Contents

☒ DEFENDER PROJECT 인수인계 문서

작성일: 2025년 5월26일 **프로젝트:** 불량고객 관리 시스템

☒ 1. 프로젝트 개요

1-1. 프로젝트 구성

- **백엔드(웹서버):** Django REST Framework 기반 API 서버
- **프론트엔드(모바일앱):** Android Kotlin 기반 네이티브 앱
- **데이터베이스:** PostgreSQL
- **서버 환경:** AWS EC2 (Ubuntu)

1-2. 주요 기능

- 사용자 관리 (로그인/회원가입)
- 고객 정보 관리 (불량고객 체크 시스템)
- 공지사항 관리
- 매장 관리

1-3. 프로젝트 구조

```
defender_project/
   defender/                  # Django
       views/                # API
       models.py             #
       migrations/           # DB
       middleware.py         #
   defender_project/         # Django
       settings.py           #
       urls.py              # URL
       defender.conf        # Nginx
   defender_app/            # Android
   defender_env/            # Python
```

☒ 2. 서버 및 네트워크 정보

2-1. 운영 서버 정보

구분	정보	비고
운영 서버 IP	`3.38.245.204`	AWS EC2 인스턴스
웹 접속 주소	`http://3.38.245.204`	포트 80 (nginx)
Django 서버	`http://127.0.0.1:8000`	내부 포트
SSH 접속	`ssh ec2-user@3.38.245.204`	AWS 기본 계정

2-2. 허용된 호스트 목록

```
ALLOWED_HOSTS = [
    'localhost',
    '127.0.0.1',
```

```
    '3.38.245.204',
    '192.168.0.4',
    '10.30.2.91'
]
```

2-3. 개발 환경 주소

- **로컬 개발:** `http://127.0.0.1:8000`
- **내부 네트워크:** `http://192.168.0.4:8000`

☒ 3. 중요 계정 및 비밀번호 정보

3-1. Django 설정 (실제 정보)

☒ 매우 중요한 보안 정보입니다!

```
# SECRET KEY (Django      )
SECRET_KEY = 'django-insecure-*(iprgwcz2774y51v2=(pojqagou*nl%mh4i1+_r7n!e#c2a1*'

#
ENCRYPTION_KEY = b'fJIz6AZnv9LBoAlDGt0WhcZ9q11WOP6fwOjJhJwo600='
```

3-2. 데이터베이스 정보 (실제 정보)

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'defender',
        'USER': 'postgres',
        'PASSWORD': '111222',
        'HOST': '3.38.245.204',
        'PORT': '5432',
    }
}
```

데이터베이스 직접 접속:

```
psql -h 3.38.245.204 -U postgres -d defender
#    : 111222
```

3-3. 앱 인증 토큰(******)

```
// Android       Bearer
bearerToken: String = "defender0651"
```

3-4. Django 관리자 계정(********)

- **접속 주소:** `http://3.38.245.204/admin`

- **계정 생성 방법:**

  ```
  python manage.py createsuperuser
  ```

- ☒ **실제 관리자 계정은 별도 전달 필요**

☒ 4. 앱(Android) 상세 정보

4-1. 앱 기본 정보

```
applicationId = "com.daiso.depender"
versionName = "1.0.2-20250507"
minSdk = 28
targetSdk = 34
namespace = "com.daiso.depender"
```

4-2. 서버 연결 설정

```
//
private var host: String = "http://3.38.245.204:80/"

//      (    )
// private var host: String = "http://localhost:8000/"
// private var host: String = "http://192.168.0.4:8000/"
```

4-3. 주요 API 엔드포인트

기능	메소드	엔드포인트	설명
로그인	POST	/api/users/login/	사용자 인증
고객 목록	POST	/api/clients/list/	고객 리스트 조회
불량고객 체크	POST	/api/clients/check-bad-client/	전화번호로 불량고객 확인
고객 등록	POST	/api/clients/save/	새 고객 정보 등록
고객 수정	PUT	/api/clients/update/	고객 정보 수정
고객 상세	POST	/api/clients/get-client/	특정 고객 상세 정보
공지사항	POST	/api/announce/list/	공지사항 목록

4-4. Android 개발 환경

```
# Android SDK
SDK_PATH = "/Users/mac_kyh/Library/Android/sdk"

#
./gradlew assembleDebug    #
./gradlew assembleRelease  #
```

☒ 5. 데이터베이스 구조

5-1. 주요 테이블

users 테이블

- id (VARCHAR): ID (Primary Key)
- pw (VARCHAR):
- isAdmin (BOOLEAN):
- email (VARCHAR):
- store_name (VARCHAR):
- phone_number (VARCHAR):
- search_count (INTEGER):
- usage_start_date (DATE):
- usage_end_date (DATE):
- createDate (DATETIME):

clients2 테이블

- id (INTEGER): ID (Primary Key)
- nickName (VARCHAR): (Unique)
- name (VARCHAR):
- phoneNumber (TEXT):
- gender (VARCHAR): (male/female)
- ages (VARCHAR):
- isBadClient (BOOLEAN):
- extra (TEXT):
- registeredBy (VARCHAR): ID
- store_name (VARCHAR):
- createDate (DATETIME):

stores 테이블

- id (INTEGER): ID (Primary Key)
- address (VARCHAR):
- ownerId (VARCHAR): ID (Foreign Key)
- province (VARCHAR):
- storeName (VARCHAR):
- isDefenderActive (BOOLEAN):
- createDate (DATETIME):

announcement 테이블

- id (INTEGER): ID (Primary Key)
- title (VARCHAR):
- userId (VARCHAR): ID

- memo (TEXT):
- createDate (DATETIME):

☒ 6. 개발 환경 설정

6-1. 백엔드 (Django) 설정

```
# 1.
cd /Users/mac_kyh/WebProjects/defender_project

# 2.
source defender_env/bin/activate

# 3.
pip install django==4.2.19
pip install djangorestframework
pip install psycopg2-binary
pip install cryptography

# 4.
python manage.py migrate

# 5.
python manage.py runserver

# 6.        (  1 )
python manage.py createsuperuser
```

6-2. 자주 사용하는 Django 명령어

```
#
python manage.py makemigrations

#
python manage.py migrate

#
python manage.py createsuperuser

#       (    )
python manage.py runserver --verbosity=2

#       ( )
python manage.py collectstatic
```

☒ 7. 서버 배포 정보

7-1. Nginx 설정 (defender.conf)

```
server {
    listen 80;
    server_name 3.38.245.204;

    location = /favicon.ico {
        access_log off;
        log_not_found off;
    }

    location /static/ {
        root /home/ec2-user/projects/defender_project;
    }

    location /media/ {
        root /home/ec2-user/projects/defender_project;
    }

    location / {
        proxy_pass http://127.0.0.1:8000;
        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;
    }
}
```

7-2. 서버 경로 정보

```
#
/home/ec2-user/projects/defender_project

#
/home/ec2-user/projects/defender_project/static/

#
/home/ec2-user/projects/defender_project/media/

# Nginx
/etc/nginx/sites-available/defender.conf
/etc/nginx/sites-enabled/defender.conf
```

7-3. 서버 관리 명령어

```
# Nginx
sudo systemctl restart nginx
```

```bash
# Nginx
sudo systemctl status nginx

# Django
nohup python manage.py runserver 127.0.0.1:8000 &

#
ps aux | grep python
```

☒ 9. 보안 주의사항

9-1. 절대 공개하면 안 되는 정보

☒ 다음 정보들은 절대 GitHub 등 공개 저장소에 올리면 안 됩니다:

- Django SECRET_KEY: `django-insecure-*(iprgwcz2774y51v2=(pojqagou*nl%mh4i1+_r7n!e#c2a1*`
- 데이터베이스 비밀번호: `111222`
- 암호화 키: `fJIz6AZnv9LBoAlDGt0WhcZ9q11WOP6fwOjJhJwo600=`
- Bearer 토큰: `defender0651`

9-2. 권장 보안 조치

1. **즉시 변경해야 할 항목:**
   - 데이터베이스 비밀번호
   - Django SECRET_KEY
   - Bearer 토큰
   - 관리자 계정 비밀번호
2. **환경변수 사용:**

   ```bash
   # .env
   import os
   from dotenv import load_dotenv

   load_dotenv()
   SECRET_KEY = os.getenv('SECRET_KEY')
   DB_PASSWORD = os.getenv('DB_PASSWORD')
   ```
3. **추가 보안 설정:**
   - HTTPS 적용 (현재 HTTP만 사용)
   - 방화벽 설정 (필요한 포트만 개방)
   - 정기적인 보안 업데이트

☒ 10. 문제 해결 가이드

10-1. 자주 발생하는 문제

서버 연결 문제

```
#  :
#  :
sudo netstat -tlnp | grep :8000
sudo systemctl status nginx
```

데이터베이스 연결 오류

```
#  : DB
#  : PostgreSQL
sudo systemctl status postgresql
sudo systemctl restart postgresql
```

앱 로그인 실패

```
#  :
#  : API
curl -X POST http://3.38.245.204/api/users/login/ \
  -H "Authorization: Bearer defender0651" \
  -H "Content-Type: application/json" \
  -d '{"id":"testuser","pw":"testpass"}'
```

10-2. 로그 확인 방법

```
# Django
python manage.py runserver --verbosity=2

# Nginx
sudo tail -f /var/log/nginx/error.log

# Nginx
sudo tail -f /var/log/nginx/access.log

#
sudo journalctl -u nginx -f
```

10-3. 디버깅 팁

```
# Django
import logging
logging.basicConfig(level=logging.DEBUG)

# API
from rest_framework.response import Response
return Response({"debug": "test"}, status=200)
```

☒ 12. 연락처 및 추가 자료

12-1. 참고 문서

- Django 공식 문서 (한글)
- Django REST Framework
- Android 개발 가이드
- PostgreSQL 문서

12-2. 유용한 도구(********)

- **API 테스트:** Postman, curl
- **데이터베이스 관리:** pgAdmin, DBeaver
- **서버 모니터링:** htop, netstat
- **로그 분석:** tail, grep, journalctl

☒ 최종 주의사항

☒ 이 문서에 포함된 모든 비밀번호, 키, 토큰은 실제 운영 정보입니다.

인수인계 완료 후 반드시 다음 작업을 수행하세요:

1. **모든 비밀번호 즉시 변경**
2. **이 문서를 안전한 곳에 보관**
3. **불필요한 복사본 삭제**
4. **새로운 보안 정책 수립**

문의사항이 있으시면 언제든 연락 주세요!

문서 작성일: 2025년 3월 22일
최종 수정일: 2025년 3월 22일
버전: 1.0