제 10 회 소프트웨어공학과 경진대회

작품 제안서

성공회대학교 투표 시스템

Producer

2018 년 3 월

서 론

블록 체인에 대한 전 세계적인 관심으로 블록 체인 기술을 도입한 새로운 서비스가 등장하고 있다. 가장 주목 받는 서비스는 전자 투표 서비스이다. 에스토니아에서는 이미 블록 체인 전자 투표를 진행하고 있다.

이미 작년부터 숭실대학교에서는 총 학생회 선거를 전자 투표 시스템으로 진행하고 있다. 이로써 종이 투표 때 보다 훨씬 효율이 증가했고, 유권자들도 빠르고 쉽게 선거 정보 및 결과를 알 수 있게 되었다.

본 프로젝트에서는 블록 체인 기술을 활용한 전자 투표 시스템을 개발한다. 또한 마이크로 서비스 아키텍처를 선택해 서비스 별로 서버를 개발해 유지 보수 및 추후 모바일 선거로의 확장에 대비한다.

블록 체인 기술의 도입으로 투표의 공정성을 높일 뿐만 아니라 선진 투표 문화 정착에 큰 기여를 할 수 있을 것으로 예상된다.

주요어: Spring-boot, JPA, Mysql, Redis, Vue.js, React.js, AWS, 블록 체인, 투표, 마이크로 서비스 아키텍처

목차

1. 개요

1.1 주제 / 작품 명

성공회대학교 전자 투표 시스템

Producer (당신의 회장에게 투표하세요)

1.2 팀원 소개 및 담당 내용

- 소프트웨어공학과 201232016 배다슬 / PM, 문서 작성, DB 설계, 투표 서버 개발, 투표 정보 및 결과 조회 서버 개발, 배포
- 소프트웨어공학과 201132034 조민국 / 투표 관리 서버 개발
- 소프트웨어공학과 201432043 정보석 / 프론트 엔드 개발, 배포

1.3 개발 동기

- 종이 투표의 불편함을 해소하고 학생들의 투표율 증가 및 선진 투표 문화 정착을 위해 전자 투표 시스템의 도입이 필요하다.

- 이미 작년부터 숭실대학교에서는 총 학생회 투표를 전자 투표 시스템으로 진행하고 있다.

• 부정 선거의 의혹을 해소하고자 블록 체인 기술을 도입한 전자 투표 시스템의 수요가 증가할 것으로 예상된다.

• 블록 체인 기술의 활용을 통해 다양한 분야에 블록 체인 기술 도입 가능성의 확인이 필요하다.

1.4 개발 목적

• 전자 투표 시스템 개발
• 블록 체인 기술의 활용

1.5 개발 목표

• Spring-boot, JPA로 투표 서버 및 투표 관리 서버 개발
• Node.js Express로 투표 정보 및 결과 조회 서버 개발
• React.js, Vue.js로 프론트 엔드 개발
• 블록 체인 구현

1.6 창의성 / 우수성

• 전자 투표의 공정성 및 의미 있는 결과를 위해 블록 체인 기술의 도입
• 모놀리식 서비스 아키텍처가 아닌 서비스 별로 서버를 분리한 마이크로 서비스 아키텍처 도입으로 유지 보수 및 향후 확장성을 증가 시켰다.

• 블록 체인 기술의 활용이 가능하다.

1.7 활용성 / 사업성

• 성공회대학교 총학생회 및 학회장 투표에 도입이 가능하다.
• REST API를 활용해 추후 모바일 선거 시스템으로의 확장이 가능하다.
• 각종 의결 투표 시 사용이 가능하다.

2. 개발

2.1 개발 일정

• 2017.12.28 ~ 2018.01.13 : 시스템 설계 및 DB 설계
• 2018.01.13 ~ 2018.02.25 : 프론트 엔드 및 백 엔드 서버 개발
• 2018.02.25 ~ 2018.03.20 : 블록 체인 기술 도입

2.2 개발 내용

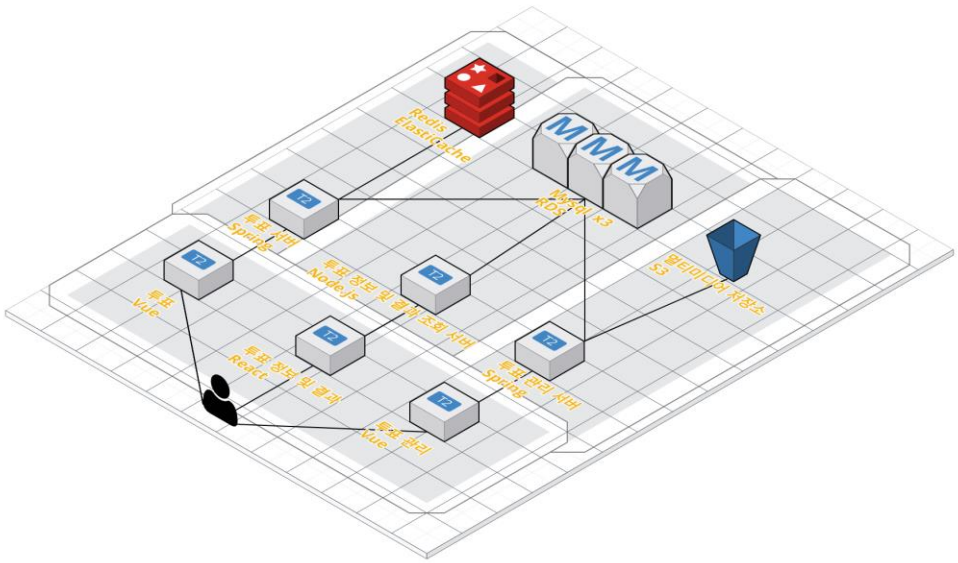
• Spring-boot, JPA로 투표 및 투표 관리 서버 개발
• My SQL로 DB 구축 및 Redis로 세션 관리 DB 구축
• React.js으로 투표 관리 프론트 엔드 개발
• Vue.js로 투표 및 투표 정보 결과 조회 프론트 엔드 개발

• Node.js로 투표 정보 및 결과 조회 서버 개발

• AWS 배포

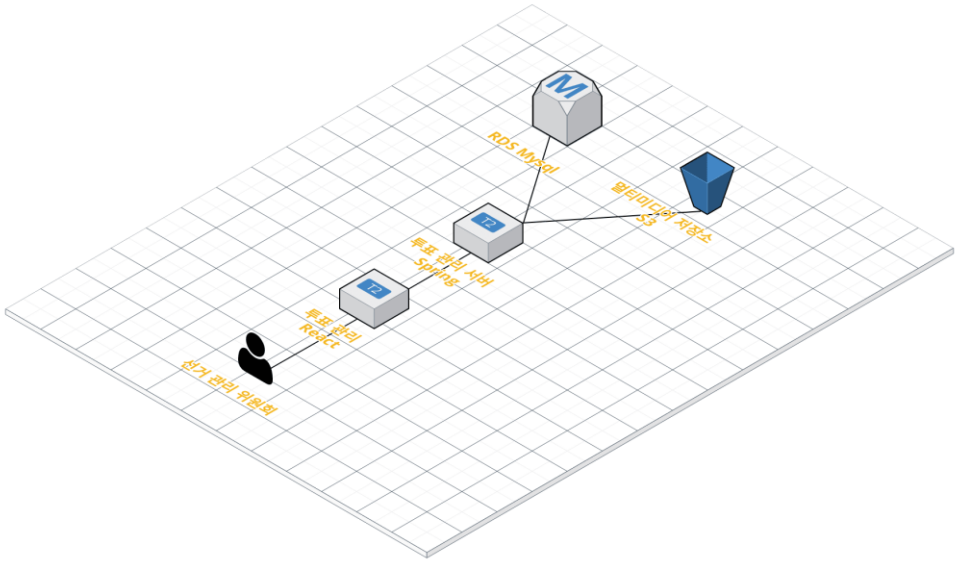
3. 구성

3.1 시스템 구성



[그림 Ⅲ-1] 전체 시스템 구성도

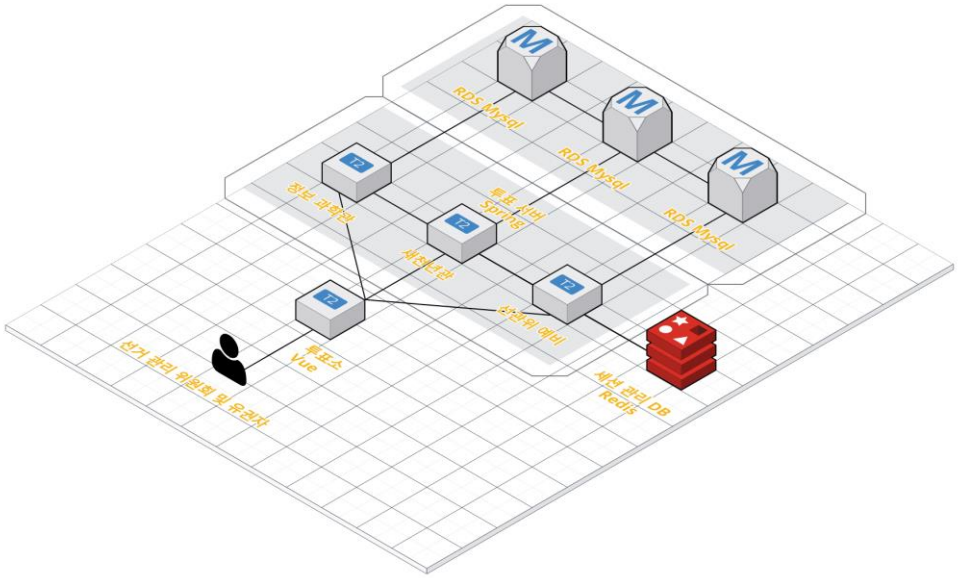
모든 서버는 AWS를 통해 서비스 된다. MySql은 AWS RDS를 통해 배포 되고, Redis는 AWS ElasticCache, 멀티미디어 저장소는 AWS S3을 사용한다. 나머지 백엔드 서버, 프론트 엔드 서버는 AWS EC2를 통해 배포 된다. 개발 단계에선 하나의 Instance를 통해 서비스 되었지만 실제는 각 Instance마다 서비스 된다.



[그림 Ⅲ-2] 투표 관리 시스템 구성도

선거 관리 위원회가 사용하는 투표 관리 시스템의 구성도이다. 선관위는 이 시스템을 이용해 투표를 등록, 수정, 삭제 등 투표 관리, 유권자 관리, 투표 시간 관리, 선관위 명단을 관리한다. 프론트 엔드는 React.js로 개발 되었으며, 서버는 Spring-boot, JPA로 개발되었다. 멀티미디어 저장은 AWS S3에 저장 되며, DB는 AWS RDS를 사용한다.

선거가 진행되기 전, 선관위는 이 시스템을 통해 미리 선거를 등록해야 한다. 선거 정보 및 대상 학과, 시간 정보 등을 설정하고 후보자 정보를 입력해야 한다. 그 다음 유권자 명단을 excel 파일로 정리해 업로드 하면 유권자 명단이 저장된다.

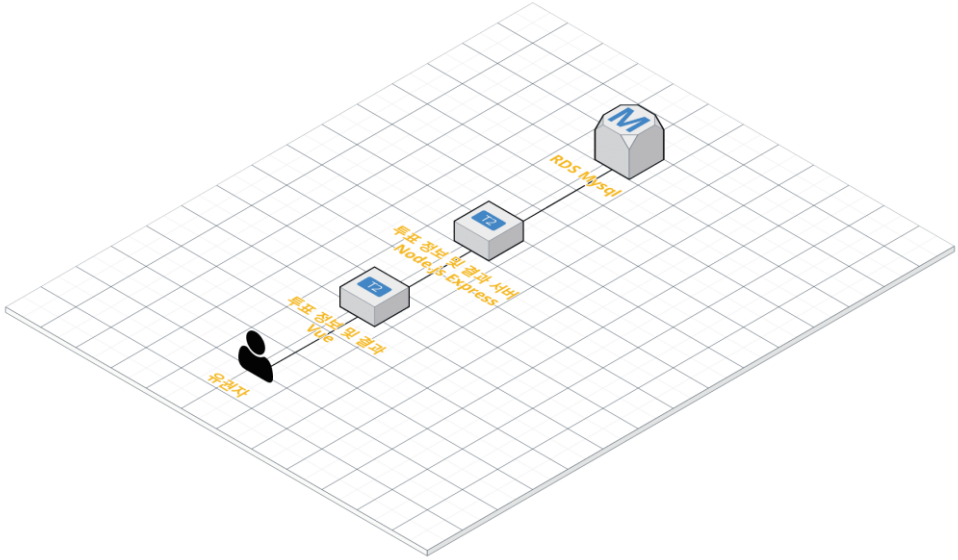


[그림 Ⅲ-3] 선거 시스템 구성도

실제 투표가 이루어지는 시스템이다. 실제 성공회대학교 총학생회 투표에선 2개의 투표소를 운영한다. 여기에 선관위 예비 서버를 포함해 총 3개의 서비스가 운영된다. 각 투표소에서 유권자들이 투표한 데이터는 블록 체인으로 분산되 저장 되어 해킹이 어려워 지게 된다.

유권자는 투표소로 가 자신의 신원을 확인하게 되면 인증번호를 발급 받게 된다. 이 인증번호는 DB에 저장이 되지만, 유권자 실제 정보와 같이 저장 되지 않기 때문에 인증번호가 누구의 인증번호인지 알 수 없다. 유권자는 해당 인증번호를 가지고 투표소의 pc를 통해 로그인을 하고 난 뒤, 후보자에 투표를 하면 선거는 종료된다.

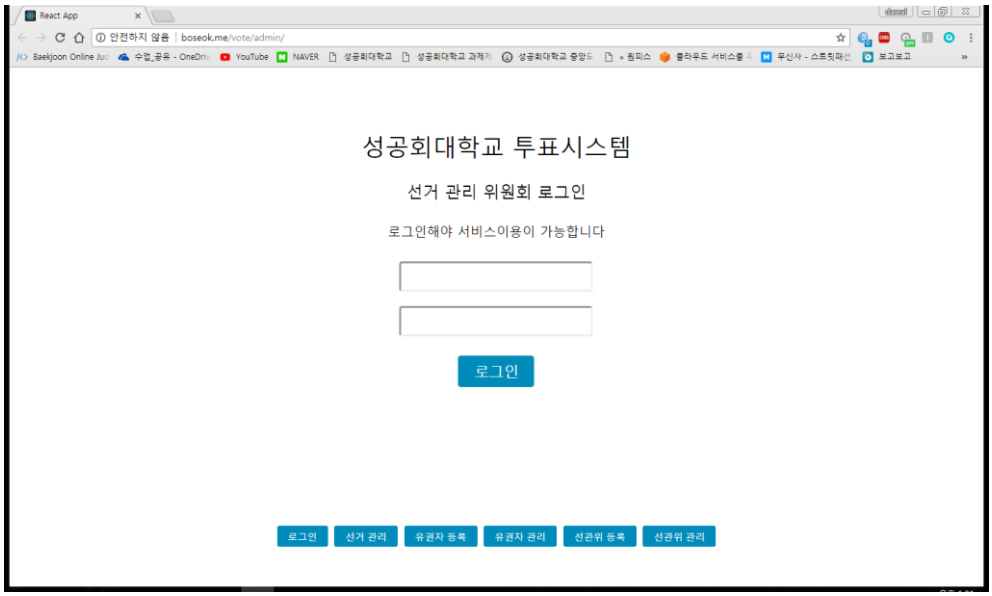
프론트 엔드는 Vue.js로 개발 되었고, 서버는 Spring-boot, JPA로 개발 되었다. 중복 로그인을 막기 위해 세션 서버로 Redis AWS ElaticChach를 사용하였다. 개발 중엔 블록체인이 저장되는 DB를 스키마로 구분하였지만, 실제 서비스에선 개별적인 DB에 저장된다.



[그림 Ⅲ-4] 투표 정보 및 결과 조회 시스템 구성도

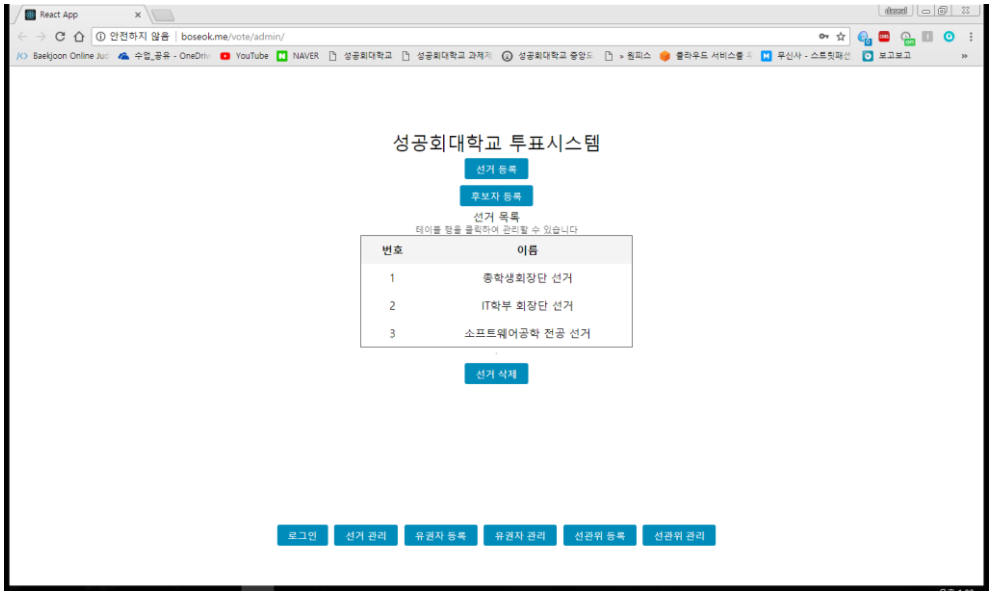
투표 중에 실시간 투표율, 나의 선거권 확인, 선거 결과 확인 등 사용자가 가장 많이 사용하는 투표 정보 및 결과 조회 시스템이다. Node.js Express 으로 개발 되었으며 프론트 엔드는 Vue.js로 개발 되었다.

3.2 프론트 엔드



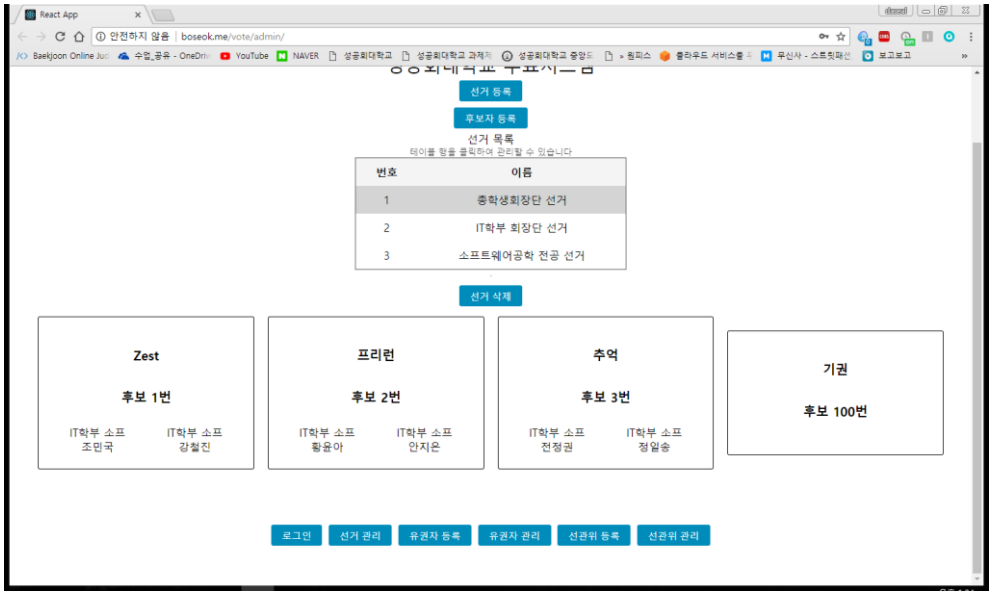
[그림 Ⅲ-5] 투표 관리 시스템 메인 화면

선관위가 사용하는 투표 관리 시스템의 메인 화면이다. 선거 관리, 유권자 관리, 유권자 등록, 선관위 관리 기능이 있다.



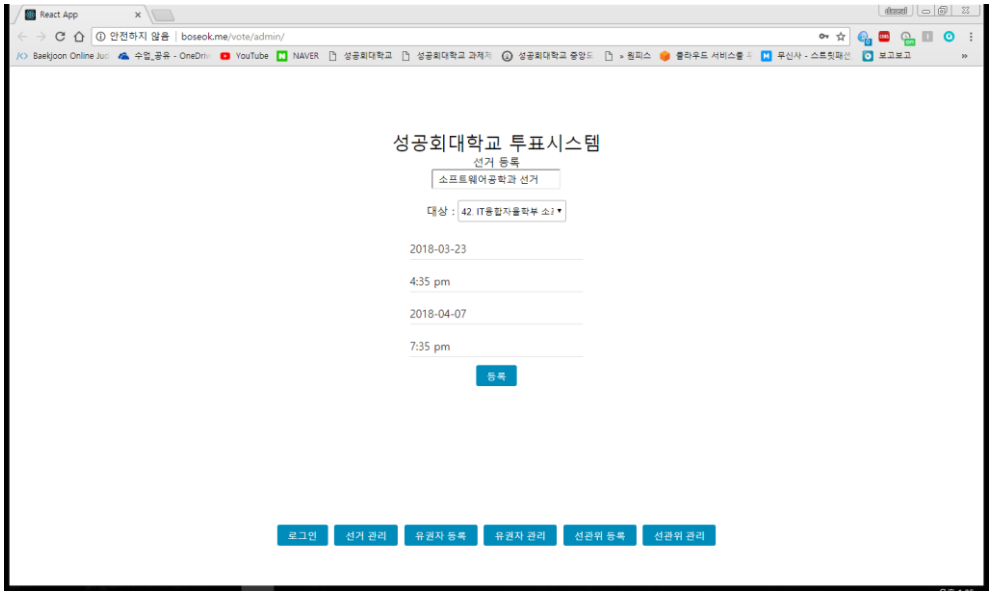
[그림 Ⅲ-6] 선거 / 후보자 등록

선거를 등록하고, 해당 선거에 후보자를 등록하는 페이지이다.



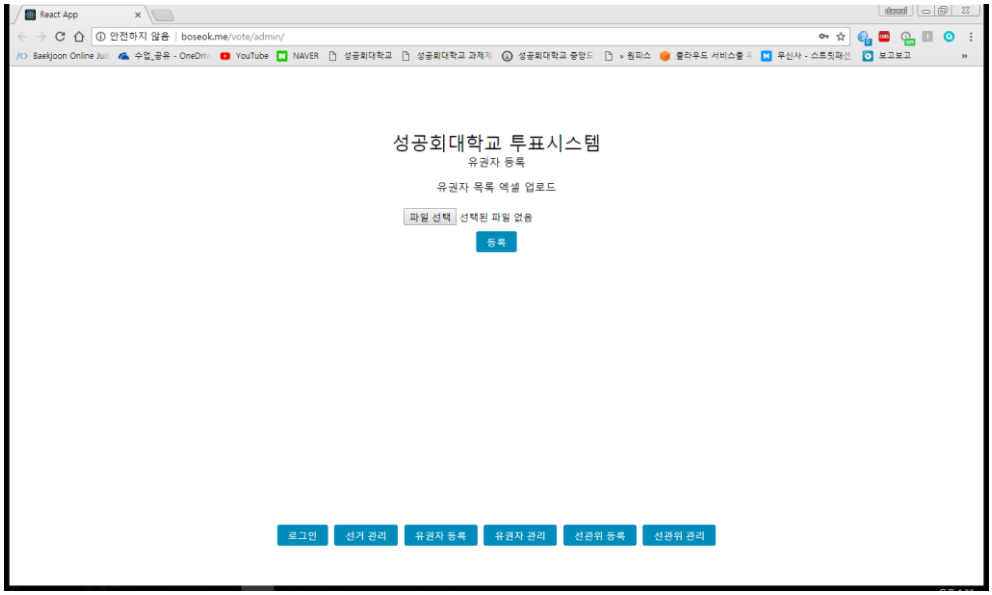
[그림 Ⅲ-7] 선거 관리

선거를 선택하면 현재 선거에 출마중인 후보자 목록과 기호가 보이게 된다.



[그림 Ⅲ-8] 선거 등록

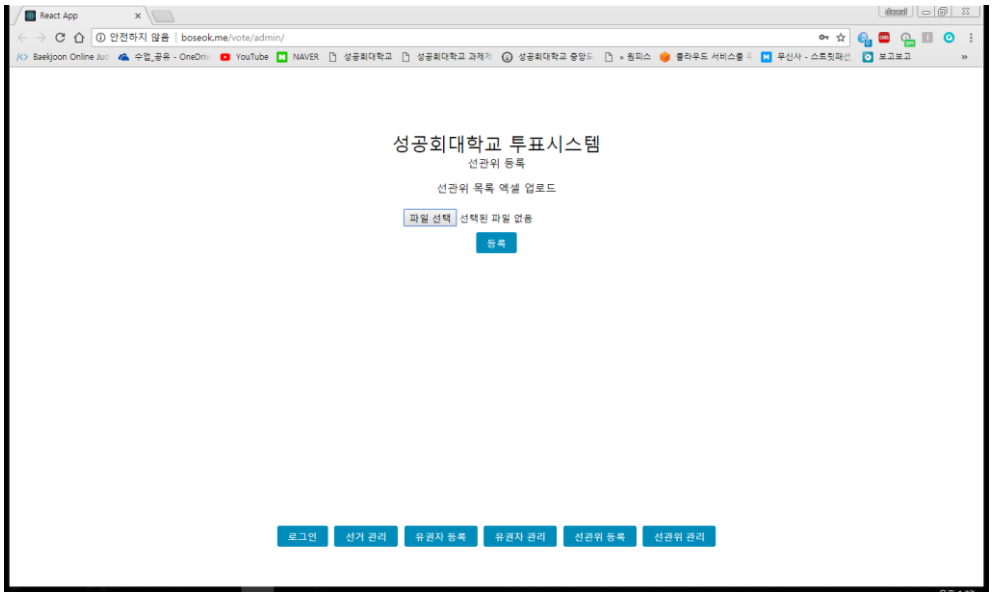
새로 선거를 등록할 수 있다. 선거 이름, 선거의 대상 학과, 선거 시간을 입력한다.



[그림 Ⅲ-9] 유권자 등록

유권자를 등록하는 페이지이다. 엑셀 파일로 유권자 명단을 업로드하면 자동으

로 DB에 저장된다.

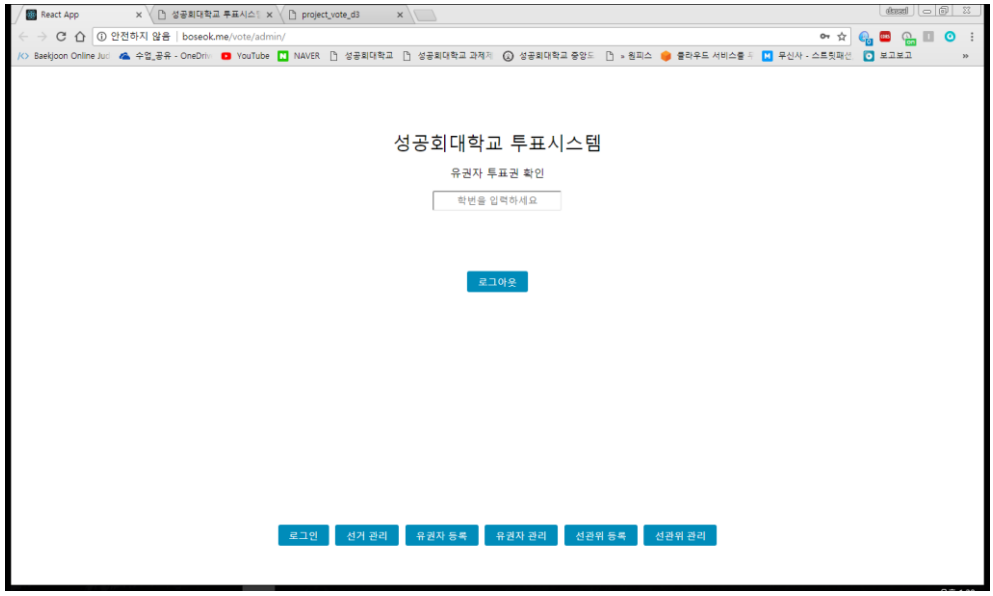


[그림 Ⅲ-10] 선관위 등록

선관위를 등록하는 페이지이다. 해당 페이지는 오직 선거관리 위원장만 이용할 수 있다. 유권자 명단과 마찬가지로 엑셀 파일 업로드를 통해 이루어진다.

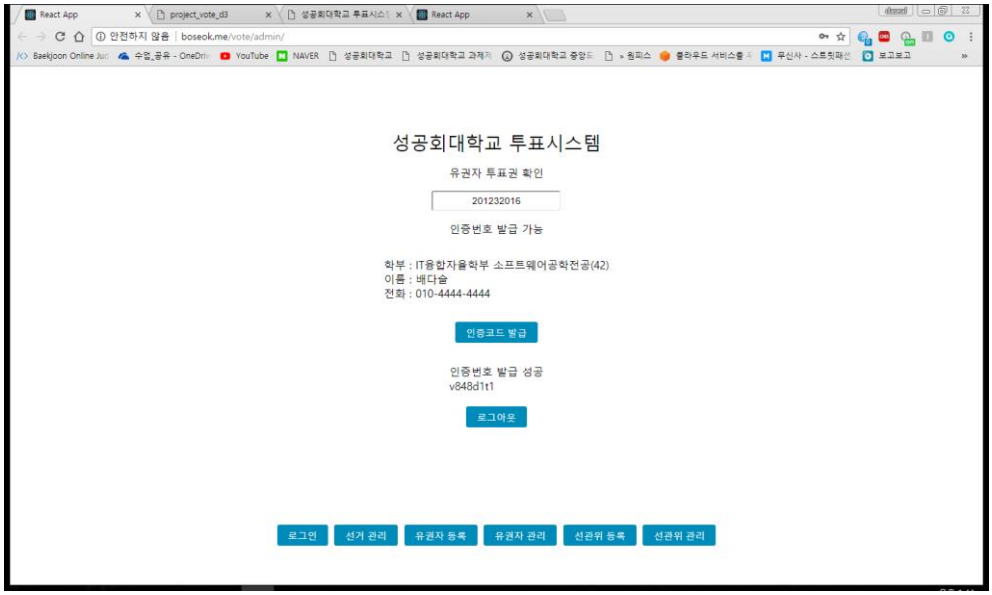
[그림 Ⅲ-11] 선거 관리 위원회 명단

현재 등록된 선거 관리 위원회 명단이다.



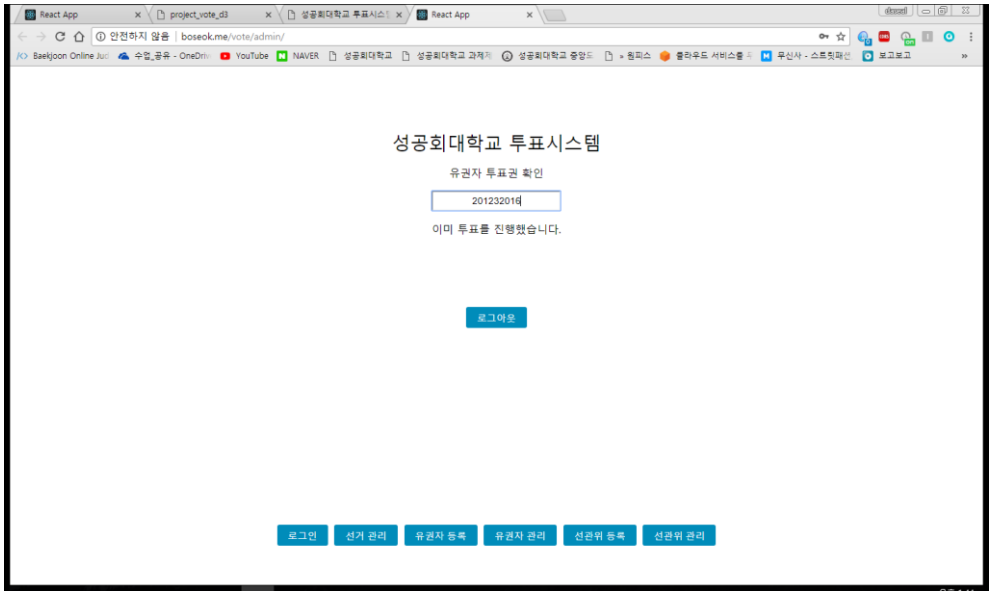
[그림 Ⅲ-12] 선관위 인증번호 발급 페이지

선관위가 유권자를 확인하고 유권자에게 인증번호를 발급하는 페이지이다.

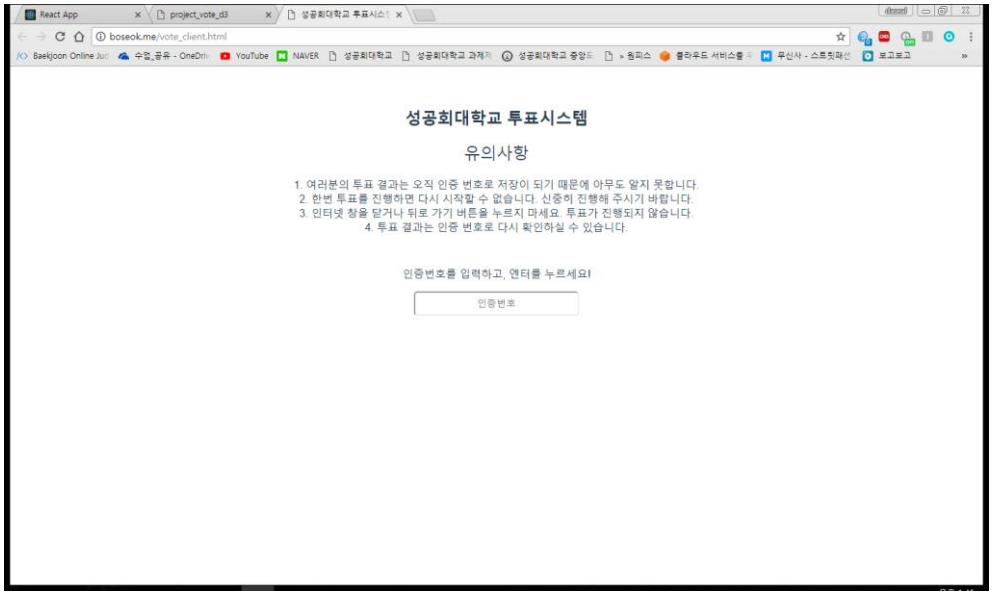


[그림 Ⅲ-13] 인증 번호 발급

유권자의 학번을 토대로 인증번호를 발급한다. 만약 유권자가 선거권이 없거나, 이미 선거를 진행하였을 경우 인증번호가 발급되지 않는다.



[그림 Ⅲ-14] 이미 투표를 진행 한 경우

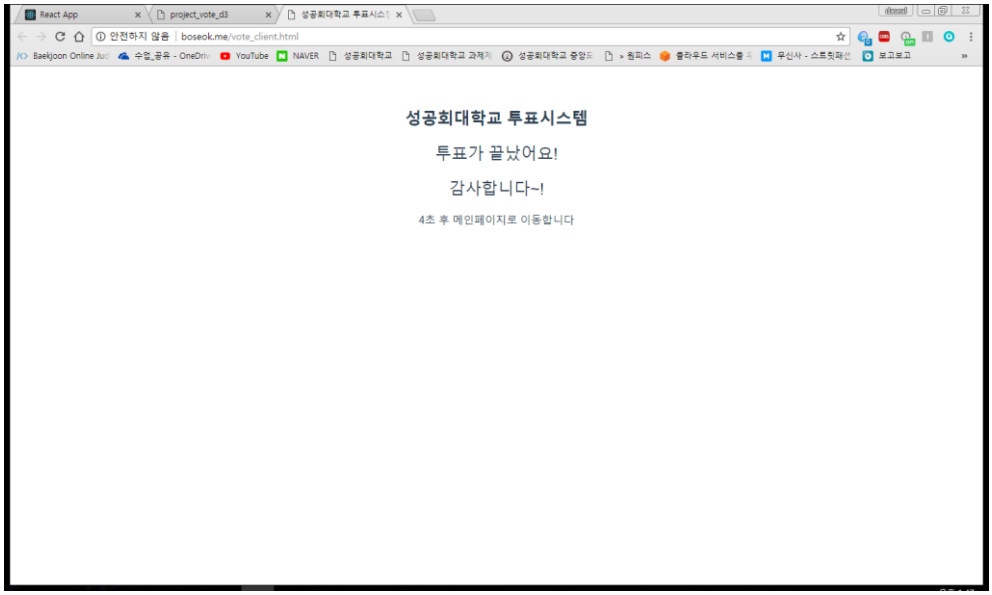


[그림 Ⅲ-15] 투표 페이지

유권자가 투표소에서 사용하는 투표 페이지이다. 선관위로부터 받아온 인증번호로 로그인을 하면 된다.

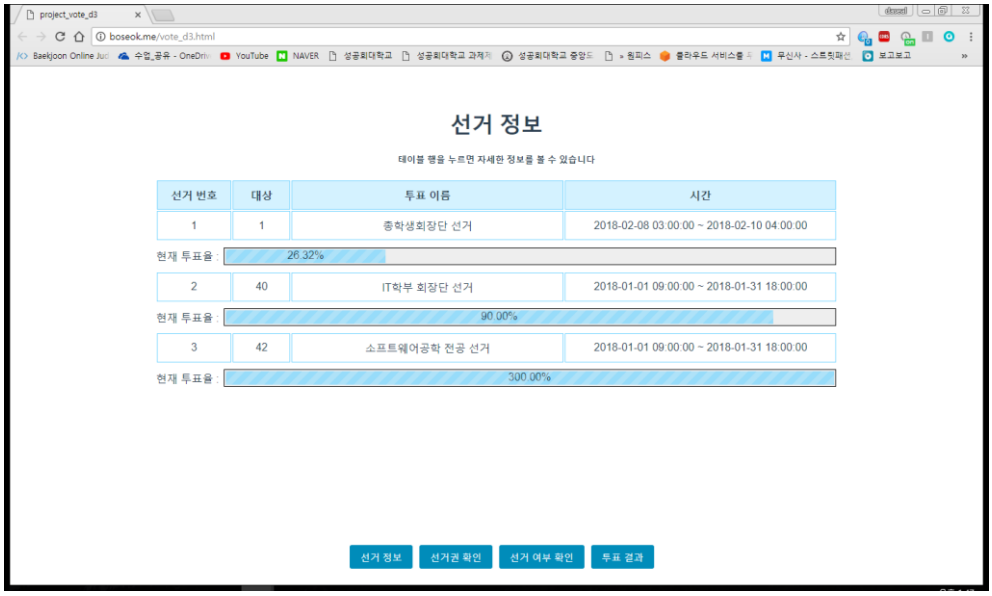
[그림 Ⅲ-16] 후보자 목록

유권자의 학과에 맞게 투표 및 후보자 목록이 보이게 된다.



[그림 Ⅲ-17] 투표 완료

투표를 완료하고 난 뒤 자동으로 메인 페이지로 이동한다.

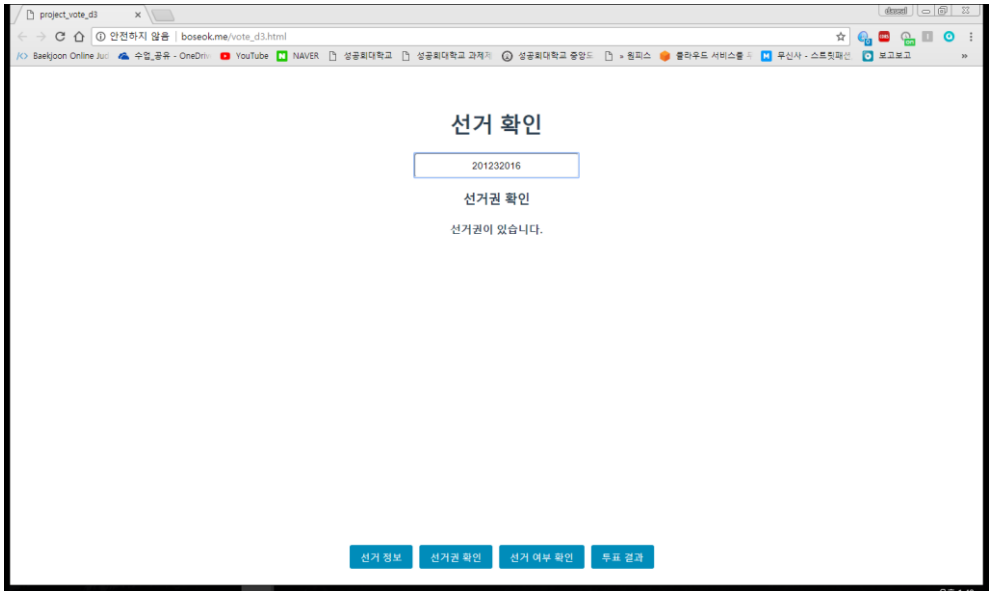


[그림 Ⅲ-18] 선거 정보 및 결과 조회 페이지

실시간 투표율, 선거 정보 및 결과 등을 조회할 수 있는 페이지이다.

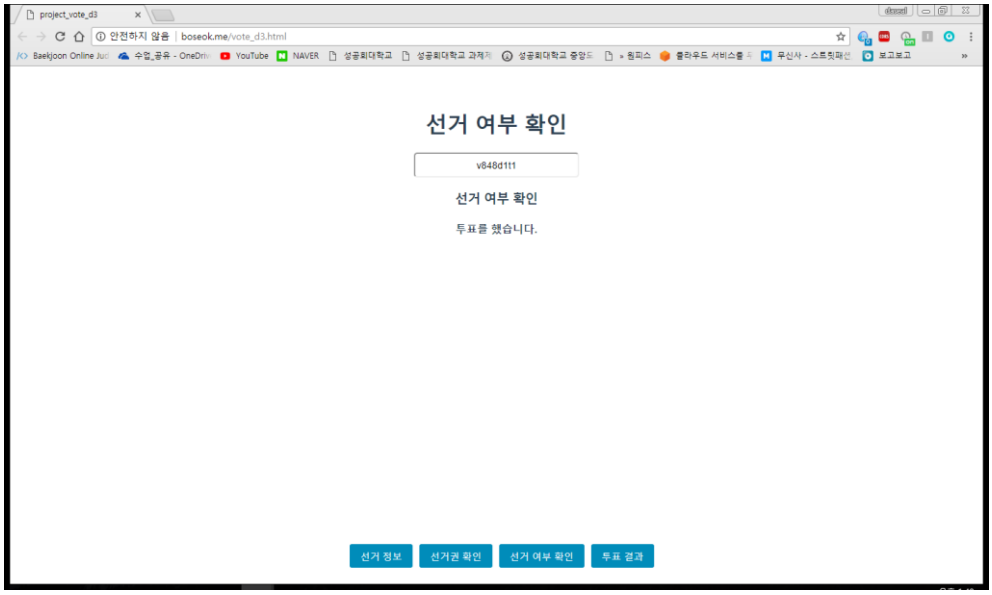
[그림 Ⅲ-19] 선거 정보 조회

각 투표에 출마한 후보자와 선거 기호를 알 수 있다.



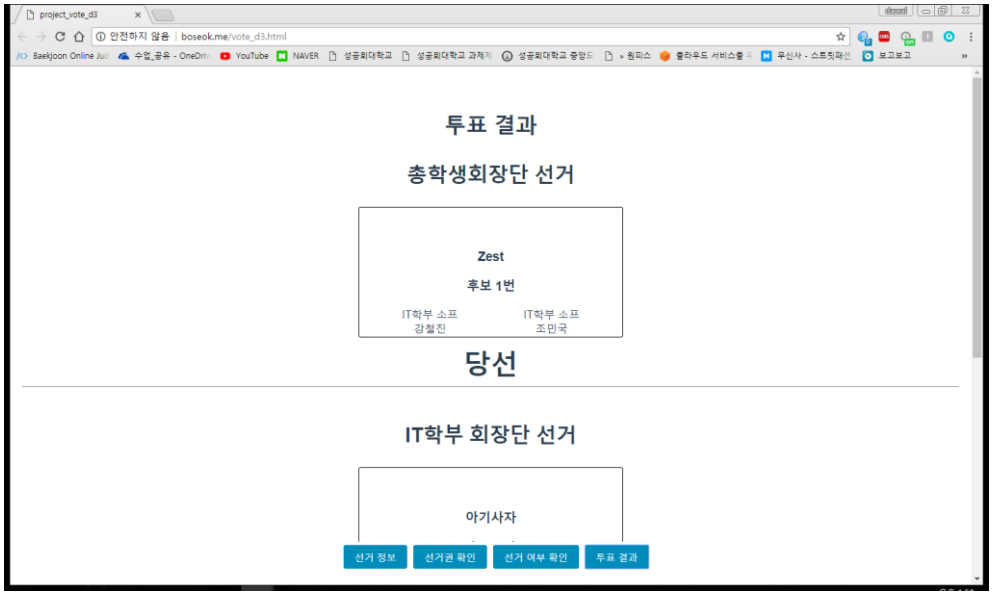
[그림 Ⅲ-20] 선거권 확인

자신이 선거권이 있는지 확인 할 수 있다.



[그림 Ⅲ-21] 성공적인 투표 여부

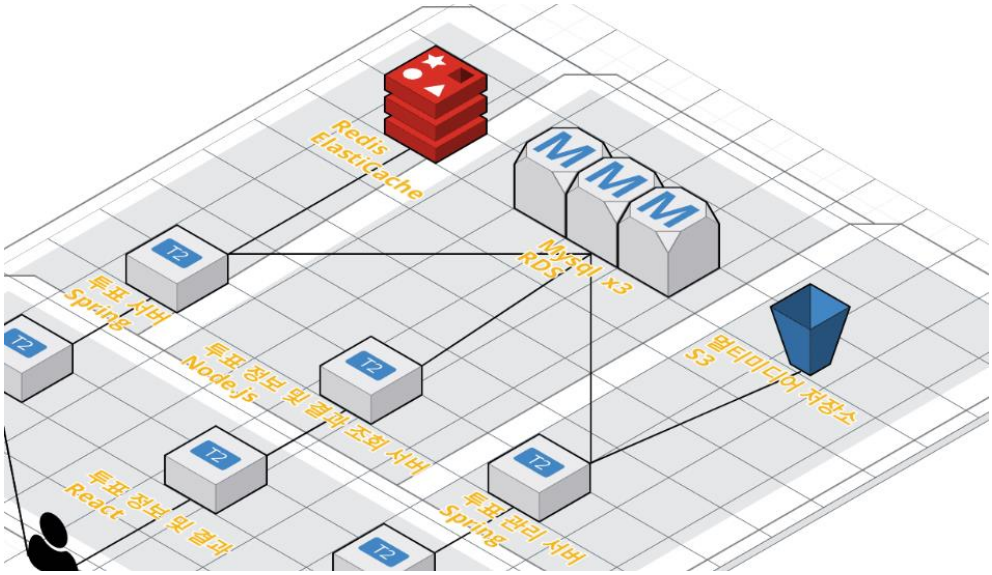
투표가 성공적으로 진행 되었다면 인증번호로 투표 완료 여부를 알 수 있다.



[그림 Ⅲ-22] 선거 결과 페이지

투표가 모두 종료 된 후 선거 결과를 조회하는 페이지이다.

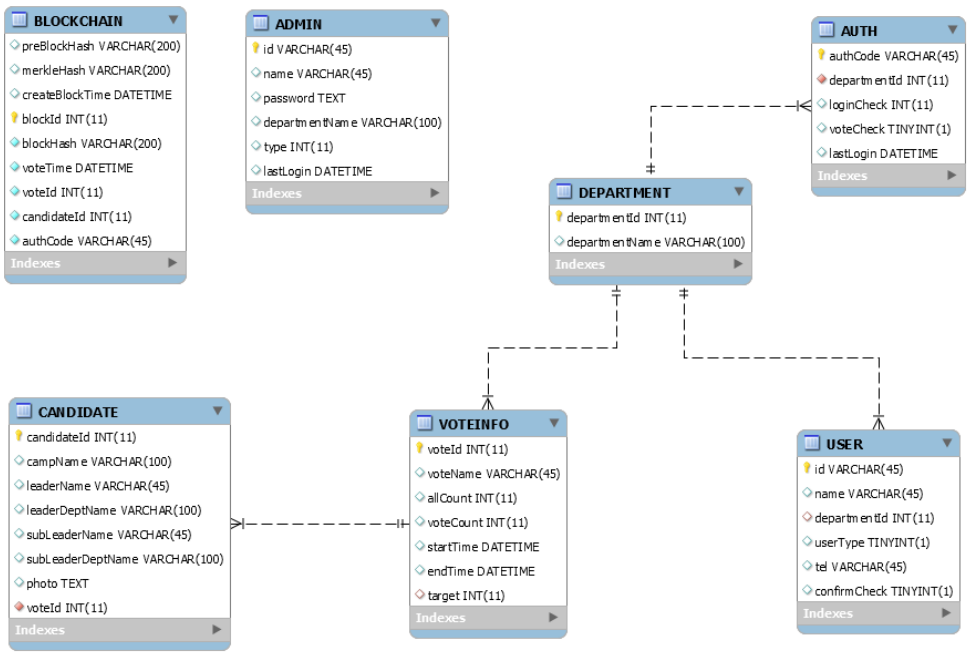
3.3 서버



[그림 Ⅲ-23] 서버

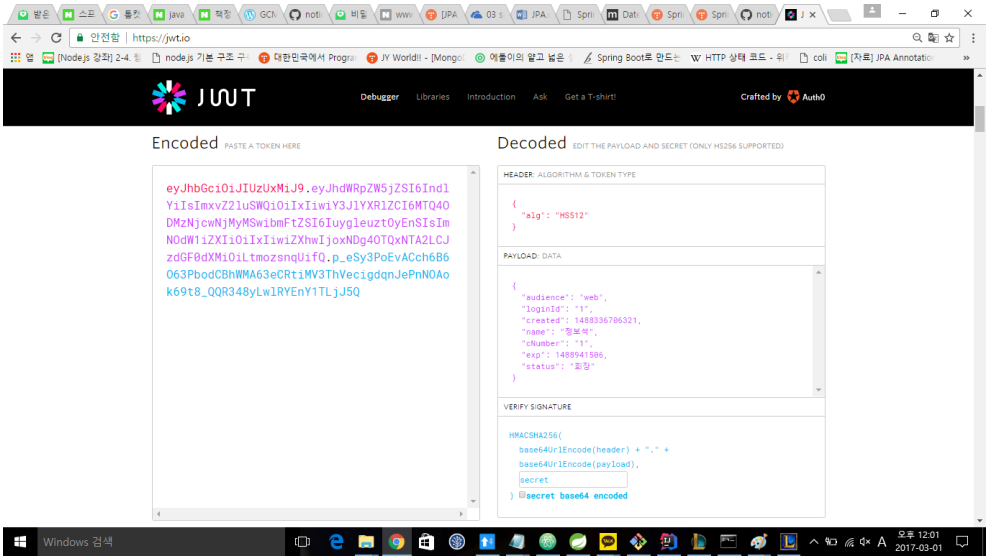
하나의 서버에 모든 서비스가 구동중인 모놀리식 서비스 아키텍처가 아닌 서비스 별로 서버를 분리한 마이크로 서비스 아키텍처 구조로써, 투표 및 투표 관리 서버는 Spring-boot를 사용해 개발했고, JPA를 사용해 MySql을 사용했다. 투표 정보 및 결과 조회 서버는 Node.js Express을 사용했다.

실제 운용 시 투표 서버는 총 3대가 운용 될 예정이지만, 개발 중엔 예산 때문에 하나의 서버에서 3개의 서비스를 실행하는 것으로 대신했다. 각 서비스에서 사용하는 블록 체인 DB도 같은 물리적 DB에서 3개의 스키마로 사용하는 것으로 대신했다.



[그림 Ⅲ-24] ERD 다이어그램

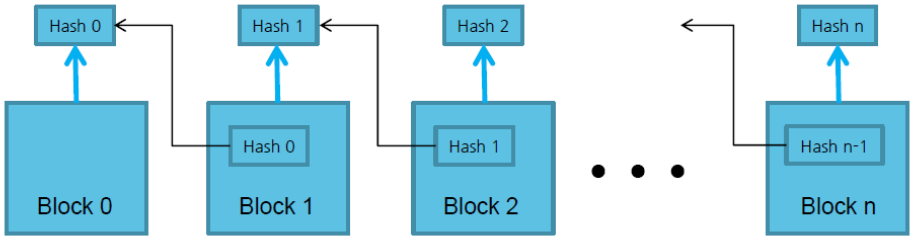
실제 서비스 운영 시 블록 체인 DB는 개별적으로 저장된다. 인증 코드는 누구의 것인지 알 수 없는 DB 구조이다. 인증 코드가 발급 될 때 학과 정보도 같이 저장이 되기 때문에 인증번호만으로도 어느 선거에 투표를 해야 하는 지 알 수 있다.



[그림 Ⅲ-25] JWT에서 생성한 토큰

로그인을 성공하면 서버는 웹으로 토큰 값을 할당하게 된다. 이후 통신을 할 때 헤더 값의 'Authorization' 항목에 토큰 값을 같이 보내줘야 한다. 또한 이 토큰 값들은 Redis 서버에 같이 저장이 되기 때문에 중복 로그인 방지나, 로그아웃의 처리가 가능해진다.

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



[그림 Ⅲ-26] 블록 체인 구성도

투표 값들은 블록 체인으로 구성되어 서버에 저장이 된다. 먼저 각 투표소의 블록 체인을 처음부터 검사를 한 뒤, 블록체인에 이상이 없다면 새로 블록 체인을 형성해 삽입 된다. 그 뒤 나머지 2 개의 투표 서버에 현재 저장된 블록 체인 데이터를 보낸다. 각 투표 서버는 다시 자신들의 블록 체인이 완벽한지 검사를 진행하고 이상이 없다면 전송 받은 블록 체인을 저장한다. 현재는 서버가 3 개라 이러한 서비스가 금방 종료되지만, 블록 체인 시스템이 거대해지게 되면 블록 체인의 분기와 병합에 신경을 써야 할 것이다.

각 블록마다 저장되는 데이터는 이중 해시를 통해 값이 암호화 되며 이 때 Key 를 섞어 해시 하기 때문에 블록의 실제 데이터를 알아내기는 어렵다. 설령 알아 냈다 하더라도 나머지 2 개의 DB 의 데이터를 모두 수정해야 실제 값이 변경되기 때문에 블록 체인의 해킹은 불가능하다. 만약 누군가 블록체인의 데이터를 임의로 변경한 흔적이 발견되면, 블록체인 검사 시 이를 확인 할 수 있고, 선거는 강제로 종료된다.

4. 참고문헌

Spring-boot

https://spring.io/docs

Spring Data JPA

https://docs.spring.io/spring-data/jpa/docs/2.0.5.RELEASE/reference/html/

React.js

https://reactjs.org/docs/hello-world.html

Vue.js

https://kr.vuejs.org/v2/guide/index.html

Node.js

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

Express

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

RDS

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

mysql

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

Redis

https://redis.io/documentation

BlockChain

https://blockchain.info/ko

JWT

https://jwt.io/