# TUGAS PENDAHULUAN
# PEMROGRAMAN PERANGKAT BERGERAK

## MODUL XIV

## DATA STORAGE
## ‘API’


**Disusun Oleh :**

**Aji Prasetyo Nugroho / 2211104049**

**S1SE-06-2**

**Asisten Praktikum :**

**Muhammad Faza Zulian Gesit Al Barru**

**Aisyah Hasna Aulia**

**Dosen Pengampu :**

**Yudha Islami Sulistya, S.Kom., M.Cs.**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**


**2024**

**TUGAS PENDAHULUAN**

**SOAL**

**a.** Sebutkan dan jelaskan dua jenis utama **Web Service** yang sering digunakan dalam pengembangan aplikasi.

**1) SOAP (Simple Object Access Protocol)**

- **Penjelasan**: SOAP adalah protokol berbasis XML yang digunakan untuk pertukaran data antara aplikasi melalui protokol seperti HTTP atau SMTP. SOAP memiliki standar yang ketat dan mendukung transaksi kompleks serta pengaturan keamanan seperti WS-Security.
- **Kelebihan**: Mendukung operasi yang kompleks, memiliki fitur keamanan bawaan, cocok untuk sistem enterprise yang membutuhkan keandalan tinggi.
- **Kekurangan**: Relatif berat karena menggunakan XML, sehingga bisa memperlambat performa.

**2) REST (Representational State Transfer)**

- **Penjelasan**: REST adalah arsitektur berbasis sumber daya (resource-oriented) yang sering menggunakan protokol HTTP. RESTful Web Services menggunakan metode HTTP seperti GET, POST, PUT, dan DELETE untuk mengelola data.
- **Kelebihan**: Ringan, mudah diimplementasikan, kompatibel dengan berbagai format data (misalnya JSON dan XML), serta sangat cocok untuk aplikasi berbasis web.
- **Kekurangan**: Kurang mendukung transaksi kompleks dibandingkan SOAP.

**b.** Apa yang dimaksud dengan **Data Storage API**, dan bagaimana API ini mempermudah pengelolaan data dalam aplikasi?

1) **Definisi**: Data Storage API adalah antarmuka pemrograman aplikasi yang memungkinkan aplikasi untuk berinteraksi dengan sistem penyimpanan data, seperti database, layanan cloud storage, atau file sistem. Contoh API ini adalah Firebase Realtime Database, Amazon S3, dan Google Cloud Storage.
2) **Manfaat**:

- **Abstraksi dan Kemudahan**: Pengembang tidak perlu memahami detail teknis penyimpanan data; cukup menggunakan fungsi API untuk menyimpan,

membaca, menghapus, atau memperbarui data.

- **Efisiensi**: Operasi data dapat dioptimalkan melalui API yang dirancang untuk performa tinggi.
- **Keamanan**: API sering kali dilengkapi fitur autentikasi dan otorisasi untuk mengamankan data.
- **Skalabilitas**: Memungkinkan aplikasi untuk dengan mudah menangani jumlah data yang besar.

**c.** Jelaskan bagaimana proses kerja komunikasi antara klien dan server dalam sebuah Web Service, mulai dari permintaan (*request*) hingga tanggapan (*response*).

1) **Permintaan (Request) dari Klien**:
   - Klien mengirimkan permintaan HTTP (misalnya, GET, POST, PUT, atau DELETE) ke server.
   - Permintaan ini berisi informasi seperti URL endpoint, header, metode HTTP, dan body data (jika ada).
2) **Pemrosesan di Server**:
   - Server menerima permintaan dari klien.
   - Server memproses permintaan tersebut, misalnya dengan memanggil fungsi yang relevan atau mengakses database.
   - Server memvalidasi data permintaan dan mengeksekusi logika bisnis yang sesuai.
3) **Pengiriman Tanggapan (Response) ke Klien**:
   - Setelah selesai memproses, server mengirimkan tanggapan HTTP ke klien.
   - Tanggapan ini berisi kode status HTTP (misalnya, 200 untuk sukses, 404 untuk tidak ditemukan, atau 500 untuk kesalahan server) serta data (jika diperlukan, biasanya dalam format JSON atau XML).
4) **Klien Menangani Tanggapan**:
   - Klien menerima dan memproses data dari tanggapan server.
   - Tergantung pada tanggapan, klien bisa menampilkan data kepada pengguna, menyimpan data, atau mengambil langkah lanjutan.

**d.** Mengapa keamanan penting dalam penggunaan **Web Service**, dan metode apa saja yang dapat diterapkan untuk memastikan data tetap aman?

1) **Alasan Keamanan Penting**:
   - **Melindungi Data Sensitif**: Data pribadi atau bisnis harus aman dari akses tidak sah.

- **Mencegah Serangan Siber**: Seperti injeksi SQL, XSS (Cross-Site Scripting), dan man-in-the-middle (MITM).
- **Menjaga Kepercayaan Pengguna**: Pengguna cenderung tidak menggunakan aplikasi yang tidak aman.

2) **Metode Keamanan**:

- **Autentikasi**:
  - Menggunakan mekanisme seperti OAuth, API key, atau token JWT untuk memastikan hanya pengguna yang berwenang yang dapat mengakses API.
- **Enkripsi**:
  - Menggunakan protokol seperti HTTPS (SSL/TLS) untuk mengenkripsi komunikasi antara klien dan server.
- **Validasi Input**:
  - Memastikan data yang diterima server aman dan bebas dari potensi eksploitasi, seperti injeksi SQL.
- **Rate Limiting**:
  - Membatasi jumlah permintaan dari klien untuk mencegah serangan DDoS.
- **Firewall dan IDS/IPS**:
  - Menggunakan sistem deteksi dan pencegahan intrusi untuk memantau aktivitas mencurigakan.
- **Pembaruan dan Patch Rutin**:
  - Memastikan sistem dan perangkat lunak selalu menggunakan versi terbaru untuk melindungi dari kerentanan yang diketahui.