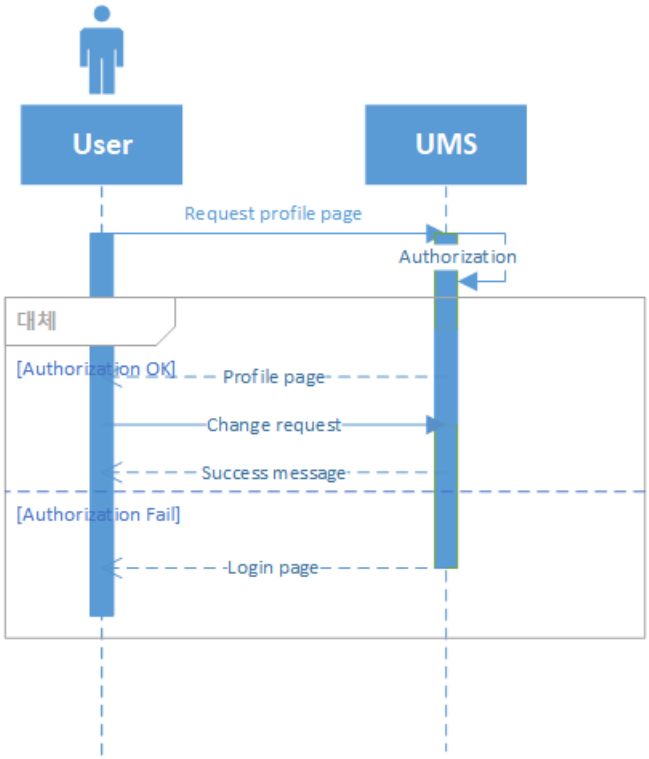
3) User 가 ID 와 Password 를 UMS 에 전달한다.

4) UMS 는 DB 를 참조하여 해당 정보를 Authenticate 한다.

4-1) User 의 정보가 Authentic 하다면 Json Wep Token 을 발급하여 해당 유저의 Login session 을 유지한다.

4-1) User 의 정보가 Authentic 하지 않다면, 실패 메시지를 보내 Login page 로 redirect 한다.

6.3.3 Profile page

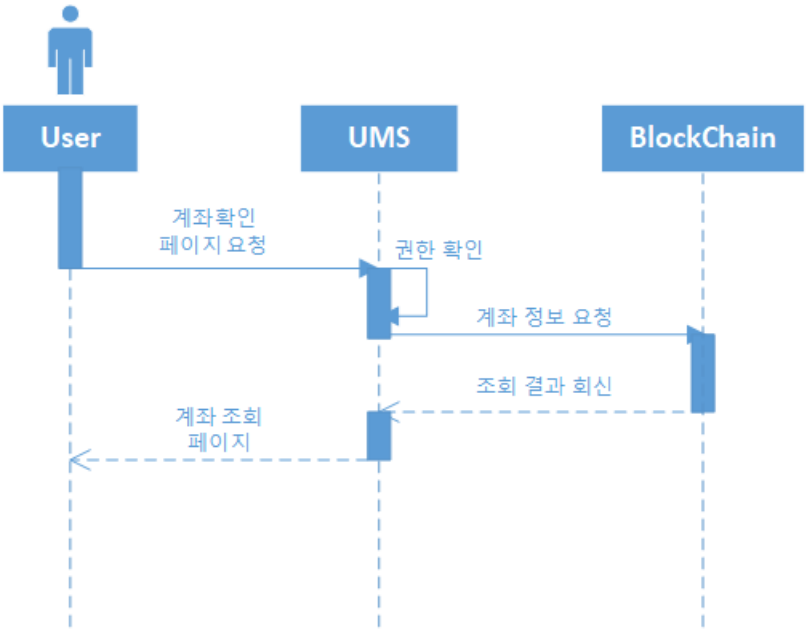


1) User 는 회원정보 수정 요청을 하고 JWT 를 통해 UMS 는 권한을 확인한다.

2-1) Authorization 이 성공적으로 이루어지면 Profile page 를 전달해 수정 혹은 확인이 가능하도록 한다.

2-2) Authorization 이 실패하면, 실패 메시지와 함께 Login page 로 redirection 한다.

6.3.4 Account check

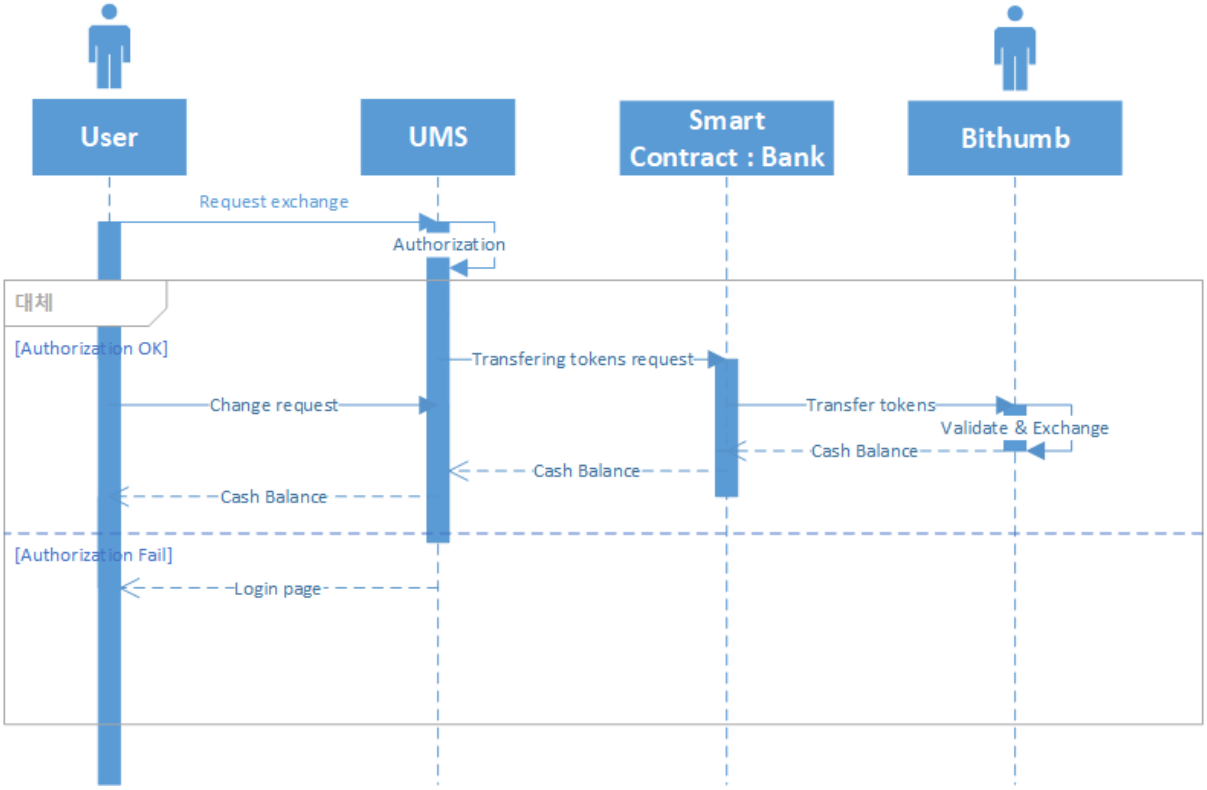


1) User 는 계좌 확인을 요청하고 UMS 는 권한을 확인한다.

2) 가상화폐의 잔고 정보는 블록체인에 기록되어 있어, UMS 는 Blockchain 에 해당 정보를 요청한다.

3) Blockchain 에 저장된 가상 화폐는 UMS 를 거쳐 User 에게 전달된다.

6.3.5 Exchange cryptocurrency



1) User 는 환전 거래를 요청한다. UMS 에서는 해당 User 에 대한 검증을 진행한다.

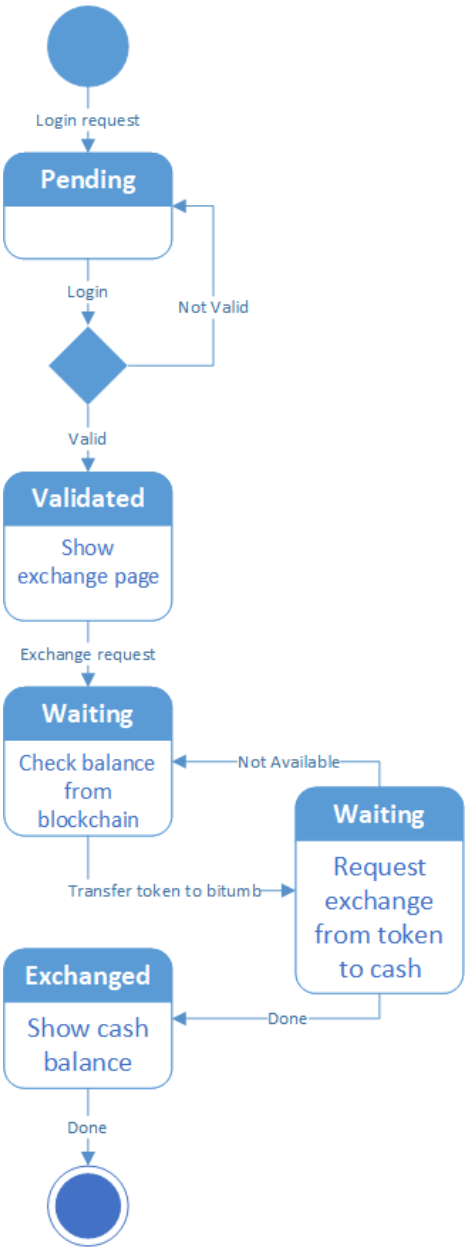
1-1) 검증이 실패하면 Login page 로 redirection 을 진행한다.

2) 거래 요청이 확인되면 UMS 는 Blockchain 상의 Smart Contract 에 거래 작업을 요청한다

3) 가상 화폐가 저장되고 기록되는 Blockchain 상의 Smart Contract 에서, Bithumb 으로 요청된 금액만큼의 화폐를 전송한다.

4) 교환된 Cash 잔고를 유저에게 전달한다.

6.4 State Diagram



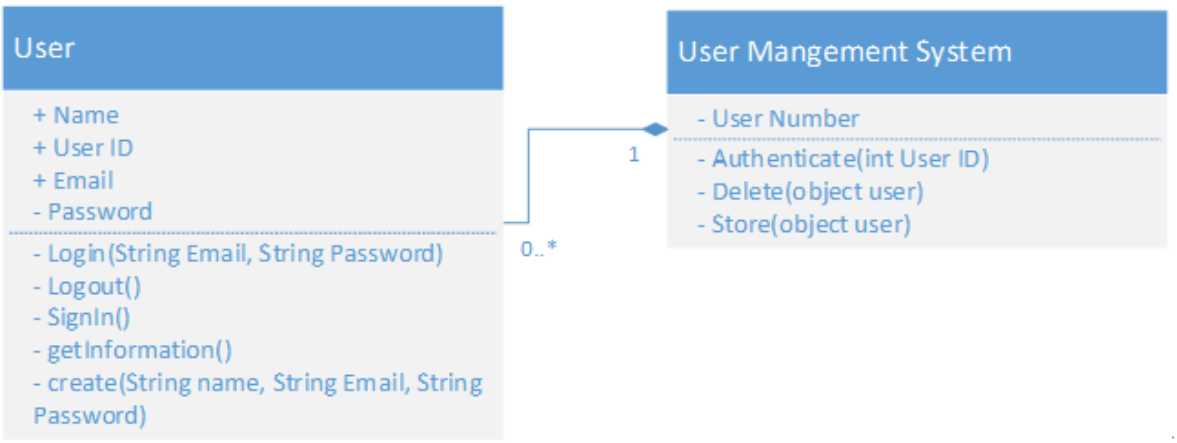
1) User 는 로그인 요청을 하고, Authorization 과정 동안 Pending 상태가 된다.

2) 이후 인증이 완료되면 User 는 JWT 를 이용해 Validated 상태로 전환된다.

3) 인증이 완료된 유저는 환전에 대한 요청을 진행하고 Waiting 상태로 전환된다.

4) 환전 작업은 Blockchain 의 Smart contract 와 Bithumb 의 Protocol 을 통해 이루어지며 완료 메시지를 받게 되면, Exchanged 상태로 전환되어 환전된 Cash balance 를 확인할 수 있다.

6.5 Class Diagram



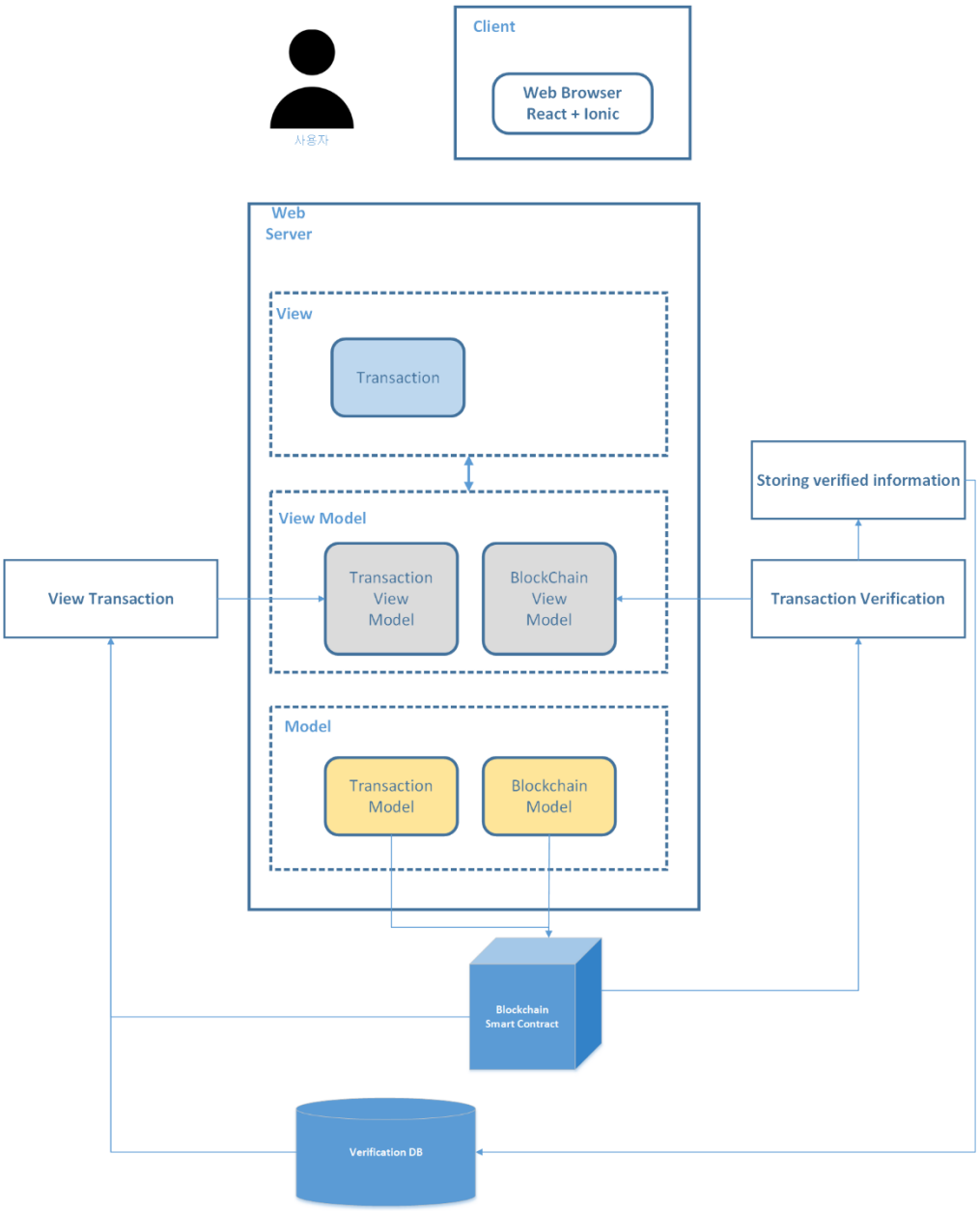
먼저 시스템의 전체적인 구조를 표현하기 위해서 Class Diagram 을 활용한다. Object 는 다음과 같이 User 와 User Management System(UMS) 2 가지가 존재한다.

User - 가장 기본적인 Object 로서 User 의 아이디, 비밀번호, 이메일 주소, 휴대폰 번호 등의 개인 정보를 담고 있다. Operation 으로는 Instance 를 생성하기 위한 "create"와 Profile 정보를 읽기 위한 "getInformation"이 존재한다.

UMS - User 를 관리하기 위한 시스템으로 User 를 인증하거나, Database 로 저장 및 삭제하는 Operation 을 수행한다.

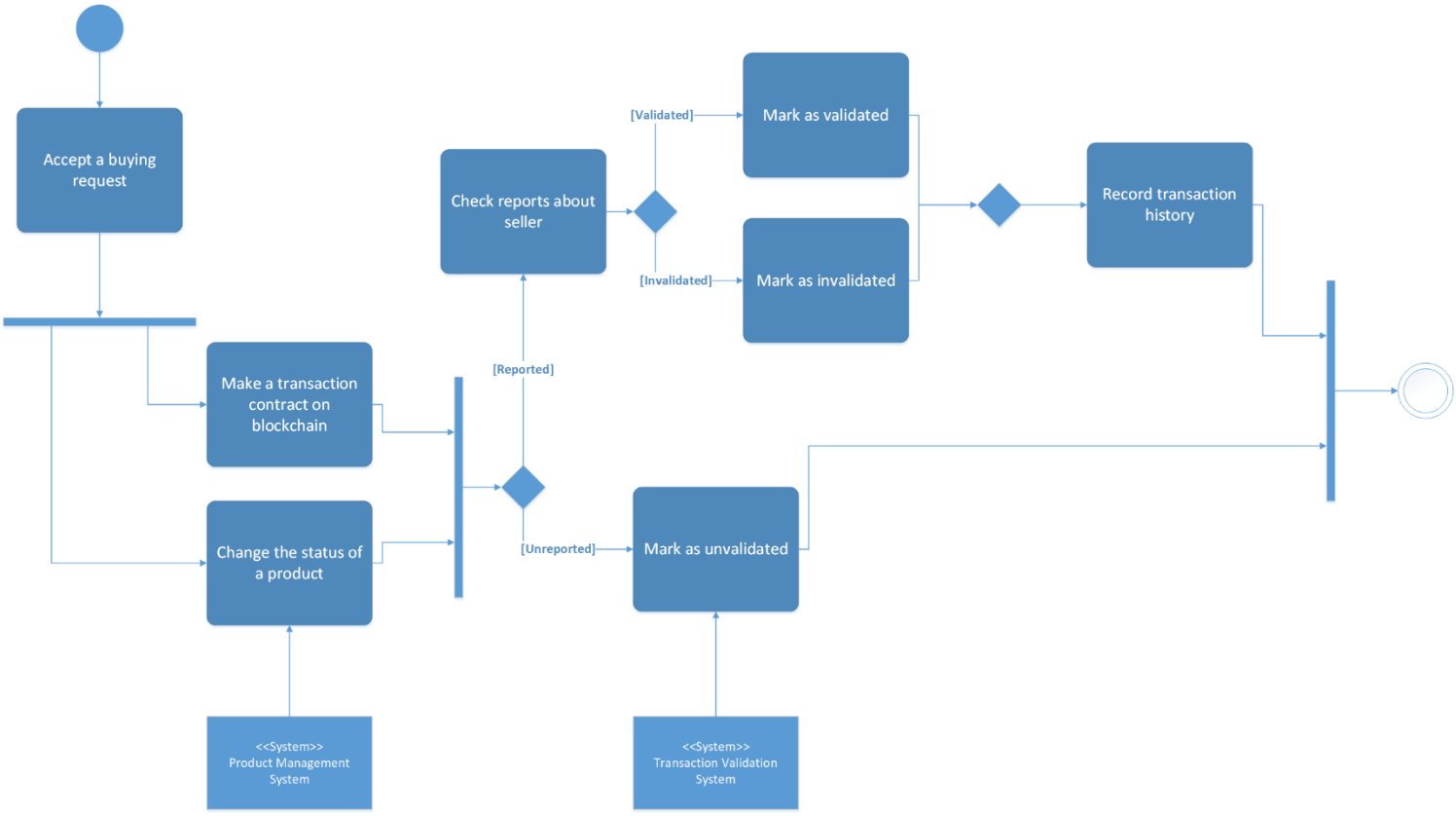
7. Transaction Validation System

7.1 System Overview



해당 시스템은 거래 검증을 위한 Transaction Validation System 이다. Buyer 와 Seller 를 통해 만들어진 Transaction 에 대해 Validator 가 검증하는 과정을 보여준다.

7.2 Activity diagram



해당 Process 는 Buyer 가 구매 요청 이후, 해당 Transaction 에 대한 검증 과정을 보여준다.

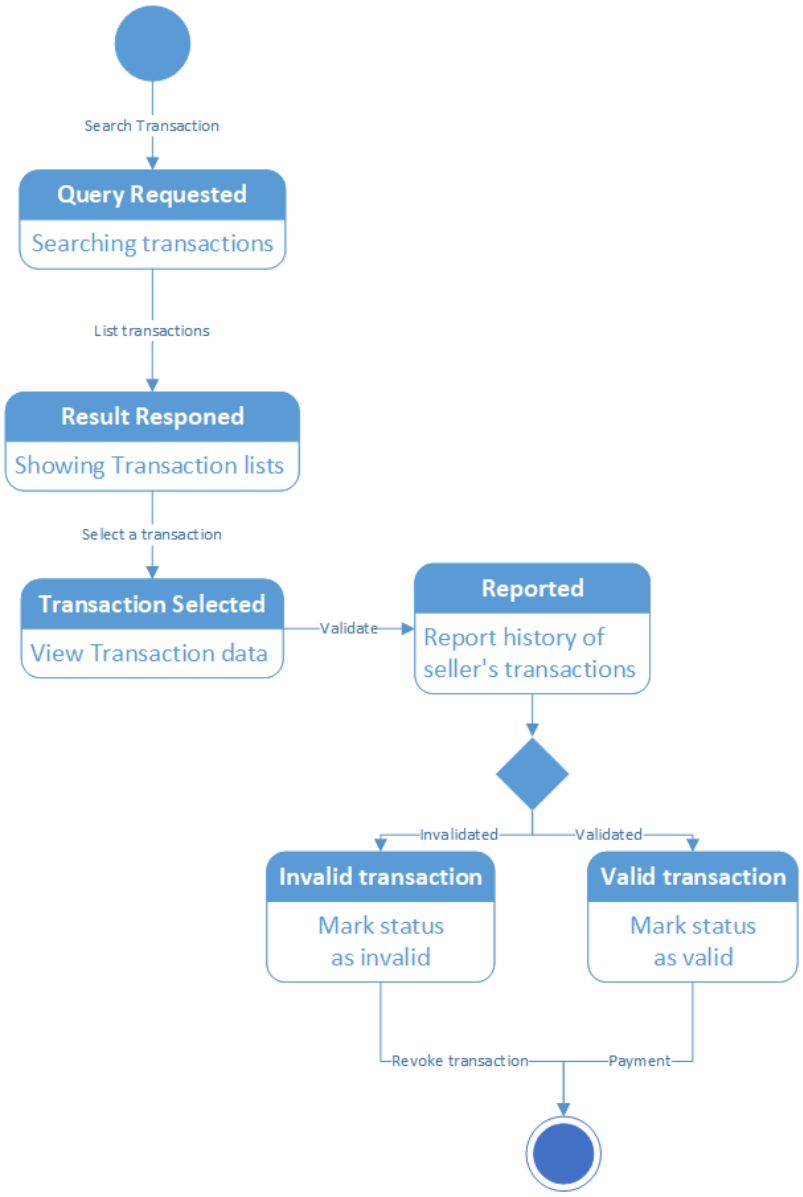
1) 구매자가 특정 Product 에 대한 구매 요청을 진행한다.

2) PMS 는 해당 Transaction 에 대한 정보를 Blockchain 에 올리고 Product 를 "Validating" 상태로 변경한다.

3) 이후 Blockchain 상에서 PoS Consensus Algorithm 과 Vaildator 들을 통한 해당 거래 증명을 기다린다.

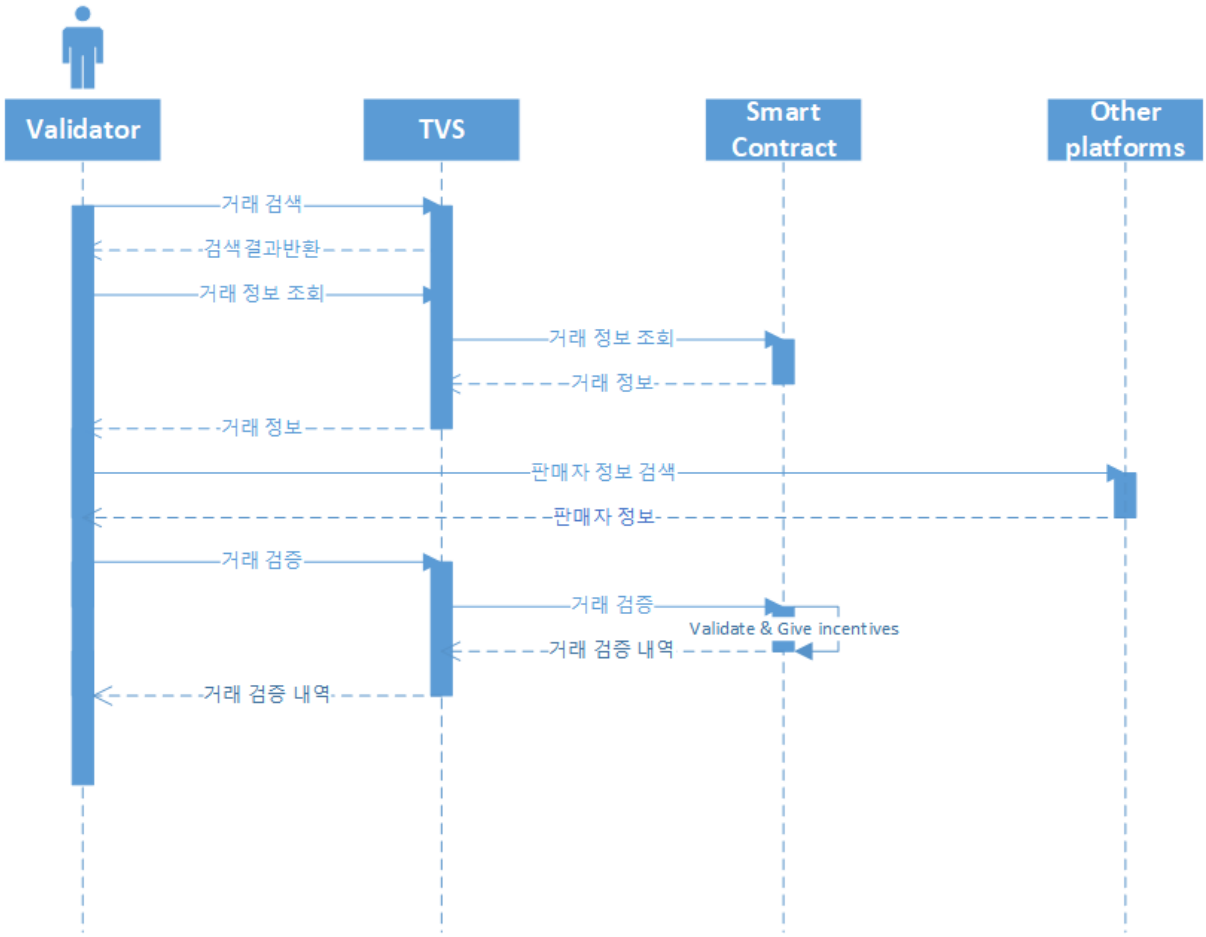
4) 거래에 대한 Report 가 발생하면, Report 의 결과에 따라 해당 거래를 "Valid" 또는 "Invalid" 상태로 변경한다.

7.3 State diagram



위 Activity diagram 과 마찬가지로 Transaction 에 대한 검증 과정을 보여주는 State diagram 이다. Validator 가 하나의 Transaction 을 골라 거래 내용을 확인하고 검증을 진행한다. 검증에 대한 내용이 Reported 상태에서 Buyer 가 확인한 뒤 해당 거래를 Valid 상태 혹은 Invalid 상태로 전환한다. Invalid 상태라면 해당 거래는 취소되어 종료되고 Valid 상태라면 해당 거래에 대한 대금을 지불하고 거래를 성사시킨다.

7.4 Sequential Diagram



1) Validator 가 Transaction Validation System 에 거래 리스트를 조회한다.

2) 거래 리스트 중 특정 거래에 대한 TVS 에 조회를 요청한다.

3) TVS 는 Blockchain 상의 Smart Contract 에 Transaction 내역을 조회해 Validator 에게 전달한다.

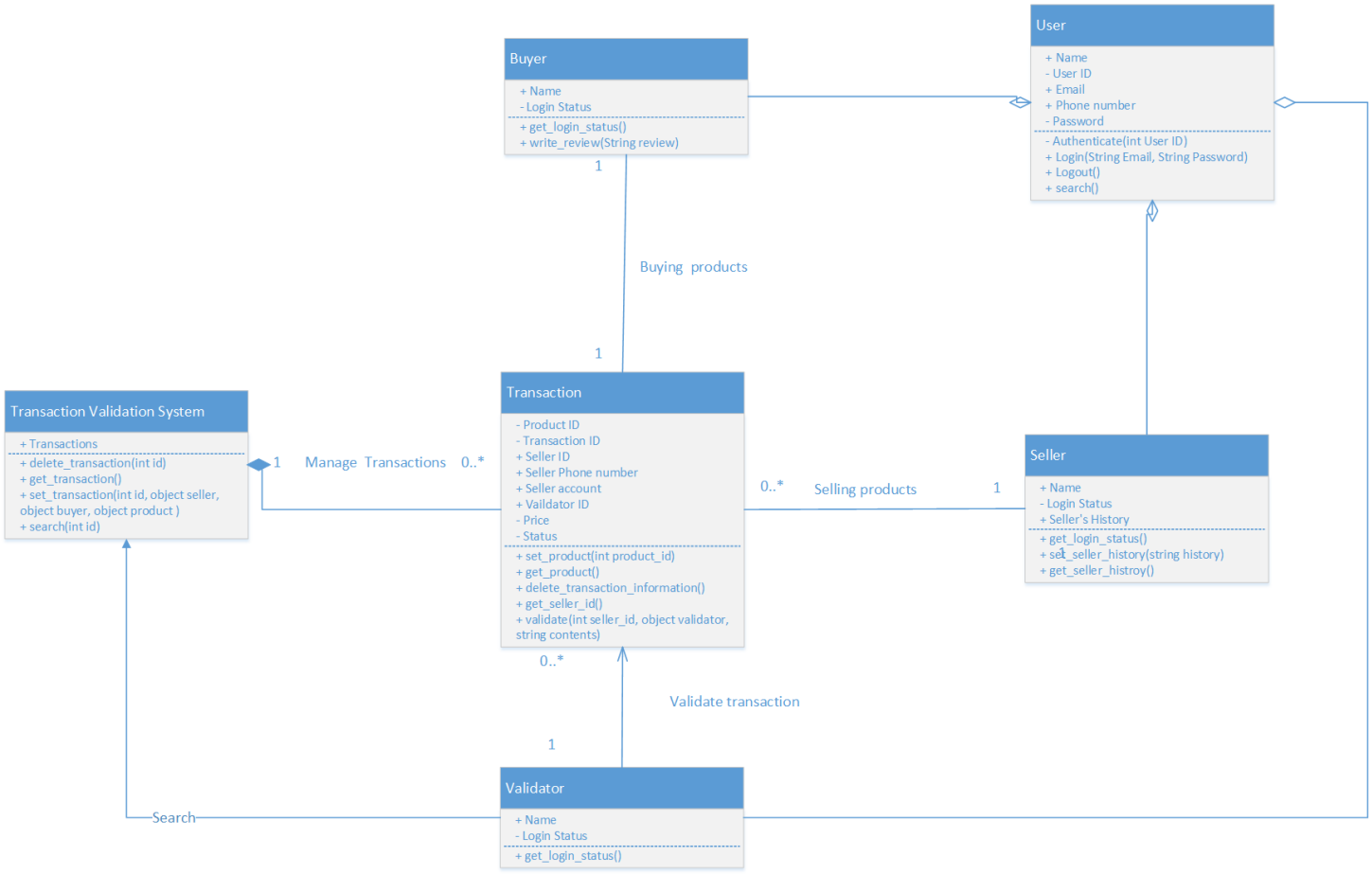
4) 해당 내역을 확인한 Validator 는 판매자에 대한 정보를 타 플랫폼을 통해 History 를 검색한다.

5) 판매자에 대한 타 플랫폼의 과거 거래 정보를 통해 TVS 에 거래 검증을 진행한다.

6) 해당 거래 검증은 Smart Contract 에 기록되고 Validator 에게 Incentive 가 지급된다. 이후 검증 내역이 TVS 에 반환된다.

7) 검증 내역이 TVS 에 기록되어 Validator 에게 보여진다.

7.5 Class diagram



해당 시스템에 관여되는 Object 는 총 6 개로 구성된다.

User - Seller 와 Buyer, Validator 의 Superclass 로서 기본적인 User 의 정보를 담고 있다.

Seller - User class 의 subclass 로서 물건을 Product Management System 에 판매할 product 를 등록할 객체이다.

Buyer - User class 의 subclass 로서 PMS 을 통해 물건을 검색 및 구매할 객체이다.

Validator - User class 의 subclass 로서 Transaction 을 검증하는 객체이다.

Transaction - 해당 시스템에서 생성되고 검증되는 객체이다. TVS 를 통해 관리되고 Validator 에게 검증된다.

TVS - Transaction 을 관리하는 시스템 객체이다. Validator 는 해당 객체를 통해 Transaction object 를 참조한다.

8. Protocol Design

8.1. Overview

The protocol design describes the protocols that must be followed in subsystem interactions. Describe the format, purpose, and meaning of data communication.

8.2. Protocol Description

8.2.1 HTTP Protocol

The request response message between the client and the server is described separately according to the protocol implementation.

8.2.1.1 User Authentication

1) Login

POST	/user/login		
Login application protocol for users.			
Category	Identifier	Description and Constraints	Example
Parameter	Id	User ID	
	Password	Password for the corresponding ID	
Response	Message	“Success” or “Failure”	
	Cookie	Authenticated JWT	
Status Code	200	Response OK	
	400	Bad Request	

Table 1: Description of Login Protocol

2) Logout

GET	/user/logout		
Log out program			
Category	Identifier	Description and Constraints	Example
Parameter	Id	User ID	
	Cookie	Authenticated JWT	
Response	Message	“Success” or “Failure”	
	File	/user/login	
Status Code	200	Response OK	
	400	Bad Request	

3) Sign-up

POST	/user/singup		
Sign-up application protocol for users.			
Category	Identifier	Description and Constraints	Example
Parameter	Email	User ID	
	Password	Password for the corresponding ID	
	Name	User Name	
	Phone number	Phone number	
Response	Message	“Success” or “Failure”	
	Redirection	/user/login	
Status Code	200	Response OK	
	400	Bad Request	

8.2.1.2 Product

1) Search

GET	/item/search		
Search products program			
Category	Identifier	Description and Constraints	Example
Parameter	Name	Product name	"Glasses"
Response	File	A page of product lists, which is from database	
Status Code	200	Response OK	
	404	404 Not Found	

2) Upload

POST	/item/upload		
Upload products program for seller			
Category	Identifier	Description and Constraints	Example
Parameter	Name	Prodcut name	
	Price	Product price	
	Description	Product descriptions	
	Authorization	JWT	
Response	Message	“Success” or “Failure”	
	File	A view page of registered item	
Status Code	200	Response OK	
	403	403 Forbidden	

3) Delete

GET	/item/delete		
Delete products program for seller			
Category	Identifier	Description and Constraints	Example
Parameter	Name	Prodcut name	
	Authorization	JWT	
Response	Message	“Success” or “Failure”	
	File	User's products page	
Status Code	200	Response OK	
	403	403 Forbidden	

4) Buy

GET	/item/buy		
Buying a product program for buyer			
Category	Identifier	Description and Constraints	Example
Parameter	Product ID	Prodcut ID	
	Authorization	JWT	
Response	Message	“Success” or “Failure”	
	File	/transaction	
Status Code	200	Response OK	
	403	403 Forbidden	
	404	Not found	

8.2.1.3 Transaction

1) Transaction list

GET	/transaction		
Transaction list page for users			
Category	Identifier	Description and Constraints	Example
Parameter	Authorization	JWT	
Response	Message	“Success” or “Failure”	
	File	List of transactions	
Status Code	200	Response OK	
	403	403 Forbidden	
	404	Not found	

2) Transaction validation

GET	/transaction/validation		
Transaction validation page			
Category	Identifier	Description and Constraints	Example
Parameter	Seller History	Seller's history found on other platform for validating transaction	
	Authorization	JWT	
Response	Message	“Success” or “Failure”	
	File	List of transactions	
Status Code	200	Response OK	
	403	403 Forbidden	
	404	Not found	

9. Database Design

9.1 Objective

요구사항 명세서를 기반으로 데이터베이스를 설계한다. 데이터베이스 내의 Entity 와 Data 의 관계를 표현하는 ER Diagram 을 제시하고, Relational Schema 를 도식화하여 서술한다. 또한 실제 테이블을 작성하고 데이터베이스를 구축할 수 있는 SQL 의 DDL 쿼리를 작성한다.

9.2 ER Diagram(사용자 및 상품)

9.3 Relational Schema(사용자 및 상품)

9.3.1. User

테이블 이름	User
테이블 설명	사용자
PRIMARY KEY	User_id
FOREIGN KEY	
INDEX	

NO	PK	AI	FK	NULL	컬럼 이름	TYPE	설명	참조 테이블
1	V				User_id	VARCHAR(15)	아이디	
2					password	VARCHAR(20)	패스워드	
3					email	VARCHAR(45)	이메일	
4					phone	INT(11)	전화번호	

5					address	VARCHAR(45)	주소	

9.3.2 UserWallet

테이블 이름	UserWallet
테이블 설명	Used2Block 에서 사용자의 가상화폐 계좌
PRIMARY KEY	User_id
FOREIGN KEY	User_id
INDEX	

NO	PK	AI	FK	NULL	컬럼 이름	TYPE	설명	참조 테이블
1	V		V		User_id	VARCHAR(15)	아이디	User
2					account	VARCHAR(45)	계좌번호	

9.3.3 Address

테이블 이름	Address
테이블 설명	주소 목록
PRIMARY KEY	address
FOREIGN KEY	address
INDEX	

NO	PK	AI	FK	NULL	컬럼 이름	TYPE	설명	참조 테이블
1	V		V		address	VARCHAR(45)	주소	User
2					city	VARCHAR(45)	시	
3					road	VARCHAR(45)	도로명	
4					building	VARCHAR(45)	건물	

9.3.4 VerificationReport

테이블 이름	VerificationReport
테이블 설명	검증 내역 보고
PRIMARY KEY	report_number
FOREIGN KEY	seller_id, validator_id
INDEX	

NO	PK	AI	FK	NULL	컬럼 이름	TYPE	설명	참조 테이블
1	V	V			report_number	INT	보고서 번호	
2			V		seller_id	VARCHAR(15)	판매자 아이디	User
3			V		validator_id	VARCHAR(15)	검증인 아이디	User
4					report_text	TEXT	보고 내용	

5					validate_upload_time	DATETIME	보고된 시간	

9.3.5 Product

테이블 이름	Product
테이블 설명	판매하기 위해 게시된 물품
PRIMARY KEY	Product_code
FOREIGN KEY	Seller_id
INDEX	

NO	PK	AI	FK	NULL	컬럼 이름	TYPE	설명	참조 테이블
1	V				Product_code	VARCHAR(45)	물품 코드명	
2					Product_name	VARCHAR(45)	물품 이름	
3					Price	INT	가격	
4					Image1	BLOB	물품 이미지	
5					Image2	BLOB		
6					Image3	BLOB		
7					Description	TEXT	물품 설명	
8			V		Seller_id	VARCHAR(15)	판매자 아이디	User
9					Upload_time	DATETIME	게시 날짜	

9.3.6 Order

테이블 이름	Order
테이블 설명	거래되고 있는 물품
PRIMARY KEY	product_code
FOREIGN KEY	product_code, seller_id, buyer_id, validator_id, upload_time
INDEX	

NO	PK	AI	FK	NULL	컬럼 이름	TYPE	설명	참조 테이블
1	V		V		product_code	VARCHAR(45)	상품 코드명	Product
2			V		seller_id	VARCHAR(15)	판매자 아이디	User
3			V	V	buyer_id	VARCHAR(15)	구매자 아이디	User
4			V	V	validator_id	VARCHAR(15)	검증인 아이디	User
5				V	ship_company	VARCHAR(45)	운송 회사	
6				V	livoice_number	VARCHAR(45)	운송장 번호	
7			V		upload_time	DATETIME	게시 날짜	Product

8				V	transaction_time	DATETIME	거래된 시간	
9				V	validate_time	DATETIME	검증 완료일	

9.4 SQL DDL

9.4.1 User

```
CREATE TABLE User
{
        'User_id'           VARCHAR(15)         NOT NULL,
        'password'      VARCHAR(20)         NOT NULL,
        'email'         VARCHAR(45)         NOT NULL,
        'phone'         INT(11)             NUT NULL,
        'address'           VARCHAR(45)         NOT NULL,
        PRIMARY KEY ('User_id')
};
```

9.4.2. UserWallet

```
CREATE TABLE UserWallet
{
        'User_id'           VARCHAR(15)         NOT NULL,
        'account'       VARCHAR(45)         NOT NULL,
        PRIMARY KEY   (User_id),
FOREIGN KEY (User_id) REFERENCES USER(User_id)
};
```

9.4.3 Address

```
CREATE TABLE Address
{
        'address'           VARCHAR(15)         NOT NULL,
        'city'          VARCHAR(45)         NOT NULL,
        'road'          VARCHAR(45)         NOT NULL,
        'building'      VARCHAR(45)         NOT NULL
```

```
        PRIMARY KEY   (address),,
        FOREIGN KEY   (address) REFERENCES User(address)
};
```

9.4.4 VerificationReport

```
CREATE TABLE VerificationReport
{
        'report_number'          VARCHAR(45)          NOT NULL
        AUTO_INCREMENT,
        'seller_id'              VARCHAR(15)          NOT NULL,
        'validator_id'           VARCHAR(15)          NOT NULL,
        'report_text'            TEXT                 NOT NULL,
        'validate_upload_time'   DATETIME             NOT NULL,
        PRIMARY KEY   (report_number),
        FOREIGN KEY   (seller_id) REFERENCES User(User_id),
        FOREIGN KEY   (validator_id) REFERENCES User(User_id)
};
```

9.4.5 Product

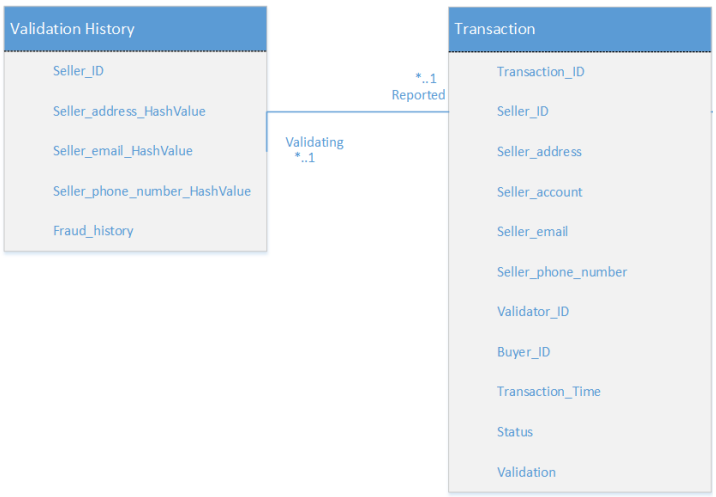
```
CREATE TABLE Product
{
        'product_code'           VARCHAR(45)          NOT NULL,
        'product_name'           VARCHAR(45)          NOT NULL,
        'price'                  VARCHAR(15)          NOT NULL,
        'image1'                       BLOB                   NOT NULL,
        'image2'                       BLOB                   NOT NULL,
        'image3'                       BLOB                   NOT NULL,
        'description      '      DATETIME             NOT NULL,
        'seller_id'              VARCHAR(15)          NOT NULL,
        'upload_name'            DATETIME             NOT NULL,
        PRIMARY KEY   (product_code),
        FOREIGN KEY (seller_id) REFERENCES User(User_id)
```

};

9.4.6 Order

```
CREATE TABLE Order
{
        'product_code'      VARCHAR(45)      NOT NULL,
        'seller_id'         VARCHAR(15)      NOT NULL,
        'buyer_id'          VARCHAR(15)      NULL,
        'validator_id'      VARCHAR(15)      NULL,
        'ship_company'      VARCHAR(45)      NULL,
        'invoice_number'          VARCHAR(45)          NULL,
        'upload_time'       DATETIME         NOT NULL,
        'transaction_time   '     DATETIME           NULL,
        'validate_time'     DATETIME         NULL,
        PRIMARY KEY   (product_code),
        FOREIGN KEY (product_code) REFERENCES Product(product_code),
        FOREIGN KEY (seller_id) REFERENCES User(User_id),
        FOREIGN KEY (buyer_id) REFERENCES User(User_id),
        FOREIGN KEY (validator_id) REFERNCES User(User_id),
        FOREIGN KEY (upload_time) REFERENCES Product(upload_time),
};
```

9.5 ER Diagram(블록체인에 저장되는 거래내역)



9.6 Relational Schema(블록체인에 저장되는 거래내역)

9.6.1 Transaction

테이블 이름	Transaction
테이블 설명	거래 기록
참조 테이블	-

NO	컬럼 이름	TYPE	설명
1	Transaction_id	uint256	거래 코드
2	Seller_id	VARCHAR(15)	판매자 아이디
3	Seller_address	VARCHAR(45)	판매자 주소
4	Seller_account	VARCHAR(45)	판매자 이메일
5	Seller_email	VARCHAR(45)	판매자 전화번호
6	Seller_phone_number	INT(11)	사기 거래 내역
7	Validator_id	VARCHAR(15)	검증인 아이디
8	Buyer_id	VARCHAR(15)	판매자 아이디
9	Transaction_time	DATETIME	거래 시간
10	status	INT	0 : 거래 전 / 1 : 검증 과정 진행 중 / 2 : 거래 완료
11	validation	TEXT	검증 내용

9.6.2 Validatioin History

테이블 이름	Validation_History
테이블 설명	검증 기록
참조 테이블	Transaction

NO	컬럼 이름	TYPE	설명
1	seller_id	uint256	판매자 아이디
2	Seller_address_HashValue	uint256	판매자 주소의 해쉬값
3	Seller_email_HashValue	uint256	판매자 이메일의 해쉬값
4	Seller_phone_number_HashValue	unit256	판매자 전화번호의 해쉬값
5	Fraud_history	uint256 array	사기 거래 내역

10. Testing Plan

10.1 Objective

시스템이 설계한 의도대로 실행되며, 시스템 내부에서 결함이 있는지 확인하기 위해 설계 단계에서 testing plan 에 대한 계획을 세우고 완성도 높은 시스템을 위해 정식 출시 전에 test 를 진행한다.

10.2 Testing Policy

Testing 과정은 Module test, Sub System Integration Test, System Integration Test, Acceptance Test 단계로 진행될 것이며, 각 test 가 이뤄지는 시기와 무엇을 test 를 할지는 각 항목에서 설명한다.

10.2.1 Module Test

서브 시스템을 이루는 각 module 을 실행하였을 때 error 가 생기는가, 정상적인 output 이 출력되는 가를 확인한다.

10.2.2 Sub System Integration Test

Used2Block 을 만들면서 개발하게 될 서브시스템은 User Management System, Product Management System, Validation System, Transaction System 4 개로 나누어진다.
여기서 Transaction System 다른 서브 시스템 및 같이 사용하게 될 existing system 들을 모두 아울러서 동작하는 시스템이기 때문에, 이 단계가 아닌 System Integration Test 파트에서 다룬다.
나머지 서브 시스템인 User Management System, Product Management System, Validation System 에 대해서는 각 항목별로 나누어 각 시스템에서 필요한 기능들이 정상적으로 작동하는지에 대해 test case 로 진행할 것이며, 각 서브 시스템을 test 하는데 쓰일 test case 는 다음과 같다.

10.2.2.1 User Management System

구성요소	설명

테스트 이름	회원가입 테스트
사전 조건	1. 사용자의 아이디, 이메일, 전화번호는 고유해야 한다. 2. 이메일은 '아이디 문자열@도메인 문자열'의 형식이어야 한다.
수행 절차	1. 아이디 중복 체크 2. 이메일 중복 체크 3. 이메일 형식 체크 4. 전화번호 중복 체크
기대되는 결과	회원 가입 후 데이터베이스에 회원 정보 저장한다. 회원에게 계좌 신설된다.

구성요소	설명
테스트 이름	로그인 테스트
사전 조건	1. 로그인 하려는 사용자가 회원가입을 한 상태이다.
수행 절차	1. 모든 양식 기입했는지 체크 2. 사용자 아이디 데이터베이스에 존재하는지 여부 체크 3. 사용자 아이디와 비밀번호 일치 여부 체크
기대되는 결과	사용자 로그인 상태로 기록된다

구성요소	설명
테스트 이름	로그인 테스트
사전 조건	1. 사용자가 로그인 한 상태이다.
수행 절차	1. 로그아웃 기능 실행
기대되는 결과	사용자 계정에서 로그아웃 되고, 새로 로그인할 수 있도록 된다.

10.2.2.2 Product Management System

Product Management System 은 판매하고자 하는 물품을 등록, 수정 및 삭제하고, 구입하고자 하는 물품을 검색하고 확인하는 기능을 가진다.

구성요소	설명
테스트 이름	상품등록 테스트
사전 조건	

수행 절차	1. 모든 양식 기입했는지 체크 2. 양식에 기입된 데이터가 정상적인지 체크
기대되는 결과	등록하고자 하는 상품이 데이터베이스에 저장된다.

구성요소	설명
테스트 이름	상품 정보 수정 테스트
사전 조건	상품이 데이터베이스에 저장된 상태여야 한다.
수행 절차	1. 상품의 등록자와 수정하고자 하는 사용자가 같은지 체크 2. 수정하고자 하는 내용이 정상적인 데이터인지 체크
기대되는 결과	상품이 수정된 내용이 데이터베이스에 새로 저장된다.

구성요소	설명
테스트 이름	상품 삭제 테스트
사전 조건	상품이 데이터베이스에 저장된 상태여야 한다.
수행 절차	1. 상품의 등록자와 삭제하고자 하는 사용자가 같은지 체크 2. 수정하고자 하는 내용이 정상적인 데이터인지 체크
기대되는 결과	삭제하고자 하는 상품이 데이터베이스에서 성공적으로 삭제된다..

구성요소	설명
테스트 이름	상품 검색 테스트
사전 조건	상품이 데이터베이스에 저장된 상태여야 한다.
수행 절차	1. 검색하고자 하는 상품의 키워드를 입력한다. 2. 검색 기능을 실행한다.
기대되는 결과	검색 조건을 충족하는 상품들만 목록으로 나타난다.

10.2.2.3 Validation System

구성요소	설명
테스트 이름	검증 내용 보고

사전 조건	검증인으로 참여하고자 하는 사용자가 가상화폐의 일정 지분 이상을 소유하고 있어야 한다(대포폰을 이용하여 허위로 가입한 사용자들의 무분별한 개입을 막기 위함).
수행 절차	1. 검증인 자격이 있는 사용자의 아이디로 로그인 2. 판매 상품을 올린 판매자에 대한 검증 보고를 작성한 후 보고 3. 검증인 자격이 없는 사용자의 아이디로 로그인 4. 2번 과정 진행 5. 사용자에게 검증인 자격이 없다는 메시지를 보여준다.
기대되는 결과	검증인으로의 자격이 있는 사용자가 보고한 내용만이 데이터베이스에 저장되고, 자격이 없는 사용자가 보고한 내용은 저장되지 않는다.

10.2.3 System Integration Test

System Integration Test 는 상기에 서술한 대로, 여러 개의 서브 시스템들이 갖춰졌을 때, 물품 거래를 위한 데이터의 이동이 원활하게 이뤄지는 지 확인하는 과정을 걸치며, 다음과 같은 단계를 거친다.

구성요소	설명
테스트 이름	거래 진행 테스트
사전 조건	상품이 데이터베이스에 저장된 상태여야 한다. 구매자 계정, 판매자 계정, 검증인 계정이 있어야 한다.
수행 절차	1. 구매자 계정으로 접속한다. 2. 구입하고자 하는 상품에 구매 신청을 한다. 3. 판매자 계정으로 접속한다. 4. 거래를 수락한다. 5. 검증 과정을 통해 판매자 계정이 정상적인 사용자라고 입력한다. 6. 물품 배송이 완료됐다고 가정한 input을 입력한다.
기대되는 결과	구매자 계좌에 있는 가상 화폐가 판매자 계좌로 전달된다.

구성요소	설명
테스트 이름	거래 파기 테스트

사전 조건	상품이 데이터베이스에 저장된 상태여야 한다. 구매자 계정, 판매자 계정, 검증인 계정이 있어야 한다.
수행 절차	1. 구매자 계정으로 접속한다. 2. 구입하고자 하는 상품에 구매 신청을 한다. 3. 판매자 계정으로 접속한다. 4. 거래를 수락한다. 5. 검증 과정을 통해 판매자 계정이 부적합한 판매자라고 입력한다.
기대되는 결과	거래에 대한 smart contract 가 파기된다. 검증인 계정에 연결된 계좌로 인센티브가 주어진다.

10.2.4 Acceptance Test

Used2Block 이 웹을 기반으로 하는 플랫폼이기 때문에, 웹에서 완성된 시스템을 실행해봤을 때, 유저 인터페이스에서 화면이 제대로 전환되는지, 화면에 원하는 데이터가 출력되는지, 버튼을 클릭했을 때 원하는 기능이 제대로 작동하는지 확인한다.

11. Development Environment

11.1 Objective

이 챕터에서는 실제 개발 단계에서 어떠한 기술과 개발 환경을 사용할 것인지에 대한 서술을 진행한다.

11.2 Ionic

스마트폰, 태블릿 등 다양한 접속 디바이스를 타겟으로 Flexible 하게 해당 디바이스 환경에 맞춘 UI 를 사용할 수 있도록 Web Application Framework 인 Ionic 을 사용한다. 이를 통해 Used2Block 은 기본적으로 웹 기반의 플랫폼이지만, Evolution 과정에서 모바일 어플리케이션으로도 출시할 계획이다.

11.3 Django

Django 는 Python 으로 작성된 오픈 소스 웹 어플리케이션 프레임워크이다.
제공하는 기능이 풍부하고, 쉽고 빠르게 웹 개발이 가능하며 사용자가 많아 정보에 대한 습득이 쉽다. 실제로 유사한 서비스인 번개장터 역시 Django 를 사용하고 있다.

11.4 Remix for ETH-net

Used2Block 에서는 Ethereum Blockchain 의 Network 를 이용한다. 이를 위해서는 해당 Network 에서 운용되는 프로그램인 Smart Contract 를 작성해야한다. Smart Contract 는 주로 Solidity 언어를 이용해 만들어지는데, Remix IDE 를 통해 쉽게 해당 언어를 작성 및 Smart Contract 를개발할 수 있다.

11.5 Amazon Web Service

Amazon Web Service 는 아마존에서 제공하는 클라우드 컴퓨팅 웹 서비스 기능이다. 개발자가 유연하고 확장 가능한 어플리케이션 구축할 수 있도록 Platform 을 제공한다. 비용적인 측면에서도 사용량만큼의 금액만을 지불하기 때문에 비용도 절감할 수 있다는 판단 하에 AWS 를 사용할 계획을 세웠다.

11.6 MySQL

MySQL 은 일반적으로 가장 많이 쓰이는 오픈 소스의 관계형 데이터베이스 관리 시스템이다. 기본적으로 MySQL 은 SQL 을 통해 데이터베이스를 구축 및 관리해야 하지만, 공식 GUI tool 인 MySQL 워크 벤치를 통해 데이터베이스 관리를 그래픽적으로 수행할 수 있다. 이를 통해 데이터베이스가 확장되더라도 GUI tool 을 이용하면 관리에 문제가 없을 것이라 판단되어 MySQL 을 사용한다.

12. Development Plan

week	6	7	8	9	10	11	12	13	14	15
Business Modeling										
Requirement Engineering										
System Modeling										
Design Architecture										
Implementation (Frontend)										
Implementation (Backend)										
Sub System Test										
System Integration										
System Test										
Deployment										

현재는 Used2Block 을 개발하는데 쓰일 existing system 을 분석하고, 개발이 필요한 sub system 들에 대해서도 구조 설계 및 개발을 진행하고 있다. 개발은 Frontend 팀과 Backend 팀으로 나누어서 parallel 하게 진행하고 있으며, sub system 의 개발 및 test 과정이 끝난 후에 시스템 통합을 한 후 시스템에 대한 test 를 진행할 것이다.
전체 시스템에도 아무런 이상이 없다고 판단이 되면 출시할 것이다.

13. Index

14. Reference

Dalbey, J. (2017). *Examples of State Transition Diagrams*. Referenced from http://Users.csc.calpoly.edu/~jdalbey/SWE/Design/STDexamples.html

Guru99. (2019). *Deployment Diagram: UML Tutorial with Example*. Referenced from https://guru99.com/deployment-diagram-uml-example.html#7

RMIT University. (n.d.). *Sequence Diagram Example*. Referenced from https://www.dlsweb.rmit.edu.au/Toolbox/knowmang/content/gathering_data/use_case/sequence_diagram_popup.htm

Salma. (2017). *UML Class Diagrams Tutorial, Step by Step*. Referenced from https://medium.com/@smagid_allThings/uml-class-diagrams-tutorial-step-by-step-520fd83b300b

University of Regina. (2017). *Data Modeling and Entity Relationship Diagram (ERD)*. Referenced from https://www.dlsweb.rmit.edu.au/Toolbox/knowmang/content/gathering_data/use_case/sequence_diagram_popup.htm

Visual Paradigm. (n.d.). *What is Activity Diagram?* Retrieved November 9, 2019, from https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/.

DjangoGirls. (n.d.). What is Django? Retrieved November 9, 2019, from https://tutorial.djangogirls.org/en/django/.