



MSIG

test-msig-developer-BE

2024

PHONEBOOK APPLICATION DOCUMENTATION

**JAVA SPRING BOOT IMPLEMENTATION
CRUD OPERATIONS, CLEAN ARCHITECTURE,
AND API INTEGRATION**

By :
MOH FAISAL AMIR

<https://github.com/mohfaisalamir>

Daftar Isi

1. Pendahuluan

- Deskripsi Aplikasi
- Fitur Utama

2. Persyaratan Sistem

- Software
- Dependencies

3. Arsitektur Aplikasi

- Clean Architecture dan Diagram Arsitektur

4. Penjelasan Fungsional dan API Endpoints (Payload/Postman Requests)

- CRUD Operations
- Search Functionality

5. Instalasi dan Konfigurasi

- Langkah-langkah Instalasi
- Konfigurasi

8. UI/Frontend

- Desain UI
- Screenshot

9. Penutup

- Kontak

1. Pendahuluan

1.1 Deskripsi Aplikasi

Aplikasi Phonebook ini adalah sebuah aplikasi berbasis web yang dikembangkan menggunakan Java Spring Boot dengan mengikuti standar J2EE. Aplikasi ini memungkinkan pengguna untuk mengelola kontak melalui fitur-fitur CRUD (Create, Read, Update, Delete) yang lengkap. Selain itu, aplikasi ini dirancang menggunakan prinsip Clean Architecture untuk memastikan bahwa setiap lapisan kode memiliki tanggung jawab yang jelas dan terpisah, sehingga memudahkan pengembangan dan pemeliharaan di masa depan. Dengan menggunakan API yang terintegrasi, aplikasi ini juga dapat diakses dan diuji melalui alat-alat seperti Postman dan Swagger.

1.2 Fitur Utama

1. **CRUD Operations:** Pengguna dapat melakukan operasi Create, Read, Update, dan Delete terhadap data kontak, dengan implementasi yang mengikuti standar J2EE dan Spring Boot.
2. **Search Functionality:** Pengguna dapat mencari kontak berdasarkan nama dengan fleksibilitas dalam variasi huruf besar/kecil, sehingga memudahkan dalam menemukan informasi kontak yang diinginkan.
3. **Clean Architecture:** Aplikasi ini dibangun dengan arsitektur bersih yang memisahkan logika bisnis, logika aplikasi, dan antarmuka pengguna. Ini memastikan kode lebih modular, mudah dikelola, dan dapat diperluas di masa depan.
4. **Integrasi API:** Aplikasi ini menyediakan endpoint API yang memungkinkan interaksi dengan data kontak melalui berbagai klien, seperti frontend React yang telah dibuat atau alat seperti Postman. API ini juga didokumentasikan menggunakan Swagger.
5. **UI Sederhana dan Fungsional:** Aplikasi ini dilengkapi dengan antarmuka pengguna yang sederhana namun efektif, memudahkan pengguna dalam mengakses dan mengelola data kontak. UI frontend dikembangkan menggunakan React.
6. **Testing dan Dokumentasi:** Aplikasi ini telah diuji menggunakan Postman, dan dokumentasi lengkap, termasuk payload/request, serta script tabel untuk pengujian, disediakan untuk memastikan aplikasi berjalan dengan baik.

2. Persyaratan Sistem

2.1 Software

1. **Sistem Operasi:** Aplikasi ini dapat dijalankan pada sistem operasi yang mendukung Java, seperti Windows, macOS, atau Linux.
2. **Java Development Kit (JDK):** Versi 21 atau lebih baru diperlukan untuk menjalankan dan mengembangkan aplikasi ini.
3. **IDE:** Menggunakan IDE seperti IntelliJ IDEA, dan VSCode yang telah terintegrasi dengan Spring Boot.
4. **Database:** PostgreSQL (versi 14.12) digunakan sebagai sistem manajemen basis data. Anda bisa menggunakan pgAdmin untuk administrasi database.
5. **Web Browser:** Untuk aplikasi frontend, menggunakan browser modern seperti Google Chrome, Mozilla Firefox, atau Microsoft Edge.

2.2 Dependencies

1. **Spring Boot:** Framework utama untuk membangun aplikasi Java, digunakan untuk implementasi backend.
2. **Spring Data JPA:** Untuk interaksi dengan basis data menggunakan Java Persistence API (JPA).
3. **PostgreSQL Driver:** Untuk koneksi aplikasi ke basis data PostgreSQL.
4. **Axios:** Untuk mengirimkan permintaan HTTP dari frontend React.
5. **React:** Library JavaScript untuk membangun antarmuka pengguna (frontend).
6. **Bulma:** Framework CSS yang digunakan untuk styling UI di frontend.
7. **Postman:** Alat untuk menguji dan memverifikasi API, digunakan untuk mengirim permintaan HTTP dan melihat respons API.

3. Arsitektur Aplikasi

3.1 Clean Architecture

Aplikasi ini dibangun mengikuti prinsip Clean Architecture untuk memastikan pemisahan yang jelas antara berbagai lapisan aplikasi, meningkatkan modularitas, dan memudahkan pemeliharaan serta pengembangan.

Struktur arsitektur aplikasi adalah sebagai berikut:

3.1.1 Backend:

1. **Config:** Mengelola konfigurasi aplikasi termasuk pengaturan database, keamanan, dan konfigurasi lainnya yang diperlukan.
2. **Controller:** Menangani permintaan HTTP dari klien, memproses input, dan meneruskannya ke layanan yang sesuai.
3. **DTO (Data Transfer Object):** Menyediakan cara untuk memindahkan data antara lapisan aplikasi tanpa mengungkapkan detail implementasi internal.
4. **Entity:** Model data yang mewakili tabel-tabel di basis data dan struktur data inti aplikasi.
5. **Repository:** Mengelola interaksi dengan basis data, termasuk operasi CRUD untuk entitas.
6. **Service:** Menyimpan logika bisnis utama aplikasi, bertindak sebagai jembatan antara controller dan repository.
7. **Utils:** Kumpulan fungsi utilitas yang mendukung berbagai aspek teknis aplikasi, seperti format data atau fungsi bantu lainnya. Berikut Diagram Backend :



3.1.2 Frontend:

Component: Komponen utama dari antarmuka pengguna (UI) meliputi:

1. **ContactList:** Menampilkan daftar kontak yang tersedia dengan opsi untuk menambah, mengedit, atau menghapus kontak.
2. **AddContact:** Formulir untuk menambah kontak baru ke dalam sistem.
3. **UpdateContact:** Formulir untuk memperbarui informasi kontak yang sudah ada.

Berikut Diagram Frontend :



4. Penjelasan Fungsional dan API Endpoints (Payload/Postman Requests)

4.1 CRUD Operations

Aplikasi Phonebook mendukung operasi CRUD (Create, Read, Update, Delete) untuk mengelola data kontak. Berikut adalah rincian dari masing-masing operasi:

4.1.1 Create (Menambah Kontak):

- **Deskripsi:** Pengguna dapat menambahkan kontak baru ke dalam sistem.
- **Endpoint:**
POST :<http://localhost:8080/api/contact>
- **Request Body:** Menggunakan DTO AddContactDTO yang berisi informasi kontak seperti nama, nomor telepon, dan alamat email.
- **Respons:** Menyediakan detail kontak yang baru saja ditambahkan, termasuk ID yang dihasilkan oleh sistem.

Contoh Request Body:

```
{
  "name": "Yono",
  "email": "yono@msig.com",
  "phone": "082200006754"
}
```

4.1.2 Read (Membaca Kontak):

- **Deskripsi:** Pengguna dapat melihat daftar semua kontak atau detail kontak tertentu.
- **Endpoint:**
GET : <http://localhost:8080/api/contacts> (untuk daftar semua kontak)
GET: <http://localhost:8080/api/contacts/{name}> (untuk kontak by Name).
- **Respons:** Mengembalikan daftar kontak atau detail kontak berdasarkan ID.

Contoh Respons untuk Daftar Kontak:

```
[
  {
    "id": "ff808181910cccf901910d35271e0001",
    "name": "Abraham",
    "email": "abraham@msig.com",
    "phone": "08454656768"
  },
  {
    "id": "ff808181910cccf901910e1717d20002",
    "name": "Alex",
    "email": "alex@msig.com",
    "phone": "082256576783"
  },
  {
    "id": "95e36dd8-3e87-4ff1-bb30-d5fb373e30d8",
    "name": "Alimin",
    "email": "alimin@msig.com",
    "phone": "081234567896"
  }
]
```

4.1.3 Update (Memperbarui Kontak):

- **Deskripsi:** Pengguna dapat memperbarui informasi kontak yang sudah ada.
- **Endpoint:**
PUT :<http://localhost:8080/api/contact/{name}>
- **Request Body:** Menggunakan DTO UpdateContactDTO yang berisi informasi baru untuk kontak.
- **Respons:** Menyediakan detail kontak yang telah diperbarui.

Contoh Request Body:

```
{
  "name": "Yono",
  "email": "yono@msig.com",
  "phone": "082200006754"
}
```

4.1.4 Delete (Menghapus Kontak):

- **Deskripsi:** Pengguna dapat menghapus kontak dari sistem.
- **Endpoint:**
DELETE:<http://localhost:8080/api/contact/{name}>
- **Respons:** Konfirmasi bahwa kontak telah dihapus.

Contoh Respons:

```
{
  "message": "Successfully deleted contact",
  "statusCode": 200,
  "data": "Contact with name 'yudi' has been deleted."
}
```

4.1.5 Search Functionality (get contact by Name):

Aplikasi Phonebook memungkinkan pengguna untuk mencari kontak berdasarkan nama. Fitur ini memudahkan dalam menemukan kontak yang diperlukan.

- **Deskripsi:** Pengguna dapat mencari kontak dengan nama yang diinputkan.
- **Endpoint:**
GET : <http://localhost:8080/api/contact/{name}>
- **Parameter:** Query parameter name untuk pencarian nama kontak.
- **Respons:** Mengembalikan daftar kontak yang cocok dengan nama yang dicari, baik sesuai nama lengkap atau sebagian nama.

Contoh Request untuk Pencarian:

GET : <http://localhost:8080/api/contact/malaka>

Respons yang ditampilkan postman:

```
{
  "message": "Successfully retrieved contacts",
  "statusCode": 200,
  "data": [
    {
      "id": "cd5be70e-062d-405f-b905-a2b5e613d6f4",
      "name": "Ibrahim Gelar Datuk Sutan Malaka",
      "email": "tanmalaka@msig.com",

```



```

        "phone": "081234567894"
    }
]
}

```

Catatan:

- Pastikan untuk mengganti {id} dalam URL dengan ID yang sesuai saat menggunakan endpoint yang memerlukan ID.
- Pastikan untuk mengganti {name} dalam URL dengan nama kontak yang sesuai saat menggunakan endpoint pencarian berdasarkan nama.
- Payload untuk request dan response harus mengikuti format JSON yang sesuai dengan contoh di atas.

4.2 Script Tabel

Untuk pengujian di lingkungan database, berikut adalah script SQL yang dapat digunakan untuk membuat tabel yang dibutuhkan dalam aplikasi Phonebook:

- Script untuk Membuat Tabel Kontak:

```

CREATE TABLE contacts (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    name VARCHAR(100) NOT NULL,
    phone_number VARCHAR(15) NOT NULL,
    email VARCHAR(100),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

```

- Script untuk Mengisi Data Uji:

```

INSERT INTO m_contact (name, phone, email) VALUES
('Soekarno', '081234567890', 'soekarno@msig.com'),
('Hatta', '081234567891', 'hatta@msig.com'),
('Sudirman', '081234567892', 'sudirman@msig.com'),
('Kartini', '081234567893', 'kartini@msig.com'),
('Tan Malaka', '081234567894', 'tanmalaka@msig.com'),
('Muso', '081234567895', 'muso@msig.com'),
('Alimin', '081234567896', 'alimin@msig.com'),
('Semaun', '081234567897', 'semaun@msig.com'),
('Hasanuddin', '081234567898', 'hasanuddin@msig.com'),
('Bung Tomo', '081234567899', 'bungtomo@msig.com'),
('Dahlan', '081234567900', 'dahlan@msig.com'),
('Salim', '081234567901', 'salim@msig.com'),
('Antasari', '081234567902', 'antasari@msig.com'),
('Cokro', '081234567903', 'cokro@msig.com'),
('Soetomo', '081234567904', 'soetomo@msig.com'),
('Siti', '081234567905', 'siti@msig.com'),
('Dewantara', '081234567906', 'dewantara@msig.com'),
('Yani', '081234567907', 'yani@msig.com'),
('Wartabone', '081234567908', 'wartabone@msig.com'),
('Raden', '081234567909', 'raden@msig.com'),
('Arief', '081234567910', 'arief@msig.com'),
('Kamil', '081234567911', 'kamil@msig.com');

```

5. Instalasi dan Konfigurasi

5.1 Langkah-langkah Instalasi

5.1.1 Persiapan Sistem

Pastikan sistem operasi kamu sudah diperbarui. Instal semua dependensi yang diperlukan, seperti Java, PostgreSQL, dan ReactJS.

Instalasi Backend

5.1.2 Clone repositori aplikasi dari GitHub:

`git clone git@github.com:mohfaisalamir/test-msig-developer-BE.git`

5.1.3 Arahkan ke direktori proyek:

`cd repository`

5.1.4 Instalasi Dependencies:

- **Untuk backend:** Pastikan Anda memiliki JDK versi 11 atau lebih baru. Jalankan perintah berikut untuk menginstal dependencies menggunakan Maven:
`./mvnw install`
- **Untuk frontend:** Masuk ke direktori frontend dan instal dependencies dengan npm:
`cd frontend`
`npm install`

5.1.5 Jalankan Aplikasi Backend:

Jalankan aplikasi backend menggunakan Maven:

`./mvnw spring-boot:run`

Aplikasi akan berjalan di <http://localhost:8080>.

5.1.6 Jalankan Aplikasi Frontend:

Di direktori frontend, jalankan aplikasi React dengan:

`npm start`

6. UI/Frontend

6.1 Desain UI:

Antarmuka pengguna (UI) untuk aplikasi Phonebook dirancang agar sederhana dan intuitif, dengan fokus pada kemudahan penggunaan. Komponen utama dalam UI meliputi:

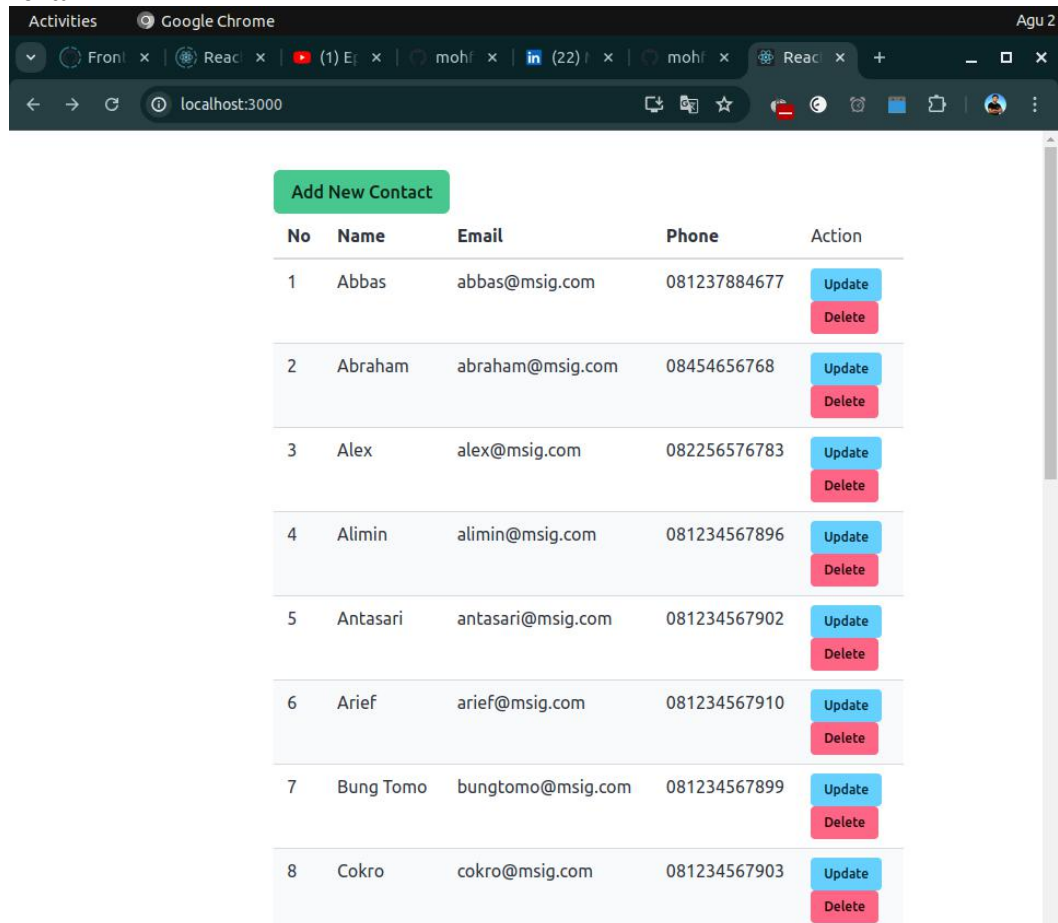
1. **ContactList:** Menampilkan daftar semua kontak yang tersimpan dalam aplikasi. Setiap kontak disajikan dengan opsi untuk mengedit atau menghapusnya.
2. **AddContact:** Formulir untuk menambahkan kontak baru. Formulir ini mencakup input untuk nama, nomor telepon, dan email.
3. **UpdateContact:** Formulir untuk memperbarui informasi kontak yang sudah ada. Pengguna dapat mengubah nama, nomor telepon, atau email dari kontak yang dipilih.

UI aplikasi menggunakan framework CSS seperti Bulma untuk memastikan tampilan yang bersih dan responsif, sehingga mudah diakses baik pada perangkat desktop maupun seluler.

6.2 Screenshot:

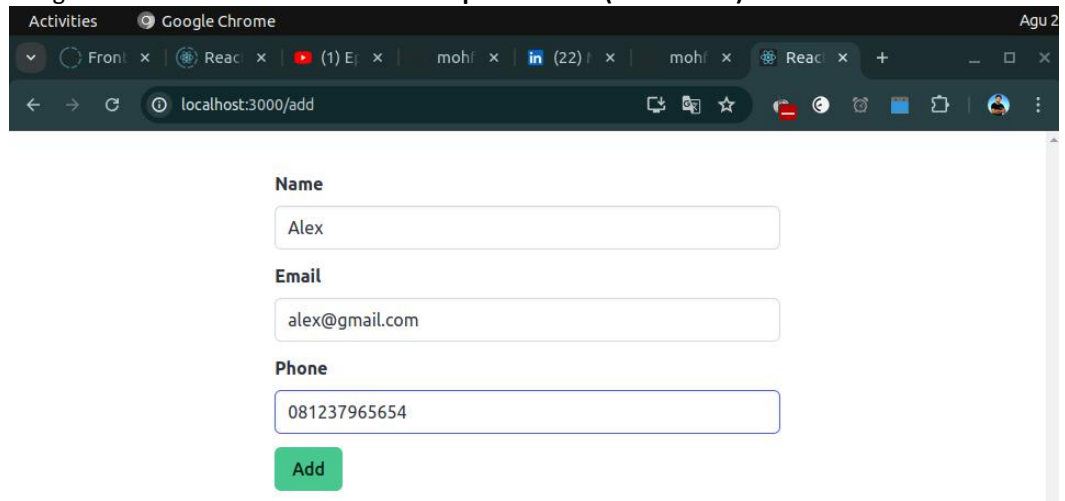
Berikut adalah beberapa tangkapan layar dari antarmuka aplikasi Phonebook:

1. **Tampilan Utama (ContactList):** Menampilkan daftar semua kontak dengan opsi untuk mengedit atau menghapus kontak.



2. **Formulir Tambah Kontak (AddContact):**

Menyediakan input untuk menambahkan nama, nomor telepon, dan email kontak baru.
Dengan 'Click' **Add New Customer** di **Tampilan Utama (ContactList)**



Activities Google Chrome Agu 2

localhost:3000/add

Name

Alex

Email

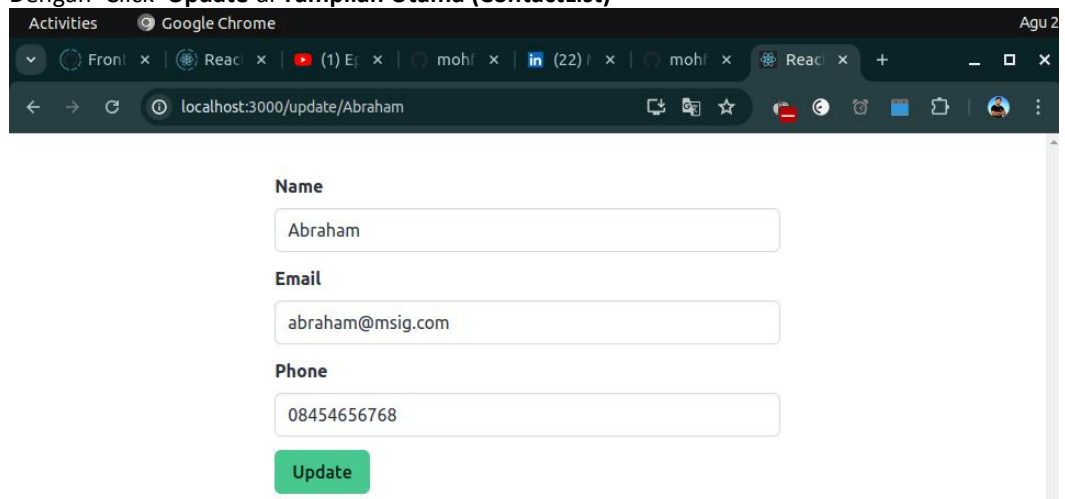
alex@gmail.com

Phone

081237965654

Add

3. **Formulir Perbarui Kontak (UpdateContact):**
Menampilkan informasi kontak yang dipilih untuk diedit dan menyimpan perubahan.
Dengan 'Click' **Update** di **Tampilan Utama (ContactList)**



Activities Google Chrome Agu 2

localhost:3000/update/Abraham

Name

Abraham

Email

abraham@msig.com

Phone

08454656768

Update

Tangkapan layar ini membantu memberikan gambaran visual tentang bagaimana pengguna dapat berinteraksi dengan fitur-fitur yang disediakan dalam aplikasi.

7. Penutup

- Kontak:

Jika ada pertanyaan lebih lanjut atau jika Anda memerlukan bantuan terkait aplikasi Phonebook ini, jangan ragu untuk menghubungi saya melalui email atau media komunikasi lainnya yang tercantum di bawah ini:

- **Nama:** MOH FAISAL AMIR
- **Email:** mohfaisalamir94@gmail.com
- **LinkedIn:** <https://www.linkedin.com/in/mohfaisalamir/>
- **GitHub:** <https://github.com/mohfaisalamir/>

Terima kasih telah meninjau dokumentasi aplikasi ini. Jika ada pertanyaan atau memerlukan bantuan lebih lanjut, mohon untuk menghubungi saya melalui kontak yang tertera. Saya berharap aplikasi ini memenuhi kriteria yang diharapkan dan dapat diterima di PT Asuransi MSIG Indonesia.