|  |
| --- |
|  |

|  |
| --- |
|  |
| 광주 2반 10조 |
| 텐텐텐 |
|  |
|  |
|  |
|  |

목차

[I. 개요 2](#_Toc96072459)

[1. 프로젝트 개요 2](#_Toc96072460)

[2. 프로젝트 사용 도구 2](#_Toc96072461)

[3. 개발환경 2](#_Toc96072462)

[4. 외부 서비스 2](#_Toc96072463)

[5. Gitgnore 처리한 핵심 키들 3](#_Toc96072464)

[II. 빌드 3](#_Toc96072465)

[1. 환경변수 형태 3](#_Toc96072466)

[2. 빌드하기 5](#_Toc96072467)

[3. 배포하기 5](#_Toc96072468)

[4. 서비스 이용 방법 7](#_Toc96072469)

[가) Firebase 실시간 알림 8](#_Toc96072471)

[나) Google Cloud Platform 8](#_Toc96072472)

# 개요

## 프로젝트 개요

특히 첫 독립을 하는 사회 초년생 1인 가구의 경우, 새로운 환경과 책임으로 인해 심리적으로 여유롭지 못하고 긴장하기 쉬워 바쁜 일상을 효율적으로 계획하기 어렵다.

이런 어려움을 극복하고자 현대 사회에서는 효율적인 시간 관리를 위해 자기계발 플래너 앱을 많이 사용한다.

이 서비스는 일반적인 플래너 앱들과는 달리, 사용자의 부모님 목소리를 샘플링하여 음성 알림으로 제공함으로써 사용자들은 익숙하고 편안한 목소리를 통해 일상 생활을 더욱 즐겁고 따뜻하게 경험할 수 있다

## 프로젝트 사용 도구

이슈 관리 : JIRA

형상 관리 : Gitlab

커뮤니케이션 : Notion, Mattermost

디자인 : Figma

UCC : 어도비 프리미어

## 개발환경

VS Code : 1.85.1

IntelliJ : 17.0.9+7-b1087.9 amd64

JVM : 17.0.9

Node.js : 18.15.0

SERVER : AWS EC2 Ubuntu 20.04.6 LTS

DB : MySql

## 외부 서비스

Firebase Realtime DB (front) :

## Gitgnore 처리한 핵심 키들

Vue : .env.loval (최상단 위치)

Spring : application.yml, serviceAccountKey.json

(\src\main\resources, 또는 classPath에 위치)

Flask : .env (최상단 위치)

# 빌드

## 환경변수 형태

/frontend/vite.config.js

manifest: {

"name": "todak",

"short\_name": "todak",

"icons": [

{

"src": "/logo/KakaoTalk\_20240208\_220849622.jpg",

"sizes": "192x192",

"type": "image/png",

"purpose": "any"

},

{

"src": "/logo/KakaoTalk\_20240208\_220849622.jpg",

"sizes": "512x512",

"type": "image/png",

"purpose": "maskable"

}

],

"start\_url": "/",

"display": "standalone",

"background\_color": "#ffffff",

"theme\_color": "#ffffff"

}

/frontend/package.json

{

"name": "frontend",

"version": "0.0.0",

"private": true,

"scripts": {

"build": "vite build",

"dev": "vite",

"serve": "vite serve",

"web-push": "web-push",

"preview": "vite preview"

},

"dependencies": {

"@popperjs/core": "^2.11.8",

"@stomp/stompjs": "^7.0.0",

"axios": "^1.6.7",

"bootstrap": "^5.3.2",

"bootstrap-vue": "^2.23.1",

"chart.js": "^4.4.1",

"firebase": "^10.8.0",

"firebase-tools": "^13.1.0",

"jwt-decode": "^4.0.0",

"moment": "^2.30.1",

"papaparse": "^5.4.1",

"pinia": "^2.1.7",

"pinia-plugin-persist": "^1.0.0",

"pinia-plugin-persistedstate": "^3.2.1",

"register-service-worker": "^1.7.2",

"socket.io-client": "^4.7.4",

"v-calendar": "^3.1.2",

"vite-plugin-pwa": "^0.17.5",

"vue": "^3.4.19",

"vue-chartjs": "^5.3.0",

"vue-chartkick": "^1.1.0",

"vue-color": "^2.8.1",

"vue-jwt-decode": "^0.1.0",

"vue-router": "^4.2.5",

"vue-socket": "^0.2.3",

"vue-webrtc": "^3.0.1",

"vue3-cookies": "^1.0.6",

"vuefire": "^3.1.22",

"vuex": "^4.1.0",

"web-audio-recorder": "^1.0.5",

"web-audio-recorder-js": "^0.0.2",

"web-push": "^3.6.7",

"webrtc-adapter": "^8.2.3",

"ws": "^8.16.0"

},

"devDependencies": {

"@vitejs/plugin-vue": "^4.5.2",

"@vue/cli-plugin-babel": "^5.0.8",

"@vue/cli-plugin-eslint": "^5.0.8",

"@vue/cli-service": "^5.0.8",

"@vue/compiler-sfc": "^3.0.0-0",

"babel-eslint": "^10.1.0",

"eslint": "^7.5.0",

"eslint-plugin-vue": "^7.0.0-0",

"vite": "^5.0.10"

},

"eslintConfig": {

"root": true,

"env": {

"node": true

},

"extends": [

"plugin:vue/vue3-essential",

"eslint:recommended"

],

"parserOptions": {

"parser": "babel-eslint"

},

"rules": {}

},

"browserslist": [

"> 1%",

"last 2 versions",

"not dead"

],

"keywords": [],

"type": "module"

}

.env

DB\_USER=DB호스트 유저 정보

DB\_PASS=DB 비밀번호

DB\_URL= DB 주소

.application.properties

# REDIS

redis:

spring.data.redis.host=i10c210.p.ssafy.io  
spring.data.redis.port=6379

# MySql DB(배포)

datasource:

url: jdbc:mysql://i10c210.p.ssafy.io:3306/todak?useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Seoul&zeroDateTimeBehavior=convertToNull&rewriteBatchedStatements=true

driver-class-name: com.mysql.cj.jdbc.Driver

username: ssafy

password: ssafy

# JWT 시크릿 키

jwt:

secret: dyAeHubOOc8KaOfYB6XEQoEj1QzRlVgtjNL8PYs1A1tymZvvqkcEU7L1imkKHeDa

## 빌드하기

1) Front

npm install

npm run build

cd functions

npm install firebase-functions

cd ..

firebase deploy

2) Back-spring

Gradle 실행

Bootjar 실행

3) Back-flask

pip install -r requirements.txt

서버 구동 위치에서 python app.py

## 배포하기

Nginx 설정

upstream jenkins {

server 127.0.0.1:4000;

keepalive 32;

}

map $http\_upgrade $connection\_upgrade {

default upgrade;

'' close;

}

server {

listen 80;

listen [::]:80;

server\_name i10c210.p.ssafy.io;

location / {

return 301 https://i10c210.p.ssafy.io;

}

}

server {

listen 443 ssl;

listen [::]:443 ssl;

server\_name tupli.kr;

ssl\_certificate /etc/letsencrypt/live/i10c210.p.ssafy.io/fullchain.pem; # managed by Certbot

ssl\_certificate\_key /etc/letsencrypt/live/i10c210.p.ssafy.io/privkey.pem; # managed by Certbot

include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot

ssl\_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

root /var/www/html/dist;

index index.html;

location / {

try\_files $uri $uri/ /index.html;

}

location /ws-stomp {

proxy\_pass http://localhost:8080/ws-stomp;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "Upgrade";

proxy\_set\_header Host $host;

proxy\_set\_header Origin "";

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-Host $host;

proxy\_set\_header X-Forwarded-Server $host;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

}

location /flask {

proxy\_pass http://localhost:5000;

proxy\_redirect off;

charset utf-8;

rewrite /flask/(.\*) /$1 break;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-NginX-Proxy true;

}

}

이후 sudo service nginx start

## 서비스 이용 방법

### Firebase 실시간 알림

1. Firebase 콘솔 프로젝트 만들기 (GCP에 프로젝트 있을 경우 그거 선택)

2. 설정 → 프로젝트 설정 → 서비스 계정 → 새 비공개 키 생성

3. realtimedatabase 생성 후 주소 확인

### Google Cloud Platform

1. Google Cloud Platform 접속 후 새 프로젝트 만들기

→ 좌측 햄버거 메뉴에서 저장소 Cloud Storage 클릭

2. 브라우저로 이동될텐데 버킷 만들기 선택 후 버킷 이름 지정하고 만들기

일단은 tupli-profile로 생성 (이 버킷 이름은 나중에 spring에서 저장 위치로 사용됨)

3. springboot 접속용 access key를 만들어야 함.

좌측 햄버거 메뉴 다시 열고 IAM 및 관리자 > 서비스 계정 으로 이동후

서비스 계정 만들기 클릭

4. 서비스 계정이름을 만들고 (tupli-profile-access),

권한 부여에 저장소 관리자, 저장소 개체 관리자 부여

5. 서비스 계정 만들고 나올 경우, 속성 등으로 키로 이동

키 추가 선택하여 json 다운로드