Security

Security adalah serangkaian teknik, proses, dan teknologi yang digunakan untuk melindungi website dari ancaman, serangan, dan akses tidak sah. Fitur keranjang adalah salah satu titik rawan dalam sistem e-commerce karena langsung terkait data pengguna, harga, transaksi, dan proses checkout. Maka perlu dirancang keamanan yang kuat dan menyeluruh.

Rancangan yang akan diterapkan :

1. Authentication & Session Security

Tujuan: memastikan keranjang hanya bisa diakses oleh pemilik akun.

Langkah Rancangannya:

- Gunakan JWT atau Session-Based Login.
- Simpan session ID atau token secara secure (HTTPOnly cookie).
- Terapkan session timeout otomatis setelah tidak aktif.
- Lakukan re-authentication saat checkout.

2. Access Control (Authorization)

Tujuan: mencegah pengguna melihat atau mengubah keranjang milik orang lain.

Rancangan:

- Setiap keranjang terikat pada **user_id**.
- Jangan gunakan ID keranjang yang bisa ditebak (gunakan UUID).

3. Input Validation & Data Sanitization

Tujuan: mencegah serangan seperti SQL Injection, XSS, LFI, RFI.

Rancangannya:

- Validasi input:
    - jumlah item (harus numeric dan batas maksimal)
    - ID produk (UUID)
    - harga tidak berasal dari frontend
- Sanitasi semua data input menggunakan:

    - prepared statements
    - escaping HTML output

4. Database Security (SQL Injection Prevention)

Tujuan: melindungi query dari manipulasi hacker.

Solusi:

- Batasi hak akses database (least privilege):
- user database untuk app hanya boleh SELECT, INSERT, UPDATE, DELETE.

5. Proteksi API (Rate Limiting & Throttling)

Tujuan :

Agar keranjang tidak bisa diserang dengan spam atau brute force.

Rancangannya:

- Limit 10–20 request / menit per user.
- Blokir IP/user jika request mencurigakan (bot).

6. Enkripsi & HTTPS

Semua data transaksi harus melalui:

- **HTTPS**
- TLS 1.2+
- HSTS Policy

Data sensitif seperti token, session, ID keranjang terenkripsi di server.

Macam macam ancaman :

- SQL Injection
- XSS (Cross-Site Scripting)
- CSRF (Cross-Site Request Forgery)
- DDoS Attack
- Data Breach (kebocoran data)

Contoh serangan yang mungkin terjadi :

SQL Injection

SQL Injection (SQLi) terjadi ketika aplikasi memasukkan input pengguna langsung ke perintah SQL tanpa pemisahan parameter sehingga penyerang bisa menyisipkan (inject) kode SQL yang dieksekusi oleh database. Hasilnya: pembacaan data sensitif, manipulasi data (ubah harga, stok), atau bahkan penghapusan tabel.

Dampak :

1. **Data Bocor (Data Breach)**

   Penyerang bisa melihat seluruh isi database:

   - Data akun
   - Email
   - Alamat
   - No. telepon
   - Data checkout
   - Data produk
   - Data admin

   Ibaratnya: orang asing bisa membuka lemari arsip kantor tanpa izin.

2. **Manipulasi Data (Mengubah Data Secara Ilegal)**

   Penyerang bisa mengubah isi database sesuka hati, misalnya:

   - Mengubah harga produk
   - Mengubah stok
   - Mengubah role user menjadi admin
   - Mengubah password akun lain
   - Mengubah status pesanan

   Ibaratnya: orang titipan masuk gudang dan menempel harga baru dengan seenaknya.

3. **Menghapus Data (Drop Table)**

   Serangan SQL Injection paling ekstrem dapat:

   - Menghapus tabel penting (users, produk, transaksi)
   - Mengosongkan database
   - Merusak seluruh sistem

4. **Login Tanpa Password (Bypass Login)**

   Penyerang bisa masuk sebagai user lain bahkan admin **tanpa password**.

Mengapa database bisa terkena SQL injection :

1. **Tidak ada filter**

   berarti website menerima input dari user apa adanya, tanpa di cek dulu apakah input itu:

- berbahaya atau tidak
- sesuai format atau tidak
- mengandung karakter aneh atau tidak
- wajar atau mencurigakan

Dengan kata lain:
Website tidak menyaring, tidak membersihkan, dan tidak memvalidasi apa pun yang dikirim oleh user.Akibatnya, user bisa memasukkan kode berbahaya dan website akan menerimanya tanpa protes.

2. **Hak akses terlalu luas**

Biasanya aplikasi menggunakan 1 user database, misalnya `app_user`.
Kalau `app_user` diberikan izin yang berlebihan seperti:

- boleh **DROP TABLE**
- boleh **DELETE semua data**
- boleh **UPDATE tabel tanpa batas**
- boleh **mengatur user database lain**
- boleh **mengakses semua tabel**, bukan hanya tabel yang diperlukan

Maka ketika terjadi SQL Injection, hacker bisa memanfaatkan hak akses ini untuk melakukan kerusakan besar.

Seharusnya itu hanya akses : membaca data produk,menambah keranjang,membuat transaksi

Cara penanganan SQL injection:

- **Validasi dan Filter Input**

  Jangan percaya input user.

  Contoh validasi umum:

  - username hanya boleh huruf/angka
  - ID harus angka atau UUID
  - jumlah barang harus integer & batas maksimal
  - tolak karakter berbahaya (`' " ; -- #`) jika tidak diperlukan

- **Batasi Hak Akses User Database (Least Privilege)**

  Jangan berikan akses database “admin” ke aplikasi.

  Pembatasan :

  - user DB **hanya boleh SELECT, INSERT, UPDATE, DELETE**
  - tidak boleh DROP TABLE
  - tidak boleh ALTER TABLE
  - tidak boleh CREATE user baru

DDoS Attack

Distributed = Terdistribusi, berasal dari banyak sumber/komputer
Denial = Penolakan
of Service = Terhadap layanan

Attack = Serangan

Distributed Denial of Service adalah serangan di mana ribuan atau jutaan komputer/bot **mengirim permintaan ke website secara bersamaan** sampai servernya (spam)

Dampak :

- Website tidak bisa diakses (down)
- Pelanggan tidak bisa checkout
- Reputasi berubah buruk
- Potensi kerugian bisnis (penjualan berhenti)
- Server perlu direstart atau dipulihkan

Mengapa bisa terkena DDOS :

1. **Server punya kapasitas terbatas**

   Setiap server (tempat website disimpan) punya batas kemampuan, sama seperti manusia punya batas tenaga.

   Server tidak bisa menangani:

   - terlalu banyak pengunjung dalam waktu bersamaan
   - terlalu banyak request (permintaan)
   - terlalu besar data yang masuk
   - terlalu banyak koneksi yang dibuka
   - penggunaan CPU/RAM yang berlebihan

   Jika batas kemampuan ini terlewati, server akan:

   - menjadi lambat
   - error
   - bahkan mati (down)

2. **Tidak ada perlindungan anti-DDoS**

   Artinya website **tidak memiliki sistem atau alat khusus** untuk:

   - mendeteksi serangan trafik berlebihan,
   - memblokir permintaan berbahaya,
   - membedakan pengunjung asli dan bot,
   - melimit permintaan yang tidak wajar,
   - melindungi server dari kelebihan beban.

   Jika tidak ada perlindungan ini, maka:

   Sedikit saja ada lonjakan trafik, server langsung down.
   Dan jika diserang DDoS, website tidak mampu melawan.

Penanganan DDoS :

1. Gunakan Cloudflare / Anti-DDoS Protection

Ini adalah cara paling efektif dan paling mudah.

Dengan mengaktifkan Anti-DDoS:

- bot akan difilter
- traffic mencurigakan diblokir
- server tidak menerima beban berlebih

Contohnya:

- Cloudflare
- Google Cloud Armor
- AWS Shield
- Akamai

2. Gunakan Rate Limiting

Rate limiting membatasi berapa banyak request yang boleh dikirim pengguna dalam hitungan detik.

Misal:

- 10 request per detik per IP
- blokir jika melewati batas

Ini mencegah:

- spam request
- bot API
- serangan flood

3. Gunakan Firewall (WAF / Network Firewall)

Firewall berfungsi menyaring traffic tidak normal.

Bisa memblokir:

- IP mencurigakan
- request yang terlalu banyak
- user-agent bot
- protokol tidak valid

Contoh firewall:

- UFW (Ubuntu)
- iptables
- Cloudflare WAF