Dependability Document 2.0 - 1조

Availability

- 프로그램 실행을 위한 manual 제공
  - **Readme.txt**

```
Windows Powershell

$env:FLASK_APP='swsec_team1'
$env:FLASK_ENV='development'
flask run

Mac, LINUX

export FLASK_APP='swsec_team1'
export FLASK_ENV = 'development'
flask run

Windows Command

set FLASK_APP='swsec_team1'
set FLASK_ENV='development'
flask run

-----------------------------------------------

when you run the server first in your PC,
do:

flask db init
flask db migrate
flask db upgrade

flask run

-----------------------------------------------

do install:

pip install Flask
pip install Flask-sqlalchemy
pip install Flask-wtf
pip install Flask-migrate
pip install werkzeug
pip install wtforms
```

- 현재 로컬 환경에서만 실행 가능

Reliability

- 운영자
  - 메뉴 등록 및 폐기, 메뉴 검색 등으로 메뉴 관리 정상적으로 가능
  - 메인 화면에 월 매출을 그래프로 나타내어 시각적으로 확인 가능
  - 각 메뉴별 판매량과 별점을 확인 가능
  - 각 메뉴별 판매량과 별점을 기준으로 프로그램이 메뉴를 정확하게 평가

- 고객
  - 메뉴 검색 기능으로 이용했던 메뉴를 쉽게 찾기 가능
  - 이용했던 메뉴에 대해 별점 평가 가능
  - 이용했던 메뉴에 대해 리뷰 작성 가능
  - Google 계정으로 별도의 회원가입 없이 쉽게 로그인 가능

Safety

- 식당 운영진을 타겟으로 구현한 프로그램이기 때문에 특정 환경에 피해를 입힐 요소는 존재하지 않음

Security

- 기능적 측면
  - DB Injection

        - 일반적인 DB는 SQL Injection과 같은 Code injection을 통한 공격이 상당수 존재함
        - 그러나 Flask에서 DB로 사용하는 SQLalchemy의 경우 DB Injection과 같은 공격에 대해 입력 값을 자동으로 필터링하는 기능을 제공하기 때문에 공격이 불가능

    - 운영진과 고객의 회원 권한 분리 (Access Control)
        - `access`라는 변수를 이용해 운영진과 고객이 접근할 수 있는 페이지를 분리
        - 운영진 페이지에서는 고객 기능을 사용할 수 없고, 고객 페이지에서는 운영진 기능을 사용할 수 없음

    - Password 암호화
        - 데이터베이스에 회원의 password를 보관할 때, 해시함수를 통해 암호화하여 보관

          ```
          from werkzeug.security import generate_password_hash, check_password_hash
          ```

        - SHA256 사용

- 관리적 측면
    - 고객의 개인정보 유출 우려
        - 회원가입 시 개인정보 제공 동의를 받는 페이지를 추가 구현하여 개인정보 관리를 위한 policy 필요 (미구현)

Resilience

- 관련된 별도 기능을 제공하지는 않음

- 데이터베이스를 백업하여 disruptive event에 대한 대책 마련

- 웹 기반 프로그램이므로 로컬 환경이 아닌 웹 서버를 별도로 구축할 시 네트워크 장애와 관련된 시스템 구축