Backend

FrontEnd vs BackEnd

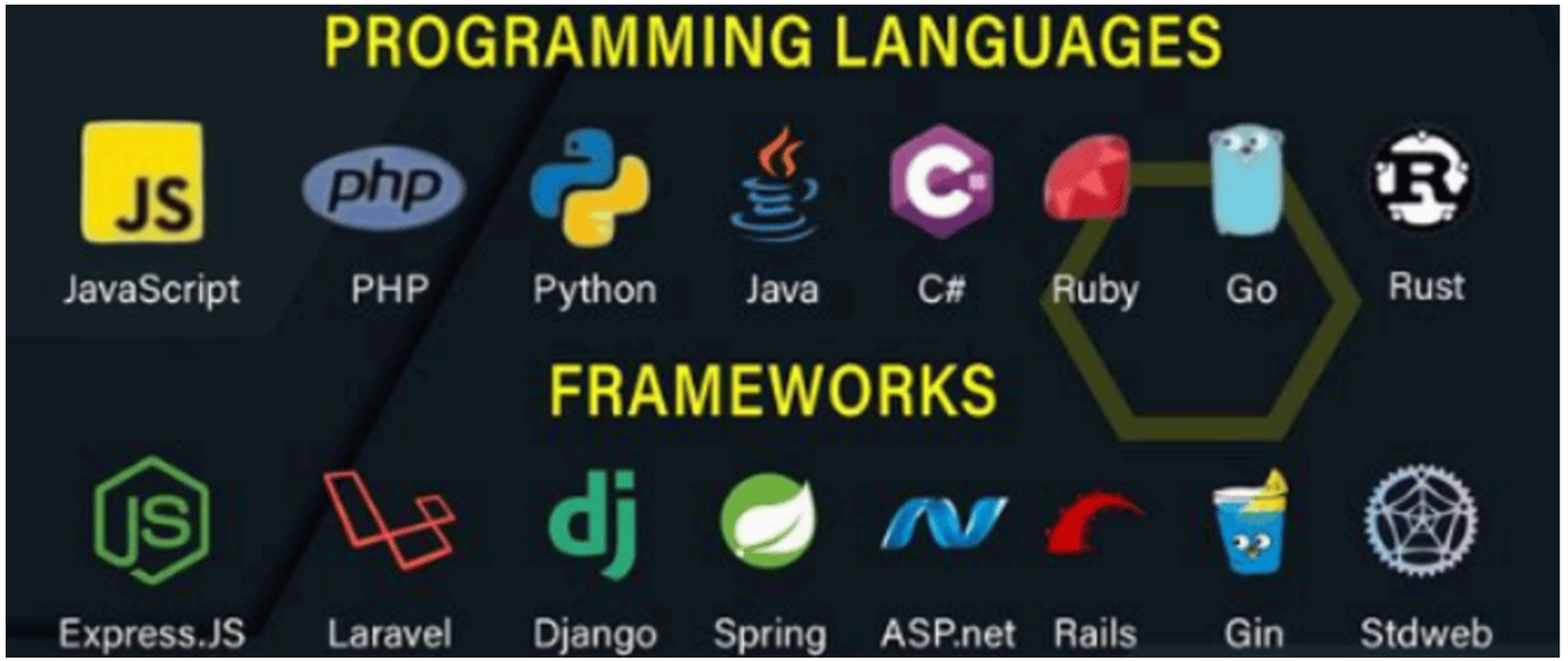
Frontend mengembangkan aspek visual dari website, seperti font, gambar, warna, dll.

Developer frontend juga harus memastikan bahwa website berjalan dan responsive sesuai dengan yang diinginkan.

Bahasa : HTML, CSS, Javascript

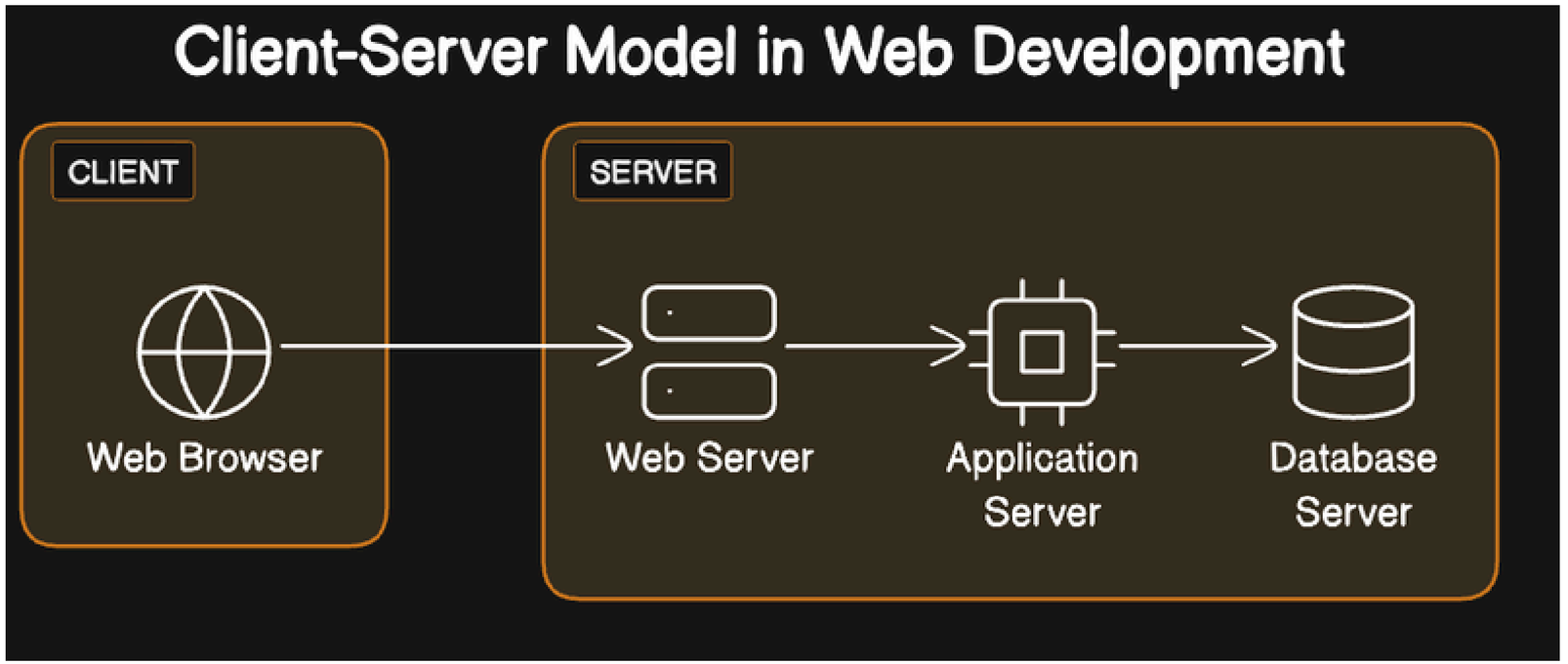
FrontEnd vs BackEnd

Backend adalah bagian dari sebuah sistem perangkat lunak atau aplikasi yang bertanggung jawab atas logika bisnis, pemrosesan data, serta komunikasi dengan database dan layanan eksternal.



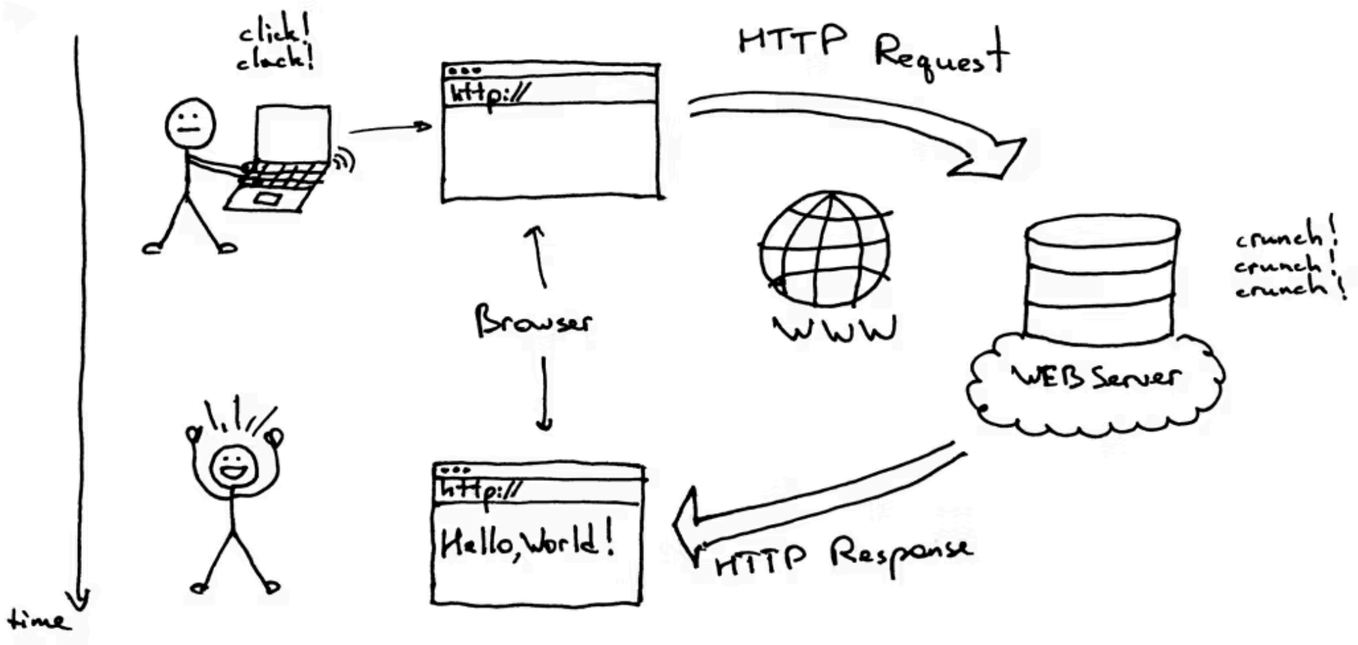
Client-Server

Client-server model adalah arsitektur jaringan yang membagi tugas antara penyedia layanan (server) dan pengguna layanan (client). Model ini digunakan dalam berbagai aplikasi, seperti web, email, dan database.



HTTP

HTTP (HyperText Transfer Protocol) adalah protokol komunikasi yang digunakan untuk mentransfer data antara client (misalnya browser) dan server dalam jaringan internet. HTTP memungkinkan pengiriman dokumen seperti HTML, CSS, JavaScript, gambar, dan video dari server ke client.



HTTP Message

HTTP bekerja berdasarkan model request-response, di mana klien (client) mengirim permintaan (request) ke server, dan server merespons dengan mengirimkan response.

1. HTTP Request

Sebuah HTTP request terdiri dari tiga bagian utama:

1. Request Line → Menentukan metode, URL, dan versi HTTP.
2. Header Fields → Berisi metadata tentang permintaan.
3. Message Body (Opsional) → Berisi data tambahan (biasanya pada metode POST/PUT).

```
POST /login HTTP/1.1
Host: www.example.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 27

username=user&password=12345
```

HTTP Message

2. HTTP Response

Setelah menerima request dari klien, server akan mengirimkan response yang berisi informasi tentang hasil permintaan.

Struktur HTTP Response

1. Status Line → Berisi versi HTTP, kode status, dan pesan status.
2. Header Fields → Berisi metadata tentang response.
3. Message Body (Opsional) → Berisi konten yang dikirim oleh server.

```
HTTP/1.1 200 OK
Date: Mon, 17 Mar 2025 10:30:00 GMT
Server: Apache/2.4.41
Content-Type: text/html
Content-Length: 1256

<html>
  <body>
    <h1>Selamat datang!</h1>
  </body>
</html>
```

HTTP Method

HTTP memiliki beberapa metode untuk mengelola sumber daya:

- GET – Mengambil data dari server.
- POST – Mengirim data ke server untuk diproses.
- PUT – Memperbarui atau mengganti sumber daya di server.
- DELETE – Menghapus sumber daya di server.
- HEAD – Mirip GET tetapi tanpa body response.
- PATCH – Memperbarui sebagian data sumber daya.
- OPTIONS – Memeriksa metode yang didukung server.

HTTP Status

HTTP menggunakan kode status untuk menunjukkan hasil permintaan:

- 1xx (Informasi) → Contoh: 100 Continue
- 2xx (Sukses) → Contoh: 200 OK, 201 Created
- 3xx (Redirect) → Contoh: 301 Moved Permanently, 302 Found
- 4xx (Client Error) → Contoh: 400 Bad Request, 404 Not Found
- 5xx (Server Error) → Contoh: 500 Internal Server Error, 503 Service Unavailable

HTTPS

HTTPS (Hypertext Transfer Protocol Secure) adalah versi HTTP yang lebih aman karena menggunakan TLS/SSL untuk mengenkripsi data. Perbedaan utama:

- HTTP: Data dikirim dalam teks biasa (tidak terenkripsi).
- HTTPS: Data dienkripsi sehingga lebih aman terhadap serangan siber seperti man-in-the-middle (MITM) attack.

URL

URL (Uniform Resource Locator) adalah alamat unik yang digunakan untuk mengakses sumber daya (resource) di internet, seperti halaman web, gambar, video, atau file lainnya.

URL terdiri dari beberapa bagian utama :

1. Skema (Protocol)
2. Hostname (domain name atau ip address)
3. Port (opsional)
4. Path (lokasi resource di server)
5. Query string (opsional)
6. Fragment (opsional)

URL

1. Skema

Menentukan protokol yang digunakan untuk mengakses sumber daya. Contoh:

- http:// → Protokol tanpa enkripsi.
- https:// → Protokol aman dengan enkripsi SSL/TLS.
- ftp:// → Digunakan untuk transfer file.
- mailto: → Untuk mengirim email.

2. Hostname

Menunjukkan lokasi server yang menyimpan sumber daya. Contoh:

- www.example.com (nama domain)
- 192.168.1.1 (alamat IP)

URL

3. Port

Menentukan port jaringan yang digunakan untuk komunikasi dengan server.

- Default: 80 untuk HTTP, 443 untuk HTTPS.
- Bisa ditentukan secara eksplisit, misalnya :8080.

Contoh : http://localhost:3000

4. Path

Menunjukkan lokasi spesifik sumber daya di dalam server.

Contoh : https://example.com/blog/article.html

URL

5. Query String
Digunakan untuk mengirim data tambahan ke server dalam format key=value.
Contoh : https://www.example.com/search?q=chatgpt&page=2

6. Fragment
Menunjukkan bagian tertentu dalam halaman.
Contoh : https://example.com/page.html#section1

Node JS

Bahasa Pemrograman Javascript awalnya hanya bisa berjalan di atas browser, karena ada runtime engine di dalamnya.
Maka Ryan Dahl menciptakan node.js sebagai runtime JavaScript yang berjalan di sisi server.
Dibangun di atas V8 JavaScript Engine (mesin yang juga digunakan oleh Google Chrome), Node.js memungkinkan eksekusi kode JavaScript di luar browser.

Node.js menggunakan event-driven architecture, yang berarti proses tidak perlu menunggu operasi selesai sebelum melanjutkan tugas lainnya.
Node.js menggunakan model single-threaded, tetapi dapat menangani banyak permintaan secara bersamaan menggunakan event loop, sehingga sangat efisien dalam menangani banyak koneksi.

Module

Module dalam Node.js adalah kumpulan kode JavaScript yang dapat digunakan kembali di berbagai bagian aplikasi. Node.js memiliki sistem modular yang memungkinkan pemisahan kode ke dalam file atau pustaka terpisah untuk meningkatkan keterbacaan, pemeliharaan, dan efisiensi kode.

Terdapat 3 jenis module :

1. Core Modules
2. Local Modules
3. Third-Party Modules (NPM)