Nama : Muh. Nur Afrizal
NIM : 222212738
Kelas / No : 3SI1 / 22

LAPORAN PRAKTIKUM PEMROGRAMAN PLATFORM KHUSUS

MODUL 5

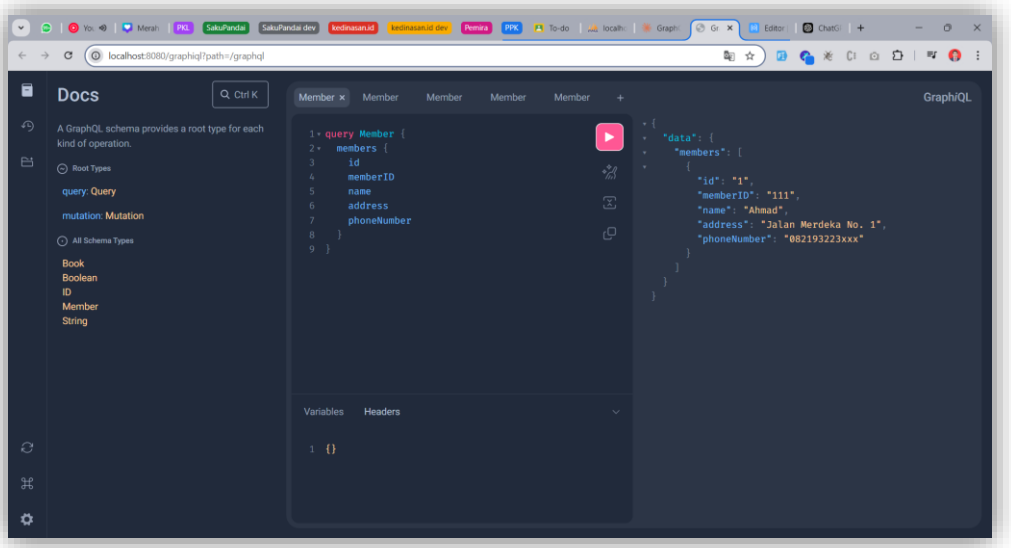
GraphQL

Lanjutkan proyek perpustakaan praktikum 5 dengan menambahkan operasi CRUD untuk kelas entity Member menggunakan GraphQL.
Buat laporan pekerjaan Anda dengan penjelasan yang meliputi penjelasan kode program yang dibuat, pengujian setiap operasi CRUD beserta output responnya.

Link repository git: https://github.com/afrzl/Pemrograman-Platform-Khusus/tree/master/Pertemuan%205/perpustakaan

A. Output

1. View All Members

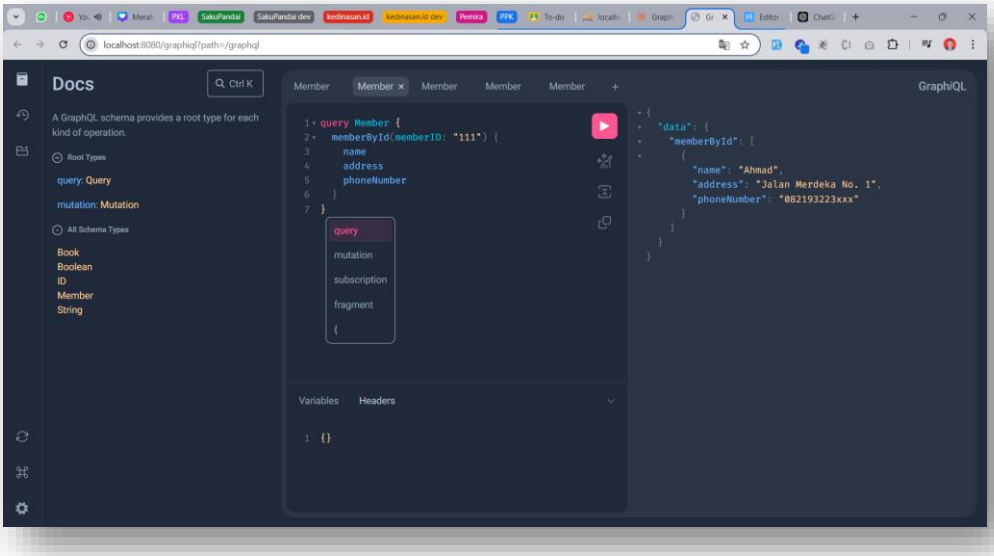


Nama : Muh. Nur Afrizal

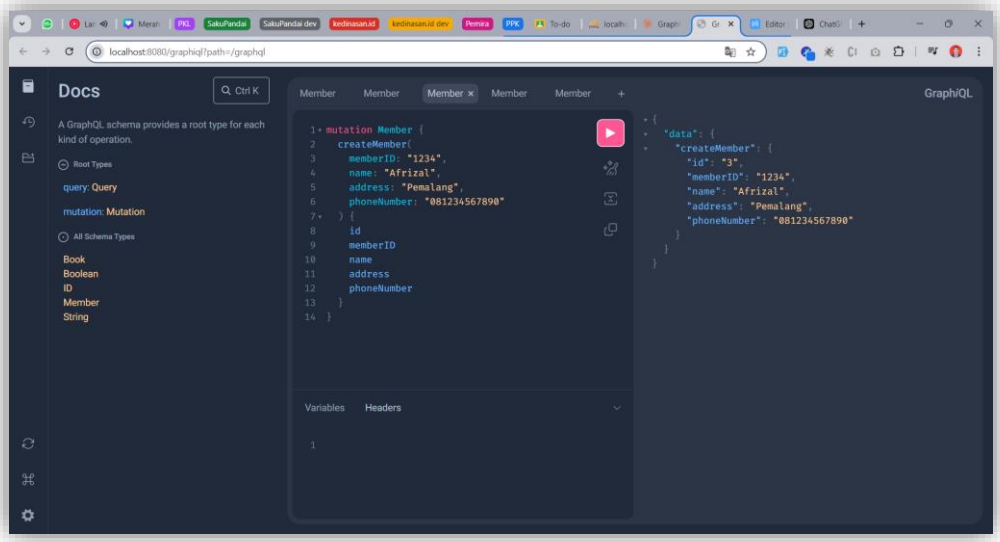
NIM : 222212738

Kelas / No : 3SI1 / 22

2. View Member By ID



3. Create Member

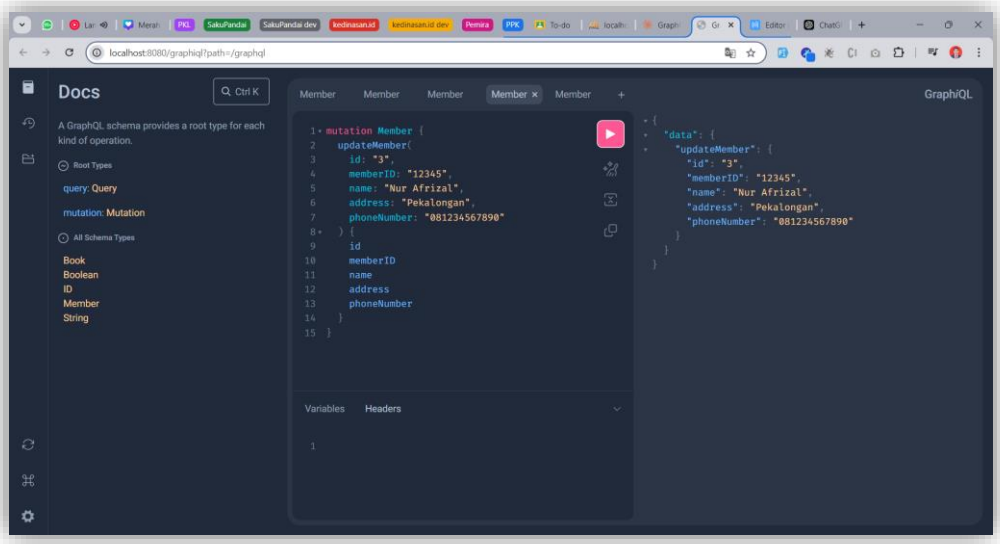


Nama : Muh. Nur Afrizal

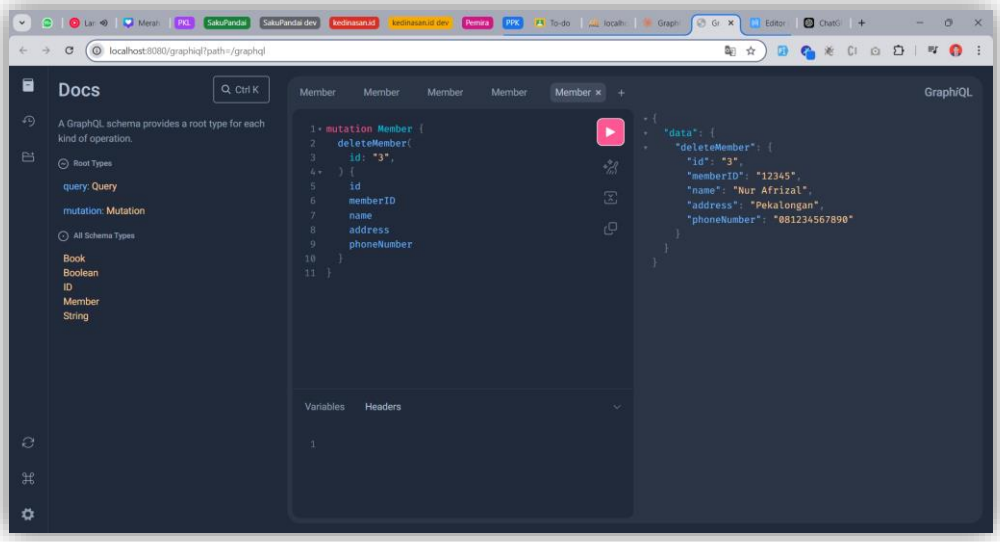
NIM : 222212738

Kelas / No : 3SI1 / 22

4. Update Member



5. Delete Member



B. schema.graphqls

```
type Book {
    id: ID
    title: String
    author: String
    description: String
}

type Member {
    id: ID
    memberID: String
    name: String
    address: String
    phoneNumber: String
}
```

Nama : Muh. Nur Afrizal
NIM : 222212738
Kelas / No : 3SI1 / 22

```
type Query {
    books:[Book]
    bookById(id: ID): Book
    members:[Member]
    memberById(memberID: String): [Member]
}

type Mutation {
    createBook(title: String!, description: String, author: String!) :
Book!
    updateBook(id:String!, title: String!, description: String, author:
String!) : Book!
    deleteBook(id:String!): Book

    createMember(memberID: String!, name: String!, address: String,
phoneNumber: String) : Member!
    updateMember(id:String!, memberID: String!, name: String!, address:
String, phoneNumber: String) : Member!
    deleteMember(id:String!): Member
}
```

Skema GraphQL ini mendefinisikan dua tipe utama, yaitu Book dan Member, dengan beberapa field seperti id, title, dan author untuk buku serta memberID, name, dan phoneNumber untuk anggota. Ada query untuk mendapatkan daftar buku dan anggota, serta detail berdasarkan ID atau memberID. Terdapat juga mutasi untuk membuat, memperbarui, dan menghapus buku serta anggota. Semua operasi mutasi membutuhkan field tertentu sebagai input untuk memodifikasi data dalam sistem.

C. MemberGraphqlController

```
package com.polstat.perpustakaan.controller;

import ...

@Controller
public class MemberGraphqlController {
    @Autowired
    private MemberRepository memberRepository;

    @QueryMapping
    public List<Member> members() {
        return (List<Member>) memberRepository.findAll();
    }

    @QueryMapping
    public List<Member> memberById(@Argument String memberID) {
        return memberRepository.findByMemberID(memberID);
    }

    @MutationMapping
    public Member createMember(@Argument String memberID,
                               @Argument String name,
                               @Argument String address,
                               @Argument String phoneNumber) {
        Member member = Member.builder()
                .memberID(memberID)
                .name(name)
```

Nama : Muh. Nur Afrizal
NIM : 222212738
Kelas / No : 3SI1 / 22

```
                .address(address)
                .phoneNumber(phoneNumber)
                .build();

        return memberRepository.save(member);
    }

    @MutationMapping
    public Member updateMember(@Argument String id,
                               @Argument String memberID,
                               @Argument String name,
                               @Argument String address,
                               @Argument String phoneNumber) {
        Member existingMember =
memberRepository.findById(Long.parseLong(id))
                .orElseThrow(() -> new RuntimeException("Member not found
with id: " + id));

        if (memberID != null) existingMember.setMemberID(memberID);
        if (name != null) existingMember.setName(name);
        if (address != null) existingMember.setAddress(address);
        if (phoneNumber != null)
existingMember.setPhoneNumber(phoneNumber);

        return memberRepository.save(existingMember);
    }

    @MutationMapping
    public Member deleteMember(@Argument String id) {
        Member existingMember =
memberRepository.findById(Long.parseLong(id))
                .orElseThrow(() -> new RuntimeException("Member not found
with id: " + id));

        memberRepository.delete(existingMember);

        return existingMember;
    }
}
```

Controller MemberGraphqlController ini mengelola operasi GraphQL untuk entitas Member dalam aplikasi Spring. Terdapat dua query, members() untuk mengambil semua anggota, dan memberById() untuk mencari anggota berdasarkan memberID. Selain itu, tiga mutasi tersedia: createMember() untuk membuat anggota baru, updateMember() untuk memperbarui data anggota berdasarkan id, dan deleteMember() untuk menghapus anggota. Operasi mutasi menerima input sebagai argumen dan memanipulasi data di repository yang terhubung ke database melalui MemberRepository.

D. Member

```
package com.polstat.perpustakaan.entity;

import ...

@Setter
@Getter
```

Nama : Muh. Nur Afrizal
NIM : 222212738
Kelas / No : 3SI1 / 22

```java
@AllArgsConstructor
@NoArgsConstructor
@Builder
@Entity
@Table(name = "members")
public class Member {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(name = "memberid", nullable = false)
    private String memberID;
    @Column(nullable = false)
    private String name;
    @Column(nullable = false)
    private String address;
    @Column(name = "phone_number", nullable = true)
    private String phoneNumber;
}
```

Kelas Member ini merupakan entitas JPA yang merepresentasikan tabel members di dalam database. Setiap anggota memiliki beberapa atribut: id yang dihasilkan secara otomatis sebagai primary key, memberID, name, address, dan phoneNumber. Atribut-atribut ini dipetakan ke kolom tabel dengan anotasi @Column, di mana memberID, name, dan address tidak boleh kosong, sedangkan phoneNumber bersifat opsional. Kelas ini juga menggunakan anotasi dari Lombok seperti @Getter, @Setter, @AllArgsConstructor, @NoArgsConstructor, dan @Builder untuk mempermudah pembuatan getter, setter, konstruktor, dan pola builder.

E. MemberRepository

```java
package com.polstat.perpustakaan.repository;

import ...

@RepositoryRestResource(collectionResourceRel = "member", path = "member")
public interface MemberRepository extends
PagingAndSortingRepository<Member, Long>, CrudRepository<Member, Long> {
    List<Member> findByName(@Param("name") String name);
    List<Member> findByMemberID(@Param("memberid") String memberID);
}
```

MemberRepository adalah interface yang bertindak sebagai repository untuk entitas Member, memperluas PagingAndSortingRepository dan CrudRepository agar mendukung operasi CRUD serta paginasi dan sorting. Repository ini menggunakan anotasi @RepositoryRestResource, yang secara otomatis membuat endpoint REST untuk entitas Member. Selain itu, dua metode query kustom disediakan: findByName() untuk mencari anggota berdasarkan nama, dan findByMemberID() untuk mencari anggota berdasarkan memberID, dengan parameter yang dipetakan melalui anotasi @Param.