

누리:봄

B303 : NURI:BOM

삼성 청년 SW아카데미 대전캠퍼스 7기

자율프로젝트 (2022. 10. 10 ~ 2022. 11. 21)

포팅 매뉴얼

김현주(팀장), 김동신, 김선후, 박정현, 전병찬, 최원재

목차

프로젝트 개요.....	3
프로젝트 기술 스택.....	3
CI/CD.....	6
빌드 상세내용.....	12
DB.....	14
외부 서비스.....	16

프로젝트 개요

누리봄은 음성 인식 스피커와 TV를 기반으로 제공되는 홀로어르신 생활 케어 서비스입니다. 음성 명령을 통해서 음악 재생, 어르신을 위한 체조, 초성 퀴즈 게임, 영상 편지 기능을 실행해서 홀로어르신의 외로움을 달래주고, 모션 인식을 통해 낙상을 감지하여 어르신이 도움이 필요한 상황에 처한 경우 자동으로 신고하여 홀로어르신을 안전하게 보살펴 드립니다.

프로젝트 기술 스택

가. 이슈 관리 : Jira

나. 형상 관리 : GitLab

다. 커뮤니케이션 : Mattermost, Notion

라. 개발환경

1) OS : Windows

2) 하드웨어

A. LattePanda 3 Delta : Ubuntu 20.04.4 LTS

B. Arduino Uno

C. Coral

3) IDE

A. IntelliJ IDEA Ultimate

B. Visual Studio Code

C. Aduino IDE

D. UI/UX : Figma

4) Database

A. MySQL Workbench

B. Firebase

5) Server : AWS EC2 (MobaXterm)

A. Ubuntu 20.04.4 LTS

B. Docker 20.10.17

C. Jenkins 2.346.2 LTS

D. Nginx 1.18.0

마. 상세 내용

1) Backend

A. JAVA (Open JDK (Zulu 8.33.0.1))

B. Spring Boot Gradle 7.5

C. Lombok 1.18.20, Swagger3.0.0, JPA, QueryDSL

D. Python 3.8

2) Frontend

A. HTML, CSS, JavaScript

B. React 18.2.0

i. axios 0.27.2

ii. npm 8.15.0

iii. react-beautiful-웅 13.1.1

iv. react-calendar 3.9.0

v. react-redux 8.0.2

vi. react-router-dom 6.0.0

- vii. redux 4.2.0
- viii. redux-persist 6.0.0
- ix. redux-thunk 2.4.1
- x. styled-components 5.3.5
- xi. moment 2.29.4
- xii. sass-loader 13.1.0
- xiii. date-fns 2.29.3
- C. Nodejs 16.18.0

3) IOT

- A. Arduino 1.0.0
- B. EDGEPU 2.11.1
- C. TensorFlow Lite 2.5.0
- D. PyCoral 2.0.0
- E. Gstreamer 1.0.0
- F. Python 3.8.0

4) 외부 서비스

- A. Naver Clova
 - i. Voice
 - ii. Chatbot
- B. Google Speech to text

CI/CD

CI/CD 작동 방식은 GitLab에서 Push Event가 발생하면 WebHook 기능을 통해 Jenkins에서 자동으로 빌드를 실행합니다. Jenkins에서 Nginx서버를 내포한 React, Spring Boot 각 디렉토리 내부의 Dockerfile을 활용하여 Docker 이미지를 생성한 후 SSH 연결을 통해 AWS에 Docker Container를 생성하는 방식입니다.

1. MobaXterm을 이용해 EC2 서버 원격 접속.
2. Ubuntu환경에서 Docker 설치.
3. Docker-compose.yml을 만들고 Docker-compose up -d 명령어를 사용하여 Jenkins와 프로젝트에 사용할 MySQL 컨테이너를 생성.

```
version: '3'


services:
  jenkins:
    image: jenkins/jenkins:lts
    container_name: jenkins
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - /jenkins:/var/jenkins_home
    ports:
      - "9090:8080"
    privileged: true
    user: root
  db:
    image: mysql:8.0.29
    container_name: mysql
    command:
      - --default-authentication-plugin=mysql_native_password
      - --character-set-server=utf8mb4
      - --collation-server=utf8mb4_unicode_ci
      - --range_optimizer_max_mem_size=16777216
    restart: always
    environment:
      MYSQL_DATABASE: nuribom
      MYSQL_USER: nuribom
      MYSQL_PASSWORD: ajrqnf1a303
      MYSQL_ROOT_PASSWORD: tkxkrnsl12
      TZ: Asia/Seoul
    volumes:
      - ./mysql/data:/var/lib/mysql
    ports:
      - 3306:3306
```

4. 새로운 Jenkins 프로젝트 생성 및 설정


Enter an item name

deploy


» Required field




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




Pipeline
 Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project
 다양한 환경에서의 테스트, 플레폼 특성 빌드, 기타 등등 처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.



Folder
 Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.



Multibranch Pipeline
 Creates a set of Pipeline projects according to detected branches in one SCM repository.

A. 구성에서 소스코드 관리 -> Git 설정에서 Git 레포지토리에 관한 설정 진행

소스 코드 관리

Git ?

Repositories ?

Repository URL ?

https://lab.ssafy.com/s07-final/S07P31B303.git

Credentials ?

wj5295@naver.com/*****

+ Add

고급...

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/master

B. 빌드 유발 탭에서 아래와 같이 체크박스 체크. 그 후 고급 버튼을 클릭하고 Secret

token을 찾아 Generate 버튼을 눌러 토큰을 생성. (GitLab과 WebHook 연결에 사용)

빌드 유발

- ☐ 빌드를 원격으로 유발 (예: 스크립트 사용) ?
- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ Build when a change is pushed to GitLab. GitLab webhook URL: http://j7b206.p.ssafy.io:9090/project/yogidicedeploy_ ?
 - Enabled GitLab triggers
 - ☒ Push Events
 - ☐ Push Events in case of branch delete
 - ☒ Opened Merge Request Events

Secret token ?

b77deda04094746641db8d550704f9f1

Generate

Clear

C. Steps에서 Execute shell을 추가하여 다음과 같이 입력 후 저장.

Build Steps

≡ Execute shell ?

×

Command

See [the list of available environment variables](#)

```
docker image prune -a --force
mkdir -p /var/jenkins_home/images_tar
cd /var/jenkins_home/workspace/nuribom/NURI-BOM/
docker build -t spring .
docker save spring > /var/jenkins_home/images_tar/spring.tar

cd /var/jenkins_home/workspace/nuribom/nuribom-display/
docker build -t react-display .
docker save react-display > /var/jenkins_home/images_tar/react-display.tar

cd /var/jenkins_home/workspace/nuribom/nuribom-admin/
docker build -t react-admin .
docker save react-admin > /var/jenkins_home/images_tar/react-admin.tar

ls /var/jenkins_home/images_tar
```


5. 자동 빌드를 위한 GitLab WebHook 연결.

- A. 배포할 프로젝트의 GitLab 레포지토리에서 Settings-> Webhooks 페이지로 이동 후 URL에는 `http://배포서버공인IP:9090/project/{생성한jenkins프로젝트명}/`을 입력, Secret token에는 Jenkins 프로젝트 생성 시 발행한 Secret token 입력.

Webhook

Webhooks enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

URL

`http://k7b303.p.ssafy.io:9090/project/nuribom`

URL must be percent-encoded if it contains one or more special characters.

Secret token

`04282595bf613b254ce93f11525add8b`

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

Trigger

☒ Push events

☐ Tag push events

☐ Comments

☐ Confidential comments

☐ Issues events

☐ Confidential issues events

☒ Merge request events

master

Push to the repository.

A new tag is pushed to the repository.

A comment is added to an issue or merge request.

A comment is added to a confidential issue.

An issue is created, updated, closed, or reopened.

A confidential issue is created, updated, closed, or reopened.

A merge request is created, updated, or merged.

- B. Webhook을 생성 후 test로 정상 작동하는지 확인.

6. Jenkins SSH 연결 설정 (Publish over SSH)

- A. Dashboard -> Jenkins 관리 -> 시스템 설정의 Publish over SSH 탭으로 이동
- B. SSH Servers 탭에 추가 버튼 클릭
- C. Name, Hostname, Username을 아래와 같이 입력 후 고급 버튼 클릭.

SSH Servers

SSH Server

Name ?

nuribom


Hostname ?

k7b303.p.ssafy.io

Username ?

ubuntu


Remote Directory ?

 고급...

- D. Use password authentication, or use different key 체크 후, 제공 받은 pem 파일의 내용을 복사 붙여 넣기.

☒ Use password authentication, or use a different key ?

Passphrase / Password ?

 Concealed

Change Password

Path to key ?

Key ?

-----BEGIN RSA PRIVATE KEY-----

7. 빌드 후 조치

- A. 프로젝트 구성 페이지의 빌드 후 조치 추가 클릭, Send build artifacts over SSH를 선택
- B. 빌드 후 실행할 명령문을 다음과 같이 작성

Send build artifacts over SSH ?

SSH Publishers

SSH Server

Name ?

deploytest

고급...

Transfers

Transfer Set

Source files ?

/README.md

Exec command ?

```
sudo docker load < /jenkins/images_tar/react-display.tar
sudo docker load < /jenkins/images_tar/react-admin.tar
sudo docker load < /jenkins/images_tar/spring.tar

if (sudo docker ps | grep "react-display"); then sudo docker stop react-display; fi
if (sudo docker ps | grep "react-admin"); then sudo docker stop react-admin; fi
if (sudo docker ps | grep "spring"); then sudo docker stop spring; fi

sudo docker run -it -d --rm -p 3000:3000 --name react-display react-display
echo "Run react-display"
sudo docker run -it -d --rm -p 80:80 -p 443:443 --name react-admin react-admin
echo "Run react-react-admin"
sudo docker run -it -d --rm -p 8081:8081 --name spring spring
echo "Run Spring"
```

8. HTTPS 적용 및 Nginx 설정

- A. 다음 명령어를 통해 Certbot container를 생성하고 인증서를 발급

```
sudo docker run -it --rm --name certbot -p 80:80 \
-v "/home/ubuntu/certbot/conf:/etc/letsencrypt" \
-v "/home/ubuntu/certbot/log:/var/log/letsencrypt" \
-v "/home/ubuntu/certbot/www:/var/www/certbot" \
certbot/certbot certonly
```

- B. standalone를 선택하고 이메일, 도메인 이름 등을 차례로 입력
- C. ubuntu환경에서 Vue 프로젝트 폴더로 이동 후 deploy_conf 디렉토리를 생성하고 그 안에서 nginx.conf 파일을 생성(설정 내용은 빌드 상세 내용 확인).

빌드 상세내용

Docker-compose와 Dockerfile 파일을 이용해 빌드합니다.

- Docker-compose.yml

```
version: '3'

services:
  jenkins:
    # 컨테이너내의 서비스
    # 서비스명 : jenkins
    image: jenkins/jenkins:lts
    # 컨테이너 생성 시 jenkins/jenkins:lts 이미지를 기반 이미지로 사용
    container_name: jenkins
    # 컨테이너 이름
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock # /var/run/docker.sock와 컨테이너 내부의 /var/run/docker.sock를 연결
      - /jenkins:/var/jenkins_home
    # /jenkins와 /var/jenkins_home연결
    ports:
      - "9090:8080"
    # 포트 매핑, AWS의 9090 포트와 컨테이너의 8080포트 연결
    privileged: true
    # 컨테이너 시스템의 주요 자원에 연결할 수 있게 설정(기본적으로 false)
    user: root
    # 젠킨스에 접속할 유저 계정(root로 할 경우 관리자)
    db:
    # 서비스명 db
    image: mysql:8.0.29
    # 컨테이너 생성 시 mysql:8.0.29 이미지를 기반 이미지로 사용
    container_name: mysql
    # 컨테이너 명 mysql
    command:
    #아래 4줄 : mysql 설정 부분, 인증 방식, encoding 방식 설정
      - --default-authentication-plugin=mysql_native_password
      - --character-set-server=utf8mb4
      - --collation-server=utf8mb4_unicode_ci
      - --range_optimizer_max_mem_size=16777216
    restart: always
    environment:
      MYSQL_DATABASE: nuribom
    # 사용하는 db
      MYSQL_USER: nuribom
    # root 패스워드
      MYSQL_PASSWORD: ajrqnf1a303
    # 커넥션 접속하는 user
      MYSQL_ROOT_PASSWORD: tkxkrns112
    # user 패스워드
      TZ: Asia/Seoul
    volumes:
      - ./mysql/data:/var/lib/mysql
    # 데이터 마운트 용도, ./mysql/data와 /var/lib/mysql 연결
      - 3306:3306
    # 포트 매핑, AWS의 3306 포트와 컨테이너의 3306 포트 연결
```

- Spring Boot Dockerfile(yogidice/Dockerfile)

```
FROM openjdk:8-jdk-alpine as build-stage
# openjdk:8-jdk-alpine 이미지를 build-stage라고 지정

COPY gradlew .
# gradlew을 이미지에 복사
COPY gradle gradle
# gradle을 이미지에 복사
COPY build.gradle .
# build.gradle을 이미지에 복사
COPY settings.gradle .
# settings.gradle을 이미지에 복사
COPY src src
# Spring 애플리케이션 소스 코드를 이미지에 복사
RUN chmod +x ./gradlew
# gradlew 실행할 수 있는 권한 부여 (없으면 permission denied 발생)
RUN ./gradlew bootJar
# gradlew를 사용해서 실행 가능한 jar 파일 생성

# Deploy Stage
FROM openjdk:8-jdk-alpine
# 새로 생성할 이미지의 기반 이미지 지정 (openjdk:8-jdk-alpine)

WORKDIR /var/jenkins_home/workspace/nuribom/NURI-BOM
#working directory를 /var/jenkins_home/workspace/nuribom/NURI-BOM로 지정
COPY --from=build-stage build/libs/*.jar app.jar
# build-stage 이미지에서 build/libs/*.jar 파일을 app.jar로 복사

EXPOSE 8081
ENTRYPOINT [ "java", "-jar", "/var/jenkins_home/workspace/nuribom/NURI-BOM/app.jar" ]
# 컨테이너 생성 후 최초 실행될 때 명령어 지정
```

- React Dockerfile(nuribom-admin)

```
FROM node:16.15.0 as build-stage
WORKDIR /var/jenkins_home/workspace/nuribom/nuribom-admin
# working directory를 /var/jenkins_home/workspace/nuribom/nuribom-display로 지정
COPY package*.json ./
# package*.json을 이미지에 복사
RUN npm install
# npm install로 필요한 모듈 설치
COPY . .
# 모두 복사
RUN npm run build
# npm run build로 build 파일 생성
FROM nginx:stable-alpine as production-stage
# nginx:stable-alpine 이미지를 production-stage로 지칭
COPY --from=build-stage /var/jenkins_home/workspace/nuribom/nuribom-admin/build /usr/share/nginx/html
#{working directory}/build에 있는 파일을 /usr/share/nginx/html로 복사
COPY --from=build-stage /var/jenkins_home/workspace/nuribom/nuribom-admin/deploy_conf/nginx.conf /etc/nginx/conf.d/default.conf
#{working directory}/deploy_conf/nginx.conf를 /etc/nginx/conf.d/default.conf로 복사
EXPOSE 80
# 80번 포트로 개방
CMD ["nginx", "-g", "daemon off;"]
# 컨테이너 실행되면 nginx 실행, global 디렉티브로 daemon off 옵션 적용
```

- React Dockerfile(nuribom-display)

```
FROM node:16.15.0 as build-stage
WORKDIR /var/jenkins_home/workspace/nuribom/nuribom-display
# working directory를 /var/jenkins_home/workspace/nuribom/nuribom-display로 지정
COPY package*.json ./
# package*.json을 이미지에 복사
RUN npm install
# npm install로 필요한 모듈 설치
COPY . .
# 모두 복사
RUN npm run build
# npm run build로 build 파일 생성
EXPOSE 3000
# 3000번 포트로 개방
CMD ["npm", "start"]
# 컨테이너 생성 후 실행
```

- Nginx 설정

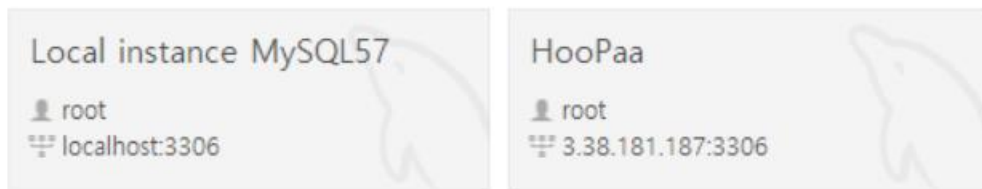
```
1 upstream backend{
2     ip_hash;
3     server 172.18.0.1:8081;
4 }
5
6 upstream worker{
7     ip_hash;
8     server 172.18.0.1:3000;
9 }
10
11 map $http_upgrade $connection_upgrade {
12     default upgrade;
13     '' close;
14 }
15
16 server {
17     listen 80;
18     server_name k7b303.p.ssafy.io;
19     location / {
20         return 301 https://$host$request_uri;
21     }
22 }
23
24 server {
25     listen 443 ssl;
26     listen [::]:443 ssl;
27     server_name k7b303.p.ssafy.io;
28     access_log /var/log/nginx/access.log;
29     error_log /var/log/nginx/error.log;
30     ssl_certificate /etc/letsencrypt/live/k7b303.p.ssafy.io/fullchain.pem;
31     ssl_certificate_key /etc/letsencrypt/live/k7b303.p.ssafy.io/privkey.pem;
32     ssl_protocols TLSv1 TLSv1.1 TLSv1.2 SSLv3;
33     ssl_ciphers ALL;
34
35     location / {
36         root /usr/share/nginx/html;
37         index index.html index.htm;
38         proxy_redirect off;
39         charset utf-8;
40         try_files $uri $uri/ /index.html;
41         proxy_http_version 1.1;
42         proxy_set_header Upgrade $http_upgrade;
43         proxy_set_header Connection "upgrade";
44         proxy_set_header Host $host;
45         proxy_set_header X-Real-IP $remote_addr;
46         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
47         proxy_set_header X-Forwarded-Proto $scheme;
48         proxy_set_header X-Nginx-Proxy true;
49     }
50
51     location /api/ {
52         proxy_pass http://backend;
53         proxy_redirect off;
54         proxy_http_version 1.1;
55         proxy_set_header Upgrade $http_upgrade;
56         proxy_set_header Connection $connection_upgrade;
57         proxy_set_header Host $http_host;
58         proxy_set_header X-Real-IP $remote_addr;
59         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
60         proxy_set_header X-Forwarded-Proto $scheme;
61         proxy_set_header X-Nginx-Proxy true;
62     }
63
64     error_page 500 502 503 504 /50x.html;
65     location = /50x.html {
66         root /usr/share/nginx/html;
67     }
68 }
```

80번 포트(http)로 들어온 요청은 443번 포트(https)로 리다이렉트 시킨다. url 에 /api가 붙어 있는 요청은 Spring Boot서버로 연결한다.

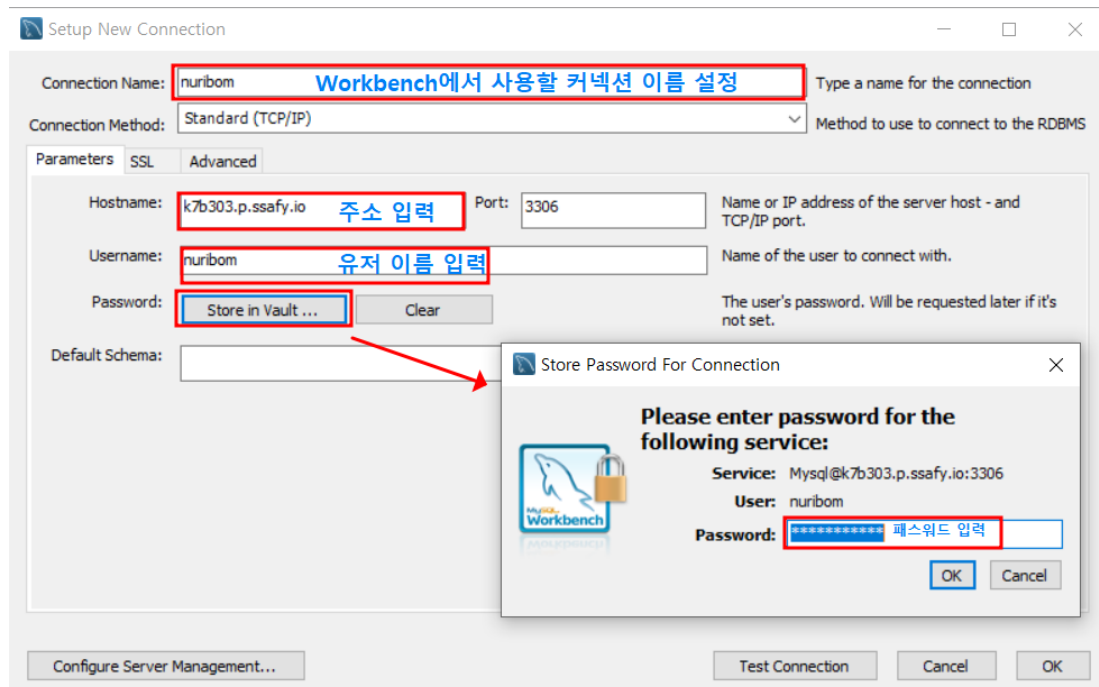
DB

1. MySQL Workbench 추가

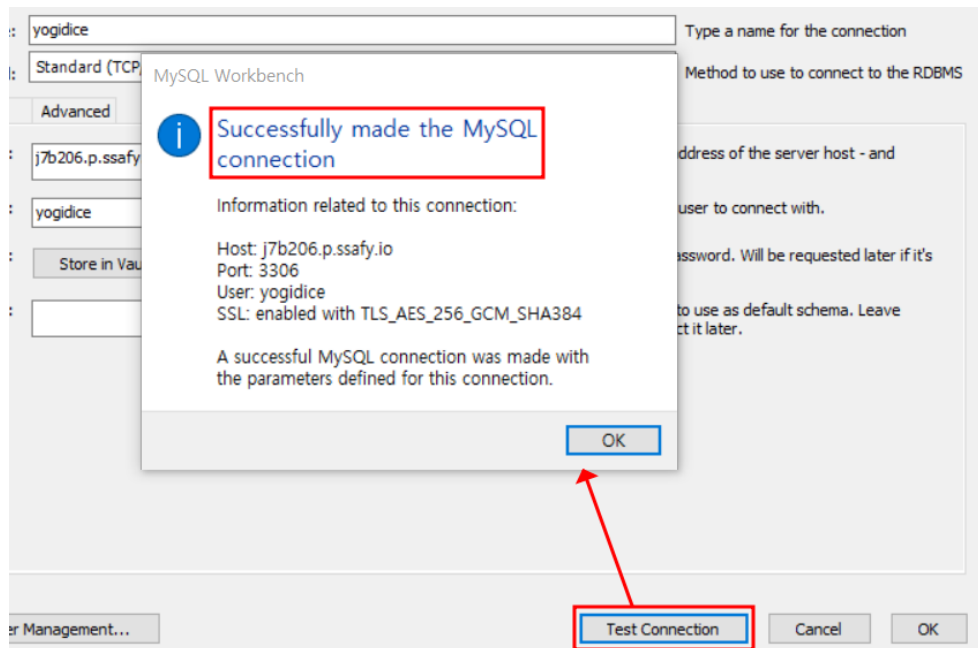
MySQL Connections



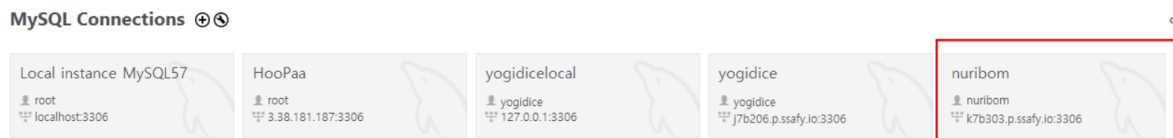
2. EC2 계정 정보 작성



3. 테스트 커넥션 확인



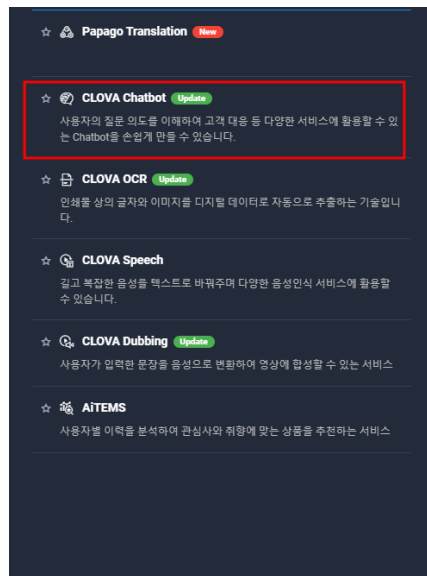
4. 커넥션 추가 확인



외부 서비스

네이버 CLOVA Chatbot

1. 네이버 클라우드 플랫폼의 콘솔에서 AI Service 카테고리의 CLOVA Chatbot 선택



2. 도메인 생성

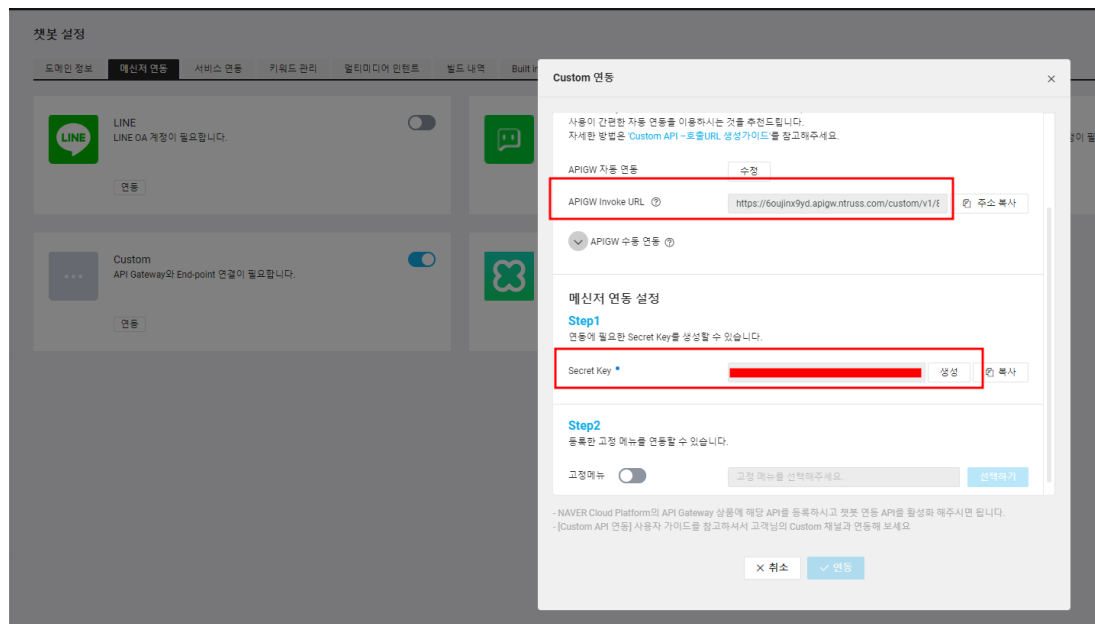
<input type="checkbox"/>	도메인 ID	도메인 이름	도메인 코드	자연어처리
<input type="checkbox"/>	8315	nuribom	nuribom	한국어

3. 대화 생성 및 빌드/배포

The screenshot shows the Naver CLOVA Chatbot console. The '대화 목록' (Conversation List) tab is selected in the top navigation bar. The main area displays a table of conversations with columns for ID, 대화 이름 (Conversation Name), 대화 태그 (Conversation Tag), 대화 위치 (Conversation Location), 질문 (Question), 질문 개수 (Question Count), 답변 (Answer), 답변 유형 (Answer Type), 빌드 상태 (Build Status), and 작성/변경일 (Created/Modified Date).

ID	대화 이름	대화 태그	대화 위치	질문	질문 개수	답변	답변 유형	빌드 상태	작성/변경일 (UTC+09:00)
5084576	배경화면	배경	단말 대화	<?>? [배경] 배경화면(배경) 배경화면 <?>?	10	bg	기본 답변	빌드 완료	2022-11-17 09:18:23 수정
5084574	홈	홈	단말 대화	나갈래(관외) <?>? >?>	14	home	기본 답변	빌드 완료	2022-11-17 09:18:50 수정
5084535	이전	이전	단말 대화	<?>? [이전(전)] [상(사진)]<?>?	7	prev	기본 답변	빌드 완료	2022-11-17 09:19:34 수정
5084531	다음	다음	단말 대화	다음 [것(개)]<?>?	7	next	기본 답변	빌드 완료	2022-11-16 17:52:27 수정
5082709	사진	사진	단말 대화	@(사진)	12	picture	기본 답변	빌드 전(수정됨)	2022-11-18 14:33:33 수정
5082705	영상변지	영상변지	단말 대화	@(영상)	12	video	기본 답변	빌드 전(수정됨)	2022-11-18 14:34:46 수정
5080616	부정	부정	단말 대화	<?>? 싫은데요 <?>?	20	negative	기본 답변	빌드 완료	2022-11-15 21:32:56 수정
5080615	긍정	긍정	단말 대화	그래	21	positive	기본 답변	빌드 완료	2022-11-15 21:29:31 수정

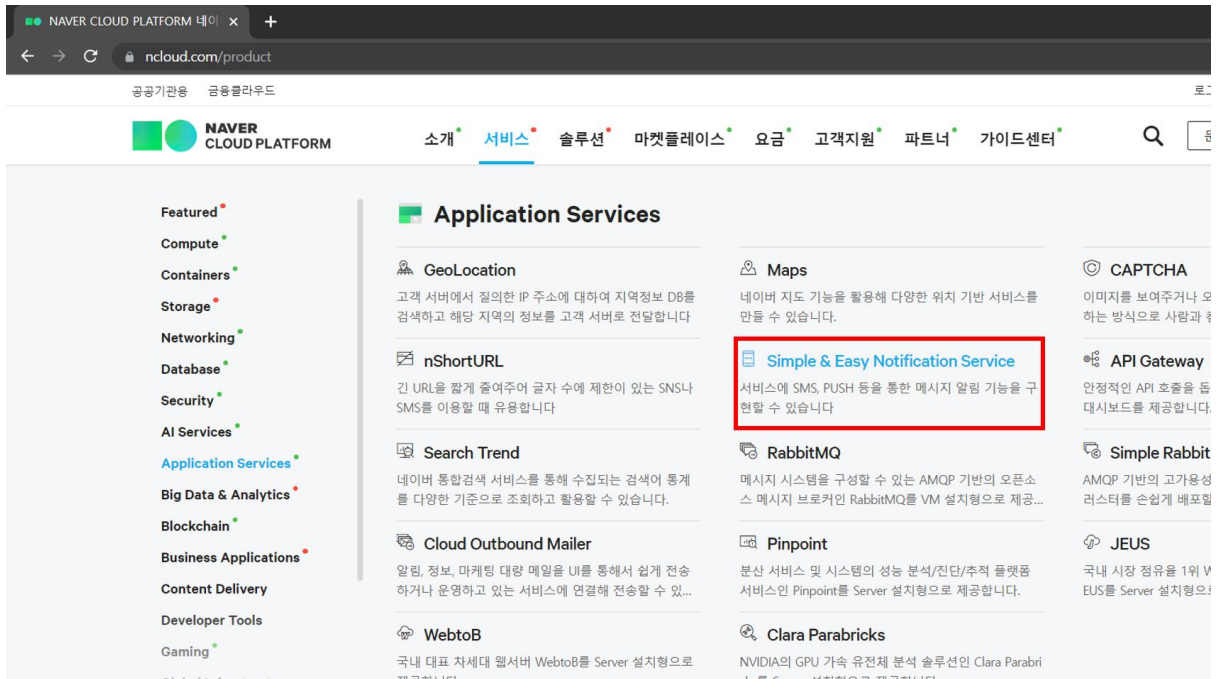
4. API 게이트웨이 url 및 Secret Key 발급



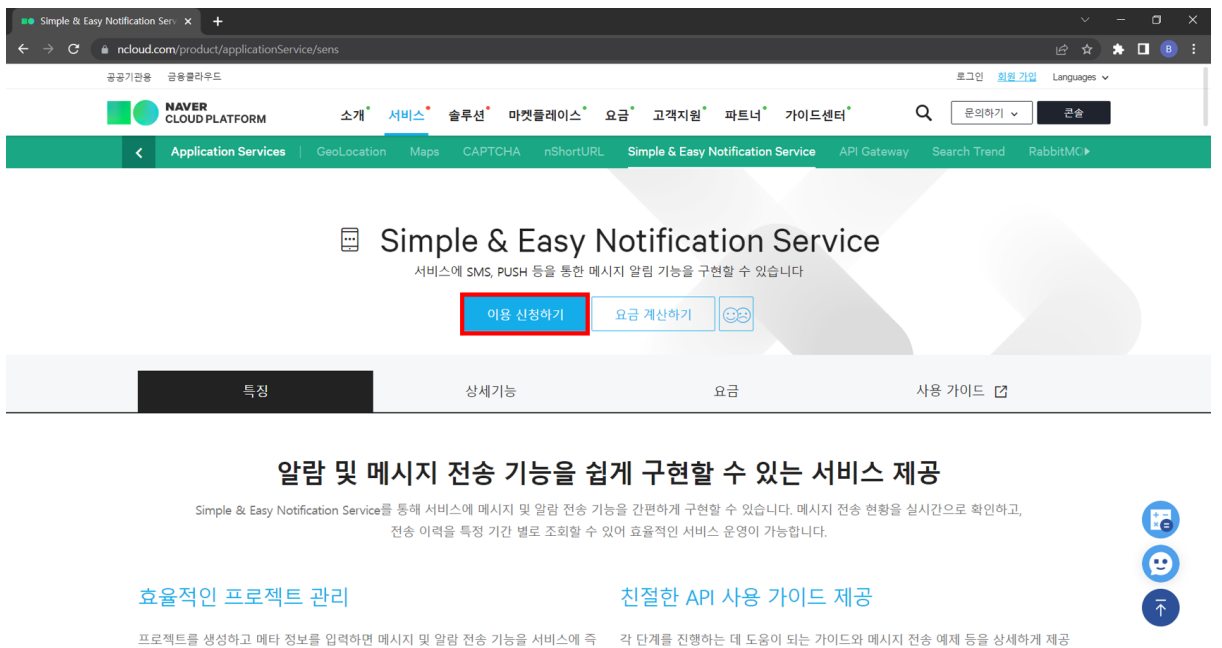
네이버 클라우드 플랫폼 - Simple & Easy Notification Service

낙상 사고 발생 시 등록된 보호자에게 긴급 메시지를 보내기 위해 네이버 클라우드 플랫폼의 Simple & Easy Notification Service를 사용하였습니다.

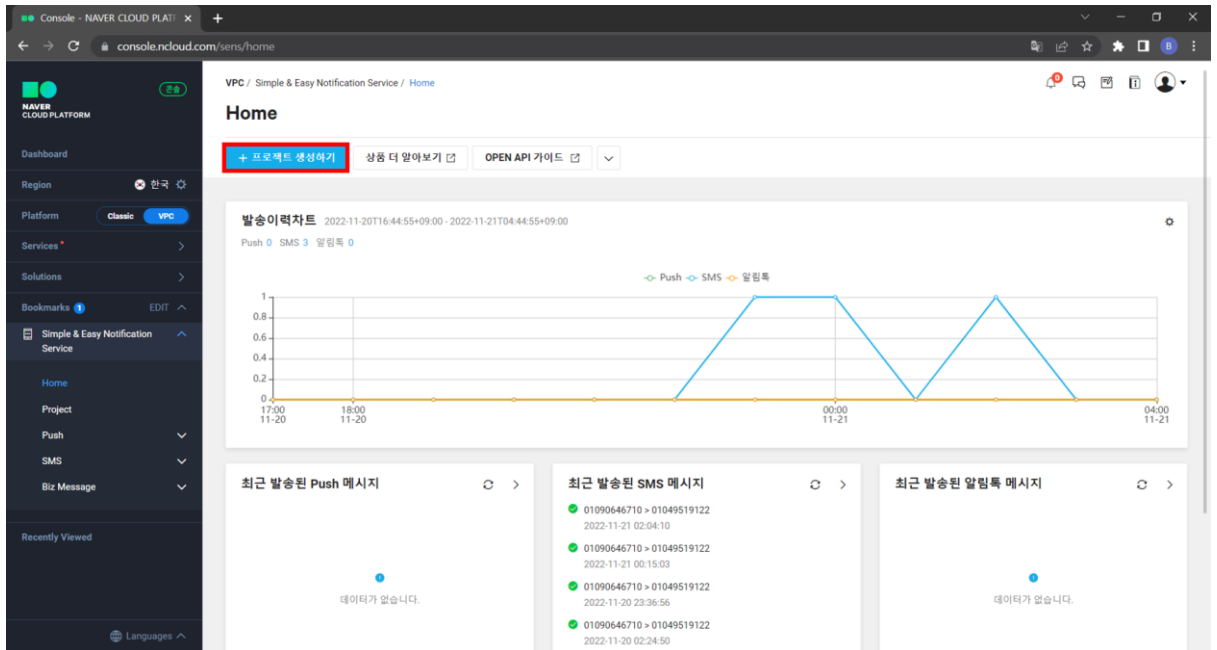
1. 서비스 선택



2. 서비스 이용 신청



3. 프로젝트 생성 (1)



4. 프로젝트 생성 (2)

The screenshot shows the same Naver Cloud Platform console interface as before, but with a '프로젝트 생성' (Create Project) dialog box open in the center. The dialog box has a title bar and a close button. Inside, there is a 'Project 설정' (Project Settings) section with a note: '(필수 입력 사항입니다.)' (Required input items). Below this, there are three checkboxes for '서비스 Type' (Service Type): 'PUSH', 'SMS' (which is checked), and 'Biz Message'. There is also a text input field for '이름' (Name) containing the value 'nuriom'. Below the name field, there is a text input field for '설명' (Description) containing the value '플로 어프리케이션을 위한 알림 서비스'. At the bottom of the dialog box, there are two buttons: 'X 취소' (Cancel) and '✓ 확인하기' (Confirm), with the '확인하기' button highlighted in red. The background of the console is dimmed.

5. 발신번호 등록 (1)

3. 대표 전화번호(15YY, 16YY, 18YY) : 번호 앞에 지역번호 사용 금지
4. 공통서비스식별번호 (ON0개월) : 번호 앞에 지역번호 사용 금지
5. 특수번호(112, 1335 등)는 해당 사용자(국가, 지방자치단체, 공공기관 등)에 한하여 사용 가능
• 스캠신호로 인해 문자 메시지 발송이 차단된 번호는 발신번호로 등록하더라도 발송되지 않습니다.

발신번호 사전등록 방법

발신번호는 회당 최대 10개까지 등록할 수 있습니다. 조과 등록이 필요할 경우, 고객지원을 통해 문의 바랍니다.

서류 인증

- 휴대폰번호 및 유선번호와 통신서비스 이용증명필 제출
 - 본인 소유의 발신번호를 등록할 경우
 - 발급된 지 1개월 이내의 통신서비스 이용증명필
 - 발신번호 대리 등록이 필요할 경우
 - 발신번호의 대리 등록이 가능한 조건(링크)
 - 위 링크를 참고하여 조건에 맞는 서류 제출
 - 내이비 클라우드 플랫폼 관리자가 확인 후 승인(영업일 기준 1~2일)
 - 등록 완료

10건 이상의 발신번호 등록이 필요할 경우, 고객지원을 통해 문의해 주세요.

[서류 인증 \(Doc\) >](#)

핸드폰 인증

- 핸드폰 SMS 본인인증
 - 인증 절차
 - 본인인증 서비스 약관 동의
 - 핸드폰 명의자 정보 및 핸드폰 번호 입력 (인증시 사용하는 번호가 발신번호로 등록됩니다.)
 - 인증번호 수신 및 입력
 - 핸드폰 본인 인증 완료
- 등록 완료
 - 회원 명의의 핸드폰으로만 인증할 수 있습니다

[본인인증 \(SMS\) >](#)

① 통신서비스 이용증명필이한?
전기통신사업자가 이용자 본인에 사용하는 전화번호임을 증명하기 위해 발급하는 서류입니다. 가입자의 성명/주소/생년월일(기업회원의 경우 사업자등록번호), 가입자 통신 서비스 종류(유선, 무선, 인터넷 전화 등) 및 전화번호가 포함되어 있어야 하며, 등록 신청일 기준으로 1개월 이내에 발급된 서류여야 합니다. 자세한 사항은 가입한 통신사 고객센터에 문의하시기 바랍니다.

6. 발신번호 등록 (2)

본인 휴대전화 인증

☒ 아래 약관에 모두 동의합니다.

- ☒ 인증시 개인정보 이용 [보기](#) ☒ 인증시 고유식별정보 처리 [보기](#)
- ☒ 통신사 이용약관 [보기](#) ☒ 인증시 이용약관 [보기](#)
- ☒ 개인정보 수집 [보기](#)

이름

내국인

생일 년(4자) 월 일

SKT 휴대전화번호

인증번호

인증비율은 내이비 클라우드 플랫폼에서 부담합니다.

[확인](#)

공공데이터포털 – 기상청 단기예보 조회서비스

어르신 분들께 현재 날씨 정보를 제공해드리기 위해 기상청의 단기예보 조회 api 를 활용하였습니다.

1. 서비스 활용 신청

The screenshot shows the 'OpenAPI 상세' (OpenAPI Details) page for the '기상청_단기예보 ((구)_동네예보) 조회서비스' (Weather Short-term Forecast ((Formerly)_Daily Forecast) Query Service). The page includes a description of the service, a '활용신청' (Apply) button, and a table of API details.

분류체계	과학기술 - 과학기술연구	제공기관	기상청
관리부서명	기상융합서비스과	관리부서 전화번호	042-481-7502
API 유형	REST	데이터포맷	JSON+XML
활용신청	12818	키워드	단기예보, 초단기실황, 초단기예보

2. Call back URL

3) [단기예보조회] 상세기능명세

a) 상세기능정보

상세기능 번호	3	상세기능 유형	조회 (상세)
상세기능명(국문)	단기예보조회		
상세기능 설명	단기예보 정보를 조회하기 위해 발표일자, 발표시각, 예보지점 X 좌표, 예보지점 Y 좌표의 조회 조건으로 발표일자, 발표시각, 좌표구분 문자, 예보값, 예보일자, 예보시각, 예보지점 X 좌표, 예보지점 Y 좌표의 정보를 조회하는 기능		
Call Back URL	http://apis.data.go.kr/1360000/VilagefcstInfoService_2.0/getVilagefcst		
최대 메시지 사이즈	[48,452] byte		
평균 응답 시간	[600] ms	초당 최대 트래픽선	[30] tps

b) 요청 메시지 명세

항목명(영문)	항목명(국문)	항목크기	항목구분	샘플데이터	항목설명
serviceKey	인증키	100	1	인증키 (URL Encode)	공공데이터포털에서 발급받은 인증키
numOfRows	한 페이지 결과 수	4	1	50	한 페이지 결과 수 Default: 10
pageNo	페이지 번호	4	1	1	페이지 번호 Default: 1
dataType	응답자료형식	4	0	XML	요청자료형식(XML/JSON) Default: XML

base_date	발표일자	8	1	20210628	'21년 6월 28일발표
base_time	발표시각	4	1	0500	05시 발표 * 하단 참고자료 참조
nx	예보지점 X 좌표	2	1	55	예보지점의 X 좌표값 *별첨 엑셀 자료 참조
ny	예보지점 Y 좌표	2	1	127	예보지점의 Y 좌표값 *별첨 엑셀 자료 참조

※ 항목구분 : 필수(1), 옵션(0), 1 건 이상 복수건(1..n), 0 건 또는 복수건(0..n)

c) 응답 메시지 명세

항목명(영문)	항목명(국문)	항목크기	항목구분	샘플데이터	항목설명
numOfRows	한 페이지 결과 수	4	1	50	한 페이지당 표출 데이터 수
pageNo	페이지 번호	4	1	1	페이지 수
totalCount	데이터 총 개수	10	1	1	데이터 총 개수
resultCode	응답메시지 코드	2	1	00	응답 메시지코드
resultMsg	응답메시지 내용	100	1	NORMAL SERVICE	응답 메시지 설명
dataType	데이터 타입	4	1	XML	응답자료형식 (XML/JSON)
baseDate	발표일자	8	1	20210628	'21년 6월 28일 발표
baseTime	발표시각	6	1	0500	05시 발표