

제 8 회 IT 경진대회 작품 제안서

보석 주머니

2017 년 9 월

서론

가상 암호 화폐의 대표적인 비트 코인은 1bit가 벌써 4백만원을 넘어 갈 정도로 가상 화폐에 대한 수요가 증가하고 있다. 이 가상 암호 화폐의 핵심은 블록 체인 기술이다.

가상 화폐 거래가 점점 많아지면서 거래소도 늘어나고 있다. 거래소 웹 페이지는 많이 존재하지만 모바일 애플리케이션은 아직 많지 않다.

본 프로젝트에서는 가상 암호 화폐의 시세 조회 및 지갑 관리가 가능한 모바일 애플리케이션을 개발한다. 또한 화폐의 거래 기능을 구현하기에는 제약이 있기에 별도의 가상 암호 화폐(통칭 : 보석 코인) 서버를 개발한다.

보석 코인 가상 화폐 서버 역시 블록 체인 기술을 이용해 구현한다. 모바일 애플리케이션에서 실제 암호 화폐 거래 기능을 이 보석 코인으로 대신한다. 이 과정을 통해 암호 화폐와 블록 체인 기술의 이해를 높인다.

주요어:React-native, Spring-boot, node.js, express, MySQL, DynamoDB, 블록 체인, 가상 암호 화폐, 애플리케이션

목차

1. 개요	1
1.1 주제 / 작품 명	1
1.2 팀원 소개 및 담당 내용	1
1.3 개발 동기	1
1.4 개발 목적	2
1.5 개발 목표	2
1.6 창의성/우수성	2
1.7 활용성/사업성	3
2. 개발	4
2.1 개발 일정	4
2.2 개발 내용	4

3. 구성	5
3.1 시스템 구성	5
3.2 모바일 앱	8
3.3 서버	21
4. 참고 문헌	27

그림 목차

[그림 III-1] 전체 네트워크 구성도	5
[그림 III-2] 모바일 앱 통신 네트워크 구성도	6
[그림 III-3] 가상 화폐 서버 통신 네트워크 구성도	7
[그림 III-4] 앱 아이콘	8
[그림 III-5] 로그인 화면 / 회원 가입 화면	8
[그림 III-6] 시세 정보	9
[그림 III-7] 내 지갑	10
[그림 III-8] 지갑 관리 메뉴	11
[그림 III-9] 지갑 관리	12
[그림 III-10] QR코드 스캐너	13
[그림 III-11] 거래 기록 조회	14
[그림 III-12] 친구 지갑 관리	15
[그림 III-13] 부가 기능	16
[그림 III-14] 거래소	17
[그림 III-15] 모의 환전	18
[그림 III-16] 커뮤니티 게시판	19
[그림 III-17] 공지사항	20
[그림 III-18] 앱 서버 구동 화면	21
[그림 III-19] ERD 다이어그램	22
[그림 III-20] JWT에서 생성한 토큰	23

[그림 III-21] https 통신 설정-----	23
[그림 III-22] 노드 서버 운영-----	24
[그림 III-23] 블록 체인 구성도-----	24
[그림 III-24] 계좌 생성시 삽입되는 데이터 구조-----	25
[그림 III-25] 보석 코인 서버 구성도-----	25

1. 개요

1.1 주제 / 작품 명

주제: 암호 화폐 관리 어플리케이션 및 암호 화폐 시스템 구현

작품 명: 보석 주머니

1.2 팀원 소개 및 담당 내용

- 소프트웨어공학과 201232016 배다슬 / PM, 문서 작성, 가상화폐 서버 개발, 물리 DB 설계
- 소프트웨어공학과 201232025 이승기 / 앱 서버 개발, 논리 DB 설계
- 소프트웨어공학과 201432043 정보석 / 하이브리드 앱 개발, 논리 DB 설계

1.3 개발 동기

- 전 세계적으로 가상 암호 화폐에 대한 수요가 증가하고 있고 그에 따라 시장이 확대되고 있다.
- 가상 암호 화폐의 핵심 기술인 블록 체인 기술의 발전 가능성이 커지고 있고, 관련 기술 숙련자의 수요도 증가하고 있다.

- 가상 암호 화폐 관련 모바일 애플리케이션은 많은 수요에 비해 제대로 된 시스템이 없다.
- 임의의 가상 암호 화폐 구축을 통해 암호 화폐 시스템을 이해하고 핵심 기술인 블록 체인의 기술의 이해가 필요하다

1.4 개발 목적

- 가상 암호 화폐 구축을 통해 암호 화폐 시스템 및 블록 체인 시스템 이해
- 가상 화폐 시세 조회 및 관리 하이브리드 앱 개발

1.5 개발 목표

- React-Native으로 하이브리드 앱 개발
- Spring framework으로 앱 서버 개발
- Node.js으로 가상 화폐 시스템 개발
- DynamoDB로 블록 체인 구현

1.6 창의성 / 우수성

- 각각의 스마트폰 운영체제의 맞는 네이티브 앱 개발이 아닌 hybrid 앱으로 개발하기 때문에, 한번의 개발로 두가지 운영체제에 맞는 앱을 개발할 수 있다. 따라서 개발 시간과 인력을 단축시킬 수 있다.

- 핸드폰으로 가상 화폐 시세 조회 및 관리가 가능하다.
- 블록 체인 기술의 활용이 가능하다.

1.7 활용성 / 사업성

- 가상 화폐 관련 앱이 많지만 뚜렷이 인기있는 애플리케이션은 아직 없다.
- 네이티브 앱에 비해 숙련된 개발자 없이 간단한 html, css, javaScript만으로도 유지 보수가 가능하다

2. 개발

2.1 개발 일정

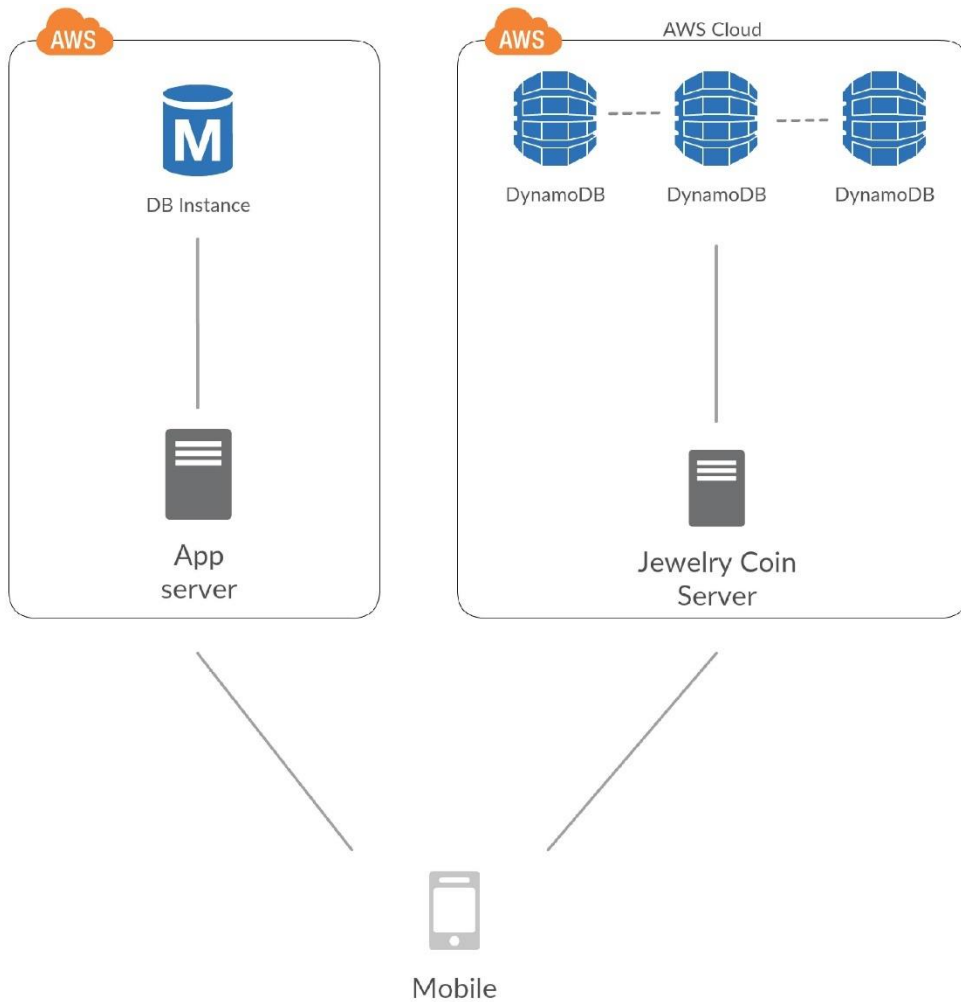
- 2017.07.17~2017.08.13 : Spring framework 앱 서버 개발 및 DB구축
- 2017.07.17~2017.09.03 : 하이브리드 앱 개발
- 2017.07.17~2017.09.03 : 가상 화폐 서버 및 블록체인 구현

2.2 개발 내용

- Spring Framework으로 앱 서버 개발
- My SQL로 앱 서버 관련 DB 구축
- React-native로 하이브리드 앱 개발
- Node.js Express로 가상 화폐 서버 개발
- AWS DyanmoDB로 블록 체인 구현
- 모든 서버와 DB는 AWS Cloud Platform 이용

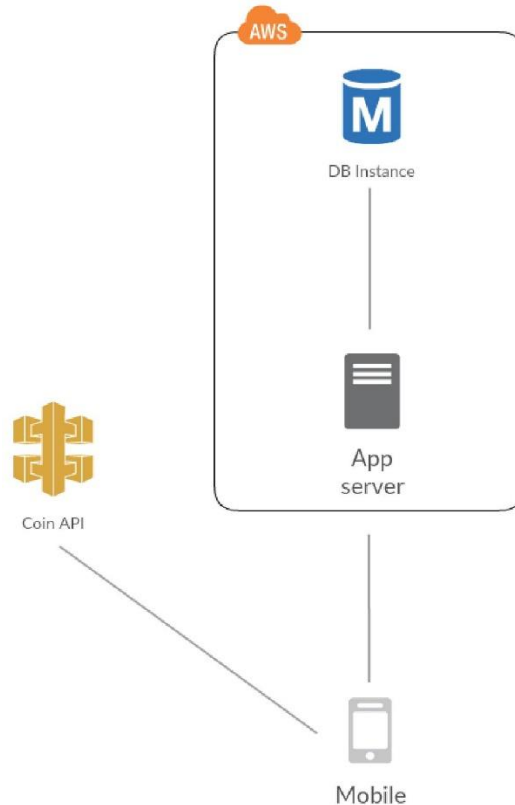
3. 구성

3.1 시스템 구성



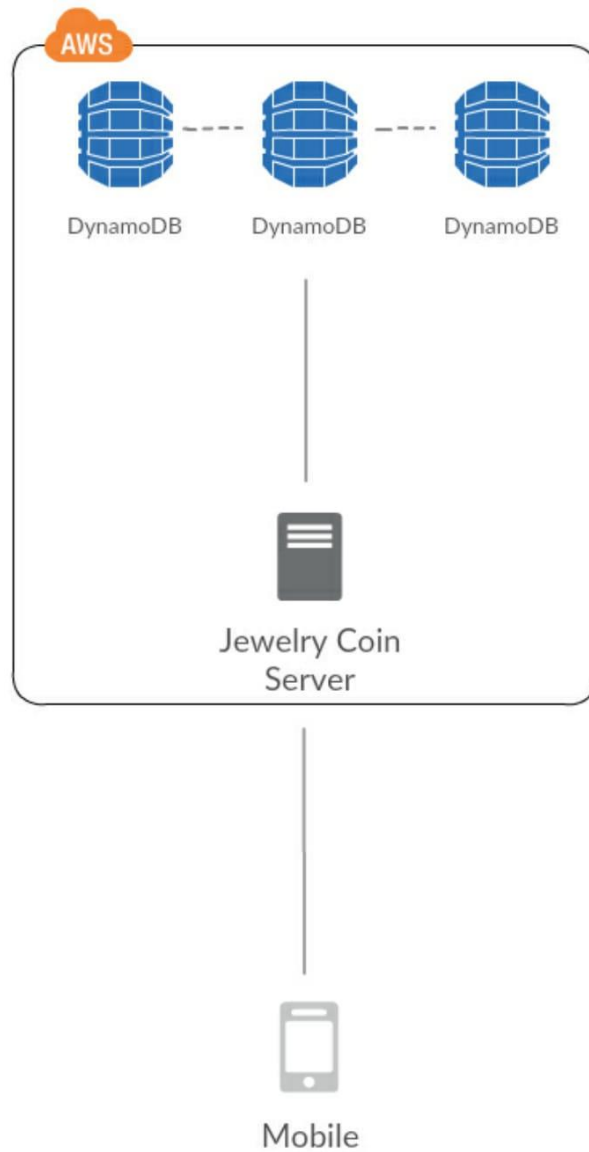
[그림 III-1] 전체 네트워크 구성도

앱 서버와 가상 화폐 서버는 서로 다른 AWSEC2에서 서비스 된다.
MySQL은 AWS RDS를 통해 서비스 되고, 3개의 DynamoDB는 주기적인 동기화를 통해 데이터 무결성을 만족하고 블록 체인 시스템을 만족 한다.



[그림 III-2] 모바일 앱 통신 네트워크 구성도

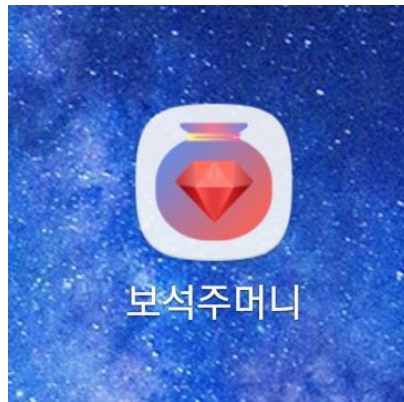
모바일 앱 통신 구성도이다. 앱 서버는 Spring boot를 이용해 개발했고, DB는 MySql을 사용했다. 모두 AWS를 통해 서비스 된다. 로그인이나 기타 커뮤니티 등의 기능은 앱 서버를 이용하고 비트 코인이나 이더리움, 리플 같은 암호 화폐의 시세 정보 등은 공식 API를 이용하거나 블록 체인을 통해 얻는다.



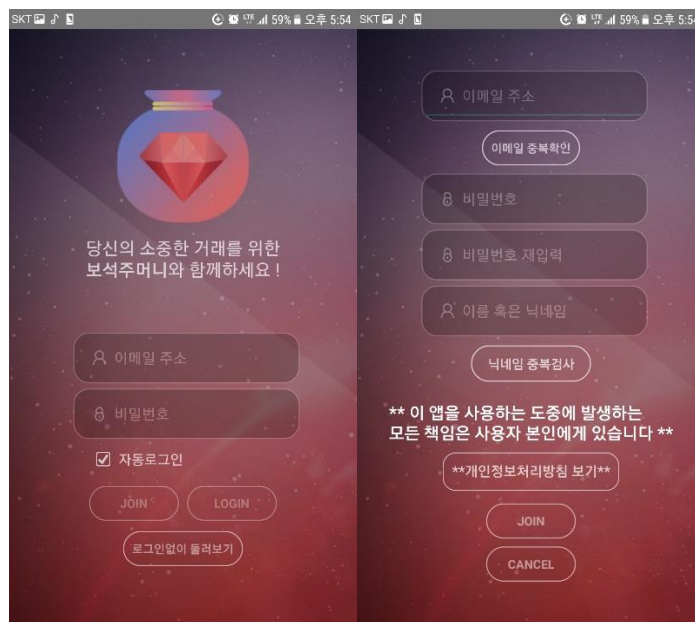
[그림 III-3] 가상 화폐 서버 통신 네트워크 구성도

프로젝트에서 Node.js express로 구축한 임의의 가상 화폐 서버이다. DB는 NoSQL 방식의 AWS에서 제공하는 DynamoDB를 사용했다. 3개의 DB를 사용해 데이터를 동기화 시켜 데이터 무결성을 유지했다. 각 DB는 모두 블록 체인 구조를 하고 있다

3.2 모바일 앱



[그림 III-4] 앱 아이콘
Android 상에서의 앱 아이콘이다.



[그림 III-5] 로그인 화면 / 회원 가입 화면

자동 로그인 기능을 지원하고 아이디는 이메일 주소를 사용한다. 닉네임을 통해 커뮤니티 기능을 이용한다.

분류	KRW	BTC
BTC	5058735.6	1
ETH	370683.04	0.07302
ETC	19916.15	0.003923
XRP	247	0.00004891
LTC	85530	0.01691
DASH	378500	0.07484000

분류	비율(BTC)	순환공급량
BTC	1.00000000	16546900.0 BTC
ETH	0.07174400	94451890.0 ETH
BCH	0.12303200	16561988.0 BCH
XRP	0.00004871	38343841883.0 XRP
LTC	0.01635330	52798057.0 LTC
XEM	0.00006560	8999999999.0 XEM

분류	KR - US	USDKRW
BTC	1055.5 (0.02%)	5057680.09
ETH	1525.2 (0.41%)	369157.87
ETC	175.8 (0.88%)	19740.34
XRP	-0.8 (-0.32%)	247.77
LTC	26.3 (0.03%)	85503.73
DASH	1452.8 (0.38%)	377047.21

분류	Price	Volume	변화율
BTC	\$ 4400.2	\$2696640000	5.15 %
ETH	\$ 318.7	\$1573550000	10.07 %
BCH	\$ 546.5	\$340645000	8.22 %
XRP	\$ 0.2	\$243680000	7.47 %
LTC	\$ 72.6	\$1032750000	13.76 %
XEM	\$ 0.3	\$7540010	8.22 %

데이터 출처: <https://www.cryptocompare.com/>

데이터 출처: <http://coinmarketcap.com/>

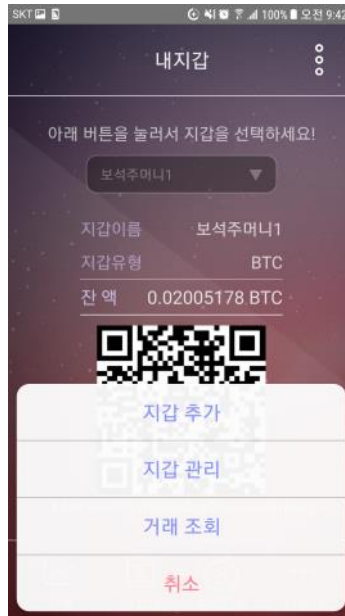
[그림 III-6] 시세 정보

로그인하고 난 뒤 메인 화면이자 시세 정보이다. 5초마다 자동으로 Refresh 되어 새로운 정보가 표시된다. 가장 인기있는 6개의 암호 화폐에 대한 KRW 환율, BTC 환율, 공급량, 시세 등을 보여준다. 이 정보들은 각 암호 화폐가 제공하는 API나 블록 체인 정보를 통해 얻는다. 각 화폐에 대한 원화, 미화의 차이도 보여주기 때문에 다양한 용도로 활용이 가능하다. 모바일 디바이스 기종에 따라 데이터가 로딩되는 시간차가 존재한다.



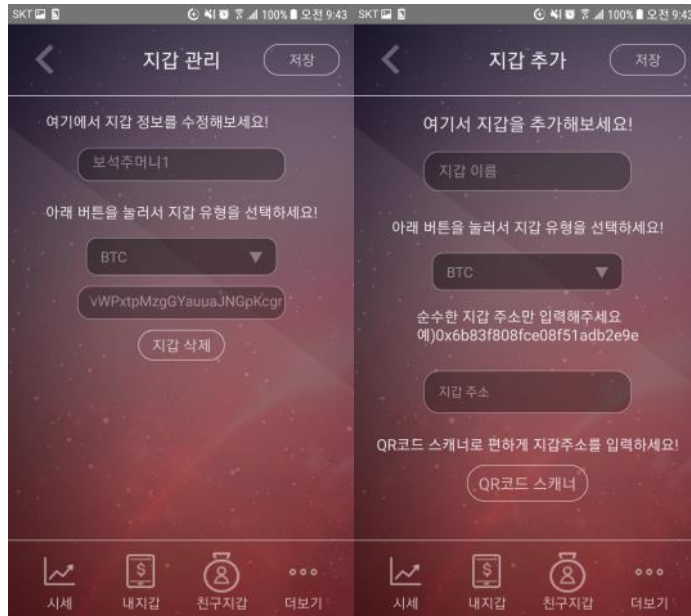
[그림 III-7] 내 지갑

내 지갑 메뉴이다. 내가 등록한 지갑의 이름, 지갑 화폐 단위 유형, 잔액, QR코드가 표시된다. 이 기능은 사용자가 다른 시스템에서 생성한, 이미 존재하는 지갑 계좌를 이곳 시스템에서 관리하는 기능이다.



[그림 III-8] 지갑 관리 메뉴

상단 우측 메뉴를 통해 지갑 관리 메뉴를 이용할 수 있다. 기존에 존재하던 지갑을 이곳 앱에 추가할 수 있고 관리할 수 있다. 또한 블록 체인을 통해 거래 기록을 조회 할 수 있다.



[그림 III-9] 지갑 관리

지갑을 추가하는 기능이다. 기존에 존재하는 지갑의 주소를 적어 넣거나 QR코드 스캐너를 통해 지갑을 등록할 수 있다. 현재 지원하는 유형은 BTC, ETH, ETC, XRP, LTC, DASH이다. 또한 자체적으로 개발한 보석 코인 계좌를 새로 생성할 수 있다.



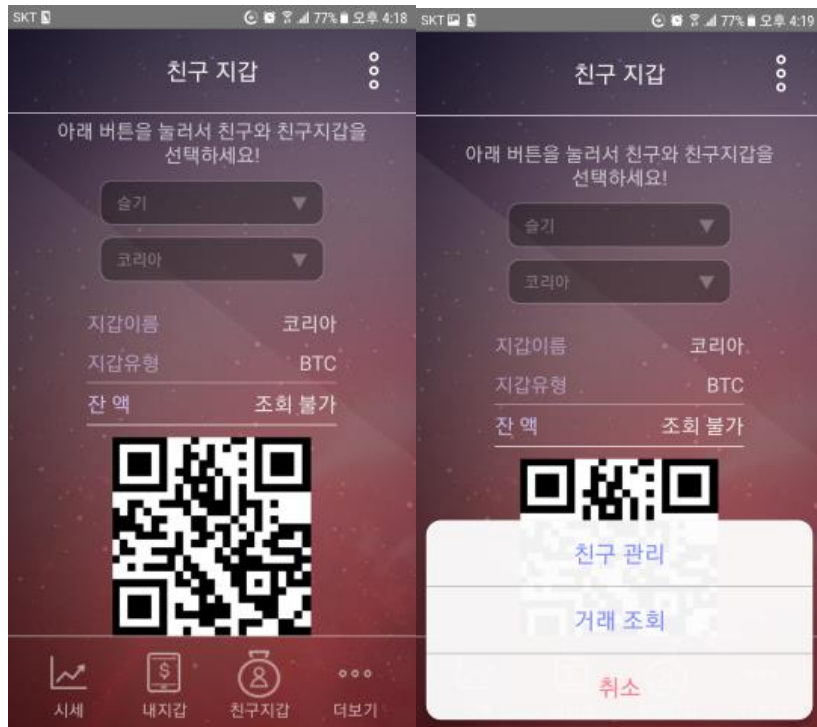
[그림 III-10] QR코드 스캐너

QR코드 스캐너를 통해 이미 존재하는 계좌를 손쉽게 추가할 수 있다.



[그림 III-11] 거래 기록 조회

해당 계좌의 모든 거래 기록을 조회할 수 있다. 이 정보는 블록체인에 저장된 정보를 가져온 것이기 때문에 누구나 계좌 주소만 알면 거래 기록을 알 수 있다. 즉 이미 공개된 정보이다.



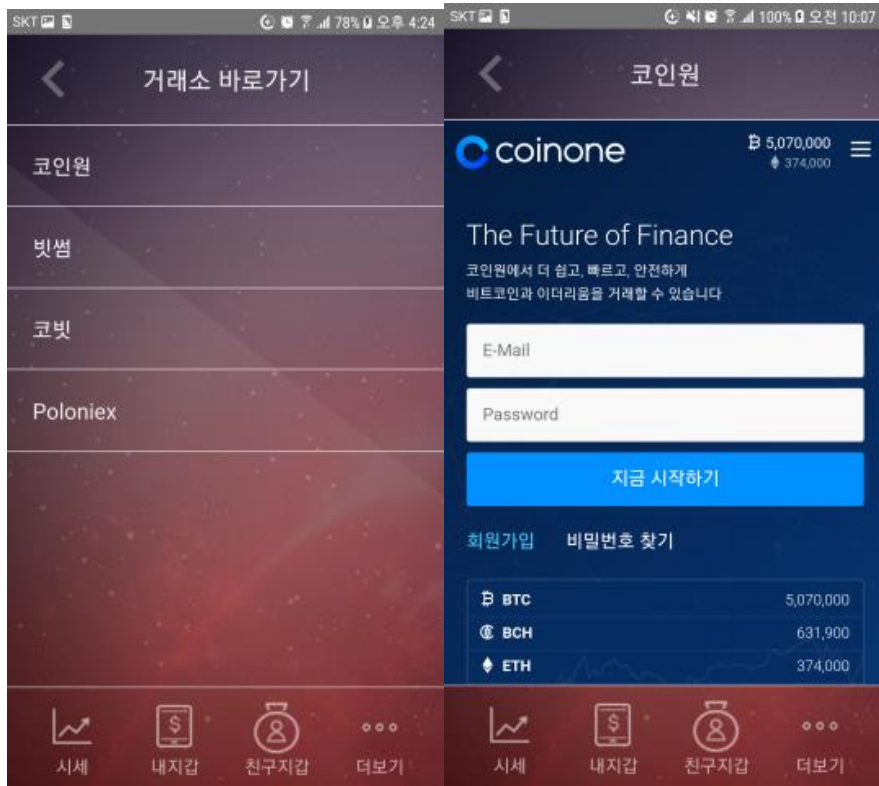
[그림 III-12] 친구 지갑 관리

커뮤니티 기능의 핵심인 친구 기능이다. 친구를 추가하고 친구의 지갑의 정보를 알 수 있다.



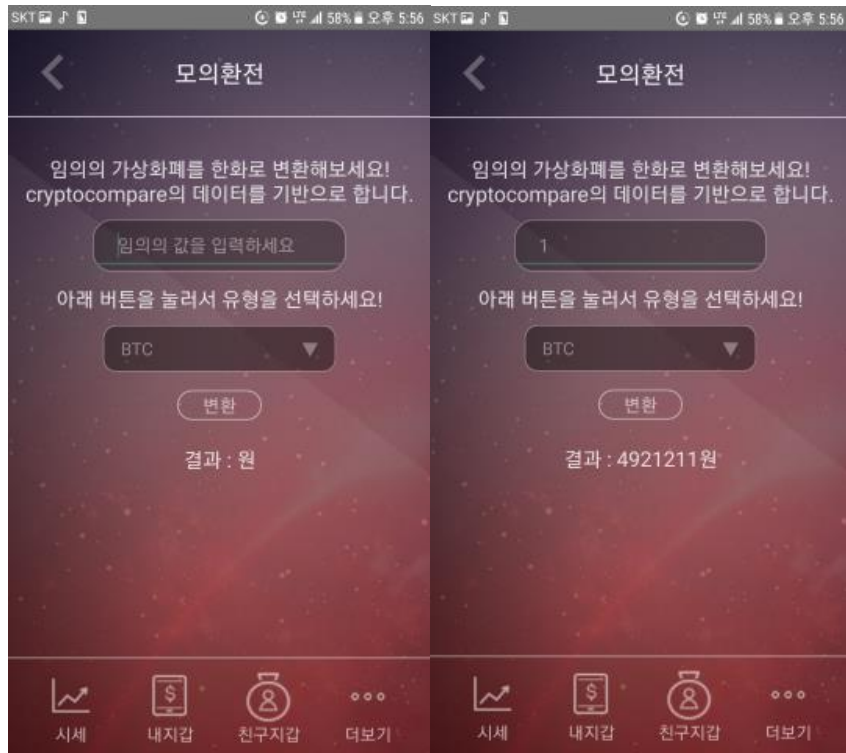
[그림 III-13] 부가 기능

부가적으로 제공하는 기능 메뉴이다. 암호 화폐 거래소, 모의 환전 커뮤니티 게시판, 공지사항 등의 기능이 있다.



[그림 III-14] 거래소

가장 많이 사용하는 거래소 4곳의 링크를 걸어놔 앱에서 바로 이용할 수 있다.
하이브리드 앱으로 개발되었기 때문에 웹 뷰를 사용할 수 있다.



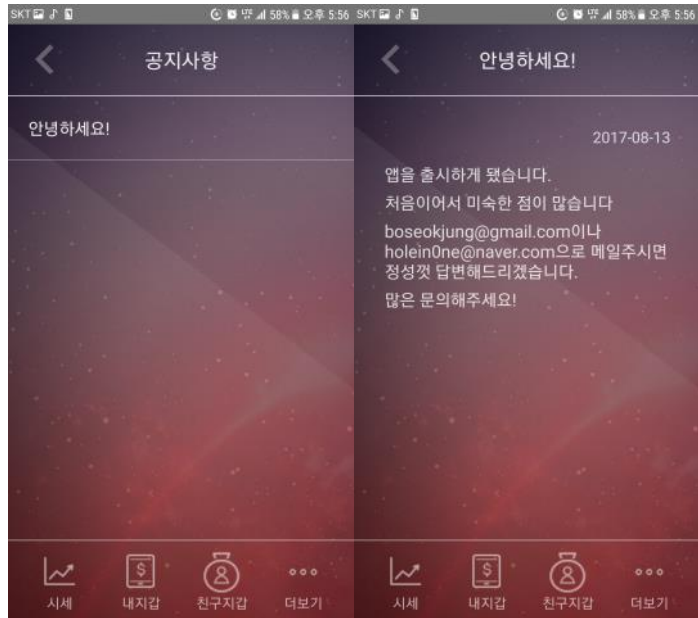
[그림 III-15] 모의 환전

모의 환전 기능이다. 여러가지 암호 화폐의 임의의 값을 입력해 해당하는 시세를 환화로 변환해 준다.



[그림 Ⅲ-16] 커뮤니티 게시판

커뮤니티 게시판 기능이다. 기본적인 게시판 기능에 추천 수가 많은 글 3개는 상단에 표시된다.

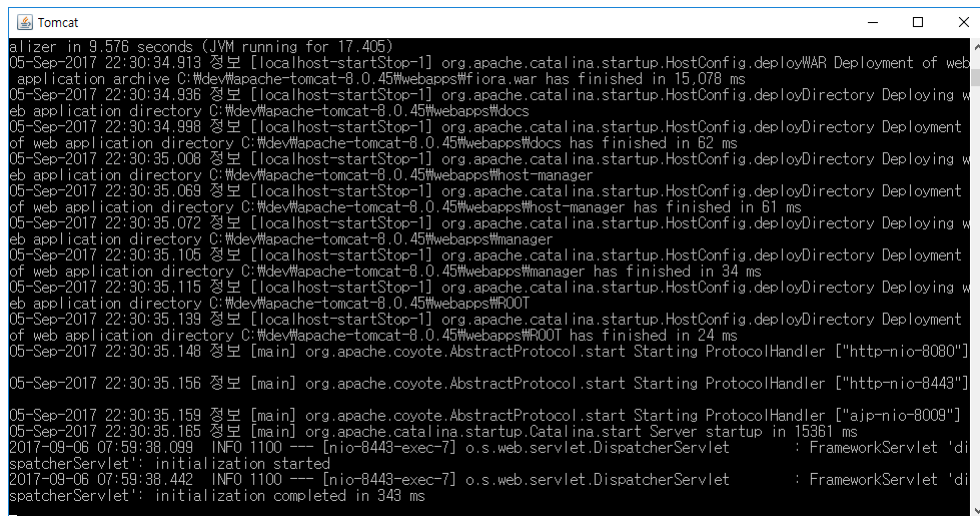


[그림 III-17] 공지사항

3.3 서버

서버는 앱 서버와 가상 화폐(명칭 : 보석 코인)서버로 나뉜다. 앱 서버는 로그인, 지갑 관리, 친구 관리, 커뮤니티 게시판 등 보석 코인 서버를 제외하고 모든 기능을 서비스 하고 있다.

앱 서버는 Spring-boot를 사용해 개발했고, ORM 프레임워크로 Mybatis를 사용했다.

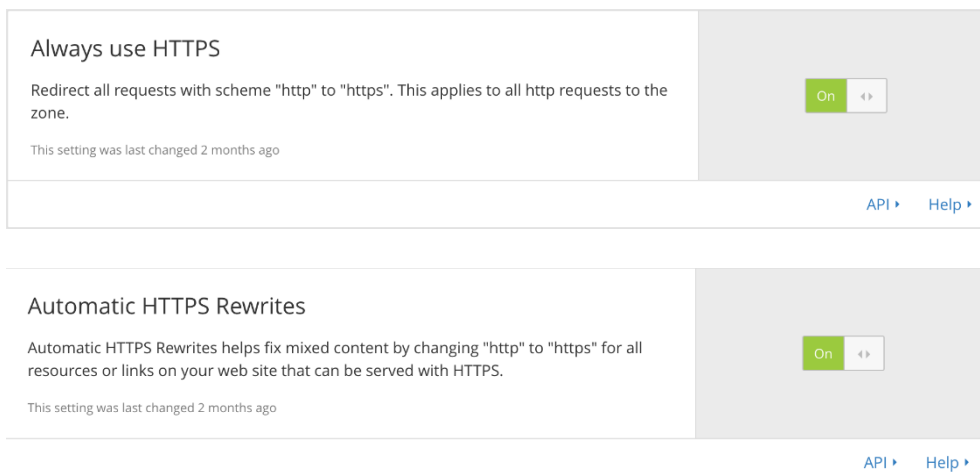
A screenshot of a Tomcat console window titled 'Tomcat'. The window shows a series of log messages from the Apache Catalina startup process. The logs indicate the deployment of web application archives and directories, including 'webapp\resources', 'webapp\WEB-INF', and 'webapp\WEB-INF\classes'. The startup process completes successfully, and the server is ready to accept connections on port 8080. The logs are in Korean and English, showing timestamps and the names of the classes involved in the startup process.

```
05-Sep-2017 22:30:34.913 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployWAR Deployment of web application archive C:\dev\workspace\spring-boot\src\main\resources\resources.war has finished in 15,078 ms
05-Sep-2017 22:30:34.936 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory C:\dev\workspace\spring-boot\src\main\resources\resources
05-Sep-2017 22:30:34.998 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory C:\dev\workspace\spring-boot\src\main\resources\resources has finished in 62 ms
05-Sep-2017 22:30:35.008 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory C:\dev\workspace\spring-boot\src\main\resources\resources
05-Sep-2017 22:30:35.069 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory C:\dev\workspace\spring-boot\src\main\resources\resources has finished in 61 ms
05-Sep-2017 22:30:35.072 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory C:\dev\workspace\spring-boot\src\main\resources\resources
05-Sep-2017 22:30:35.105 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory C:\dev\workspace\spring-boot\src\main\resources\resources has finished in 34 ms
05-Sep-2017 22:30:35.115 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application directory C:\dev\workspace\spring-boot\src\main\resources\resources
05-Sep-2017 22:30:35.139 정보 [localhost-startStop-1] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web application directory C:\dev\workspace\spring-boot\src\main\resources\resources has finished in 24 ms
05-Sep-2017 22:30:35.148 정보 [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8080"]
05-Sep-2017 22:30:35.156 정보 [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8443"]
05-Sep-2017 22:30:35.159 정보 [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["ajp-nio-8009"]
05-Sep-2017 22:30:35.165 정보 [main] org.apache.catalina.startup.Catalina.start Server startup in 15361 ms
2017-09-06 07:59:38.099 INFO 1100 --- [nio-8443-exec-7] o.s.web.servlet.DispatcherServlet : FrameworkServlet 'dispatcherServlet': initialization started
2017-09-06 07:59:38.442 INFO 1100 --- [nio-8443-exec-7] o.s.web.servlet.DispatcherServlet : FrameworkServlet 'dispatcherServlet': initialization completed in 343 ms
```

[그림 III-18] 앱 서버 구동 화면

로그인을 성공하면 서버는 앱으로 토큰 값을 할당하게 된다. 이후 통신을 할 때 헤더 값의 'Authorization' 항목에 토큰 값을 같이 보내줘야 한다.

나머지 기능 통신 할 경우 앱에서 토큰 값과 나머지 data 값을 입력 해 서버로 보내면 REST API 가 요청된 값을 응답해 준다.



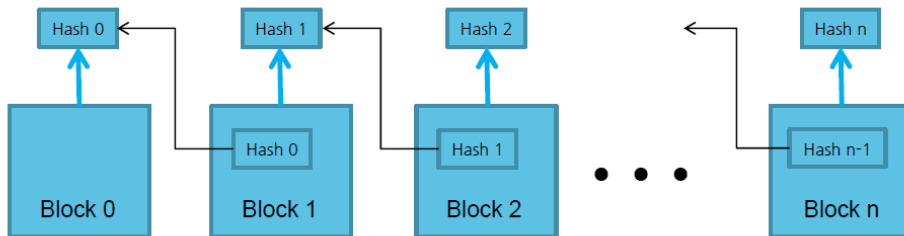
[그림 III-21] https 통신 설정

보안을 위해 도메인 관리 사이트에서 https 통신 설정을 했다. http 로 접속을 해도 https 로 자동으로 리 다이렉트 된다.

보석 코인 서버는 Node.js Express 로 개발되었다. AWS EC2 에 올려 배포 하고, DB 는 AWS DynamoDB 를 사용한다. 노드 서버 역시 API 통신을 통해 앱과 통신하게 된다.

App name	id	mode	pid	status	restart	uptime	cpu	mem	watching
Jewelry	0	fork	13226	online	0	65s	0%	49.8 MB	disabled

[그림 III-22] 노드 서버 운영



[그림 III-23] 블록 체인 구성도

자체 개발한 보석 코인의 블록 체인 구성도이다. DynamoDB 의 기본적인 구조는 저 블록 데이터로 이루어져 있다. 이전 블록의 데이터를 수정하기 위해선 해쉬값도 같이 변경해야 하는데 이렇게 되면 연쇄적으로 모든 데이터를 수정해야 하기 때문에 사실상 블록의 데이터의 변경이 힘들다.

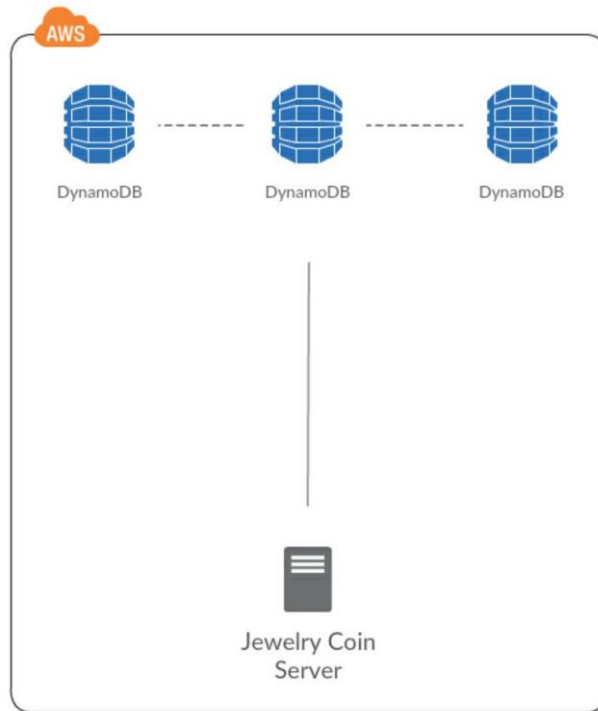
```

let params = {
  TableName: "coin",
  Item: {
    "sequenceNum": uuid1, //시퀀스 인덱스
    "parent_hash": parentHash.getParentHash(), //이전 노드 해쉬값
    data: {
      "address": address, //주소
      "balance": 0, //잔고
      "date": date //만든 날짜
    },
    "hash": h //이번 노드 해쉬값
  }
};

```

[그림 III-24] 계좌 생성시 삽입되는 데이터 구조

보석 코인의 계좌를 생성할 때 블록 체인에 저장되는 데이터의 구조이다. 항상 이전 노드의 해쉬 값이 있어야 되고 이전 노드의 해쉬값과 데이터의 암호화를 통해 이번 노드의 해쉬 값을 생성한다. 다음 노드는 이 해쉬값을 받아서 다시 데이터를 생성한다.



[그림 III-25] 보석 코인 서버 구성도

3개의 DynamoDB가 새로운 블록이 삽입 될 때마다 서로를 확인하고 동기화 한다. 그렇기 때문에 데이터를 조작하려면 3개의 DB를 모두 조작해야 한다. 실제 사용하는 블록 체인에서는 그동안 새로운 데이터가 삽입되기 때문에 사실상 해킹은 불가능 하다.

4. 참고문헌

스프링부트

<https://spring.io/docs>

리액트 네이티브

<https://facebook.github.io/react-native/docs/getting-started.html>

노드js

<https://nodejs.org/ko/docs/>

익스프레스

<http://expressjs.com/ko/4x/api.html>

dynamo db

http://docs.aws.amazon.com/ko_kr/amazondynamodb/latest/developerguide/Introduction.html

RDS

<https://aws.amazon.com/ko/rds/getting-started/>

mysql

<https://dev.mysql.com/doc/>

블록 체인

<https://blockchain.info/ko>

비트 코인

<https://bitcoin.org/ko/bitcoin-for-developers>