Panduan Peningkatan Keamanan Backend Flask

1. Konsistensi Arsitektur

Gunakan hanya satu framework, yaitu Flask. Hindari mencampur struktur Django. Ganti semua class-based views menjadi function-based menggunakan @app.route dan gunakan Blueprint untuk modularisasi.

2. CSRF Protection

Gunakan Flask-WTF dan aktifkan CSRF protection agar semua form POST aman dari serangan Cross-Site Request Forgery.

3. Validasi Input dengan Flask-WTF

Buat file forms.py dengan validator seperti DataRequired dan NumberRange dari wtforms.validators. Gunakan form.validate_on_submit() untuk memverifikasi data dari pengguna.

4. Amankan Integrasi Stripe

Validasi jumlah pembayaran dan data produk hanya dari sisi server. Jangan percaya data dari frontend. Gunakan Stripe Checkout Sessions untuk keamanan lebih.

5. Pengaturan Session yang Aman

Gunakan app.permanent_session_lifetime untuk membatasi durasi session. Hindari session yang berlaku selamanya.

6. Validasi Akses Data (IDOR Protection)

Selalu cek apakah user saat ini memiliki akses terhadap objek. Gunakan filter_by(id=..., user_id=current_user.id) untuk menghindari akses tidak sah.

7. Manajemen API Key dan Secret

Simpan API key dan secret dalam file .env dan ambil menggunakan os.getenv(). Jangan hardcode key dalam source code.

Panduan Peningkatan Keamanan Backend Flask

8. Modularisasi dengan Blueprint

Pisahkan route berdasarkan fitur dengan menggunakan Flask Blueprint agar aplikasi lebih terstruktur dan mudah dikelola.

9. Headers Keamanan

Tambahkan headers seperti Content-Security-Policy, X-Frame-Options, dan X-Content-Type-Options di after_request untuk mencegah serangan berbasis browser.

10. Cookie dan HTTPS

Aktifkan SESSION_COOKIE_SECURE, SESSION_COOKIE_HTTPONLY, dan SESSION_COOKIE_SAMESITE di konfigurasi Flask agar session cookie aman saat dikirim.