

안녕하세요!
끊임없는 도전과 몰입으로
성장하는 조용현입니다

고객에게 새롭고

고객이 신뢰할 수 있는

서비스를 개발하는 Backend Engineer

Projects

01.

Mokkozi

webRTC 기반 소개팅 서비스

2021.10.11 ~ 2021.11.19

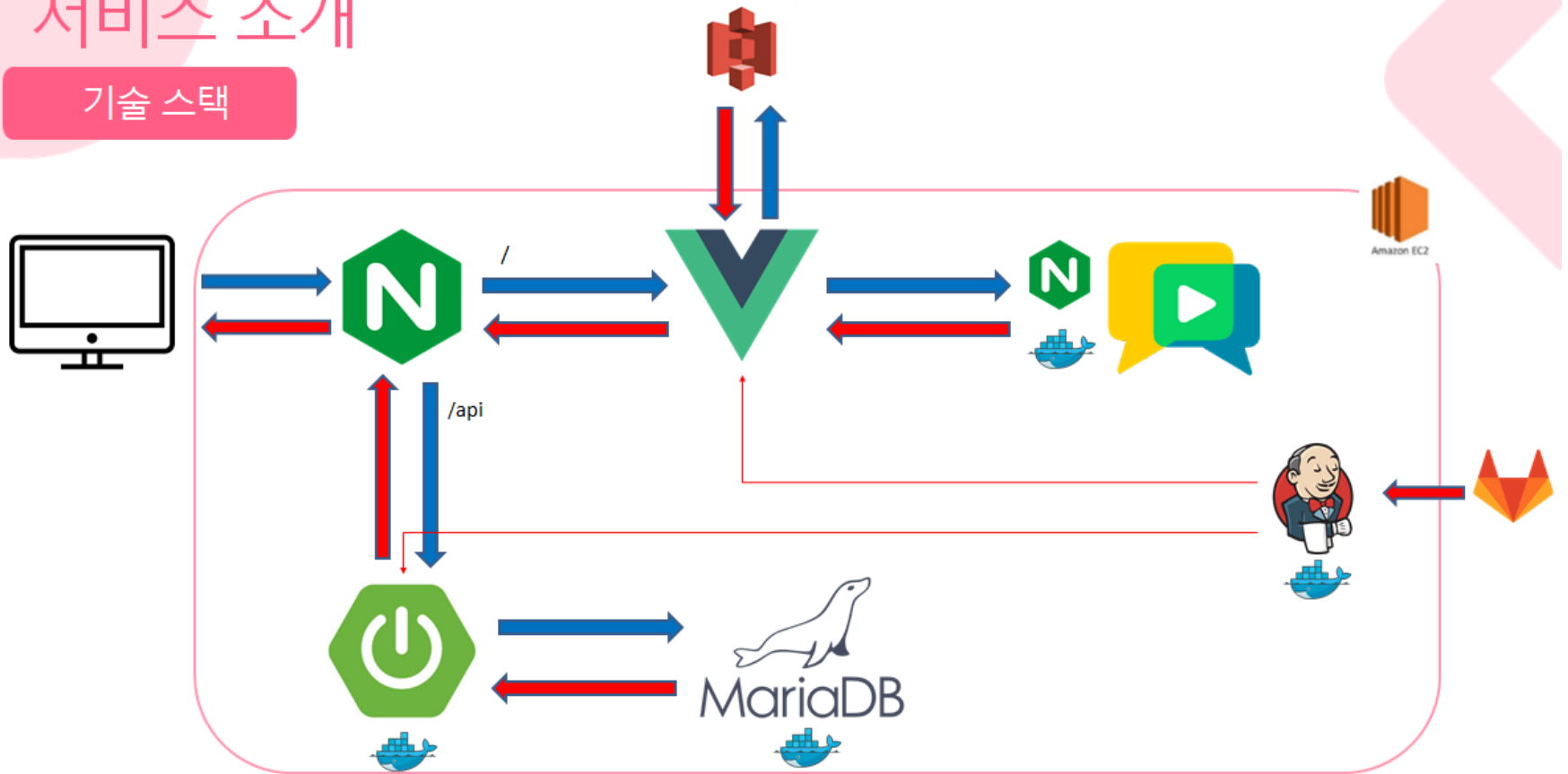
Project Detail

기간	2021.10.11 ~ 2021.11.19
참여 인원	5명
개요	20대 남녀를 잠재고객으로 설정한 소개팅 서비스
주요 기능	<ul style="list-style-type: none">- webRTC 라이브러리(openVidu)를 활용한 남녀간 화상 채팅- 남녀간 일상 공유가 가능한 커뮤니티 게시판- 팔로우, 팔로잉을 통한 회원 간 소통- 임의 기준을 통한 랜덤 회원 추천 기능
역할	팀장 / 풀스택 개발
개발 환경	openVidu, AWS EC2, AWS S3, JWT, Springboot, Vue.js(2.x), JPA, IntelliJ, Jira
담당 기능	<ul style="list-style-type: none">- AWS EC2 서버 내 OpenVidu 빌드- Spring Security와 JWT를 활용한 Authentication 구현- AWS S3를 통한 다중 이미지 업로드 기능 Backend API 작성- vue.js(2.x)와 vuetify(2.x)를 활용한 로그인 및 회원가입 페이지 구현- 로그인 및 회원가입 Backend API 작성

Project Architecture

서비스 소개

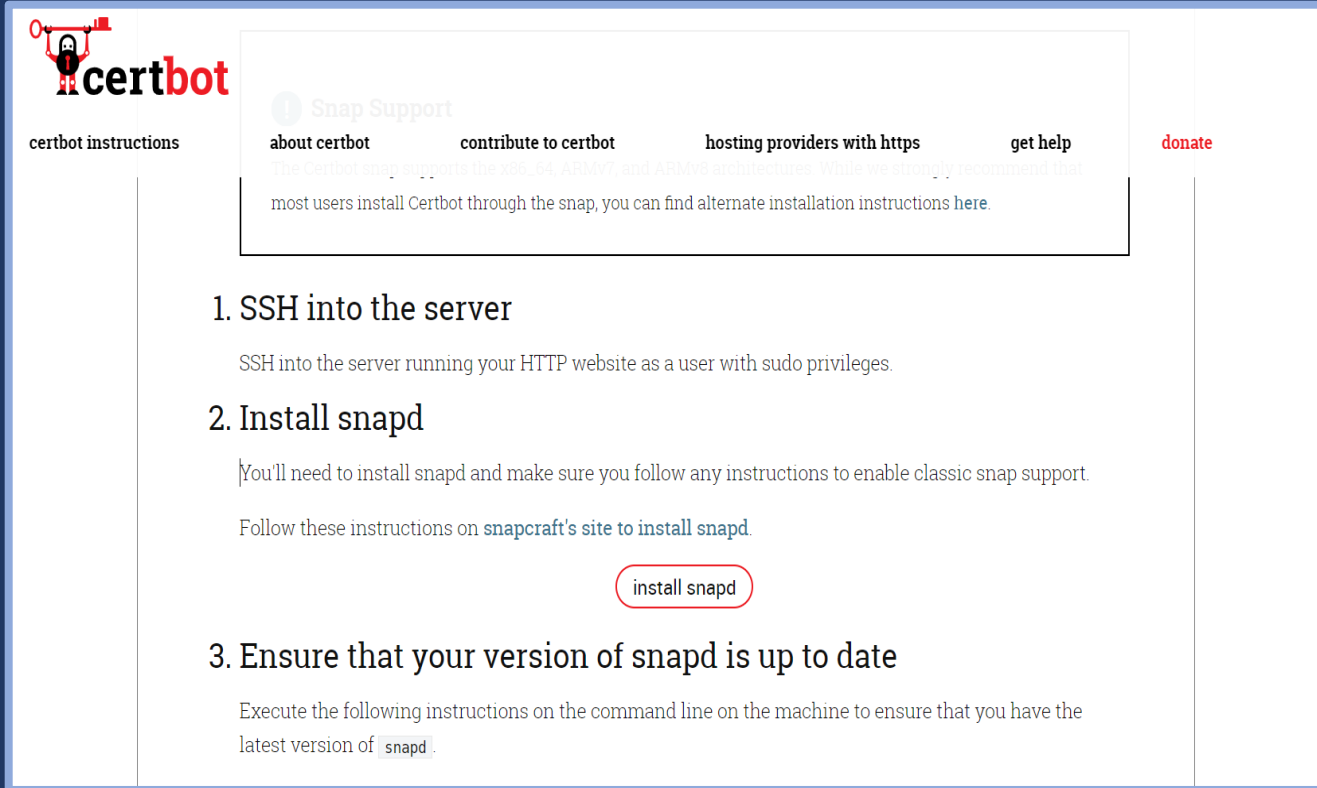
기술 스택



Nginx를 이용한 openVidu build

openVidu는 HTTPS에서
동작하기 때문에, certbot을
이용해 SSL 인증서를 발급

SSL 인증서를 Ubuntu Server에
저장함으로서 HTTPS 통신이 가능



The screenshot shows the Certbot website. At the top left is the Certbot logo. Below it are links for 'certbot instructions', 'about certbot', 'contribute to certbot', 'hosting providers with https', 'get help', and 'donate'. A 'Snap Support' section is highlighted, stating that the Certbot snap supports x86_64, ARMv7, and ARMv8 architectures and recommending that most users install Certbot through the snap, with a link to 'here' for alternate installation instructions. Below this, the first step '1. SSH into the server' is shown, followed by the instruction to SSH into the server running your HTTP website as a user with sudo privileges. The second step '2. Install snapd' is shown, followed by the instruction that you'll need to install snapd and make sure you follow any instructions to enable classic snap support, and a link to 'snapcraft's site to install snapd'. A red button labeled 'install snapd' is visible. The third step '3. Ensure that your version of snapd is up to date' is shown, followed by the instruction to execute the following instructions on the command line on the machine to ensure that you have the latest version of 'snapd'.

certbot instructions

about certbot contribute to certbot hosting providers with https get help donate

Snap Support
The Certbot snap supports the x86_64, ARMv7, and ARMv8 architectures. While we strongly recommend that most users install Certbot through the snap, you can find alternate installation instructions [here](#).

1. SSH into the server
SSH into the server running your HTTP website as a user with sudo privileges.

2. Install snapd
You'll need to install snapd and make sure you follow any instructions to enable classic snap support.
Follow these instructions on [snapcraft's site to install snapd](#).

install snapd

3. Ensure that your version of snapd is up to date
Execute the following instructions on the command line on the machine to ensure that you have the latest version of `snapd`.

Certbot을 이용한 SSL 인증서의 발급

Nginx를 이용한 openVidu build



성공적으로 배포된 openVidu

Ufw(ubuntu firewall)를 통해
springboot, vue.js, openVidu의
포트를 설정

초기 설치한 Kurento Media
Server로 인해 COTURN 서버의
충돌이 발생, 해결하기까지 오랜
시간 소요

Takeaway

HTTP와 HTTPS의 차이

SSL 인증서의 역할

Nginx의 역할

Tomcat과의 차이점

Reverse Proxy

webRTC의 정의

STUN과 TURN

Ufw를 활용한 PORT 설정

JWT를 활용한 사용자 Authentication

```
@Override
public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain)
    throws IOException, ServletException {
    logger.info("JwtFilter.doFilter 36 : JwtFilter 수행");
    HttpServletRequest httpServletRequest = (HttpServletRequest) servletRequest;
    String jwt = resolveToken(httpServletRequest);
    logger.info("JwtFilter.doFilter 39 : Token String : {}", jwt);

    // 유효한 토큰인지 검증하는 필터를 Spring Security에 등록한다.
    if (StringUtils.isNotBlank(jwt) && tokenProvider.validateToken(jwt)) {
        logger.info("JwtFilter.doFilter 44 : 토큰 유효함");
        Authentication authentication = tokenProvider.getAuthentication(jwt);
        logger.info("JwtFilter.doFilter 46 : getAuthenticaiton 수행 완료. {}", authentication);
        logger.info("JwtFilter.doFilter 47 : 생성한 authentication 객체를 SecurityContextHolder에 저장합니다.");
        SecurityContextHolder.getContext().setAuthentication(authentication);
        logger.info("JwtFilter.doFilter 49 : Security Context에 '{}' 인증 정보를 저장했습니다. uri: {}", authentication.getName(), requestURI);
    } else {
        logger.debug("JwtFilter.doFilter 51 : 유효한 JWT 토큰이 없습니다, uri: {}", requestURI);
    }

    // 필터 연쇄 적용.
    filterChain.doFilter(servletRequest, servletResponse);
}
```

유효한 토큰인지 검증하는
필터를 Spring Security에
등록한다.

유효하다면
SpringContext에 인증
정보를 저장한다.

유효하지 않다면
log 출력

JWT를 활용한 사용자 Authentication

```
public String createToken(Authentication authentication, String authorities) {  
    // 사용자의 권한 정보를 불러온다.  
    // 아래 코드는 권한을 자동으로 생성해주는 것.  
  
    // String authorities = authentication.getAuthorities().stream()  
    //     .map(GrantedAuthority::getAuthority)  
    //     .collect(Collectors.joining(","));  
  
    // 토큰의 만료기한을 설정한다.  
    long now = (new Date()).getTime();  
    Date validity = new Date(now + this.tokenValidityInMilliseconds);  
  
    // 토큰을 생성하여 return한다.  
    return Jwts.builder()  
        // 아래의 내용은 registered claim을 설정하는 과정이다.  
        // claim은? JWT내 payload에 들어가는 데이터의 형태이다. 하나의 claim은 key-value 쌍으로 이루어져 있다.  
        // claim의 종류로는 registered claim, public claim, private claim이 존재한다.  
        .claim("email", authentication.getName())  
        .setHeader("authorities", authorities) // 토큰의 권한 정보  
        .setHeader("issuer", "mokkozi.com") // 해당 토큰을 발급한 곳의 정보  
        .setIssuedAt(Date.from(LocalDate.now().atZone(ZoneId.systemDefault()).toInstant())) // 해당 토큰의 생성일  
        .signWith(SignatureAlgorithm.HS512) // 서명  
        .setExpiration(validity) // 해당 토큰의 만료일  
        .compact() // 이를 조합하여 JWT를 만들고, 해당하는 String을 반환한다.
```

토큰의 유효기한 설정

Jwt 라이브러리 통해

- 토큰의 권한 정보
 - 토큰의 생성일
 - 서명 정보
 - 만료일
- 등을 토큰 내부에 저장

Takeaway

JWT를 통한
Authentication의 순서와
구조

Filter

Spring Security의 구조

Frontend → Backend로
오는 데이터의 처리 순서

로그인 & 회원가입 페이지 구현

게시판

랜선미팅

mokkozi

아이디

비밀번호

로그인

GOOGLE 계정으로 로그인

아직 계정이 없다면? 회원가입 하세요!

Home

Login

게시판

랜선미팅

안녕하세요! 모꼬지에 오신걸 환영합니다☺
기본 인적 사항을 입력해 주세요!

이메일

닉네임

비밀번호

생년월일

성별

다음으로 현재 거주중인 주소지를 입력해 주세요!
(주변에 있는 이성을 추천할 때 활용됩니다)

주소 입력을 위해 클릭해 주세요

로그인 / 회원가입 안내

Home

Login

AWS S3를 통한 이미지 업로드



1

S3 Bucket 생성

2

Springboot에 spring-cloud-starter 라이브러리 추가

3

Frontend Server에서
MultipartFile 형태로 파일
전달

4

amazonS3Client.putObject()
통해 S3 Bucket에 파일
저장

5

저장된 파일의 고유
URL을 DB에 저장

6

사용자에게 이미지 출력

Takeaway

Spring Transaction의
종류와 과정

S3 Bucket을 활용한 이미지
업로드

Projects

02.

Blocket

블록체인 기반 자격증명 서비스

2021.08.30 ~ 2021.10.08

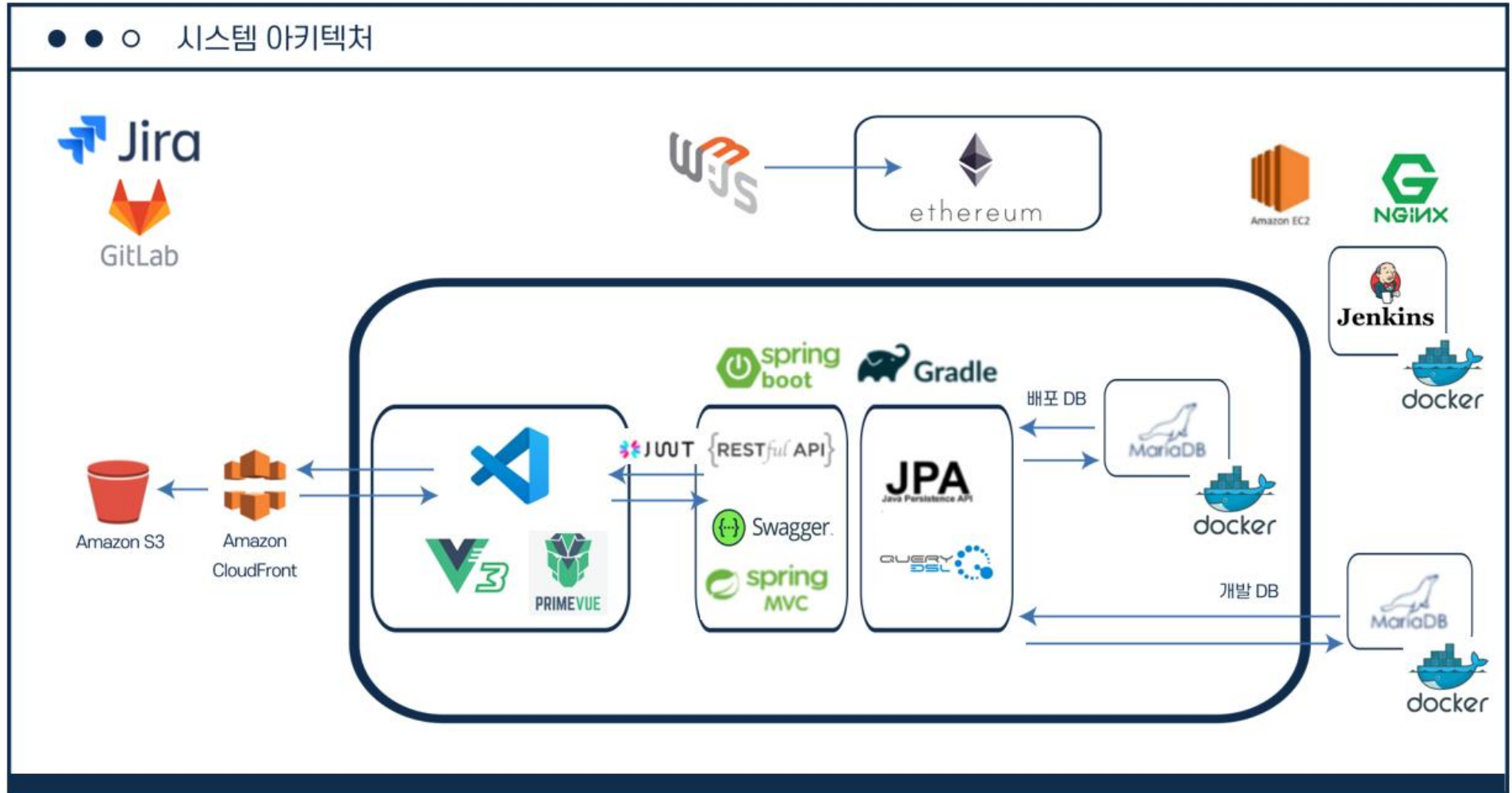
Project Detail



Project Detail

기간	2021.08.30 ~ 2021.10.08
참여 인원	5명
개요	지원자에겐 블록체인을 도입해 서류 제출 과정을 간편화하고 기업에겐 지원자의 검증된 서류를 제공하는 서비스
주요 기능	- (지원자) 관련 문서 서비스에 업로드 - (관리자) 지원자의 문서 검토 및 블록체인 업로드 - (채용담당자) 검증된 서류를 관리자로부터 제공받아 검토 절차 간소화
역할	팀장 / 풀스택 개발 / 블록체인 기능 담당
개발 환경	Ethereum Network, web3.js, springboot, Vue.js(3.x), AWS S3
담당 기능	- web3.js를 이용한 block data 저장 기능 구현 - Amazon S3 이미지 업로드 Front-end 코드 작성 - 공공데이터 API를 이용한 대학교 이름 검색 기능 구현 - async-await을 이용한 Javascript 비동기 처리 함수 구현 - 서류 제출 Front-end 페이지 구현 - 이미지 저장 관련 Backend API 작성

Project Architecture



Project 주요기능

개인정보 검증 서비스

블록체인 기반

블록체인에 데이터를 저장하여 서비스 내에서 검증 후, 위변조가 불가능한 데이터로 안전하게 통제 및 관리합니다.



- 01** 자격증, 성적증명서 등 채용 과정에서 필요한 개인정보를 서비스 내 제출
- 02** 관리자가 데이터 검증 후 블록체인에 해시값으로 저장
- 03** 기업들은 서비스의 지원자 계정을 통해 블록체인 내에서 정보를 손쉽게 조회

Web3.js 통한 블록 데이터 저장

최종 학력

최종 학력 :

아직 입력

학교명 :

아직 입력

학점 :

아직 입력

[새로 입력하기](#)

사용자는 입사 지원에 필요한 다양한 문서
(학력, 성적, 자격증, 활동사항 등..)
를 업로드할 수 있다.

활동 사항

입력한 활동사항이 없습니다. [새로 입력하기](#)

어학, 자격증

입력한 자격증이 없습니다. [새로 입력하기](#)

병역 사항

병역 사항을 입력해주세요.

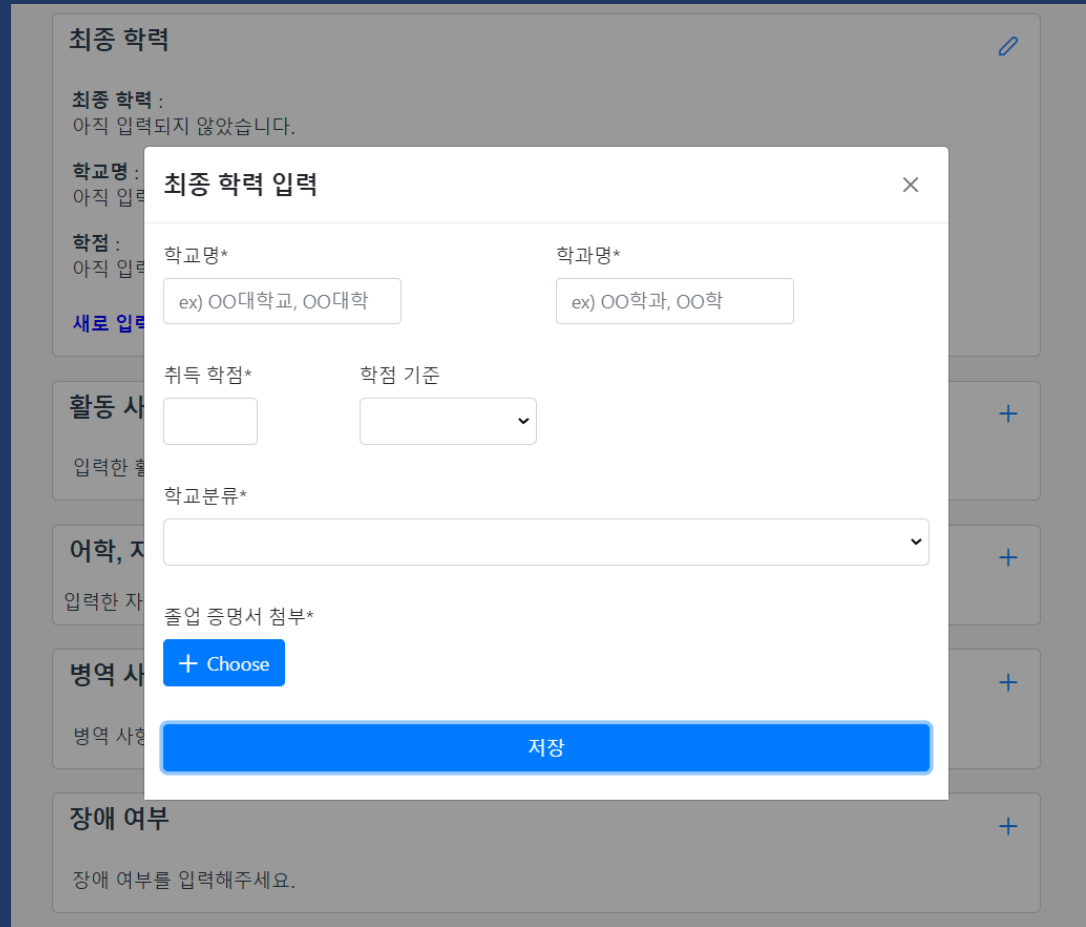
장애 여부

장애 여부를 입력해주세요.

Web3.js 통한 블록 데이터 저장


사용자는 성적 증명서, 졸업
증명서 등 입사 지원에 필요한
이미지를 설명과 함께 제출

관리자는 해당 이미지를 상세하게
검토하여 문서의 유효성을 검증
검증이 완료됐다면 이미지 파일을
해시값으로 전환하여 이더리움
네트워크에 저장



The image shows a web application interface for managing user profiles. The main form is titled '최종 학력' (Final Education) and contains fields for '최종 학력' (Final Education), '학교명' (School Name), '학점' (Credits), '활동 사항' (Activities), '어학, 자격' (Languages, Certificates), '병역 사항' (Military Service), and '장애 여부' (Disability Status). A modal window titled '최종 학력 입력' (Final Education Input) is open, showing fields for '학교명*' (School Name*), '학과명*' (Department*), '취득 학점*' (Credits*), '학점 기준' (Credit Standard), '학교분류*' (School Classification*), and '졸업 증명서 첨부*' (Attach Graduation Certificate*). The modal also includes a '+ Choose' button and a '저장' (Save) button.

Web3.js 통한 블록 데이터 저장

Etherscan

All Filters ▾ Search by Address / Txn Hash / Block / Token / Ens 🔍

Ropsten Testnet Network Home Blockchain ▾ Tokens ▾ Misc ▾ Ropsten

Transaction Details

Overview State

[This is a Ropsten **Testnet** transaction only]

Transaction Hash:	0x4950e633a959acb9402fbbb270d892fadb2e1e6282fafaab3b917c83893bab39 🔗
Status:	✓ Success
Block:	11184441 6 Block Confirmations
Timestamp:	⌚ 1 min ago (Oct-07-2021 10:22:55 PM +UTC)
From:	0xf255fc9ef3778e688950649547d398b027d8b999 🔗
To:	0xf255fc9ef3778e688950649547d398b027d8b999 🔗
Value:	0 Ether (\$0.00)
Transaction Fee:	0.00021512 Ether (\$0.00)
Gas Price:	0.00000001 Ether (10 Gwei)
Txn Type:	0 (Legacy)

[Click to see More](#) ⬇

이더리움 네트워크에 저장된
블록 데이터는 고유의 트랜잭션
해시값을 통해 확인이 가능하며,
해시값으로 전환된 이미지
고유의 값 또한 함께 확인
가능하다

Takeaway

블록체인의 정의

Accounts

Wallet

합의 알고리즘

Transaction

Smart Contract

Web3.js

지금까지 Backend Engineer
조웅현이었습니다.
귀중한 시간 투자하여 읽어주셔서
진심으로 감사드립니다.
