Resume NodeJS Dasar & RESTful API

1. Pengenalan NodeJS

- NodeJS adalah runtime JavaScript berbasis **V8 Engine** untuk menjalankan JavaScript di luar browser.
- Kelebihan utama NodeJS:
    - Non-blocking I/O dan event-driven.
    - Cocok untuk aplikasi real-time, API server, CLI tools, hingga aplikasi streaming.

2. Instalasi & Konfigurasi

- Instalasi manual dengan mengunduh dari situs resmi NodeJS.
- Bisa juga menggunakan package manager seperti **nvm** (Linux/Mac), **Chocolatey** (Windows), dan **Homebrew** (Mac).
- Setting PATH diperlukan agar NodeJS dapat diakses dari terminal.
- NodeJS menyediakan **REPL (Read Eval Print Loop)** untuk menjalankan perintah interaktif langsung di terminal.

3. Modul & Require

- NodeJS mendukung dua sistem modul:
    - **ES Modules (import/export)** → lebih modern dan direkomendasikan.
    - **CommonJS (require)** → masih digunakan untuk kompatibilitas.
- File dengan ekstensi .mjs digunakan untuk modul modern.
- Pada module, kode global mendukung await langsung karena secara default bersifat async.

4. Standard Library NodeJS

NodeJS memiliki banyak library bawaan untuk mempermudah pengembangan, di antaranya:

- **OS** → memberikan informasi sistem operasi (CPU, memori).
- **Path** → manipulasi path file atau folder.
- **File System (fs)** → membaca, menulis, dan mengelola file.
- **Debugger** → membantu debugging dengan perintah node inspect.
- **DNS** → bekerja dengan domain dan IP.
- **Events** → menangani event listener dengan EventEmitter.
- **Globals** → variabel global seperti console, setTimeout.
- **Process** → informasi tentang proses NodeJS yang berjalan.
- **Readline** → membaca input dari command line.
- **Report** → membuat laporan otomatis saat error.

- **Buffer** → bekerja dengan data biner dan encoding.
- **Stream** → membaca dan menulis data bertahap (streaming).
- **Timer** → penjadwalan dengan setTimeout, setInterval.
- **Net** → membuat TCP client dan server.
- **URL** → manipulasi alamat URL.
- **Util** → kumpulan fungsi utilitas tambahan.
- **Zlib** → melakukan kompresi atau dekompresi (gzip).
- **Console** → logging atau mencetak teks ke terminal.
- **Worker Threads** → menjalankan task CPU-intensive secara paralel.
- **HTTP/HTTPS** → membuat client atau server berbasis HTTP/HTTPS.
- **Cluster** → menjalankan beberapa proses NodeJS untuk memanfaatkan multi-core CPU.

5. Concurrency & Multithreading

- NodeJS berjalan secara default menggunakan single-threaded event loop.
- **Worker Threads** digunakan untuk mengeksekusi proses berat secara paralel.
- **Cluster** memungkinkan menjalankan banyak proses NodeJS sekaligus untuk memaksimalkan penggunaan semua core CPU.

6. Networking & Server

- **Net Module** digunakan untuk membuat TCP client dan server.
- **HTTP Module** digunakan untuk membuat server HTTP atau melakukan request HTTP client.
- **HTTPS Module** menyediakan versi aman dengan SSL/TLS.

7. Debugging & Monitoring

- Mode debugging dijalankan dengan perintah node inspect namafile.js.
- Breakpoint dapat ditambahkan dengan keyword debugger dalam kode.
- NodeJS juga mendukung pembuatan **laporan otomatis (report)** ketika terjadi error.

8. Todolist RESTful API

Contoh implementasi nyata NodeJS adalah membangun **Todolist RESTful API**.

- **GET /todos** → menampilkan daftar todo.
- **POST /todos** → menambahkan todo baru.
- **PUT /todos** → memperbarui todo berdasarkan ID.
- **DELETE /todos** → menghapus todo berdasarkan ID.