

Porting Manual

목차

목차

개발 환경

서버 인스턴스 사양

형상 관리

UI / UX

OS

이슈 관리

Communication

Front-end

DB

Back-end

IDE

Infra

기타 편의 툴

시스템 아키텍처

배포 과정

1. SSAFY EC2(서버) 접속

1. WSL 사용하여 EC2에 SSH 연결

2. EC2 초기 설정

3. 한국으로 시간 설정

2. EC2 환경 설정

1. Docker 설치

2. Docker Maria DB

3. Docker Redis

3. Dockerfile로 Jenkins images 받기 (Docker in Docker, DinD 방식)

3. Jenkins 초기 세팅 및 테스트

Jenkins 플러그인 설정

Gitlab

Docker

6. CI/CD (빌드 및 배포) 세팅

빌드 확인

Webhook 설정

배포 환경 구성

도메인 구매

SSL 발급 받기

HTTPS 적용

Front https

Back https

배포 shell 파일

docker-compose-prod.yml

빌드 절차

배포시 주의 사항

DB 정보

Maria DB

Redis + mongoDB

외부 서비스

1. Google 로그인

2. Kakao 로그인

3. Naver 로그인

4. application.yml 설정

OAuth2 설정

SMTP

S3 Bucket

서비스 이용 방법

Google Cloud Platform

SMTP - 구글 메일 설정

S3 버킷

시연 시나리오

- [메인 페이지 소개](#)
- [자세 교정 페이지](#)
- [자세 기록 페이지](#)
- [목표 페이지](#)
- [갤러리 페이지](#)

개발 환경

서버 인스턴스 사양

- CPU 정보: Intel Xeon(R) Core 4개
- RAM : 16GB
- Disk : 300GB

```
processor       : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 79
model name   : Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
stepping      : 1
microcode    : 0x000040
bus freq     : 1000.270
cache size   : 48996 KB
physical id  : 0
siblings      : 4
core id      : 0
cpu cores    : 4
apicid        : 0
```

Mem Usage: Total 100.0B used 100.0B Free 0.0B shared 1.0M buff/cache 4.501 available 4.701

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/root	311G	30G	281G	10%	/

형상 관리

- Gitlab

UI / UX

- Figma

OS

- Ubuntu 20.04
- Windows 10

이슈 관리

- Jira

Communication

- Mattermost
- Notion
- Webex

Front-end

Node	18.13.0
React	18.2.0
React-router-dom:	6.2.1
Styled-components	5.3.3
Axios	0.25.0

Back-end

java	17
Springboot	3.0.1
jUnit	5
gradle	7.6
swagger	3
mariaDB	10.11.1 RC

DB

- MariaDB : 10.10.2
- Redis : 7.0.8
- mongo : 6.0.4

IDE

- IntelliJ : 2022.3.1
- Visual Studio Code : 1.75.0

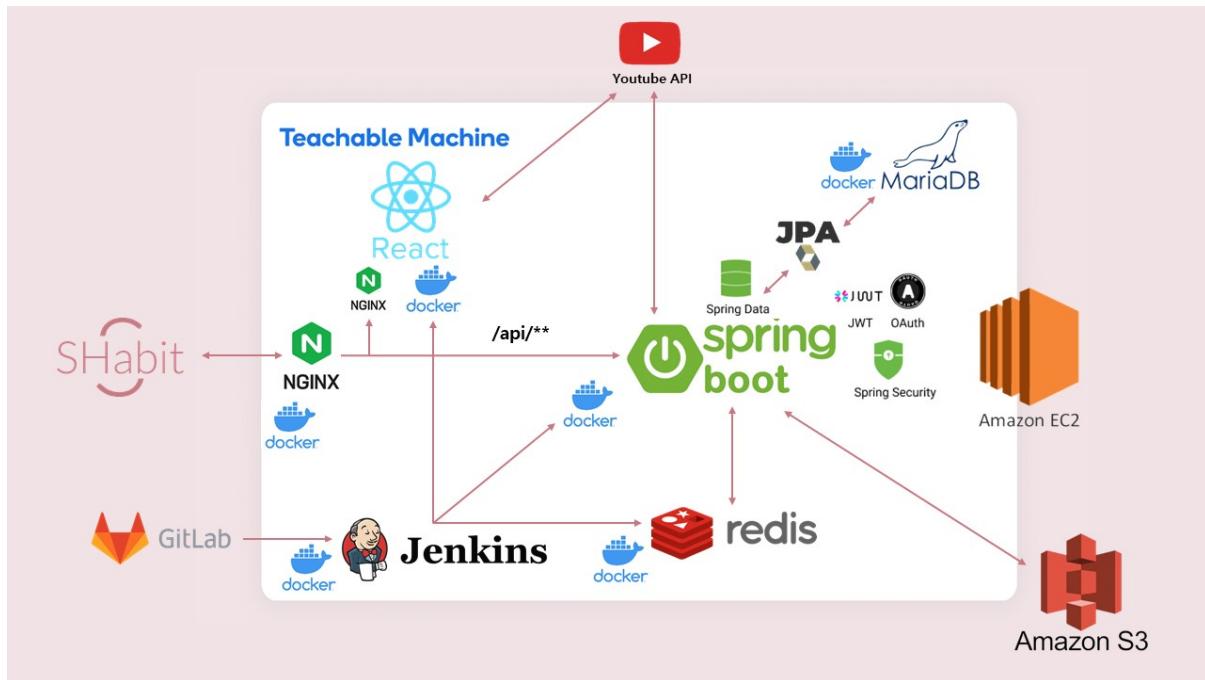
Infra

- Web Server : Nginx
- Jenkins : 2.388
- Docker : 20.10.23
- Docker-compose : 2.15.1

기타 편의 툴

- WSL2
- Postman
- Termius
- MobaXterm

시스템 아키텍처



배포 과정

- EC2 인스턴스는 이미 가지고 있다고 가정하겠습니다.

1. SSAFY EC2(서버) 접속

1. WSL 사용하여 EC2에 SSH 연결

- SSH 접속

 - 최초 접속 시 권한 요구하면 'yes' 입력

```
# sudo ssh -i [pem키 위치] [접속 계정]@[접속할 도메인]
$ sudo ssh -i I8A601T.pem ubuntu@i8a601.p.ssafy.io
```

2. EC2 편리하게 접속하는 법

- EC2 정보가 담긴 config 파일을 만들어 번거롭게 pem과 도메인 경로를 쓰지 않고 접속할 수 있다.
 - ssh 전용 폴더 생성

```
mkdir ~/.ssh
cd ~/.ssh // ssh 폴더 생성 및 이동
cp [로컬 pem 키 위치] ~/.ssh // pem 키 옮기기
vi config // config 파일 생성
```

○ config 내용 추가

```
Host ssafy
HostName [서버 ip 주소]
User ubuntu
IdentityFile ~/.ssh/[pem키 파일 명].pem
```

○ ssafy 계정에 접속

```
ssh ssafy
```

2. EC2 초기 설정

```
$ sudo apt update  
$ sudo apt upgrade  
$ sudo apt install build-essential
```

3. 한국으로 시간 설정

```
$ sudo ln -sf /usr/share/zoneinfo/Asia/Seoul /etc/localtime  
# 시간 확인  
$ date
```

```
:~$ date  
Fri May 14 14:12:02 KST 2021
```

2. EC2 환경 설정

1. Docker 설치

1. 기본 설정, 사전 설치

```
$ sudo apt update  
$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

2. 자동 설치 스크립트 활용

- 리눅스 배포판 종류를 자동으로 인식하여 Docker 패키지를 설치해주는 스크립트를 제공

```
$ sudo wget -qO- https://get.docker.com/ | sh
```

3. Docker 서비스 실행하기 및 부팅 시 자동 실행 설정

```
$ sudo systemctl start docker  
$ sudo systemctl enable docker
```

4. Docker 그룹에 현재 계정 추가

```
$ sudo usermod -aG docker ${USER}  
$ sudo systemctl restart docker
```

- sudo를 사용하지 않고 docker를 사용할 수 있다.
- docker 그룹은 root 권한과 동일하므로 꼭 필요한 계정만 포함
- 현재 계정에서 로그아웃한 뒤 다시 로그인

5. Docker 설치 확인

```
$ docker -v
```

2. Docker Maria DB

1. Docker Maria DB 이미지 다운로드 받기

```
$ docker pull mariadb
```

2. Docker에 Maria DB 컨테이너 만들고 실행하기

```
$ docker run --name mariadb -d -p 3306:3306 -v /var/lib/mysql_main:/var/lib/mysql --restart=always -e MYSQL_ROOT_PASSWORD=root mariadb
```

▼ 옵션 설명

- -v : 마운트 설정, host 의 /var/lib/mysql 과 mariadb의 /var/lib/mysql의 파일들을 동기화
- -name: 만들어서 사용할 컨테이너의 이름을 정의
- d: 컨테이너를 백그라운드에서 실행
- p: 호스트와 컨테이너 간의 포트를 연결 (host-port:container-port) // 호스트에서 3306 포트 연결 시 컨테이너 3306 포트로 포워딩
- -restart=always: 도커가 실행되는 경우 항상 컨테이너를 실행
- e: 기타 환경설정(Enviorment)
- MYSQL_ROOT_PASSWORD=root // mariadb의 root 사용자 초기 비밀번호를 설정
- mariadb: 컨테이너를 만들 때 사용할 이미지 이름

3. Maria DB에 database를 추가하고 user 권한 설정

- Docker - Mariadb 컨테이너 접속하기

```
docker exec -it mariadb /bin/bash
```

- Mariadb - 루트 계정으로 데이터베이스 접속하기

```
mysql -u root -p
```

비밀번호는 "root"

Mariadb 사용자 추가하기

```
예시) create user 'user_name'@'XXX.XXX.XXX.XXX' identified by 'user_password';
create user 'ssafy601'@'%' identified by 'ssafy601';
```

Mariadb - 사용자 권한 부여하기

```
예시) grant all privileges on db_name.* to 'user_name'@'XXX.XXX.XXX.XXX';
flush privileges;

grant all privileges on *.* to 'ssafy601'@'%';
flush privileges;
```

Mariadb - 데이터 베이스 만들기

```
예시) create database [db_name];
create database ssafy601;
```

3. Docker Redis

- Redis 이미지 받기

```
docker pull redis:alpine
```

- 도커 네트워크 생성 [디폴트값]

```
docker network create redis-network
```

- 도커 네트워크 상세정보 확인

```
docker inspect redis-network
```

- local-redis라는 이름으로 로컬-docker 간 6379 포트 개방

```
docker run --name local-redis -p 6379:6379 --network redis-network -v /redis_temp:/data -d redis:alpine redis-server --appendonly yes
```

- Docker 컨테이너 확인

```
docker ps -a
```

- 컨테이너 진입

```
# 실행 중인 redis 컨테이너에 대해 docker redis-cli 로 직접 진입  
docker run -it --network redis-network --rm redis:alpine redis-cli -h local-redis  
  
# bash로도 진입 가능하다.  
docker run -it --network redis-network --rm redis:alpine bash  
redis-cli
```

- 권한 추가

```
# slaveof no one : 현재 슬레이브(복제)인 자신을 마스터로 만듭니다.  
127.0.0.1:6379> slaveof no one
```

- 테스트

- OK 가 뜨면 성공

```
127.0.0.1:6379> slaveof no one  
OK  
127.0.0.1:6379> set apple 100  
OK  
127.0.0.1:6379> get apple  
"100"
```

3. Dockerfile로 Jenkins images 밟기 (Docker in Docker, DinD 방식)

- Dockerfile 작성

```
# 폴더 생성  
mkdir config && cd config  
  
# 아래 내용 작성  
$ vi Dockerfile  
  
FROM jenkins/jenkins:jdk17  
  
#도커를 실행하기 위한 root 계정으로 전환  
USER root  
  
#도커 설치  
COPY docker_install.sh /docker_install.sh  
RUN chmod +x /docker_install.sh  
RUN /docker_install.sh  
  
#설치 후 도커그룹의 jenkins 계정 생성 후 해당 계정으로 변경  
RUN groupadd -f docker  
RUN usermod -aG docker jenkins  
USER jenkins
```

- docker 설치 shell 파일

```
$ vi docker_install.sh

#!/bin/sh
apt-get update && \
apt-get -y install apt-transport-https \
ca-certificates \
curl \
gnupg2 \
zip \
unzip \
software-properties-common && \
curl -fsSL https://download.docker.com/linux/$(. /etc/os-release; echo "$ID")/gpg > /tmp/dkey; apt-key add /tmp/dkey && \
add-apt-repository \
"deb [arch=amd64] https://download.docker.com/linux/$(. /etc/os-release; echo "$ID") \
${(lsb_release -cs)} \
stable" && \
apt-get update && \
```

- Docker 이미지 생성

```
docker build -t jenkins/myjenkins .
```

- Jenkins 컨테이너 생성

```
docker run -d -p 9090:8080 --name=jenkinscicd \
-e TZ=Asia/Seoul \
-v /var/jenkinsDir:/var/jenkins_home \
-v /var/run/docker.sock:/var/run/docker.sock \
jenkins/myjenkins
```

- 옵션 설명

-d : 는 백그라운드에서 실행을 의미

-p 는 매핑할 포트를 의미합니다. (p가 port의 단축어가 아니었음 ..)

-v : 기준으로 왼쪽은 로컬포트, 오른쪽은 도커 이미지의 포트를 의미합니다. 도커 이미지에서의 8080 포트를 로컬 포트 9090으로 매핑한다는 뜻입니다.

```
-v /var/run/docker.sock:/var/run/docker.sock \
jenkins/myjenkins
```

이 옵션은 로컬의 도커와 젠킨스 내에서 사용할 도커 엔진을 동일한 것으로 사용하겠다는 의미입니다.

v 옵션은 ":"를 기준으로 왼쪽의 로컬 경로를 오른쪽의 컨테이너 경로로 마운트 해줍니다.

즉, 제 컴퓨터의 **사용자경로/jenkinsDir** 을 컨테이너의 **/var/jenkins_home** 과 바인드 시켜준다는 것입니다. 물론, 양방향으로 연결됩니다.

컨테이너가 종료되거나 알 수 없는 오류로 정지되어도, **jenkins_home**에 남아있는 소중한 설정 파일들은 로컬 경로에 남아있게 됩니다.

3. Jenkins 초기 세팅 및 테스트

- 젠킨스에 접속하기 전에 **/var/run/docker.sock** 에 대한 권한을 설정해주어야 합니다.
- 초기 **/var/run/docker.sock** 의 권한이 소유자와 그룹 모두 root였기 때문에 이제 그룹을 root에서 **docker** 로 변경해줄겁니다.
- 먼저, jenkins로 실행됐던 컨테이너의 bash를 root 계정으로 로그인 하기전에, 현재 실행되고 있는 컨테이너의 정보들을 확인할 수 있는 명령어를 입력해 아이디를 확인하겠습니다.

```
docker ps -a
```

```
> docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
0bcd8291015        jenkins/myjenkins   "/sbin/tini -- /usr/..."   23 seconds ago    Up 21 seconds      50000/tcp, 0.0.0.0:9090->8080/tcp   jenkinscicd
```

- 우리가 방금 생성한 컨테이너의 ID는 **0bcd8~**입니다. 도커는 다른 컨테이너 ID와 겹치지 않는 부분까지 입력하면 해당 컨테이너로 알아서 매핑해줍니다.

```
docker exec -it -u root 컨테이너ID /bin/bash
```

exec 는 컨테이너에 명령어를 실행시키는 명령어인데, `/bin/bash`와 옵션 `-it`를 줌으로써 컨테이너의 쉘에 접속할 수 있습니다.

이제 정말로 root 계정으로 컨테이너에 접속하기 위해 컨테이너ID에 `0bc`를 입력해 실행합니다.

```
> docker exec -it -u root 0bc /bin/bash
root@0bcd8291015:/# █
```

- root 계정으로 로그인이 잘 되었습니다. 이제 그룹을 바꾸기 위해 다음 명령어를 실행해줍니다.

```
chown root:docker /var/run/docker.sock
```

- 그리고 이제 쉘을 `exit` 명령어로 빠져나온 후 다음 명령어를 실행해 컨테이너를 재실행해줍니다.

```
docker restart [컨테이너 ID]
```

- Jenkins 패스워드 확인

```
docker logs bd2
```

- `docker logs` 컨테이너 id를 입력해 로그를 출력하면 `initialAdminPassword`가 출력됩니다. 이 패스워드를 입력해주면 됩니다.

```
*****
*****
*****
Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:
b99f67a2cf054174a73017e4498ce87d
This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
*****
*****
*****
```

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

- 정상적으로 입력했다면 플러그인 설치가 나오는데, 우리는 `Install suggested plugins`를 선택합니다.

Folders	OWASP Markup Formatter	Build Timeout	Credentials Binding	** SSH server Folders ** Trilead API
Timestamper	Workspace Cleanup	Ant	Gradle	
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View	
Git	SSH Build Agents	Matrix Authorization Strategy	PAM Authentication	
LDAP	Email Extension	Mailer		

- 설치가 완료되면, 어드민 계정 생성창이 나오고, 본인이 사용하실 정보들을 입력해줍시다.

Create First Admin User

계정명:

암호:

암호 확인:

이름:

이메일 주소:

Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

- 앞으로 이 url로 젠킨스에 접속하시면 됩니다.

Jenkins 플러그인 설정

- Gitlab, Docker 플러그인을 받습니다.

Gitlab

Docker

The screenshot shows two separate Jenkins plugin search results pages.

Left Column (Search: Gitlab):

- Generic Webhook Trigger Plugin** 1.86.2
Can receive any HTTP request, extract any values from JSON and many more.
[Report an issue with this plugin](#)
- GitLab** 1.6.0
This plugin allows **GitLab** to trigger Jenkins builds and display reports.
[Report an issue with this plugin](#)
- Gitlab API Plugin** 5.0.1-78.v47a_45b_9f78b_7
This plugin provides **GitLab API** for other plugins.
[Report an issue with this plugin](#)
- GitLab Authentication plugin** 1.16
This is the an authentication plugin using gitlab OAuth.
[Report an issue with this plugin](#)

A yellow callout box at the bottom states: "This plugin is up for adoption! We are looking for new maintainers."

Right Column (Search: Docker):

- Docker API Plugin** 3.2.13-37.vf3411c9828b9
This plugin provides **docker-java** API for other plugins.
[Report an issue with this plugin](#)
- Docker Commons Plugin** 1.21
Provides the common shared functionality for various Docker related plugins.
[Report an issue with this plugin](#)
- Docker Compose Build Step Plugin** 1.0
Docker Compose plugin for Jenkins.
[Report an issue with this plugin](#)
- Docker Pipeline** 563.vd5d2e5c4007f
Build and use Docker containers from pipelines.
[Report an issue with this plugin](#)
- Docker plugin** 1.3.0
This plugin integrates Jenkins with **Docker**.
[Report an issue with this plugin](#)
- docker-build-step** 2.9
This plugin allows to add various docker commands to Jenkins pipeline.
[Report an issue with this plugin](#)

A yellow callout box at the bottom states: "This plugin is up for adoption! We are looking for new maintainers."

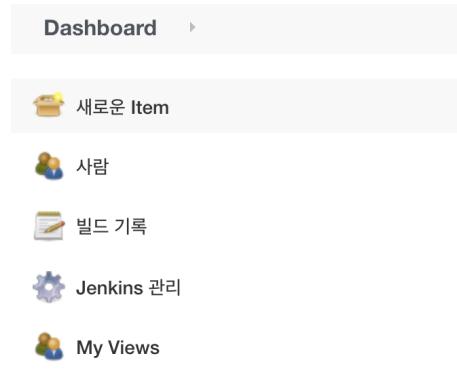
Jenkins is ready!

Your Jenkins setup is complete.

[Start using Jenkins](#)

- 여기까지 오셨다면, 젠킨스 설치 및 초기 세팅 완료!

6. CI/CD (빌드 및 배포) 세팅



- 먼저, 대쉬보드의 새로운 아이템을 클릭합니다.

Enter an item name

* Required field

Freestyle project

이것은 Jenkins의 주요 기능입니다. Jenkins는 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.

- 아이템 이름을 자유롭게 입력해주시고, Freestyle project를 선택하고, OK로 생성합니다.
- 이제 빌드 설정창이 뜰텐데, 소스 코드 관리쪽에서 Git을 선택하고, Repository URL에 다음과 같이 입력해줍니다.
- Gitlab 저장소를 입력하시면 됩니다.

소스 코드 관리

None
 Git

Repositories

Repository URL

Credentials

- Credentials에서 Username / password로 선택하고 SSAFY Email로 두가지 모두 입력하고 계정 인증을 진행합니다.
- 이제 Build에서 Execute shell을 선택해줍니다.

Build

Add build step ▲

Execute Windows batch command

Execute shell

- 폴더 내에 prod와 dev 두가지 버전이 존재하며 prod 환경 기준으로 설명하겠습니다.
- Gitlab의 저장소와 연동되어 폴더 내의 start-prod.sh 파일이 실행 됩니다.

```
bash start-prod.sh
```

빌드 확인

- 이제 드디어 세팅한 값들을 확인해볼 차례입니다. 위의 내용들을 저장하고 Build Now를 눌러봅니다.

Webhook 설정

- Jenkins 트리거 체크

Build when a change is pushed to GitLab. GitLab webhook URL: http://i8a60...

- Jenkins에서 **빌드 유발** → **Build when ...** → **고급** → 하단에 **Secret token Generate** → 토큰 발급 완료!
- Gitlab Webhook에서 해당 토큰을 등록합니다.
 1. lab.ssafy.com Gitlab 프로젝트 접속
 2. 좌측 Settings → Webhook 접속
 3. 아래와 같이 URL 란에 Jenkins에서의 Item URL를 입력하고 Save를 눌러줍니다.

Build when a change is pushed to GitLab. GitLab webhook URL: http://ssafy.io: /project/shabit-main ?

URL
http://example.com/trigger-ci.json
URL must be percent-encoded if it contains one or more special characters.

Secret token

Used to validate received payloads. Sent with the request in the X-Gitlab-Token HTTP header.

Trigger
 Push events
Branch name or wildcard pattern to trigger on (leave blank for all)

배포 환경 구성

도메인 구매

- 가비아 접속

웹을 넘어 클라우드로. 가비아
그룹웨어부터 멀티클라우드까지 하나의 클라우드 허브

q. <https://www.gabia.com/>



- 도메인 선택

!!아래는 예시이고 실제 구매한 도메인은 “shabit.site”입니다.



- 도메인 결제

- DNS 설정
 - My 가비아 → DNS 관리를 → DNS 관리에서 호스트 / 값 설정
 - 호스트에는 @, 값/위치에는 domain 주소를 작성합니다.

SSL 발급 받기

- <https://www.sslforfree.com/> 접속
- 도메인 입력

- 회원가입 및 로그인

Sign Up

Sign up for a free account to create and manage SSL certificates.

Domain: **shabit.site**

Email Address

Password

Register

I agree to receive important service updates.

[Login](#)

- SSL 발급 셋팅

SSL Certificate Setup
You're on your way to issuing a brand-new SSL certificate for one or multiple domains.
Before you can install your new certificate, please complete the steps below.

Domains
 I need a wildcard certificate PRO

Please enter at least one domain to secure. For single-domain certificates
the WWW-version of your domain will always be included at no extra charge.

Enter Domains

shabit.site shabit.site www.shabit.site

[+ Add Domain PRO](#) [Next Step →](#)

Validity
[CSR & Contact](#)
[Finalize Your Order](#)

SSL Certificate Setup

You're on your way to issuing a brand-new SSL certificate for one or multiple domains. Before you can install your new certificate, please complete the steps below.

Domains

Validity

You can now choose between generating 90-day or one-year certificate validity. To keep manual work at a minimum, we recommend 1-year certificates.

90-Day Certificate

1-Year Certificate PRO

Next Step →

CSR & Contact

> Finalize Your Order

Domains

Validity

CSR & Contact

Finalize Your Order

Based on your selection of a **90-Day SSL Certificate** you are fine staying on the **Free Plan**. To create and validate your SSL Certificate, please click "Next Step" below.

Free \$0 / month <input checked="" type="checkbox"/> Selected	Basic \$10 / month or \$8 if billed yearly <input type="checkbox"/> Select	Premium \$50 / month or \$40 if billed yearly <input type="checkbox"/> Select	Business \$100 / month or \$80 if billed yearly <input type="checkbox"/> Select
<ul style="list-style-type: none"> <input checked="" type="checkbox"/> 3 90-Day Certificates <input type="checkbox"/> 1-Year Certificates <input type="checkbox"/> Multi-Domain Certs <input type="checkbox"/> 90-Day Wildcards <input type="checkbox"/> 1-Year Wildcards <input type="checkbox"/> REST API Access <input type="checkbox"/> Technical Support 	<ul style="list-style-type: none"> <input type="checkbox"/> 90-Day Certificates <input checked="" type="checkbox"/> 3 1-Year Certificates <input checked="" type="checkbox"/> Multi-Domain Certs <input type="checkbox"/> 90-Day Wildcards <input type="checkbox"/> 1-Year Wildcards <input checked="" type="checkbox"/> REST API Access <input checked="" type="checkbox"/> Technical Support 	<ul style="list-style-type: none"> <input type="checkbox"/> 90-Day Certificates <input checked="" type="checkbox"/> 10 1-Year Certificates <input checked="" type="checkbox"/> Multi-Domain Certs <input type="checkbox"/> 90-Day Wildcards <input checked="" type="checkbox"/> 1 1-Year Wildcards <input checked="" type="checkbox"/> REST API Access <input checked="" type="checkbox"/> Technical Support 	<ul style="list-style-type: none"> <input type="checkbox"/> 90-Day Certificates <input checked="" type="checkbox"/> 25 1-Year Certificates <input checked="" type="checkbox"/> Multi-Domain Certs <input type="checkbox"/> 90-Day Wildcards <input checked="" type="checkbox"/> 3 1-Year Wildcards <input checked="" type="checkbox"/> REST API Access <input checked="" type="checkbox"/> Technical Support

- SSL 인증서를 받을 때 google.com 같은 사이트의 인증서 발급을 막기 위해서 도메인 인증을 해야합니다.

shabit.site

Congratulations, your SSL certificate is en route! However, you need to verify ownership of your domain before installing your certificate. Please follow the steps below.

Verification Method for shabit.site

We need you to verify ownership of each domain in your certificate. Please select your preferred verification method and click "Next Step".

Email Verification
 DNS (CNAME)

Follow the steps below

To verify your domain using a CNAME record, please follow the steps below:

- 1 Sign in to your DNS provider, typically the registrar of your domain.
- 2 Navigate to the section where DNS records are managed.
- 3 Add the following CNAME record:

Name: _8C€

Point To: 9669C6B165E260AE2D1
fc.comodoca.com

TTL: 3600 (or lower)

- 4 Save your CNAME record and click "Next Step" to continue.

HTTP File Upload

Next Step →

- 위에서 **Name** 과 **Point To**의 값을 가비아 DNS 관리툴에서 **호스트** / **값**에 추가 해줍니다.

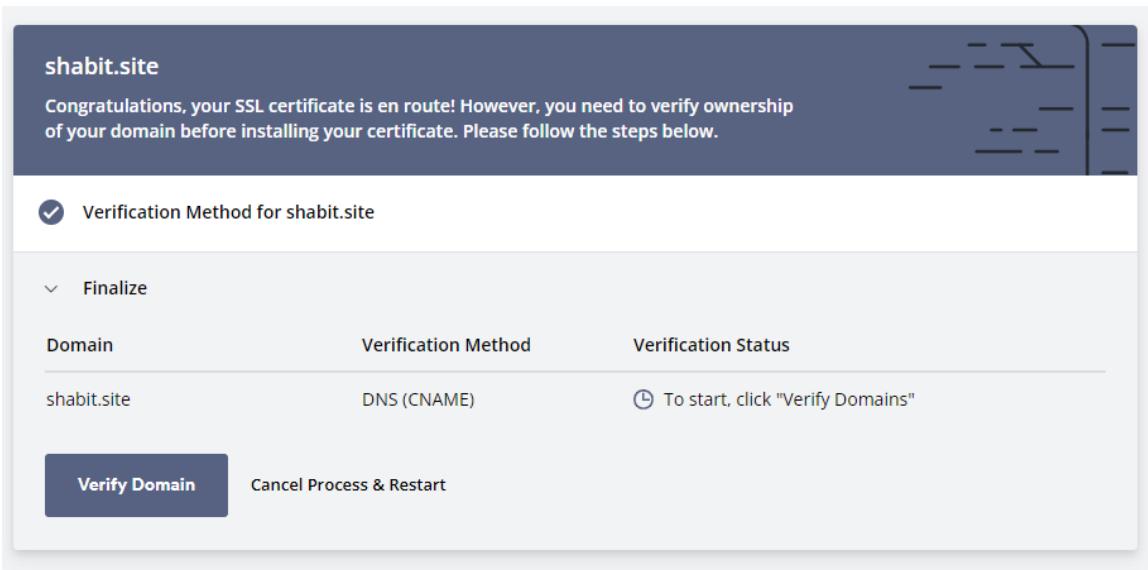
DNS 관리

shabit.site

• 레코드 개수 : 3개 • 최근 업데이트 : 2023-02-07 22:36:25 • 네임서버 : ns.gabia.co.kr

타입	호스트	값/위치
CNAME	@	l8a601.p.ssafy.io.
CNAME	_0fa955987da3d abb889	9c2fe23e41d84ee0e3e20628efe1ef90.e21fd6610e06a
CNAME	www	l8a601.p.ssafy.io.

- 인증 후 인증서 압축 파일을 발급받습니다.



HTTPS 적용

Front https

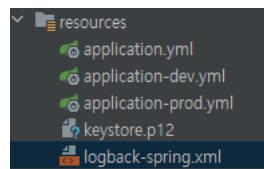
- Verify Domain 버튼을 눌러 도메인을 인증
- 성공한다면, Server Type을 Nginx로 선택한 후, 인증서를 다운로드 받습니다.
- 압축을 풀고 /.nginx/cert 폴더에 저장합니다.

Back https

- 백엔드에서 Https를 적용하기 위해서는 인증서를 pem키로 변환 해주어야 합니다.

```
sudo openssl pkcs12 -export -out keystore.p12 -inkey private.key -in certificate.crt -certfile ca_bundle.crt
```

- 이후 생성된 keystore.p12 파일을 resources에 추가합니다.



- yaml 파일에 ssl 설정 값을 추가합니다.

```
server:
  ssl:
    key-store: classpath:keystore.p12
    key-store-password: ssafy
    key-store-type: PKCS12
```

배포 shell 파일

```
docker-compose -f docker-compose-prod.yml pull
COMPOSE_DOCKER_CLI_BUILD=1 DOCKER_BUILDKIT=1 docker-compose -f docker-compose-prod.yml up --build -d
docker rmi -f $(docker images -f "dangling=true" -q) || true
```

docker-compose-prod.yml

```

version: "3"
services:
  server:
    container_name: server
    build:
      context: ./SHabit-back
      args:
        SERVER_MODE: prod
    ports:
      - 8080:8080
    environment:
      - TZ=Asia/Seoul
  client:
    container_name: client
    build:
      context: ./shabit-front
      dockerfile: Dockerfile.dev
    ports:
      - 3000:3000
    depends_on:
      - server
  nginx:
    container_name: nginx
    build: ./nginx
    depends_on:
      - server
      - client
    volumes:
      - .nginx/conf.d:/etc/nginx/conf.d
      - .nginx/zeroSSL:/var/www/zeroSSL/.well-known/pki-validation
      - .nginx/cert:/cert
    ports:
      - 80:80
      - 443:443

```

빌드 절차

- Gitalb에서 Jenkins로 Webhooks을 연동한 다음 해당 브랜치에 push, merge 를 진행합니다.
 - start-prod.sh 쉘 파일 실행
 - docker-compose 실행
 - Dockerfile 실행
 - Server → Front 순으로 배포 진행

배포시 주의 사항

1. 프론트 엔드에서 서버 주소는 shabit.site 이므로 로컬 환경에서 구동할 시 localhost로 변경해야합니다.
2. 백엔드는 prod, dev, local 세 가지 파일로 분리하여 ip 주소는 건들이지 않고 db 주소 및 계정 정보를 변경하면 됩니다.

DB 정보

Maria DB

```

datasource:
  driver-class-name: org.mariadb.jdbc.Driver
  url: jdbc:mariadb://localhost:3306/ssafy
  username: easypeasy
  password: lemon

```

Redis + mongoDB

```
data:  
  redis:  
    host: localhost  
    port: 6379  
  
  mongodb:  
    host: localhost  
    port: 27017  
    authentication-database: admin  
    username: root  
    password: root  
    database: ssafy
```

외부 서비스

1. Google 로그인

1. Google Cloud 접속

Cloud Computing Services | Google Cloud

Meet your business challenges head on with cloud computing services from Google, including data management, hybrid & multi-cloud, and AI & ML.

 <https://cloud.google.com/>

2. 최초 가입이라면 카드 정보 등록
3. API 및 서비스
4. 프로젝트 생성
5. 사용자 인증 정보 - OAuth2 클라이언트 ID - 웹 어플리케이션 선택

API API 및 서비스	사용자 인증 정보	 사용자 인증 정보 만들기	삭제				
<ul style="list-style-type: none">❖ 사용 설정된 API 및 서비스■ 라이브러리● 사용자 인증 정보▷ OAuth 동의 화면≡ 페이지 사용 동의	<p>사용 설정한 API에 액세스하려면 사용자 인증 정보를 만드세요. 자세히 알아보기.</p> <p>API 키</p> <table border="1"><thead><tr><th>□ 이름</th><th>생성일</th></tr></thead><tbody><tr><td>표시할 API 키가 없습니다.</td><td></td></tr></tbody></table> <p>OAuth 2.0 클라이언트 ID</p>	□ 이름	생성일	표시할 API 키가 없습니다.			
□ 이름	생성일						
표시할 API 키가 없습니다.							

1. 서버 URI 추가

- ex) <https://shabit.site/login/oauth2/code/google>

승인된 리디렉션 URI ?

웹 서버의 요청에 사용

URI 1 *

! 올바르지 않은 리디렉션 URI는 비워 둘 수 없습니다.

[+ URI 추가](#)

참고: 설정이 적용되는 데 5분에서 몇 시간이 걸릴 수 있습니다.

[만들기](#) [취소](#)

- 생성 후 클라이언트 ID 및 Secret 키 저장



2. Kakao 로그인

- 카카오 개발자 사이트 접속

Kakao Developers
카카오 API를 활용하여 다양한 어플리케이션을 개발해보세요. 카카오 로그인, 메시지 보내기, 친구 API, 인공지능 API 등을 제공합니다.

<https://developers.kakao.com/>

kakao developers

- 새 애플리케이션 생성
- 좌측 제품 설정 → 카카오 로그인
- 활성화 설정
- 하단 Redirect URI에 서버 주소 추가
 - <https://shabit.site/login/oauth2/code/kakao>
- 카카오 이메일을 필수 항목으로 지정할 경우 비즈니스 앱 등록을 해야한다.

동의항목

카카오 로그인으로 서비스를 시작할 때 동의 받는 항목을 설정
비즈니스 채널을 연결하면 권한이 없는 동의 항목에 대한 검수

카카오비즈니스 관리자센터에서 비즈니스 채널 연결 →

7. 클릭 후 비즈 - 앱 연결 확인 후 목적은 **[이메일 필수 항목]** 으로 설정

8. 카카오 클라이언트 ID ⇒ 앱키 - REST API 키

앱 키

플랫폼	앱 키
네이티브 앱 키	5b5f7
REST API 키	7b4cc
JavaScript 키	8555e
Admin 키	36c4c

9. 카카오 클라이언트 Secret 키 : 제품 설정 → 보안 → 키 발급

Client Secret

토肯 발급 시, 보안을 강화하기 위해 Client

코드	yj34MVM
활성화 상태	사용안함

- 활성화 상태는 '사용안함'로 설정!

3. Naver 로그인

- Naver 로그인은 앱 개발자 등록 및 해당 검수 요청을 신청해 인증을 받아야만 사용할 수 있다.
- 네이버 개발자 사이트 접속

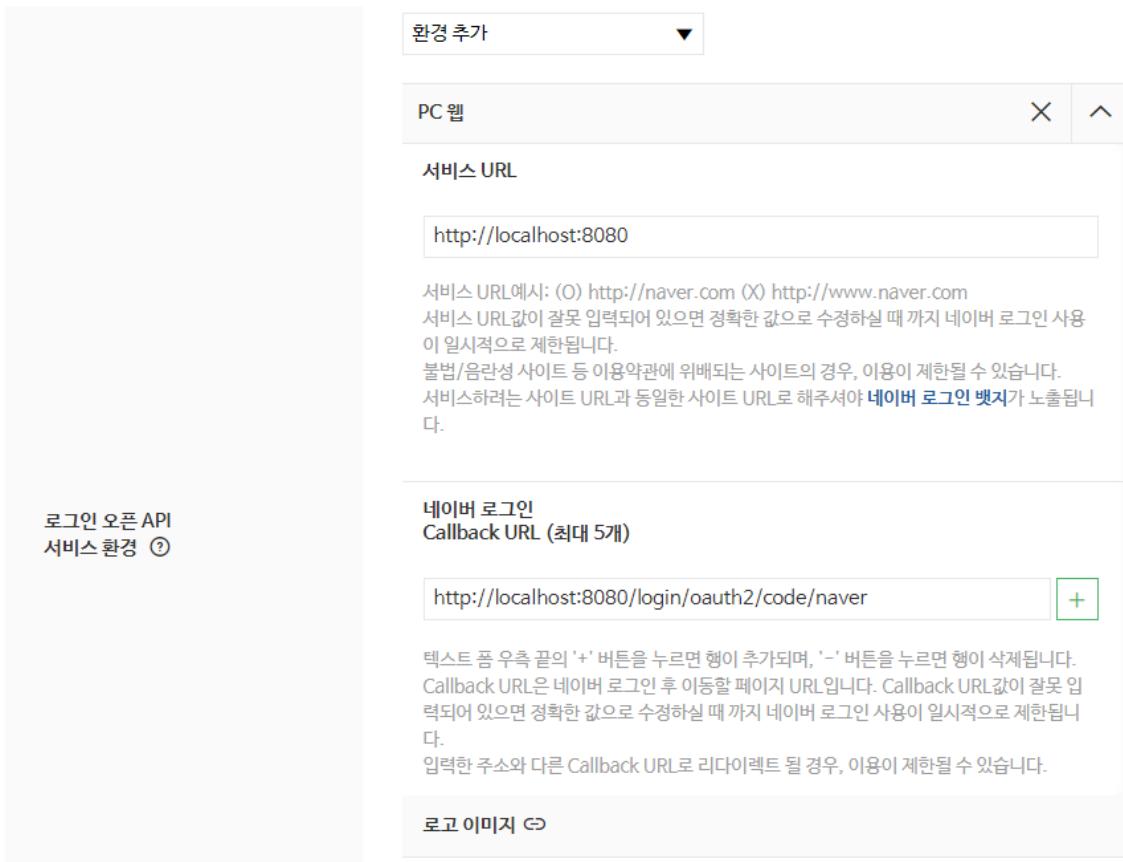
NAVER Developers

네이버 오픈 API들을 활용해 개발자들이 다양한 애플리케이션을 개발할 수 있도록 API 가이드와 SDK를 제공합니다.
제공중인 오픈 API에는 네이버 로그인, 검색, 단축URL, 캡차를 비롯 기계번역, 음성인식, 음성합성 등이 있습니다.

<https://developers.naver.com/main/>



- 상단 Application → 내 애플리케이션 접속
- 프로젝트 선택
- API 설정
- 로그인 오픈 API서비스 환경 설정



6. 앱 로고 등록 후 프로젝트 템 상단에 네이버 로그인 검수 상태 신청
7. 요구하는 이미지를 등록하고 검수 요청 하면 1~2일 이내로 완료됩니다.

<input checked="" type="checkbox"/> 검수 신청 전 가이드 확인	<input checked="" type="checkbox"/> 검수 요청 가이드 확인
--	--

아래 정보들은 [API 설정] 탭에서 조회할 정보로 선택하신 항목입니다.

네이버 로그인을 적용하려는 서비스 페이지에서 각각의 항목이 보여지는 화면을 캡처하여 업로드하고, 첨부 여부 체크박스를 체크하여 모든 항목이 업로드 되었는지 다시 한번 확인해 주세요.

만약 사용하지 않는 정보가 있다면 [API 설정] 탭으로 이동하여 해당 항목의 체크박스를 해제하고 수정 내역을 저장해 주세요.

① 제공 정보 활용처 확인 ↗	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">제공 정보(권한)</th> <th style="width: 30%;">필수/추가 여부</th> <th style="width: 40%;">첨부 여부</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">이메일 주소</td> <td style="padding: 5px;">필수</td> <td style="padding: 5px;"><input type="checkbox"/></td> </tr> <tr> <td style="padding: 5px;">회원이름</td> <td style="padding: 5px;">필수</td> <td style="padding: 5px;"><input type="checkbox"/></td> </tr> </tbody> </table>	제공 정보(권한)	필수/추가 여부	첨부 여부	이메일 주소	필수	<input type="checkbox"/>	회원이름	필수	<input type="checkbox"/>
제공 정보(권한)	필수/추가 여부	첨부 여부								
이메일 주소	필수	<input type="checkbox"/>								
회원이름	필수	<input type="checkbox"/>								

캡처 파일 업로드 ①

- pdf, jpg, gif, png 및 오피스 파일 형식을 첨부하실 수 있습니다.
- 5MB 이하의 파일을 최대 5개까지 업로드 할 수 있습니다.

네이버 로그인을 통한 신규 회원가입에 적용
 회원가입 없이 단순 로그인 인증에 적용
 게시판 글쓰기 또는 댓글 작성 등 일부 메뉴에 적용
 기타

- “카페24” 호스팅사를 통해 사이트를 만드시는 경우, 기타에 체크하고 “카페24”를 적어주세요.
- 네이버 로그인을 신규 회원가입에 적용하실 경우, 네이버 로그인을 통한 신규 회원가입 과정에서 별도의 비밀번호를 요구한다면 검수는 거부됩니다. 관련하여 자세한 사항은 [네이버 로그인 검수 가이드](#)를 참고해 주세요.

② 서비스 적용 형태 확인 ↗

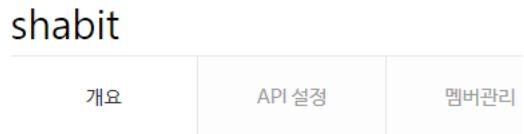
네이버 로그인 이용 절차를 확인할 수 있는 캡처파일 첨부

- 서비스에서 “네이버 로그인” 버튼을 클릭하여 로그인/회원가입을 완료하기까지 노출되는 화면을 단계별로 캡처하여 첨부해 주세요.
- 자세한 내용은 [\[검수 가이드 - 네이버 로그인 적용 형태 확인을 위한 자료 제출 방법\]](#)을 참고해 주세요.
- pdf, jpg, gif, png 및 오피스 파일 형식을 첨부하실 수 있습니다.
- 5MB 이하의 파일을 최대 5개까지 업로드 할 수 있습니다.

아래 내용 중 하나를 선택해 주세요. 해당되는 것이 없으면 “없음”으로 선택해 주세요.

스포츠 게임 배팅, 투표권발행 유사행위를 제공하는 서비스
 온라인으로 주류를 판매하는 서비스

1. 프로젝트 상단에 개요 탭에서 클라이언트 ID, Secret 키 확인 가능



4. application.yml 설정

OAuth2 설정

```
spring:
  security:
    oauth2:
      client:
        registration:
          google:
            client-id: 구글 Client ID
            client-secret: 구글 Client Secret
            redirect-uri: '{baseUrl}/{action}/oauth2/code/{registrationId}'
            scope: profile, email // 사용할 필수 정보들
          naver:
            client-id: 네이버 Client ID
            client-secret: 네이버 Client Secret
            redirect-uri: '{baseUrl}/{action}/oauth2/code/{registrationId}'
            authorization-grant-type: authorization_code
            scope:
              - name
              - email
              - profile_image
            client-name: Naver
          kakao:
            client-id: 카카오 Client ID
            client-secret: 카카오 Client Secret
            scope:
              - profile.nickname
              - account_email
              - profile.image
            client-name: Kakao
            authorization-grant-type: authorization_code
            redirect-uri: '{baseUrl}/{action}/oauth2/code/{registrationId}'
            client-authentication-method: POST

        provider:
          naver:
            authorization-uri: https://nid.naver.com/oauth2.0/authorize
            token-uri: https://nid.naver.com/oauth2.0/token
            user-info-uri: https://openapi.naver.com/v1/nid/me
            user-name-attribut: response
          kakao:
            authorization-uri: https://kauth.kakao.com/oauth/authorize
            token-uri: https://kauth.kakao.com/oauth/token
            user-info-uri: https://kapi.kakao.com/v2/user/me
            user-name-attribut: id

      app: // 클라이언트 인가 코드
        oauth2:
          authorizedRedirectUris: https://shabit.site/oauth/redirect
```

SMTP

```

spring:
  mail:
    host: smtp.gmail.com
    port: 587
    username: {google 메일 아이디}
    password: {google 계정 앱 key}
    properties:
      mail:
        smtp:
          auth: true
          starttls:
            enable: true

    mail:
      setFrom: {google 이메일}

```

S3 Bucket

```

cloud:
  aws:
    s3:
      bucket: 버킷 이름
      region:
        static: ap-northeast-2 #Asia Pacific -> seoul
      stack:
        auto: false
      credentials:
        access-key: S3 사용자 access-key
        secret-key: S3 사용자 secret-key

```

서비스 이용 방법

Google Cloud Platform

1. Google Cloud Platform 접속 후 새 프로젝트 만들기 → 좌측 햄버거 메뉴에서 API 및 서비스 클릭 → 라이브러리 클릭
 2. YouTube Data API v3 검색 → 사용 클릭 → 관리 클릭 → 사용자 인증 정보 클릭
 3. 사용자 인증 정보 만들기 → API 키 클릭
2. 발급받은 API 키를 application.yml에 다음과 같이 입력

```

key:
  youtube: {API 키}

```

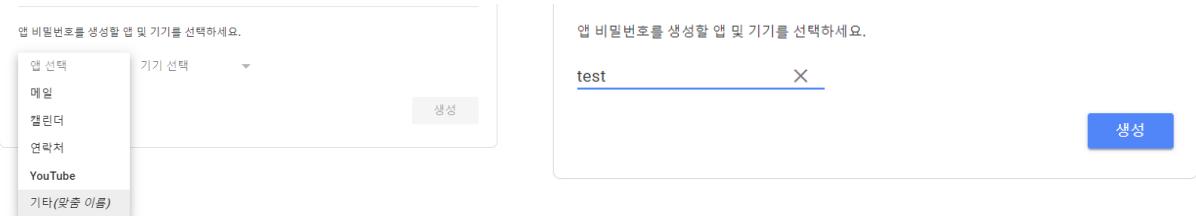
SMTP - 구글 메일 설정

1. 구글 계정 생성
2. 프로필 클릭 → 계정 설정 → 2단계 인증 활성화



3. 2단계 인증 활성화 이후 앱 비밀번호 발급

- 앱 선택 : 기타(맞춤 이름)
- 앱 비밀번호를 생성할 이름 입력 후 생성



4. 앱 비밀번호 발급 후 앱 key 입력

생성된 앱 비밀번호

기기용 앱 비밀번호

[Redacted]

사용 방법

설정하려는 애플리케이션 또는 기기의 Google 계정 설정으로 이동합니다. 비밀번호를 위에 표시된 16자리 비밀번호로 교체합니다.
일반적인 비밀번호와 마찬가지로 이 앱 비밀번호는 Google 계정에 대한 완전한 액세스 권한을 부여합니다. 비밀번호를 기억하지 않아도 되므로 적어 놓거나 다른 사용자와 공유하지 마세요.

확인

```
spring:  
  mail:  
    host: smtp.gmail.com  
    port: 587  
    username: {google 메일 아이디}  
    password: {google 계정 앱 key}
```

5. G메일 → 상단 설정 버튼 클릭 → 빠른 설정/모든 설정 보기 클릭 → 전달 및 POP/IMAP 클릭 → 이메일 보내기 설정

기본설정 라벨 받은편지함 계정 및 가져오기 끌티 및 차단된 주소 **전달 및 POP/IMAP** 부가기능 채팅 및 Meet 고급 오프라인 테마

전달:
자세히 알아보기

전달 주소 추가

도움말: 필터를 만들면 매 일 중 일부만 전달할 수도 있습니다.

POP 다운로드:
자세히 알아보기

1. 상태: 모든 메일에 대해 POP을 사용 설정되어 있습니다.
 이미 다운로드 된 메일을 포함하여 모든 메일에 POP을 활성화 하기
 지금부터 수신되는 메일에만 POP을 사용하기
 POP 사용 안함

2. POP로 메시지를 여는 경우 **Gmail 사본을 받은편지함에 보관하기**

3. 이메일 클라이언트 구성 (예: Outlook, Eudora, Netscape Mail)
설정 방법

IMAP 액세스:
(IMAP를 사용하여 다른 클라이언트에서 Gmail에 액세스)
자세히 알아보기

상태: IMAP를 사용할 수 있습니다.
 IMAP 사용
 IMAP 사용 안함

IMAP에서 메일을 삭제된 것으로 표시하는 경우:
 자동 삭제 사용 - 서버를 즉시 업데이트(기본값)
 메일을 휴지통으로 이동
 메일을 즉시 완전삭제

폴더 크기 제한
 IMAP 폴더에서 메일 수를 제한하지 않습니다(기본값).
 이만큼의 메일만 포함하도록 IMAP 폴더를 제한합니다. **1,000**

이메일 클라이언트 구성(예: Outlook, Thunderbird, iPhone)
설정 방법

변경사항 저장 **취소**

S3 버킷

1. AWS 계정 생성

2. AWS 서비스 검색창에 S3 검색 및 이동

3. 버킷 클릭

4. 버킷 정보 입력

일반 구성

버킷 이름
shabit-test

버킷 이름은 전역에서 고유해야 하며 공백 또는 대문자를 포함할 수 없습니다. [버킷 이름 지정 규칙 참조](#)

AWS 리전
아시아 태평양(서울) ap-northeast-2

기존 버킷에서 설정 복사 - 선택 사항
다음 구성의 버킷 설정만 복사됩니다.

버킷 선택

액체 소유권 Info

다른 AWS 계정에서 이 버킷에 작성한 객체의 소유권 및 액세스 제어 목록(ACL)의 사용을 제어합니다. 객체 소유권은 객체에 대한 액세스를 지정할 수 있는 사용자를 결정합니다.

ACL 비활성화됨(경고)
이 버킷의 모든 객체는 이 계정이 소유합니다. 이 버킷과 그 객체에 대한 액세스는 정책을 통해서만 지정됩니다.

ACL 활성화됨
이 버킷의 객체는 다른 AWS 계정에서 소유할 수 있습니다. 이 버킷 및 객체에 대한 액세스는 ACL을 사용하여 지정할 수 있습니다.

액체 소유권
버킷 소유자 적용

ACL 비활성화 관련 항후 권한 변경 사항
2023년 4월부터는 S3 콘솔을 사용하여 버킷을 생성할 때 ACL을 비활성화하기 위해 더 이상 s3:PutBucketOwnershipControls 권한이 필요하지 않습니다. [자세히 알아보기](#)

이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지정 정책 또는 오류를 통해 버킷 및 객체에 부여됩니다. 이 버킷 및 해당 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 모든 퍼블릭 액세스 차단을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지정에만 적용됩니다. AWS 서비스는 모든 퍼블릭 액세스 차단을 활성화하는 경우, 해당 서비스는 해당 버킷에 대한 퍼블릭 액세스를 제공하지 않습니다. 이 설정은 모든 퍼블릭 액세스 차단에 대한 정도로, 다른 모든 퍼블릭 액세스가 활성화되는 경우 흥행 스트리밍 서비스에 의해 차단 설정을 사용자 지정할 수 있습니다. [자세히 알아보기](#)

모든 퍼블릭 액세스 차단

이 설정을 활성화하면 아래 4개의 설정을 모두 활성화한 것과 같습니다. 다음 설정 각각은 서로 독립적입니다.

- 새 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단 S3는 모든 퍼블릭 버킷 및 객체에 적용되는 퍼블릭 액세스 설정을 차단하여, 기본 버킷 및 객체에 대한 새 퍼블릭 액세스 ACL 생성을 금지합니다. 이 설정은 ACL을 사용하여 S3 리소스에 대한 퍼블릭 액세스를 적용하는 기준을 한정하지 않습니다.
- 임의의 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단 S3는 버킷 및 객체에 대한 고급 퍼블릭 액세스를 부여하는 모든 ACL을 무시합니다.
- 새 퍼블릭 버킷 또는 액세스 지정 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단 S3는 버킷 및 객체에 대한 고급 퍼블릭 액세스를 부여하는 정책을 사용하는 버킷 또는 액세스 지정에 대한 퍼블릭 및 고급 액세스 설정을 무시합니다.
- 임의의 퍼블릭 버킷 또는 액세스 지정 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 및 고급 액세스 차단 S3는 버킷 및 객체에 대한 고급 퍼블릭 액세스를 부여하는 정책을 사용하는 버킷 또는 액세스 지정에 대한 퍼블릭 및 고급 액세스 설정을 무시합니다.

모든 퍼블릭 액세스 차단을 비활성화하면 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있습니다.

정책을 사이트 호스팅과 같은 구체적으로 확인된 사용 사례에서 퍼블릭 액세스가 필요한 경우가 아니면 모든 퍼블릭 액세스 차단을 활성화하는 것이 좋습니다.

현재 설정으로 인해 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있음을 알고 있습니다.

모든 퍼블릭 액세스 차단을 비활성화하면 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있습니다.

정책을 사이트 호스팅과 같은 구체적으로 확인된 사용 사례에서 퍼블릭 액세스가 필요한 경우가 아니면 모든 퍼블릭 액세스 차단을 활성화하는 것이 좋습니다.

현재 설정으로 인해 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있음을 알고 있습니다.

5. 버킷 설정 - 권한 설정

- 권한 클릭 → 버킷 정책 편집 클릭

shabit-test Info

권한 선택 지표 관리 액세스 지정

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. [자세히 알아보기](#)

버킷 ARN
arn:aws:s3:::shabit-test

정책

정책 예제 정책 생성기

정책 생성기

정책 예제

- 권한 클릭 → 버킷 정책 편집 클릭 → 정책 생성기 클릭

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. [자세히 알아보기](#)

버킷 ARN
arn:aws:s3:::shabit-test

정책

- Action에는 `GetObject`, `PutObject`, `DeleteObject` 3개를 체크하고 ARN에는 복사해둔 ARN값을 입력한다.
- ARN값을 입력하되 /* 값도 추가 해줘야한다. ARN 값이 `arn:aws:s3:::test/*` 라고 입력해주면 된다.

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an IAM Policy, an S3 Bucket Policy, an SQS Queue Policy.



Step 2: Add Statement(s)

A statement is the formal description of a single permission. See a [description of elements](#) that you can use in statements.

Effect Allow Deny

Principal *

Use a comma to separate multiple values.

AWS Service Amazon S3 All Services (*)

Use multiple statements to add permissions for more than one service.

Actions 3 Action(s) Selected All Actions (*)

Amazon Resource Name (ARN) arn:aws:s3:::habit-test/*

ARN should follow the following format: arn:aws:s3:::\${BucketName}/\${KeyName}.
Use a comma to separate multiple values.

Add Conditions (Optional)

Add Statement Add Statement

- 스크롤을 아래로 내려 **Generate Policy** 버튼을 클릭한다.
- 생성된 정책을 복사한뒤 정책 편집 부분에 붙여넣은 후 **변경** **사항 저장** 버튼을 눌러 저장한다.

Policy JSON Document
Click below to edit. To save the policy, copy the text below to a text editor.
Changes made below will not be reflected in the policy generator tool.

```
{
  "Id": "Policy1676275934590",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1676275934597",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::habit-test/*",
      "Principal": "*"
    }
  ]
}
```

버킷 정책
JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. 자세히 알아보기

```
버킷 ARN
arnaws3::habit-test

정책
1 - [
2   {
3     "Sid": "Stmt1676275934597",
4     "Action": [
5       "s3:DeleteObject",
6       "s3:GetObject",
7       "s3:PutObject"
8     ],
9     "Effect": "Allow",
10    "Resource": "arn:aws:s3:::habit-test/*",
11    "Principal": "*"
12  }
13]
```

6. 사용자 등록

- AWS 서비스 검색창에 **IAM** 검색 → **사용자** 클릭

Identity and Access Management(IAM)

서비스 (7)
특정 (19)
리소스 **New**
블로그 (56)
설명서 (118,605)
자신 기사 (50)

IAM ★
IAM은 AWS와 같은 서비스에 대한 액세스 권한을 관리하는 서비스입니다.

주요 기능
그룹 **사용자** **액세스 키** **정책** **액세스 분석기**

IAM > 사용자

사용자 (1) 정보
IAM 사용자는 계정에서 AWS와 상호 작용하는 데 사용되는 장기 자격 증명을 가진 자격증명입니다.

사용자 이름	그룹	마지막 활동	MFA
aws-user	없음	9분 전	없음

7. 사용자 생성

사용자 세부 정보

사용자 이름
shabit-user

사용자 이름은 최대 64자까지 가능합니다. 유효한 문자: A~Z, a~z, 0~9 및 + = , . -(하이픈)

콘솔 액세스 활성화 - 선택 사항
사용자가 AWS 관리 콘솔에 로그인할 수 있도록 허용하는 암호를 활성화합니다.

콘솔 암호

자동 생성된 암호
사용자를 생성한 후 암호를 볼 수 있습니다.

사용자 지정 암호
사용자의 사용자 지정 암호를 입력합니다.

- 8자 이상이어야 합니다.
- 다음 문자 유형 중 세 가지 이상을 조합하여 포함해야 합니다. 대문자(A~Z), 소문자(a~z), 숫자(0~9), 기호 ! @ # \$ % ^ & * () _ + -(하이픈) = [] { } | '

암호 표시

사용자는 다음 로그인 시 새 암호를 생성해야 합니다(권장).
사용자는 IAMUserChangePassword [정책](#)을 자동으로 가져와 암호를 변경할 수 있도록 허용합니다.

① 프로그래밍 방식의 액세스의 경우 사용자를 생성한 후 액세스 키를 생성할 수 있습니다. 자세히 알아보기 [\[\]](#)

8. 사용자 설정

- 보안 자격 증명 클릭 → 액세스 키 발급 클릭

shabit-user

삭제

요약		
ARN arn:aws:iam::120626585696:user/shabit-user	콘솔 액세스 ⚠️ MFA 없이 활성화됨	액세스 키 1 활성화되지 않음
생성됨 February 13, 2023, 17:33 (UTC+09:00)	마지막 콘솔 로그인 ① 안 함	액세스 키 2 활성화되지 않음
권한 그룹 태그 보안 자격 증명 액세스 관리자		

권한 정책 (1)
사용자에게 직접 연결된 정책을 통해 또는 그룹을 통해 권한을 정의합니다.
[\[\]](#) [제거](#) [권한 추가 ▾](#)

액세스 키 (0)

액세스 키를 사용하여 AWS CLI, AWS Tools for PowerShell, AWS SDK 또는 직접 AWS API 호출을 통해 AWS에 프로그래밍 방식 호출을 전송합니다. 한번에 최대 두 개의 액세스 키(활성 또는 비활성)를 가질 수 있습니다. [Learn more \[\]](#)

[액세스 키 만들기](#)

액세스 키 없음

액세스 키와 같은 장기 자격 증명을 사용하지 않는 것이 모범 사례입니다. 대신 단기 자격 증명을 제공하는 도구를 사용하세요. [Learn more \[\]](#)

[액세스 키 만들기](#)

액세스 키 모범 사례 및 대안

보안 개선을 위해 액세스 키와 같은 장기 자격 증명을 사용하지 마세요. 다음과 같은 사용 사례와 대안을 고려하세요.

- Command Line Interface(CLI)**
AWS CLI를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- 로컬 코드**
노출 가능한 애플리케이션 코드를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- AWS 컴퓨팅 서비스에서 실행되는 애플리케이션**
Amazon ECS 또는 AWS Lambda와 같은 AWS 컴퓨팅 서비스에서 실행되는 애플리케이션 코드를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- 서드 파티 서비스**
AWS 리소스를 보니워킹 또는 관리하는 서드 파티 애플리케이션 또는 서비스에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- AWS 외부에서 실행되는 애플리케이션**
애플리케이션을 온프레미스 호스트에서 실행하거나 로컬 AWS 클라이언트 또는 서드 파티 AWS 클라이언트 사용을 사용할 수 있도록 이 액세스 키를 사용할 것입니다.
- 기타**
귀족의 사용 사례가 여기에 나열되어 있지 않습니다.

설명 태그 설정 - 선택 사항

이 액세스 키에 대한 설명은 이 사용자에게 태그로 연결되고, 액세스 키와 함께 표시됩니다.

설명 태그 값
이 액세스 키의 풍도와 사용 위치를 설명합니다. 좋은 설명은 나중에 이 액세스 키를 자신 있게 교체하는데 유용합니다.
최대 256자까지 가능합니다. 허용되는 문자, 숫자, UTF-8로 표현할 수 있는 공백 및 _ . : / = + - @입니다.

취소 이전 액세스 키 만들기

액세스 키 검색

액세스 키	비밀 액세스 키
○	○

액세스 키 모범 사례

- 액세스 키를 일반 텍스트, 코드 라이브러리 또는 코드로 저장해서는 안됩니다.
- 더 이상 필요 없는 경우 액세스 키를 비활성화하거나 삭제합니다.
- 최근 클릭을 활성화합니다.
- 액세스 키를 장기적으로 고지합니다.

액세스 키 관리에 대한 자세한 내용은 AWS 액세스 키 관리 모범 사례를 참조하세요.

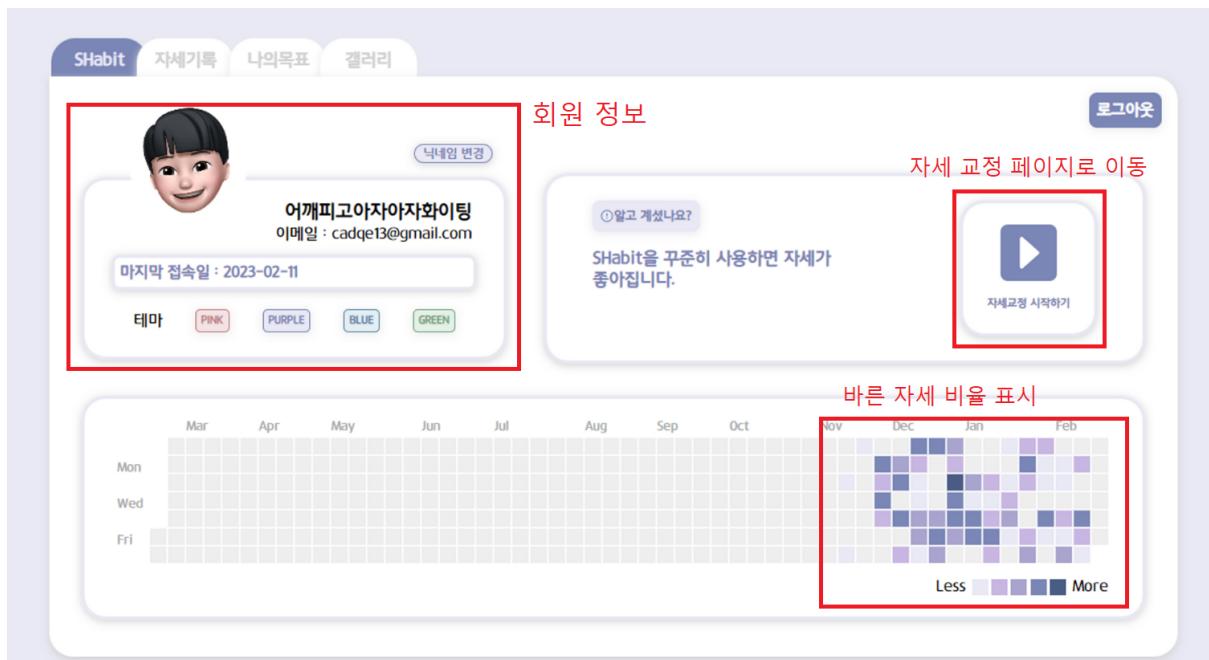
.csv 파일 다운로드 | 엔터

9. 발급받은 `access-key`, `secret-key` 입력

```
cloud:  
aws:  
  s3:  
    bucket: 버킷 이름  
region:  
  static: ap-northeast-2 #Asia Pacific -> seoul  
stack:  
  auto: false  
credentials:  
  access-key: S3 사용자 access-key  
  secret-key: S3 사용자 secret-key
```

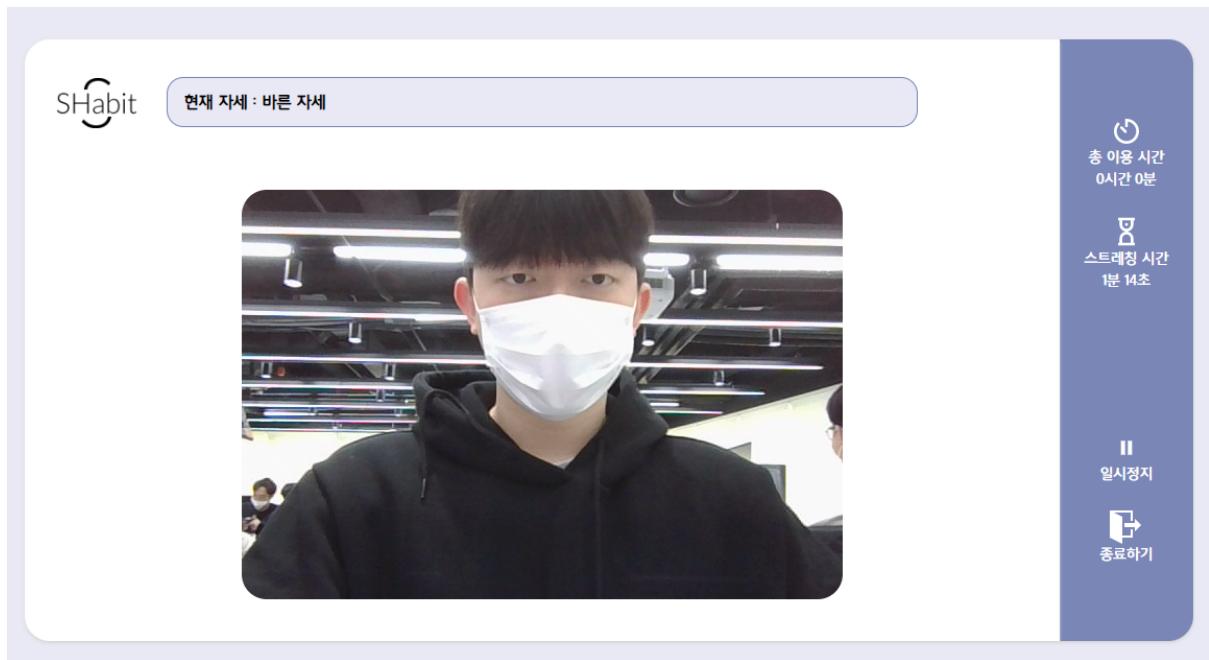
시연 시나리오

메인 페이지 소개

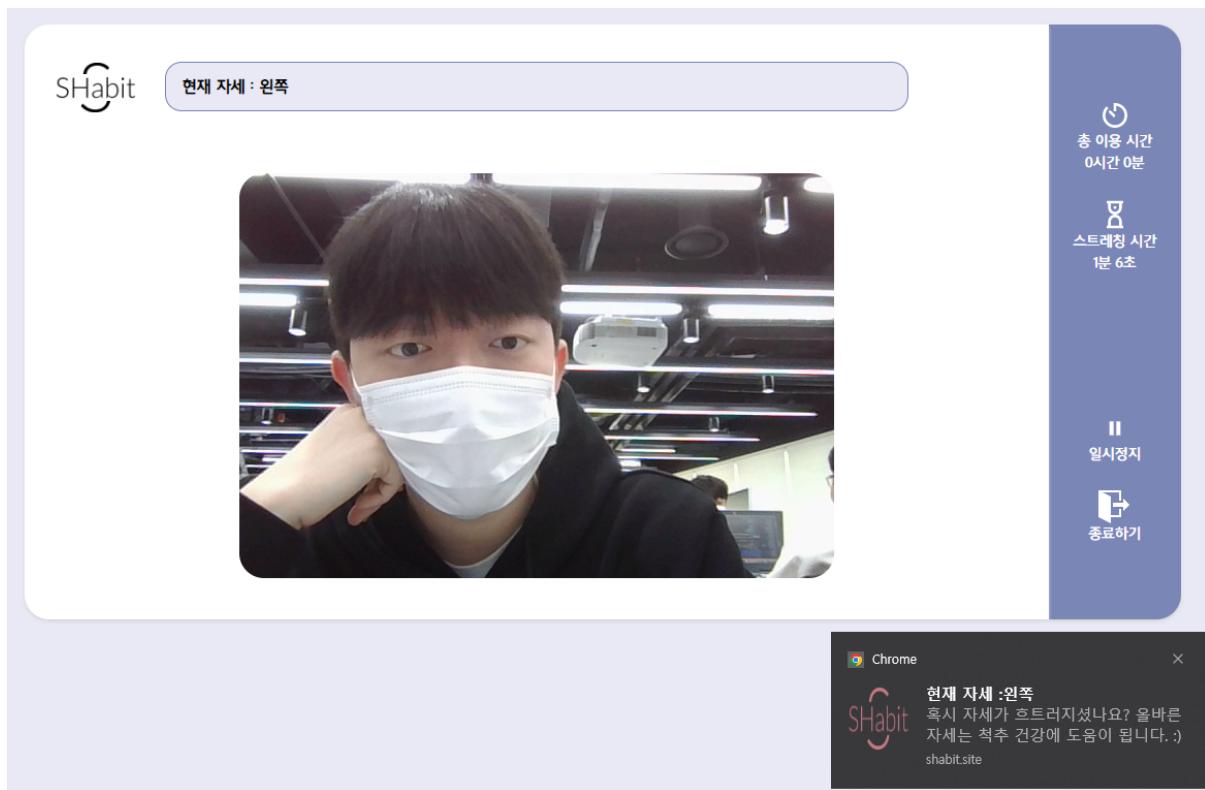


- 메인 페이지에서는 회원 정보 제공, 바른 자세를 유지한 비율을 하단에 진한 정도로 표시해주고 있습니다.
- 자세 교정 시작하기 버튼을 눌러 자세 교정 페이지로 진입합니다.
- 상단의 메뉴 부분에서 자세 기록, 나의 목표, 갤러리로 이동할 수 있습니다.

자세 교정 페이지



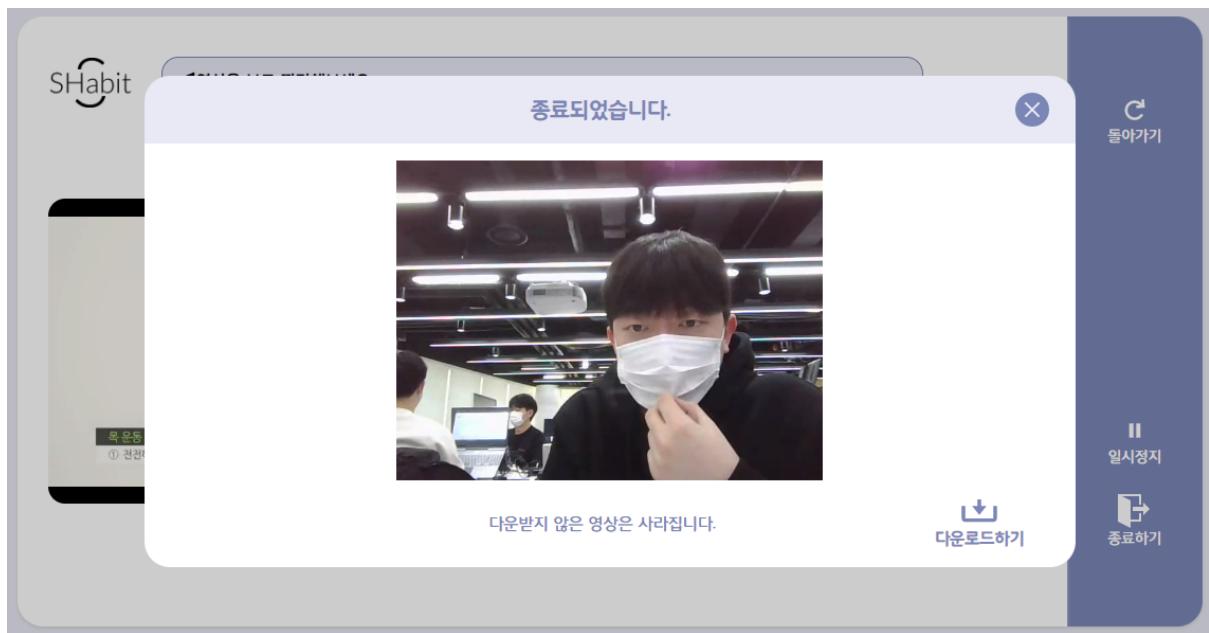
- 자세 교정 페이지에서는 중앙에 현재 사용자의 상태가 카메라로 표시됩니다.
- 상단에는 유지 중인 자세를 텍스트로 표시하고 자세가 바뀌면 실시간으로 표시합니다.
- 우측 사이드 바에서는 총 이용시간, 스트레칭까지 남은 시간, 일시정지, 종료하기 등 기능을 제공합니다.



- 바른 자세를 3분간 유지하지 않을 경우 알람을 통해 사용자에게 현재 자세를 경고 알람으로 띄워 줍니다.
- 알람을 통해 바른 자세를 유지할 수 있도록 합니다.

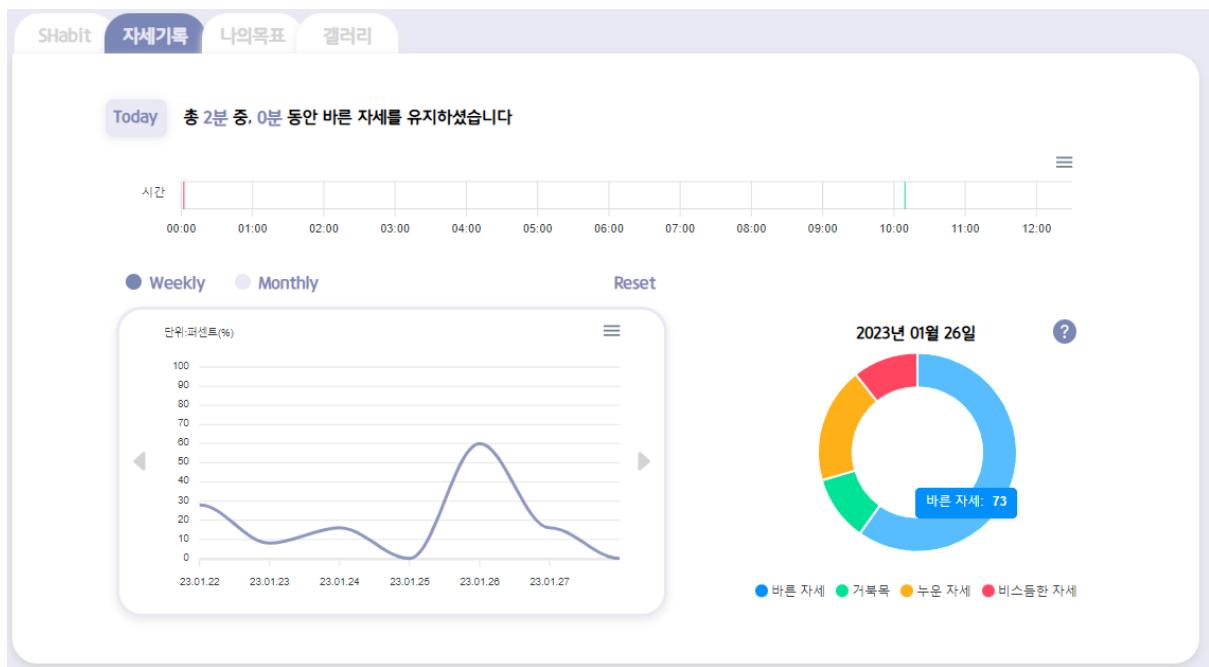


- 스트레칭 시간이 되면 사용자가 오래 유지한 바르지 않은 자세 데이터를 기반으로 스트레칭 영상을 추천해줍니다.
- 영상을 선택하게 되면 유튜브 영상을 띄워주면서 스트레칭을 할 수 있습니다.



- 작업을 마치고 종료를 하게 되면 자세 교정을 하면서 취한 자세를 짧은 영상으로 편집하여 제공합니다.
- 이를 통해 자신이 취한 자세를 간단하게 확인할 수 있습니다.

자세 기록 페이지



- 사용자의 자세 데이터를 기반으로 수치화하여 날짜별로 시각화시켜 표시해줍니다.
- 상단에는 오늘 사용한 자세 데이터를 자세하게 보여주고 하단에는 주간, 월간 별로 바른 자세를 취한 비율을 기반으로 그래프를 제공하고 있습니다.

목표 페이지



- 효과적인 바른 자세를 위해 목표를 설정하고 바른 자세를 유지한 시간을 목표 수치에 따라 확인해볼 수 있습니다.
- 우측 상단에는 해당 페이지를 스크린샷으로 찍어 SNS 등 친구들과 공유할 수 있습니다.

갤러리 페이지



- 자신이 작업하며 무의식적으로 취했던 자세를 볼 수 있는 페이지입니다.
- 갤러리 페이지에서는 사용자가 취했던 자세를 캡쳐하여 목록 형식으로 보여줍니다.