Bagaskoro Meyca Dwiyananda Putra
1506736530

Table of Contents

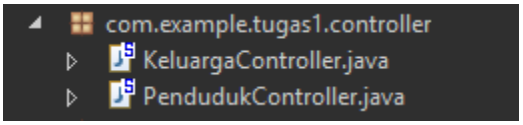
Bagaskoro Meyca Dwiyananda Putra
1506736530

Structur Project, Package, serta implementasi

MVC

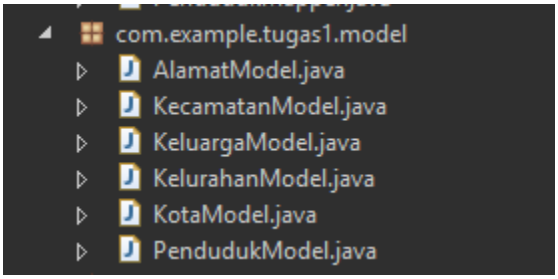
Controller

Saya menggunakan dua controller yaitu keluarga dan penduduk dikarenakan dari semua fungsi yang diberikan saya kategorikan menjadi dua. Index saya masukkan ke dalam PendudukController karena Penduduk Controller merupakan yang paling utama.



Model

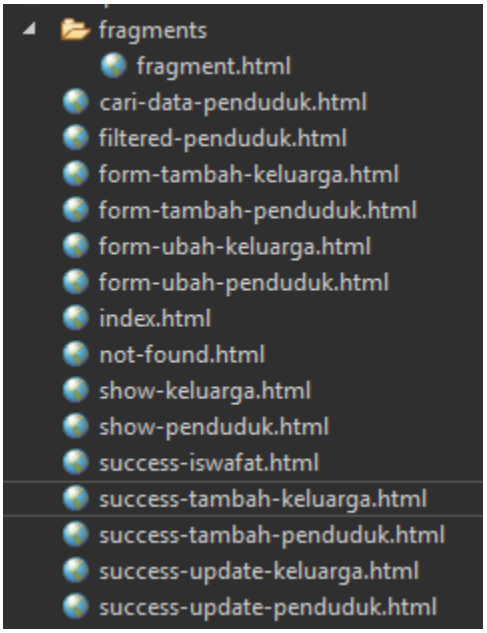
Untuk model saya gunakan daftar berikut



Alamat adalah class yang dimiliki oleh keluarga dan alamat sendiri memiliki unsur kelas lain yaitu kecamatan, keluarahan dan kota. Alamat juga memiliki penduduk sebagai anggota keluarga dan alamat yang dimiliki keluarga otomatis menjadi lamat anggota keluarga

View

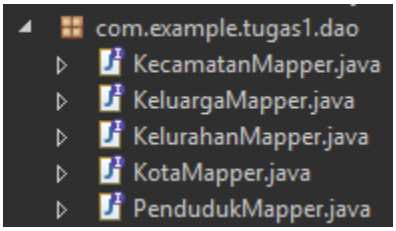
Untuk tampilan saya menggunakan satu fragment sebagai template utama dan 4 halaman utama yaitu index, cari-data-penduduk, form-tambah=keluarga dan form-tambah-penduduk



Lain-lain

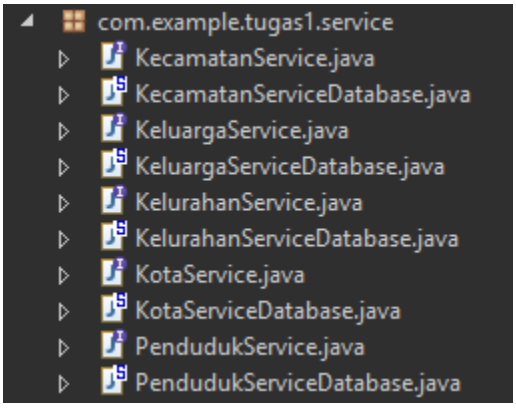
Mapper

Untuk kelas Mapper yang ada dalam package .dao saya menggunakan lima kelas yang sama dengan model hanya tidak memasukkan alamat.



Service

Untuk Service saya gunakan 10 yang berpasangan dengan mapper yaitu 5 interface dan 5 class implementasinya



Bagaskoro Meyca Dwiyananda Putra
1506736530

Implementasi

Pada fungsi cari penduduk berdasarkan NIK dan ubah penduduk saya menggunakan tambahan fungsi sebagai enkapsulasi yaitu NIK Generator

```
public String nikGenerator(AlamatModel alamat, String tanggal_lahir, int jenis_kelamin) {
    String nik = Integer.toString(alamat.getKode_kecamatan());
    nik = nik.substring(0, nik.length()-1);

    int tanggal = Integer.parseInt(tanggal_lahir.substring(tanggal_lahir.length()-2,tanggal_
    tanggal = (jenis_kelamin == 0)? tanggal : tanggal + 40;
    nik += String.format("%02d", tanggal)  + tanggal_lahir.substring(5,7) + tanggal_lahir.su

    int urutan = pendudukService.countNIK("%"+nik+"%") + 1;
    nik += String.format("%04d", urutan);
    urutan = pendudukService.countNIK("%"+nik+"%");
    if(urutan >= 1){
        nik = "" + (Integer.parseInt(nik) + 1);
    }

    return nik;
}
```

Fungsi itu digunakan oleh dua fungsi lainnya sebagai pembantu dalam mendapatkan format NIK. Penggunaannya pada fungsi berikut ini:

```
@RequestMapping(value = "/penduduk", method = RequestMethod.GET)
public String findPenduduk (@RequestParam(value = "nik", required = false, defaultValue="0")
{
    PendudukModel penduduk = pendudukService.getPenduduk(nik);

    if(penduduk == null){
        return "not-found";

    } else{
        AlamatModel alamat = pendudukService.getAlamatPenduduk(nik);
        String kewarganegaraan = (penduduk.getIs_wni() == 1? "WNI":"WNA");
        String statusWafat = (penduduk.getIs_wafat() == 1? "Wafat":"Hidup");

        model.addAttribute("penduduk", penduduk);
        model.addAttribute("alamat", alamat);
        model.addAttribute("kewarganegaraan",kewarganegaraan);
        model.addAttribute("wafat", statusWafat);

        return "show-penduduk";
    }
```

```
@RequestMapping(value = "/penduduk/ubah/{nik}", method = RequestMethod.POST)
public String updatePendudukPOST(Model model, PendudukModel penduduk, @RequestParam(value =
        @RequestParam(value = "jenis_kelamin_lama", required = false) int jenis_kelamin_lama
        @RequestParam(value = "tanggal_lahir_lama", required = false) String tanggal_lahir_l

    if(id_keluarga_lama != penduduk.getId_keluarga() || penduduk.getJenis_kelamin() != jenis
            !penduduk.getTanggal_lahir().equalsIgnoreCase(tanggal_lahir_lama)){

        AlamatModel alamat = pendudukService.getAlamatKeluarga(penduduk.getId_keluarga());

        String nik = nikGenerator(alamat, penduduk.getTanggal_lahir(), penduduk.getJenis_kel

        int id = pendudukService.getId(penduduk.getNik());
        model.addAttribute("nik", penduduk.getNik());
        penduduk.setNik(nik);

        pendudukService.updatePenduduk(id, penduduk.getNik(), penduduk.getNama(), penduduk.g
                , penduduk.getJenis_kelamin(), penduduk.getIs_wni(), penduduk.getId_keluarga
                penduduk.getStatus_perkawinan(), penduduk.getStatus_dalam_keluarga(), pendud
        model.addAttribute("flag",true);
        model.addAttribute("penduduk", pendudukService.getPenduduk(penduduk.getNik()));
        model.addAttribute("nik", nik);
        return "form-ubah-penduduk";
    } else{
        int id = pendudukService.getId(penduduk.getNik());
        model.addAttribute("nik", penduduk.getNik());
        pendudukService.updatePenduduk(id, penduduk.getNik(), penduduk.getNama(), penduduk.g
                , penduduk.getJenis_kelamin(), penduduk.getIs_wni(), penduduk.getId_keluarga
                penduduk.getStatus_perkawinan(), penduduk.getStatus_dalam_keluarga(), pendud
        model.addAttribute("penduduk", pendudukService.getPenduduk(penduduk.getNik()));
        model.addAttribute("nik", penduduk.getNik());
        model.addAttribute("flag",true);
        return "form-ubah-penduduk";
    }
}
```

Enkapsulasi tersebut juga digunakan pada class keluarga yaitu nkkGenerator.

Bagaskoro Meyca Dwiyananda Putra
1506736530

Optimasi Database

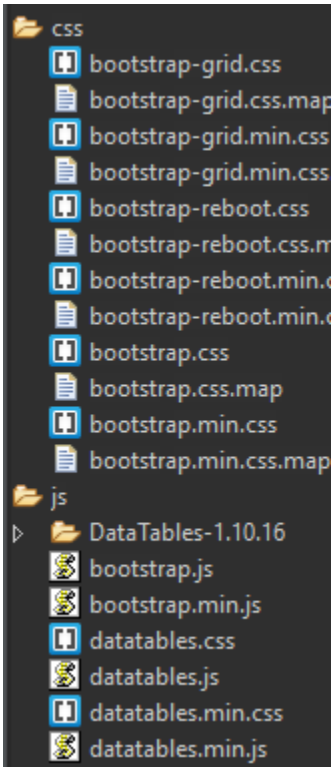
Optimasi database yang saya lakukan adalah dengan menambahkan primary key pada setiap atribut id dan index pada atribut nomor seperti nik dan nkk. Selain itu saya menambahkan hubungan foreign key pada atribut-atribut yang berhubungan di table-tabel yang berbeda seperti atribut id_kecamatan di kelurahan dengan id pada kecamatan.

Bagaskoro Meyca Dwiyananda Putra
1506736530

Stress Testing

Stress testing saya lakukan di kasus mencari penduduk di kelurahan tertentu, data yang dipanggil bervariasi dan cukup banyak. Waktu yang tercatat adalah 0.8 s sehingga bisa dikategorikan cepat tapi tidak reliable karena bisa sangat lambat di keadaan banyak user yang mengakses.

Tampilan



Untuk tampilan saya menggunakan bootstrap dan datatables

Bagaskoro Meyca Dwiyananda Putra
1506736530

Fitur Lain

Fitur yang saya tambahkan adalah pagination di bagian pencarian penduduk berdasarkan kelurahan