**Application Phonebook Service**

1. **Struktur Aplikasi**

Aplikasi Phonebook ini adalah aplikasi service yang memungkinkan pengguna untuk mengelola data kontak mereka. Fungsionalitas utama aplikasi mencakup operasi CRUD (Create, Read, Update, Delete) + Search untuk data kontak. Berikut adalah komponen utama aplikasi:

1. **Controller**: Mengelola permintaan HTTP dan memberikan respons yang sesuai.

2. **Service**: Mengandung logika bisnis dan interaksi dengan database.

3. **Repository**: Mengelola akses data ke database.

4. **Model**: Mewakili struktur data yang disimpan dalam database.

5. **Configuration**: Mengatur pengaturan aplikasi, seperti koneksi database dan akses dari

aplikasi lain (web dan mobile).

1. **Penjelasan Fungsi**
2. **Model**

Model mewakili struktur data untuk kontak. Biasanya, ini adalah kelas Java yang mendefinisikan atribut yang akan disimpan dalam database.

**Contact**: Kelas ini mungkin memiliki atribut seperti id, name, email, dan phoneNumber.

1. **Repository**

Repository mengelola akses data ke database. Spring Data JPA menyediakan antarmuka dasar yang dapat Anda gunakan untuk mengakses data tanpa perlu menulis banyak kode SQL secara manual.

**ContactRepository**: Antarmuka ini menghubungkan model Contact dengan operasi database.

1. **Service**

Service berisi logika bisnis dan berinteraksi dengan repository untuk mendapatkan atau memodifikasi data.  
**ContactService**: Kelas ini menyediakan metode untuk operasi CRUD seperti menambah, menghapus, memperbarui, dan mencari kontak.

1. **Controller**

Controller mengelola permintaan HTTP dari pengguna, menerjemahkannya menjadi panggilan layanan, dan mengembalikan hasilnya dalam format yang sesuai (misalnya JSON).

**ContactController**: Kelas ini menangani permintaan REST untuk operasi CRUD dan menyediakan endpoint API untuk mengelola kontak.

1. **Configuration**

Configuration berisi pengaturan untuk aplikasi, termasuk pengaturan database dan akses dari aplikasi lain (web & mobile).

**application.properties**: File ini berisi konfigurasi untuk menghubungkan aplikasi dengan database.

**WebConfig:** Kelas ini menangani konfigurasi agar aplikasi lain bisa akses ke aplikasi service ini.

1. **Alur Kerja**
2. **Membuat Kontak:** Pengguna mengirimkan permintaan POST ke /api/contacts dengan data kontak baru. ContactController meneruskan data ini ke ContactService, yang kemudian menyimpan data di database melalui ContactRepository.
3. **Mendapatkan Kontak**: Pengguna mengirimkan permintaan GET ke /api/contacts untuk mendapatkan daftar semua kontak. ContactController meminta data dari ContactService, yang kemudian mengambil data dari database dan mengembalikannya kepada pengguna.
4. **Memperbarui Kontak**: Pengguna mengirimkan permintaan PUT ke /api/contacts/{id} dengan data kontak yang diperbarui. ContactController meneruskan data ini ke ContactService, yang memperbarui data di database melalui ContactRepository.
5. **Menghapus Kontak**: Pengguna mengirimkan permintaan DELETE ke /api/contacts/{id}. ContactController memanggil ContactService untuk menghapus data kontak dari database.
6. **Mencari Kontak**: Pengguna mengirimkan permintaan GET ke /api/contacts/search?name={name}. ContactController meneruskan pencarian ini ke ContactService, yang melakukan pencarian di database berdasarkan nama kontak.