

포팅 메뉴얼 E105

개요

본 메뉴얼은 부울경 E105 팀의 공통 프로젝트 “낙낙”의 산출물 접속 방법 배포 환경 및 아키텍처 설계를 제공합니다.

배포 과정

현재 낙낙 프로젝트의 빌드는 gitlab에 올라가 있는 **S09P12E105** 프로젝트의 deploy 브랜치에 push event가 일어나면 등록된 webhook 트리거에 따라 <http://i9e105.p.ssafy.io:9000> 에서 실행중인 젠킨스 빌드 프로젝트 NakNak에 요청이 날라갑니다.

AWS ec2 배포서버의 /var/lib/jenkins/workspace/NakNak\$ 젠킨스 프로젝트 경로에서 코드를 최신화 하고 아래 명령을 자동으로 수행합니다.

빌드와 실행은 dockerfile과 docker-compose.yml 파일로 수행됩니다.

변경점이 없는 컨테이너 mongodb, mysql, redis 는 ec2에서 상시 실행되고,

변경점이 생기는 nginx, react, spring-main, spring-log server는 push 이벤트마다 새로 빌드 됩니다.

```
sudo docker-compose down -v

#!/bin/bash

# 이미지가 존재하는지 확인하고, 존재하면 삭제하는 함수
delete_image_if_exists() {
    local image=$1
    if sudo docker images --format "{{.Repository}}:{{.Tag}}" | grep -q "^$image$"; then
        sudo docker rmi $image
    fi
}

sudo docker rm -f spring
sudo docker rm -f logserver
# 각 이미지에 대해 함수를 호출
delete_image_if_exists "spring-image"
delete_image_if_exists "logserver-image/logserver"
delete_image_if_exists "react-image/react"
delete_image_if_exists "nginx-image/nginx"

if [ -f docker-compose.yml ]; then
    sudo docker-compose up -d
else
    echo "docker-compose.yml not found!"
    exit 1
fi
cd back/fisher-log-server

sudo docker build -t logserver-image/logserver .
```

```

sudo docker run --network=spring-redis -d -p 8081:8081 --name logserver logserver-image/logserver

cd ..

cd Spring/fisher

sudo docker build -t spring-image .
sudo docker run --network=spring-redis -d -p 8080:8080 --name spring spring-image

```

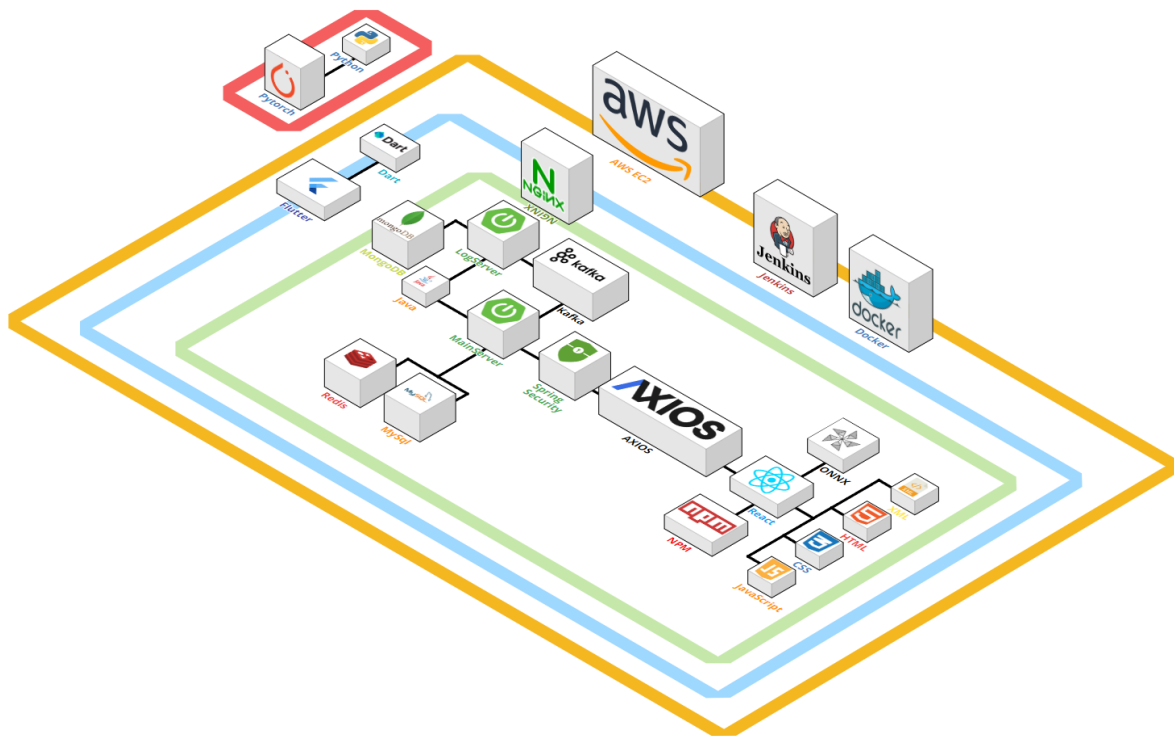
접속 계정

웹페이지(모바일 전용) : <https://i9e105.p.ssafy.io/>

아이디: ssafy@ssafy.com

비밀번호: cjswo1!

Architecture



Frontend

	버전	
IDE	vsCode	
Node.JS	18.16.1 LTS	
Flutter	flutter 3.10.6	

React	18.2.0	
-------	--------	--

- 리액트
- Node.js : 18.16.1 LTS (React를 사용하기 더 쉽게 해주는 도구들을 내장하고 있는 오픈소스이자 JavaScript runtime environment. NPM을 통해서 리액트 개발에 필요한 다양한 모듈들을 다운받아 사용. node.js가 있으면 jsx 사용 가능.(html코드를 사용할 수 있다))
 - 리코일 : 상태관리 도구 : recoil
- axios : 브라우저, Node.js를 위한 Promise API를 활용하는 HTTP 비동기 통신 라이브러리, 백과 프론트 간의 통신을 쉽게 하기 위해서 사용
- 플러터 : 모바일/웹/데스크톱 크로스 플랫폼 GUI SDK이다. 하나의 코드 베이스로 안드로이드, 리눅스, 윈도우, 맥, ios 및 웹 브라우저에서 모두 동작. 사용되는 언어는 dart를 사용한다. 리액트로 구성된 코드를 통해 개발된 서비스를 웹뷰의 형태로 앱으로 제공하기 위해서 사용

사용 외부 api

1. 기상청 api

날씨 하러가기 좋은 날씨를 구분하기 위한 예보정보를 불러옵니다.

<https://www.data.go.kr/data/15084084/openapi.do>

코드	분류	단위	압축bit수
T1H	기온	C(섭씨)	10
RN1	1시간 강수량	1mm	8
SKY	하늘상태	코드값	4
UUU	동서바람성분	m/s	12
VVV	남북바람성분	m/s	12
REH	습도	%	8
PTY	강수형태	코드값	4
VEC	풍향	deg	10
WSD	풍속	m/s	10

2. 카카오맵 api

낙시터 추천 서비스를 위한 지도api입니다.

<https://apis.map.kakao.com/web/guide/>



3. AI 음성 생성기

새로운 가입자 뉴비 가이드 소개 기능에서 고양이 안내문을 읽어 주기 위해 사용하는 tts api입니다.

<https://azure.microsoft.com/ko-kr/products/cognitive-services/text-to-speech>

Backend

프레임워크명	버전	사용 포트
MySql	mysql 8.0.33	3036:3036
MongoDB	mongo 6.0	27017:27017
Redis	redis 6.2.13-alpine	6379:6379
kafka	kafka 3.5.0	9092
nginx	nginx:latest	443:443 (https) 80:80 (http)
SpringBoot	3.1.1	8080:8080(backend server) 8081:8081(log server)
IDE	IntelliJ 2023 professional	

- Spring Boot Main Server (백엔드와 프론트엔드 간의 요청 응답처리, 메인 기능과 인증 인가를 담당하는 메인서버)
- Spring Boot Logging Server (로깅 처리를 위한 백엔드 비동기 처리 서버, 카프카 메세지 수신 하여 비동기로 로깅, 수평적 확장 가능)
- ZooKeeper (분산 애플리케이션을 위한 코디네이션 시스템 -> 카프카 분산 메세지 큐의 정보 관리)
- Kafka (쌓이는 로그 기록 요청 등을 분산하여 관리 혹은 비동기 처리로 메인 서버의 부담 줄임)
- MongoDB (로그 기록을 위한 NoSQL 데이터베이스 시스템)
- MySQL (일반적인 메인 서버의 관리를 위한 RDBMS)
- 레디스 (리프레시 토큰 저장을 위한 NoSQL 인메모리 DBMS)

DevOps

프레임워크명	버전	사용 포트
docker	24.0.5	
docker-compose	2.20.2	
nginx	nginx:latest	443:443 (https) 80:80 (http)
Jenkins	Jenkins/Jenkins 2.419	9000

- 젠킨스

자동 빌드, 자동 배포를 수행하기 위해 사용하는 파이프라인 배포 툴입니다. gitlab과 연동하여 gitlab의 특정 브랜치에 push event가 일어난 aws 배포서버에 최신 코드로 새로 빌드&실행을 수행하여 개발자가 배포에 별 다른 노력을 하지 않도록 인프라를 구축하는데 사용합니다.

- NGINX

동시접속 처리를 원활하게 수행하기 위한 웹 서버 프로그램입니다. 현재 nginx를 이용해 1024개의 동시 접속을 원활하게 수행하도록 설정하였습니다.

또한 https로 들어오는 통신의 SSL 인증을 수행하고 http로 들어오는 요청은 자동으로 https로 변환해 보안 사고를 예방하도록 유도하고 있습니다.

443 https 기본 포트로 들어오는 요청을 url 별로 프록시하여 요청이 가야할 서버(spring, react, mysql 등)로 로드밸런싱을 수행하고 있습니다.

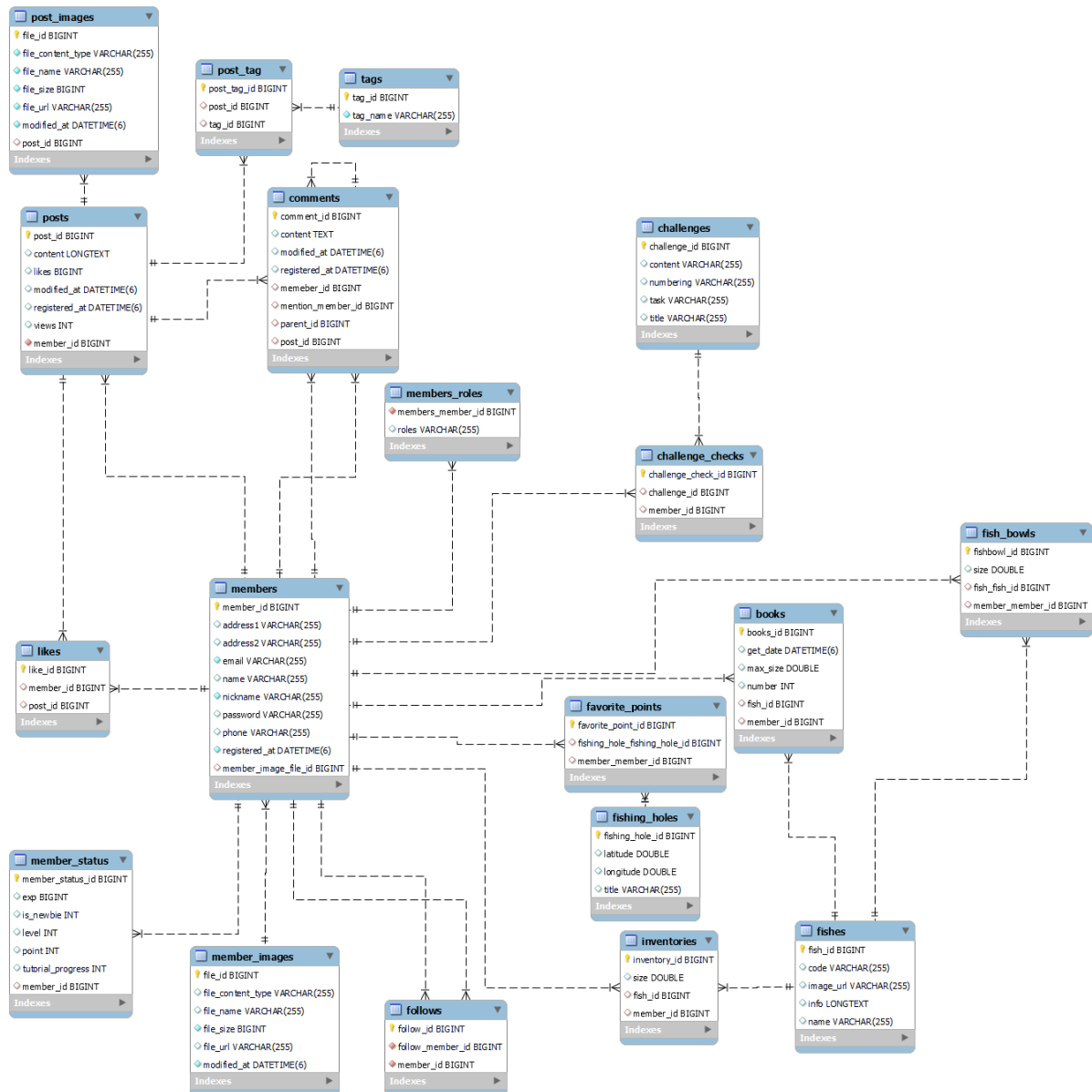
- AWS

무중단 배포 서버를 운용하기 위한 클라우드 컴퓨팅 플랫폼입니다. ssafy에서 제공하는 ec2 우분투 20.04 버전을 활용했습니다.

- DOCKER

프로세스를 컨테이너라는 독립적인 가상환경에서 실행시키기 위한 프레임워크입니다. 현재 낱낱 프로젝트의 개발 과정은 각자의 노트북에서 윈도우 로컬환경으로 디버깅을 수행하고, 개발 서버에서 결과를 테스트하고 ec2 서버에서 최종 배포를 수행하는 과정에 있습니다. 여러 환경에서 동작하는 애플리케이션의 통일성을 위해 docker image로 서버 애플리케이션들을 패키징하여 컨테이너별로 프로세스들을 독립적으로 실행합니다.

DB 명세서



DB 스키마명 : NakNak


시연 시나리오

동작 기능 설명

| 로그인



| 회원가입




이메일
cwoh9002@hanmail.net

비밀번호

이름
오철원

닉네임
sdfefwefwef



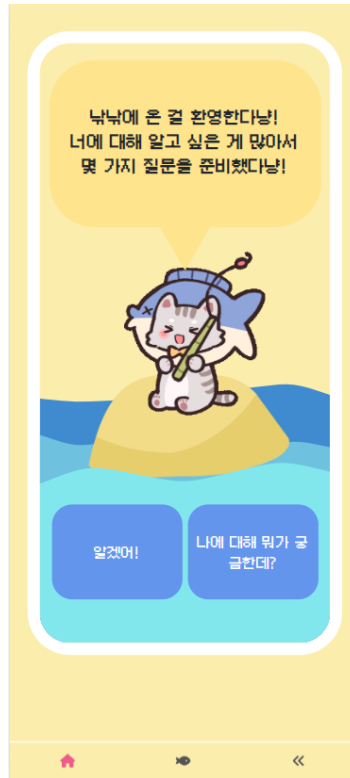
파일선택

프로필 이미지가 없으면 낭낭이 사진으로 대체됩니다.

회원가입

최초 가입자를 대상으로 하는 뉴비 가이드를 확인하기 위해 회원가입 진행

| 뉴비 가이드 진행

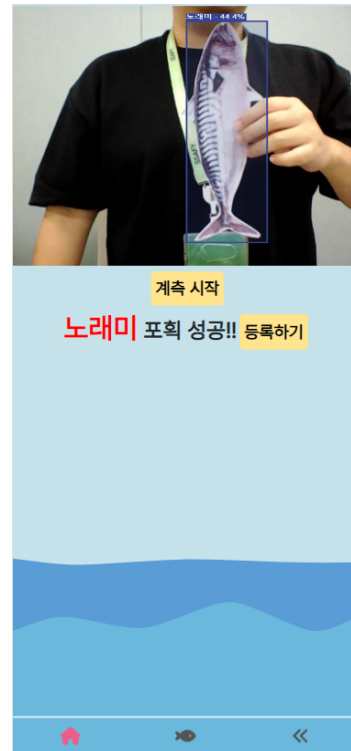


안내에 따라 선택지를 누르면 어플과 낙시 사용법에 대해 알려줍니다.

메인화면

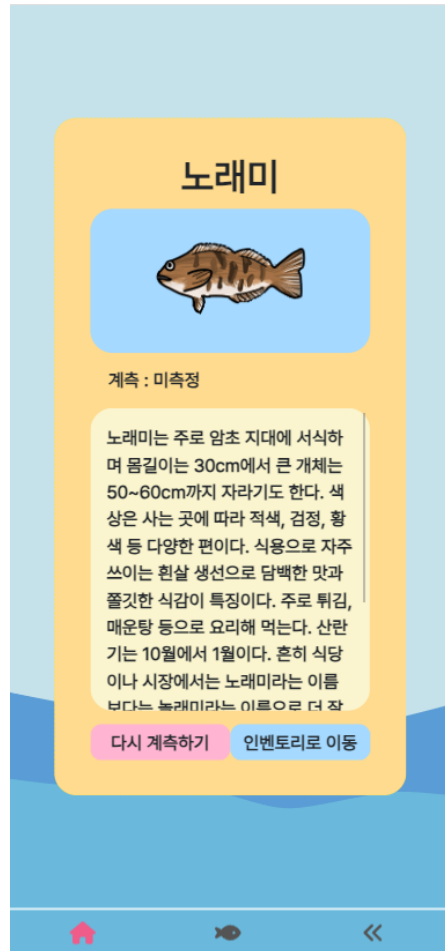


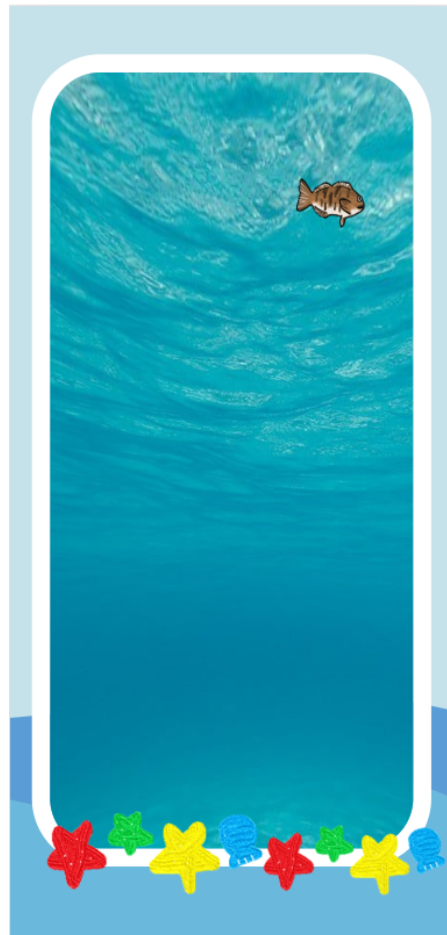
| 포획 어종 확인 카메라 기능



메인화면에서 카메라를 클릭하면 사진 촬영이 가능합니다 (위 사진은 PC 기준) 카메라는 자동으로 어종을 포착하고 물고기 어종과 길이를 분석해 줍니다.

| 포획 어종 등록



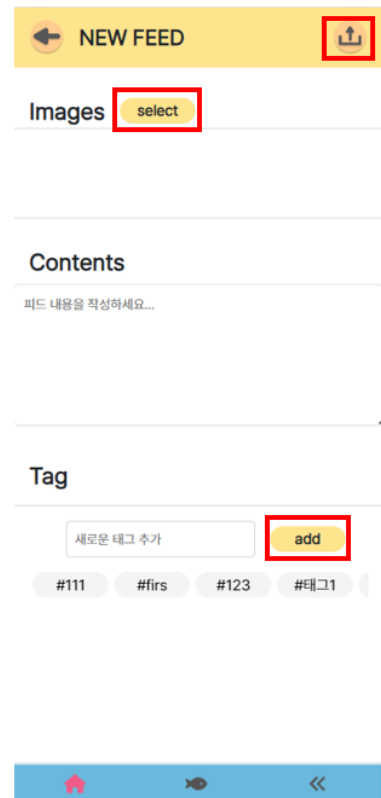
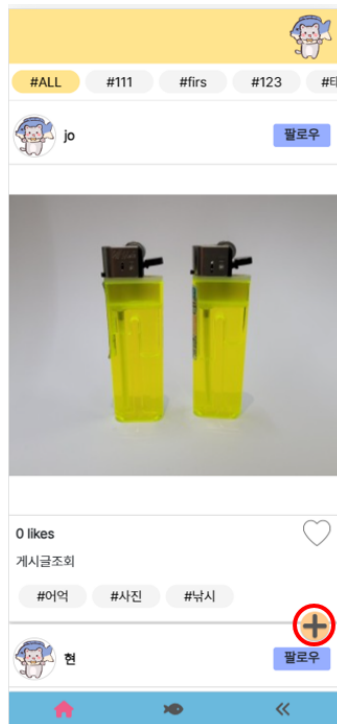


포획한 어종을 등록해 도감 등록과 수족관 채우기 기능을 수행할 수 있습니다.

카메라의 등록하기 기능 또는 메인화면에서 인벤토리로 들어가면 촬영한 물고기가 존재합니다.

인벤토리의 물고기를 클릭하고 어항으로 보내기를 누르면 수족관을 해엄치는 물고기들을 보실 수 있습니다.

게시물 확인 & 작성



SNS기능을 이용해 다른 사람이 잡은 물고기나 게시글들을 확인할 수 있고 사용자가 자신의 피드를 작성할 수도 있습니다.