



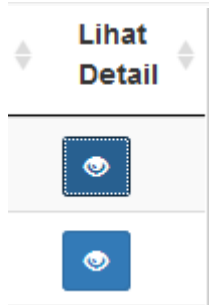
Konsep: Membuat SIDUK dengan Springboot, My Sql untuk database serta Bootstrap untuk Tampilan

Manfaat: Mempermudah pekerjaan pemerintah DKI Jakarta dan Bapak Gubernur

Fitur lain(selain fitur bonus):

1. Dependency dropdown pada fitur formulir 4,6 dan 8 (akan dijelaskan pada penjelasan fitur)
2. Fitur lihat detail penduduk pada fitur 8, yakni terdapat tombol yang akan membantu user melihat detail penduduk saat menampilkan daftar penduduk pada suatu kelurahan, kecamatan dan kota tertentu (simbol mata).

No.	Nama Lengkap	NIK	Jenis Kelamin	Tempat Lahir	Tanggal Lahir	Agama	Pekerjaan	Status Perkawinan	Kewarganegaraan	Lihat Detail
1	Heru Haryanto	3101012407110003	Pria	Jakarta	2011-07-24	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	WNA	
2	Taswir Rajata	3171020303890001	Pria	Jakarta	1989-03-03	Islam	KARYAWAN HONORER	Kawin	WNI	



3. Fitur lihat data, ketika sukses menambahkan penduduk (fitur 3), dan lihat detail keluarga ketika sukses menambahkan keluarga (fitur 4)

**Sukses!**

Penduduk dengan NIK = 3101015212170001 berhasil ditambahkan!








[Lihat Data](#)




































**Sukses!**

Keluarga dengan NKK = 3171022210170002 berhasil ditambahkan!

[Lihat Data](#)

## Struktur Project :

- Model : Penduduk, Keluarga, Kelurahan, Kecamatan, Kota
    - ▼  com.example.demo.model
      - >  KecamatanModel.java
      - ▼  KeluargaModel.java
        - >  KeluargaModel
      - >  KelurahanModel.java
      - >  KotaModel.java
      - >  PendudukModel.java
  - View :

 src/main/resources
    - ▼  static
      - >  css
      - >  fonts
      - >  images
      - >  js
    - ▼  templates
      - ▼  error
        -  404.html
        -  500.html
        -  error-page.html
        -  not-found.html
        -  detail-keluarga.html
        -  detail-penduduk.html
        -  form-cari-penduduk.html
        -  form-tambah-keluarga.html
        -  form-tambah-penduduk.html
        -  form-update-keluarga.html
        -  form-update-penduduk.html
        -  index.html
        -  list-penduduk.html
        -  sukses-tambah-keluarga.html
        -  sukses-tambah-penduduk.html
        -  sukses-update-keluarga.html
        -  sukses-update-penduduk.html
- Controller : SidukController, SidukService, SidukServiceDatabase, SidukMapper
  - >  com.example.demo
  - ▼  com.example.demo.controller
    - >  SidukController.java
  - ▼  com.example.demo.dao
    - ▼  SidukMapper.java
      - >  SidukMapper
  - >  com.example.demo.model
  - ▼  com.example.demo.service
    - >  SidukService.java
    - >  SidukServiceDatabase.java

- Optimasi : Memberi atribut, primary key, auto increment, unik, foreign key, dan index pada tabel

Pada tabel Penduduk:

## Table: penduduk

[Select data](#)[Show structure](#)[Alter table](#)[New item](#)

Column	Type	Comment
<b>id</b>	bigint(20) unsigned <i>Auto Increment</i>	
<b>nik</b>	char(16)	
<b>nama</b>	varchar(128)	
<b>tempat_lahir</b>	varchar(128)	
<b>tanggal_lahir</b>	date	
<b>jenis_kelamin</b>	int(1)	
<b>is_wni</b>	tinyint(1)	
<b>id_keluarga</b>	bigint(20) unsigned	
<b>agama</b>	varchar(64)	
<b>pekerjaan</b>	varchar(64)	
<b>status_perkawinan</b>	varchar(64)	
<b>status_dalam_keluarga</b>	varchar(64)	
<b>golongan_darah</b>	varchar(32)	
<b>is_wafat</b>	tinyint(1) [0]	

## Indexes

<b>PRIMARY</b>	<i>id</i>
<b>UNIQUE</b>	<i>nik</i>
<b>INDEX</b>	<i>id_keluarga</i>

[Alter indexes](#)

## Foreign keys

Source	Target	ON DELETE	ON UPDATE	
<b><i>id_keluarga</i></b>	<i>keluarga(id)</i>	RESTRICT	CASCADE	<a href="#">Alter</a>

Pada tabel Keluarga

### Table: keluarga

[Select data](#) [Show structure](#) [Alter table](#) [New item](#)

Column	Type	Comment
<b>id</b>	bigint(20) unsigned <i>Auto Increment</i>	
<b>nomor_kk</b>	char(16)	
<b>alamat</b>	varchar(256)	
<b>RT</b>	char(3)	
<b>RW</b>	char(3)	
<b>id_kelurahan</b>	bigint(20) unsigned	
<b>is_tidak_berlaku</b>	tinyint(1) [0]	

### Indexes

<b>PRIMARY</b>	<i>id</i>
<b>UNIQUE</b>	<i>nomor_kk</i>
<b>INDEX</b>	<i>id_kelurahan</i>

[Alter indexes](#)

### Foreign keys

Source	Target	ON DELETE	ON UPDATE	
<b>id_kelurahan</b>	kelurahan( <i>id</i> )	RESTRICT	CASCADE	<a href="#">Alter</a>

Pada tabel Kelurahan

### Table: kelurahan

[Select data](#) [Show structure](#) [Alter table](#) [New item](#)

Column	Type	Comment
<b>id</b>	bigint(20) unsigned	
<b>id_kecamatan</b>	bigint(20) unsigned	
<b>kode_kelurahan</b>	char(10)	
<b>nama_kelurahan</b>	varchar(255)	
<b>kode_pos</b>	char(5)	

### Indexes

<b>PRIMARY</b>	<i>id</i>
<b>UNIQUE</b>	<i>kode_kelurahan</i>
<b>INDEX</b>	<i>id_kecamatan</i>

[Alter indexes](#)

### Foreign keys

Source	Target	ON DELETE	ON UPDATE	
<b>id_kecamatan</b>	kecamatan( <i>id</i> )	RESTRICT	CASCADE	<a href="#">Alter</a>

Pada tabel kecamatan

## Table: kecamatan

[Select data](#) [Show structure](#) [Alter table](#) [New item](#)

Column	Type	Comment
<b>id</b>	bigint(20) unsigned	
<b>id_kota</b>	bigint(20) unsigned	
<b>kode_kecamatan</b>	char(7)	
<b>nama_kecamatan</b>	varchar(255)	

## Indexes

<b>PRIMARY</b>	<i>id</i>
<b>UNIQUE</b>	<i>kode_kecamatan</i>
<b>INDEX</b>	<i>id_kota</i>

[Alter indexes](#)

## Foreign keys

Source	Target	ON DELETE	ON UPDATE	
<i>id_kota</i>	<i>kota(id)</i>	RESTRICT	CASCADE	<a href="#">Alter</a>

[Add foreign key](#)

Pada tabel kota

## Table: kota

[Select data](#) [Show structure](#) [Alter table](#) [New item](#)

Column	Type	Comment
<b>id</b>	bigint(20) unsigned	
<b>kode_kota</b>	char(4)	
<b>nama_kota</b>	varchar(255)	

## Indexes

<b>PRIMARY</b>	<i>id</i>
<b>UNIQUE</b>	<i>kode_kota</i>

[Alter indexes](#)

## 1. Fitur 1 : Tampilkan Data Penduduk Berdasarkan NIK

User dapat memasukkan nik pada halaman index, dan melakukan pencarian pada field "Cari NIK", dengan menggunakan form request method get.

**Cari NIK**

Pada controller akan menerima @RequestMapping (value="/penduduk") dan melaksanakan method detailPenduduk().

```
@RequestMapping(value= "/penduduk", method = RequestMethod.GET)
public String detailPenduduk (Model model,
    @RequestParam(value = "nik", required=false) String nik
{
    if(nik==""){
        String errorArgs= "NIK harus diisi!";
        model.addAttribute("errorNIKArgs", errorArgs);
        return "index";
    }

    if(!nik.matches("\\d*")){
        String errorArgs= "NIK tidak boleh mengandung huruf!";
        model.addAttribute("errorNIKArgs", errorArgs);
        return "index";
    }

    PendudukModel penduduk = sidukDAO.selectPenduduk(nik);

    if(penduduk!= null){
        model.addAttribute ("penduduk", penduduk);
        return "detail-penduduk";
    }
    else{
        model.addAttribute("nomor",nik);
        return "error/not-found";
    }
}
```

Method tersebut menerima parameter nik yang dimasukkan oleh user, apabila nik null, maka akan dikembalikan warning label sebagai berikut.

**Cari NIK****NIK harus diisi!**

apabila nik mengandung karakter, maka akan dikembalikan warning label sebagai berikut.

**Cari NIK****NIK tidak boleh mengandung huruf!**

Apa bila nik valid dan penduduk ada maka akan dipanggil method selectPenduduk dengan parameter nik di kelas sidukServiceDatabase.java.

```

@Override
public PendudukModel selectPenduduk(String nik) {
    log.info ("select penduduk with nik {}", nik);
    return sidukMapper.selectPenduduk (nik);
}

```

Method `selectPenduduk` pada `sidukServiceDatabase.java` akan memanggil `selectPenduduk(nik)` dengan parameter `nik` pada kelas `sidukMapper.java`, method tersebut akan melakukan *'chaining'* dengan memanggil `selectKelurahan`, kemudian method `selectKelurahan` akan memanggil method `selectKecamatan` dan method `selectKecamatan` akan memanggil method `selectKota`.

```

@Select("select * from penduduk where nik = #{nik}")
@Results(value = { @Result(property = "nik", column = "nik"),
    @Result(property = "nama", column = "nama"),
    @Result(property = "tempat_lahir", column = "tempat_lahir"),
    @Result(property = "jenis_kelamin", column = "jenis_kelamin"),
    @Result(property = "is_wni", column = "is_wni"),
    @Result(property = "id_keluarga", column = "id_keluarga"),
    @Result(property = "agama", column = "agama"),
    @Result(property = "pekerjaan", column = "pekerjaan"),
    @Result(property = "status_perkawinan", column = "status_perkawinan"),
    @Result(property = "status_dalam_keluarga", column = "status_dalam_keluarga"),
    @Result(property = "golongan_darah", column = "golongan_darah"),
    @Result(property = "is_wafat", column = "is_wafat"),
    @Result(property = "keluarga", column = "id_keluarga",
        javaType = KeluargaModel.class,
        many = @Many(select = "selectKeluarga")) })
PendudukModel selectPenduduk(@Param("nik") String nik);

```

Kemudian apabila penduduk ditemukan, akan dikembalikan objek penduduk dengan atribut-atributnya dan ditampilkan pada view `detail-penduduk.html`

### Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#) / [Data Penduduk](#)

#### Lihat Data Penduduk - 3101012407110001

NIK	3101012407110001
Nama	Heru Haryanto
Tempat/Tanggal Lahir	Jakarta, 2011-07-24
Jenis Kelamin	Pria
Alamat	Ds. Adisumarmo No. 43
RT/RW	079/025
Kelurahan	PULAU TIDUNG
Kecamatan	KEPULAUAN SERIBU SELATAN
Kota	KABUPATEN KEPULAUAN SERIBU
Golongan Darah	A+
Agama	Islam
Status Perkawinan	Belum Kawin
Pekerjaan	BELUM/TIDAK BEKERJA
Kewarganegaraan	WNI
Status Kematian	Hidup

Namun bila penduduk tidak ditemukan akan ditampilkan view not-found.html

## Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#)

Data Nomor = 31010114051270001 tidak ditemukan

[Kembali ke Halaman Utama](#)

2. Fitur 2 : Tampilkan Data Keluarga Beserta Daftar Anggota Berdasarkan Nomor KK  
User dapat memasukkan nik pada halaman index, dan melakukan pencarian pada field "Cari NKK", dengan menggunakan form request method get.

Cari NKK	<input type="text" value="Masukkan NKK"/>
	<input type="button" value="Lihat"/>

Pada controller akan menerima @RequestMapping (value="/keluarga") dan melaksanakan method anggotaKeluarga().

```
@RequestMapping(value = "/keluarga",method = RequestMethod.GET)
public String anggotaKeluarga(Model model,
    @RequestParam(value="nkk") String nkk){
    String errorArgs;
    if(nkk==""){
        errorArgs= "NKK harus diisi!";
        model.addAttribute("errorNKKArgs", errorArgs);
        return "index";
    }

    if(!nkk.matches("\\d*")){
        errorArgs= "NKK tidak boleh mengandung huruf!";
        model.addAttribute("errorNKKArgs", errorArgs);
        return "index";
    }
    KeluargaModel keluarga = sidukDAO.selectKartuKeluarga(nkk);

    if(keluarga!= null){
        List<PendudukModel> anggotaKeluarga = sidukDAO.selectAnggotaKeluarga(nkk);
        model.addAttribute("anggota", anggotaKeluarga);
        model.addAttribute("keluarga", keluarga);
        return "detail-keluarga";
    }
    else{
        model.addAttribute("nomor",nkk);
        return "error/not-found";
    }
}
```



Method tersebut menerima parameter nik yang dimasukkan oleh user, apabila nik null, maka akan dikembalikan warning label sebagai berikut.

Cari NKK

Lihat **NKK harus diisi!**

apabila nik mengandung karakter, maka akan dikembalikan warning label sebagai berikut.

Cari NKK

Lihat **NKK tidak boleh mengandung huruf!**

Apa bila nik valid dan penduduk ada maka akan dipanggil method selectPenduduk dengan parameter nik di kelas sidukServiceDatabase.java.

```
@Override
public KeluargaModel selectKartuKeluarga(String nkk) {
    log.info ("select keluarga with nkk {}", nkk);
    return sidukMapper.selectKartuKeluarga (nkk);
}
```

Method selectKartuKeluarga pada sidukServiceDatabase.java akan memanggil selectKartuKeluarga(nkk) dengan parameter nomor kk pada kelas sidukMapper.java, method tersebut akan melakukan 'chaining' dengan memanggil selectKelurahan, kemudian method selectKelurahan akan memanggil method selectKecamatan dan method selectKecamatan akan memanggil method selectKota.

```
@Select("select * "
+ "from keluarga "
+ "where keluarga.nomor_kk = #{nkk}")
@Results(value = { @Result(property = "id", column = "id"),
@Result(property = "nomor_kk", column = "nomor_kk"),
@Result(property = "alamat", column = "alamat"),
@Result(property = "rt", column = "rt"),
@Result(property = "rw", column = "rw"),
@Result(property = "id_kelurahan", column = "id_kelurahan"),
@Result(property = "is_tidak_berlaku", column = "is_tidak_berlaku"),
@Result(property = "kelurahan", column = "id_kelurahan",
javaType = KelurahanModel.class,
many = @Many(select = "selectKelurahan")) })
KeluargaModel selectKartuKeluarga(@Param("nkk") String nkk);
```

Kemudian apabila Keluarga ditemukan, akan dipanggil method selectAnggotaKeluarga pada kelas sidukServiceDatabase.java.

```
@Override
public List<PendudukModel> selectAnggotaKeluarga(String nkk) {
    log.info ("select keluarga with nkk {}", nkk);
    return sidukMapper.selectAnggotaKeluarga(nkk);
}
```

Method selectAnggotaKeluarga pada sidukServiceDatabase.java akan memanggil selectAnggotaKeluarga(nkk) dengan parameter nomor kk pada kelas sidukMapper.java, method tersebut akan mencari dan menyimpan daftar anggota keluarga berdasarkan nkk yang dimasukkan

```
@Select("select * "
      + "from penduduk, keluarga "
      + "where keluarga.nomor_kk= #{nkk} and penduduk.id_keluarga = keluarga.id")
List<PendudukModel> selectAnggotaKeluarga(@Param("nkk") String nkk);
```

Kemudian akan dikembalikan arraylist anggota keluarga beserta atributnya dan ditampilkan pada view detail-keluarga.html

Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#) / [Data Keluarga](#)

Lihat Data Keluarga - 3101010101050001

NKK = 3101010101050001

Alamat = Jln. Cikapayang No. 594

RT/RW = 149/090

Kelurahan/Desa = PULAU PARI

Kecamatan = KEPULAUAN SERIBU SELATAN

Kota = KABUPATEN KEPULAUAN SERIBU

Nama Lengkap	NIK	Jenis Kelamin	Tempat Lahir	Tanggal Lahir	Agama	Pekerjaan	Status Perkawinan	Status dalam Keluarga	Kewarganegaraan
Taswir Rajata	3101010303890001	0	Jakarta	1989-03-03	Islam	KARYAWAN HONORER	Kawin	Kepala Keluarga	1
Mulyono Hutagalung S.Gz	3101011408150001	0	Jakarta	2015-08-14	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	Anak	1
Yuliana Wahyuni	3101016312160001	1	Jakarta	2016-12-23	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	Anak	1
Ifa Novitasari	3101014712830001	1	Jakarta	1983-12-07	Islam	PETANI/PEKEBUN	Kawin	Istri	1

Apabila keluarga tidak ditemukan akan ditampilkan view not-found.html

Data Nomor = 31010100101050001 tidak ditemukan

[Kembali ke Halaman Utama](#)

- Fitur 3: Menambahkan Penduduk Baru Sebagai Anggota Keluarga  
User dapat menambahkan penduduk baru sebagai anggota keluarga dengan menekan tombol "Tambah Penduduk" pada halaman index.

Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#)

[Tambah Penduduk](#)

Cari NIK

[Lihat](#)

Pada controller akan melaksanakan method addPenduduk().

```
@RequestMapping(value="/penduduk/tambah", method = RequestMethod.GET)
public String addPenduduk (Model model, PendudukModel penduduk)
{
    return "form-tambah-penduduk";
}
```

Method `addPenduduk` akan mengembalikan formulir tambah penduduk. Misalkan user mengisi data sebagai berikut

Nama	<input type="text" value="Aku"/>
Tempat Lahir	<input type="text" value="Jakartaeeee"/>
Tanggal Lahir	<input type="text" value="101-00-01"/>
Jenis Kelamin	<input type="text" value="Pilih Jenis Kelamin"/>
Status Dalam Keluarga	<input type="text"/>
Golongan Darah	<input type="text" value="Pilih Golongan Darah"/>
Agama	<input type="text"/>
Status Perkawinan	<input type="text" value="Pilih Status Perkawinan"/>
Pekerjaan	<input type="text"/>
Kewarganegaraan	<input type="text" value="Pilih Kewarganegaraan"/>
Status Kematian	<input type="text" value="Pilih Status Kematian"/>
Id Keluarga	<input type="text"/>
<input type="button" value="Submit"/>	

Pada saat submit akan dipanggil method `addPendudukSubmit` pada controller yang memiliki RequestMapping `/penduduk/tambah` dengan method request POST. Method tersebut akan melakukan validasi data yang dimasukkan. Pengecekan dengan menggunakan `BindingResult`

```
if(bindingResult.hasErrors()) {
    model.addAttribute("pendudukModel", penduduk);
    return "form-tambah-penduduk";
}
```

Apabila binding result memiliki error seperti yang sudah didefinisikan pada `PendudukModel`, maka akan terjadi error. Berikut adalah beberapa validasi yang didefinisikan di `PendudukModel`.

```
@NotNull(message="Wajib diisi!")
@Size(min=1, message="Wajib diisi!")
private String nama;

@NotNull(message="Wajib diisi!")
@Size(min=1, message="Wajib diisi!")
private String tempat_lahir;

@NotNull(message="Wajib diisi!")
@Pattern(regexp = "([12]\\d{3})-(0[1-9]|1[0-2])-(0[1-9]|1[2]\\d|3[01])", message = "Format tttt-bb-hh")
private String tanggal_lahir;

@NotNull(message="Wajib diisi!")
private Integer jenis_kelamin;

@NotNull(message="Wajib diisi!")
private Integer is_wni;
```

Apabila terdapat data yang tidak valid maka method `addPendudukSubmit()` akan mengembalikan view `form-tambah-penduduk` dan mengembalikan tampilan berikut.

Tanggal Lahir	<input type="text" value="101-00-01"/>
	Format tttt-bb-hh
Jenis Kelamin	<input type="text" value="Pilih Jenis Kelamin"/>
	Wajib diisi!
Status Dalam Keluarga	<input type="text"/>
	Wajib diisi!
Golongan Darah	<input type="text" value="Pilih Golongan Darah"/>
	Wajib diisi!
Agama	<input type="text"/>
	Wajib diisi!
Status Perkawinan	<input type="text" value="Pilih Status Perkawinan"/>
	Wajib diisi!
Pekerjaan	<input type="text"/>
	Wajib diisi!
Kewarganegaraan	<input type="text" value="Pilih Kewarganegaraan"/>
	Wajib diisi!

Apabila user memasukkan keluarga dengan id yang tidak ada, maka akan keluar warning berikut.

Id Keluarga	<input type="text" value="0"/>
	Keluarga dengan ID 0 tidak terdaftar

Berikut adalah cara validasi id keluarga yang tidak ada pada method addPendudukSubmit.

```
String errorArgs;
KeluargaModel keluarga = sidukDAO.selectKeluarga(penduduk.getId_keluarga());
if(keluarga==null){
    errorArgs= "Keluarga dengan ID " + penduduk.getId_keluarga() + " tidak ada";
    model.addAttribute("pendudukModel",penduduk);
    model.addAttribute("errorIDArgs", errorArgs);
    return "form-tambah-penduduk";
}
```

Nama	<input type="text" value="Aku"/>
Tempat Lahir	<input type="text" value="Jakartaeeee"/>
Tanggal Lahir	<input type="text" value="2017-12-12"/> Format tttt-bb-hh
Jenis Kelamin	<input type="text" value="Wanita"/> Wajib diisi!
Status Dalam Keluarga	<input type="text" value="Anak"/> Wajib diisi!
Golongan Darah	<input type="text" value="A+"/> Wajib diisi!
Agama	<input type="text" value="islam"/> Wajib diisi!
Status Perkawinan	<input type="text" value="Belum Kawin"/> Wajib diisi!
Pekerjaan	<input type="text" value="Gabut"/> Wajib diisi!
Kewarganegaraan	<input type="text" value="WNI"/> Wajib diisi!
Status Kematian	<input type="text" value="Hidup"/> Wajib diisi!
Id Keluarga	<input type="text" value="1"/> Wajib diisi!

Karena data diatas sudah valid maka method addPendudukSubmit akan melakukan penambahan penduduk.

```

String kodeKecamatan = keluarga.getKelurahan().getKecamatan().getKode_kecamatan().substring(0, 6);
Date ymd = null;
try {
    ymd = new SimpleDateFormat("yyyy-MM-dd").parse(penduduk.getTanggal_lahir());
} catch (Exception e) {

}

String tglLahir = new SimpleDateFormat("dd-MM-yyyy").format(ymd);
String nikTglLahir = tglLahir.replace("-", "");
nikTglLahir = nikTglLahir.substring(0,4) + nikTglLahir.substring(6,8);

if(penduduk.getJenis_kelamin() == 1){
    Integer tanggal = Integer.parseInt(nikTglLahir.substring(0,2));
    tanggal +=40;
    nikTglLahir = String.valueOf(tanggal) + nikTglLahir.substring(2,6);
}

String nikMin= kodeKecamatan + nikTglLahir + "0001";
String nikMax= kodeKecamatan + nikTglLahir + "9999";
PendudukModel cekPenduduk = sidukDAO.selectPendudukTerakhir(nikMin, nikMax);

if(cekPenduduk == null){
    penduduk.setNik(nikMin);
    sidukDAO.addPenduduk(penduduk);
    model.addAttribute("nomor", nikMin);
}else{
    Long nikBaru = Long.parseLong(cekPenduduk.getNik());
    String nikPendudukBaru = String.valueOf(nikBaru+1);
    penduduk.setNik(nikPendudukBaru);
    sidukDAO.addPenduduk(penduduk);
    model.addAttribute("nomor", nikPendudukBaru);
}
return "sukses-tambah-penduduk";

```

Diawali dengan method addPendudukSubmit akan mengambil kode kecamatan dari keluarga (cara mengambil keluarga lihat pada screenshot validasi id keluarga). Kemudian setelah mendapatkan kode kecamatan, akan dilakukan reformat tanggal lahir penduduk, untuk menghilangkan '-' dan hanya mengambil 2 angka terakhir pada tahun. Kemudian akan dilakukan pengecekan untuk penduduk berjenis kelamin wanita, apabila penduduk memiliki jenis kelamin wanita maka tanggal lahir akan ditambah 40. Setelah itu akan dibuat variabel nik minimal dan nik maksimal, kedua variabel tersebut akan dijadikan parameter pada method selectPendudukTerakhir() pada kelas sidukServiceDatabase.java.

```

@Override
public PendudukModel selectPendudukTerakhir(String nikMin, String nikMax) {
    return sidukMapper.selectPendudukTerakhir(nikMin, nikMax);
}

```

Method ini berfungsi untuk mengecek nik penduduk dengan kode kecamatan, tanggal lahir dan nomor urut terakhir dengan memanggil method selectPendudukTerakhir pada kelas sidukMapper.

```

@Select("select * "
        + "from penduduk "
        + "where nik BETWEEN #{nikMin} and #{nikMax}"
        + "order by nik "
        + "DESC LIMIT 1 ")
PendudukModel selectPendudukTerakhir(@Param("nikMin")String nikMin,@Param("nikMax") String nikMax);

```

Apabila selectPendudukTerakhir mengembalikan nilai null hal ini menandakan bahwa nik dengan kode kecamatan dan tanggal lahir belum ada pada database, sehingga nilai variabel nikMin akan menjadi NIK penduduk tersebut. Namun apabila penduduk dengan kode kecamatan dan tanggal lahir tersebut sudah ada, maka mapper akan mengembalikan

penduduk dengan nik nomor urut terakhir. Kemudian pada controller, nik penduduk dengan nomor urut terakhir akan ditambah 1.

Setelah itu penduduk dengan nik tersebut serta data yang dimasukkan melalui form tadi akan ditambahkan ke database, dan method akan mengembalikan view bahwa penambahan penduduk sukses. User juga melihat detail penduduk yang baru dimasukkan dengan menekan tombol lihat data

## Sukses!

Penduduk dengan NIK = 3101015212170001 berhasil ditambahkan!

Lihat Data

4. Fitur 4: Menambahkan Keluarga Baru Sebagai Anggota Keluarga  
User dapat menambahkan keluarga baru sebagai anggota keluarga dengan menekan tombol "Tambah Keluarga" pada halaman index.

Tambah Keluarga

Cari NKK

Lihat

Pada controller akan melaksanakan method addKeluarga().

```
@RequestMapping(value="/keluarga/tambah", method = RequestMethod.GET)
public String addKeluarga (Model model, KeluargaModel keluarga)
{
    List<KotaModel> daftarKota = sidukDAO.selectDaftarKota();
    List<KecamatanModel> daftarKecamatan = sidukDAO.selectDaftarKecamatan();
    List<KelurahanModel> daftarKelurahan = sidukDAO.selectDaftarKelurahan();

    model.addAttribute("daftarKota", daftarKota);
    model.addAttribute("daftarKecamatan", daftarKecamatan);
    model.addAttribute("daftarKelurahan", daftarKelurahan);
    return "form-tambah-keluarga";
}
```

## Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#) / Register Keluarga

Alamat

Wajib diisi!

RT

Wajib diisi!

RW

Wajib diisi!

Apabila isian data tidak valid, maka akan dikeluarkan argumen wajib diisi, validasi dihandle pada BindingResult.

```
@RequestMapping(value= "/keluarga/tambah", method = RequestMethod.POST)
public String addKeluargaSubmit (Model model, @Valid KeluargaModel keluarga, BindingResult bindingResult,
    @RequestParam(value = "kota") Integer idKota,
    @RequestParam(value = "kecamatan") Integer idKecamatan,
    @RequestParam(value = "kelurahan") Integer idKelurahan){

    if(bindingResult.hasErrors()) {
        return "form-tambah-keluarga";
    }
}
```

Binding result mengambil argumen dari keluargaModel

```
public class KeluargaModel {

    private Integer id;
    private String nomor_kk;

    @NotNull(message="Wajib diisi!")
    @Size(min=1, message="Wajib diisi!")
    private String alamat,rt, rw;

    private Integer id_kelurahan;
    private Integer is_tidak_berlaku;
    private KelurahanModel kelurahan;
    private List<PendudukModel> anggotaKeluarga;
}
```

Method addKeluarga akan mengambil terlebih dahulu seluruh daftar kota, kecamatan dan kelurahan pada database dan mengembalikan view form-tambah-keluarga.html. daftar kota, kecamatan dan kelurahan tersebut berguna untuk drop down pada field kota, kecamatan dan kelurahan. Field kota akan menampilkan seluruh kota, saat kota dipilih, field kecamatan akan menampilkan daftar kecamatan pada kota tersebut, kemudian field kelurahan akan menampilkan daftar kelurahan sesuai pada kecamatan dan kota yang dipilih. Sehingga dapat dikatakan kecamatan dan kelurahan merupakan dependency dropdown. Setelah itu akan dibuat variabel nkk minimal dan nkk maksimal, kedua variabel tersebut akan dijadikan parameter pada method selectPendudukTerakhir() pada kelas sidukServiceDatabase.java.



Alamat	<input type="text" value="Jl. gitta"/>
RT	<input type="text" value="14"/>
RW	<input type="text" value="09"/>
Kota	<input type="text" value="KOTA JAKARTA SELATAN"/>
Kecamatan	<input type="text" value="PASAR MINGGU"/>
Kelurahan	<input type="text" value="PASAR MINGGU"/>

Submit

Berikut adalah contoh data yang valid, saat user menekan tombol submit, maka akan dipanggil method `addKeluargaSubmit()`.

```

Date date = new Date();
String currentDate = new SimpleDateFormat("dd-MM-yyyy").format(date);
currentDate= currentDate.replace("-", "");
currentDate= currentDate.substring(0,4) + currentDate.substring(6,8);
KelurahanModel kelurahan = sidukDAO.selectKelurahan(idKelurahan);
String kodeKecamatan = kelurahan.getKecamatan().getKode_kecamatan().substring(0, 6);

String nkkMin= kodeKecamatan + currentDate + "0001";
String nkkMax= kodeKecamatan + currentDate + "9999";
KeluargaModel cekKeluarga = sidukDAO.selectKeluargaTerakhir(nkkMin, nkkMax);

if(cekKeluarga == null){
    keluarga.setNomor_kk(nkkMin);
    model.addAttribute("nomor", nkkMin);
}else{
    Long nkkBaru = Long.parseLong(cekKeluarga.getNomor_kk());
    String nkkKeluargaBaru = String.valueOf(nkkBaru+1);
    keluarga.setNomor_kk(nkkKeluargaBaru);
    model.addAttribute("nomor", nkkKeluargaBaru);
}

keluarga.setId_kelurahan(idKelurahan);
sidukDAO.addKeluarga(keluarga);
return "sukses-tambah-keluarga";

```

Pada method `addKeluargaSubmit`, pertama akan dibuat terlebih dahulu variable `date` yang menyimpan tanggal terkini, kemudian diubah formatnya dengan menghilangkan '-' dan mengilangkan 2 digit pertama pada tahun. Setelah itu akan diambil kelurahan sesuai id kelurahan yang dimasukkan pada dropdown. Setelah itu akan dibuat variabel `nkk` minimal dan `nkk` maksimal, kedua variabel tersebut akan dijadikan parameter pada method `selectKeluargaTerakhir()` pada kelas `sidukServiceDatabase.java`.

```

@Override
public KeluargaModel selectKeluargaTerakhir(String nkkMin, String nkkMax) {
    return sidukMapper.selectKeluargaTerakhir(nkkMin, nkkMax);
}

```

Method ini berfungsi untuk mengecek `nkk` keluarga dengan kode kecamatan, tanggal terkini dan nomor urut terakhir dengan memanggil method `selectKeluargaTerakhir` pada kelas `sidukMapper`.

```
@Select("select * "
      + "from keluarga "
      + "where nomor_kk BETWEEN #{nkkMin} and #{nkkMax}"
      + "order by nomor_kk "
      + "DESC LIMIT 1 ")
KeluargaModel selectKeluargaTerakhir(@Param("nkkMin")String nkkMin,@Param("nkkMax") String nkkMax);
```

Apabila selectKeluargaTerakhir mengembalikan nilai null hal ini menandakan bahwa nkk dengan kode kecamatan dan tanggal terkini belum ada pada database, sehingga nilai variabel nkkMin akan menjadi NKK keluarga tersebut. Namun apabila keluarga dengan kode kecamatan dan tanggal terkini tersebut sudah ada, maka mapper akan mengembalikan keluarga dengan nkk nomor urut terakhir. Kemudian pada controller, nkk keluarga dengan nomor urut terakhir akan ditambah 1.

Setelah itu keluarga dengan nkk tersebut serta data yang dimasukkan melalui form tadi akan ditambahkan ke database, dan method akan mengembalikan view bahwa penambahan keluarga sukses. . User juga melihat detail keluarga yang baru dimasukkan dengan menekan tombol lihat data

## Sukses!

**Keluarga dengan NKK = 3171022210170002 berhasil ditambahkan!**

Lihat Data

### 5. Fitur 5 : Mengubah Data Penduduk

User dapat mengubah penduduk dengan memasukkan nik penduduk pada field cari NIK yang akan mengembalikan detail penduduk dan menekan tombol Ubah Data Penduduk.

**Lihat Data Penduduk - 3101012407110001**

NIK	3101012407110001
Nama	Heru Haryanto
Tempat/Tanggal Lahir	Jakarta, 2011-07-24
Jenis Kelamin	Pria
Alamat	Ds. Adisumarmo No. 43
RT/RW	079/025
Kelurahan	PULAU TIDUNG
Kecamatan	KEPULAUAN SERIBU SELATAN
Kota	KABUPATEN KEPULAUAN SERIBU
Golongan Darah	A+
Agama	Islam
Status Perkawinan	Belum Kawin
Pekerjaan	BELUM/TIDAK BEKERJA
Kewarganegaraan	WNI
Status Kematian	Hidup

Ubah Status Kematian

Ubah Data Penduduk

Pada saat tombol 'Ubah Data Penduduk' ditekan, maka akan dipanggil method `updateKeluarga()` pada controller.

```
@RequestMapping(value= "/penduduk/ubah/{nik}", method = RequestMethod.GET)
public String updatePenduduk (Model model, @PathVariable(value = "nik") String nik)
{
    PendudukModel penduduk = sidukDAO.selectPenduduk(nik);

    if(penduduk==null){
        model.addAttribute("nomor",nik);
        return "error/not-found";
    }else{
        model.addAttribute("pendudukModel", penduduk);
    }

    return "form-update-penduduk";
}
```

Method tersebut melakukan validasi apabila user mengakses halaman ubah penduduk melalui url namun nik null, maka akan mengembalikan halaman berikut.

## Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#)

Data Nomor = 31010124071100011 tidak ditemukan

[Kembali ke Halaman Utama](#)

Namun apabila penduduk tidak null maka method akan mengembalikan formulir ubah penduduk yang sudah otomatis terisi data penduduk tersebut (autofill).

Nama	<input type="text" value="Heru Haryanto"/>
Tempat Lahir	<input type="text" value="Jakarta"/>
Tanggal Lahir	<input type="text" value="2011-07-24"/>
Jenis Kelamin	<input type="text" value="Pria"/>
Status Dalam Keluarga	<input type="text" value="Anak"/>
Golongan Darah	<input type="text" value="A+"/>
Agama	<input type="text" value="Islam"/>
Status Perkawinan	<input type="text" value="Belum Kawin"/>
Pekerjaan	<input type="text" value="BELUM/TIDAK BEKERJA"/>
Kewarganegaraan	<input type="text" value="WNI"/>
Status Kematian	<input type="text" value="Hidup"/>
Id Keluarga	<input type="text" value="172"/>

Pada saat submit, method `updatePendudukSubmit` pada controller akan melakukan validasi terlebih dahulu, validasi **sama persis** dengan fitur 3 yakni validasi form dan id keluarga terdaftar. Apabila data telah valid, maka method `addPendudukSubmit` akan dieksekusi lebih lanjut.

```

PendudukModel arsipPenduduk = sidukDAO.selectPenduduk(nik);
penduduk.setId(arsipPenduduk.getId());
if(arsipPenduduk.getTanggal_lahir() == penduduk.getTanggal_lahir()){
    String nikBaru;

    if(penduduk.getJenis_kelamin() != arsipPenduduk.getJenis_kelamin()){
        if(penduduk.getJenis_kelamin() == 1){
            Integer tanggal = Integer.parseInt(penduduk.getNik().substring(6,8));
            tanggal +=40;
            nikBaru = penduduk.getNik().substring(0, 6)+String.valueOf(tanggal)+penduduk.getNik().substring(8, 16);
        }else{
            Integer tanggal = Integer.parseInt(penduduk.getNik().substring(6,8));
            tanggal -=40;
            nikBaru = penduduk.getNik().substring(0, 6)+String.valueOf(tanggal)+penduduk.getNik().substring(8, 16);
        }
        penduduk.setNik(nikBaru);
    }

    sidukDAO.updatePenduduk(penduduk);
}else{
    Date ymd = null;
    try {
        ymd = new SimpleDateFormat("yyyy-MM-dd").parse(penduduk.getTanggal_lahir());
    } catch (Exception e) {
    }

    String tgllahir = new SimpleDateFormat("dd-MM-yyyy").format(ymd);
    String nikTgllahir = tgllahir.replace("-", "");
    nikTgllahir = nikTgllahir.substring(0,4) + nikTgllahir.substring(6,8);

    if(penduduk.getJenis_kelamin() == 1){
        Integer tanggal = Integer.parseInt(nikTgllahir.substring(0,2));
        tanggal +=40;
        nikTgllahir = String.valueOf(tanggal) + nikTgllahir.substring(2,6);
    }
    String nikMin= penduduk.getNik().substring(0, 6) + nikTgllahir + "0001";
    String nikMax= penduduk.getNik().substring(0, 6) + nikTgllahir + "9999";
    PendudukModel cekPenduduk = sidukDAO.selectPendudukTerakhir(nikMin, nikMax);
}

```

Diawali dengan mengambil data tersebut dengan menggunakan method selectPenduduk(nik) pada kelas sidukServiceDatabase.java

```

@Override
public PendudukModel selectPenduduk(String nik) {
    log.info ("select penduduk with nik {}", nik);
    return sidukMapper.selectPenduduk (nik);
}

```

Kemudian akan dilakukan pengecekan apakah user mengubah tanggal lahir penduduk, jika tidak akan dilakukan pengecekan apakah user mengganti jenis kelamin penduduk, jika jenis kelamin berubah maka akan dikurangi 40 untuk perubahan dari wanita menjadi pria, dan ditambah 40 untuk pria menjadi wanita. Namun jika tanggal lahir penduduk diperbarui, maka proses pembuatan nik akan sama persis seperti saat menambah data penduduk baru pada fitur 3. Lalu setelah nik diperbarui maka akan dilakukan perbaruan data usia dengan form yang diisi pada kelas mapper.

```

@update("update penduduk set nik = #{nik}, nama= #{nama}, tempat_lahir = #{tempat_lahir}, tanggal_lahir = #{tanggal_lahir}, "
+ "jenis_kelamin = #{jenis_kelamin}, is_wni = #{is_wni}, is_wafat = #{is_wafat}, "
+ "id_keluarga = #{id_keluarga}, agama = #{agama}, pekerjaan = #{pekerjaan}, "
+ "status_perkawinan = #{status_perkawinan}, status_dalam_keluarga = #{status_dalam_keluarga}, golongan_darah = #{golongan_darah}"
+ "WHERE id=#{id}")
void updatePenduduk(PendudukModel penduduk);

```

Dan akan mengembalikan view sukses.

## Sukses!

Penduduk dengan NIK = 3101012407110002 berhasil diubah!

## 6. Fitur 6: Mengubah Data Keluarga

User dapat mengubah keluarga dengan memasukkan nkk penduduk pada field cari NKK yang akan mengembalikan detail keluarga dan menekan tombol Ubah Data Keluarga.

Sistem Kependudukan Online Provinsi DKI Jakarta

Home / Data Keluarga

Lihat Data Keluarga - 3171022210170001

NKK = 3171022210170001

Alamat = Jln. Cikapayang No. 594

RT/RW = 091/090

Kelurahan/Desa = PULAU PARI

Kecamatan = KEPULAUAN SERIBU SELATAN

Kota = KABUPATEN KEPULAUAN SERIBU

Nama Lengkap	NIK	Jenis Kelamin	Tempat Lahir	Tanggal Lahir	Agama	Pekerjaan	Status Perkawinan	Status dalam Keluarga	Kewarganegaraan
Heru Haryanto	3101012407110003	0	Jakarta	2011-07-24	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	Anak	0
Taswir Rajata	3171020303890001	0	Jakarta	1989-03-03	Islam	KARYAWAN HONORER	Kawin	Kepala Keluarga	1
Mulyono Hutagalung S.Gz	3171021408150001	0	Jakarta	2015-08-14	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	Anak	1
Yuliana Wahyuni	3171026312160001	1	Jakarta	2016-12-23	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	Anak	1
Ifa Novitasari	3171024712830001	1	Jakarta	1983-12-07	Islam	PETANI/PEKEBUN	Kawin	Istri	1
Aku	3101015212170001	1	Jakarta	2017-12-12	Islam	Gabut	Belum Kawin	Anak	0

Ubah Data Keluarga

Pada saat tombol 'Ubah Data Keluarga' ditekan, maka akan dipanggil method `updateKeluarga()` pada controller.

```
@RequestMapping(value= "/keluarga/ubah/{nkk}", method = RequestMethod.GET)
public String updateKeluarga (Model model, @PathVariable(value = "nkk") String nkk)
{
    KeluargaModel keluarga = sidukDAO.selectKartuKeluarga(nkk);
    if(keluarga==null){
        model.addAttribute("nomor",nkk);
        return "error/not-found";
    }else{
        List<KotaModel> daftarKota = sidukDAO.selectDaftarKota();
        List<KecamatanModel> daftarKecamatan = sidukDAO.selectDaftarKecamatan();
        List<KecamatanModel> daftarKecamatanSaya = new ArrayList<>();
        List<Integer> temp = new ArrayList<>();
        for(KecamatanModel e: daftarKecamatan) {
            if (e.getId_kota() == keluarga.getKelurahan().getKecamatan().getId_kota()) {
                daftarKecamatanSaya.add(e);
                temp.add(e.getId());
            }
        }
        List<KelurahanModel> daftarKelurahan = sidukDAO.selectDaftarKelurahan();
        List<KelurahanModel> daftarKelurahanSaya = new ArrayList<>();
        for(KelurahanModel e: daftarKelurahan) {
            if (temp.contains(e.getId_kecamatan())) {
                daftarKelurahanSaya.add(e);
            }
        }
        model.addAttribute("daftarKota", daftarKota);
        model.addAttribute("daftarKecamatan", daftarKecamatan);
        model.addAttribute("daftarKecamatanSaya", daftarKecamatanSaya);
        model.addAttribute("daftarKelurahan", daftarKelurahan);
        model.addAttribute("daftarKelurahanSaya", daftarKelurahanSaya);
        model.addAttribute("keluargaModel", keluarga);
    }
}
```

Method tersebut melakukan validasi apabila user mengakses halaman ubah keluarga melalui url namun nik null, maka akan mengembalikan halaman berikut.

## Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#)

Data Nomor = 3171022210170003 tidak ditemukan

[Kembali ke Halaman Utama](#)

Namun apabila keluarga tidak null maka method akan mengembalikan formulir ubah penduduk yang sudah otomatis terisi data penduduk tersebut (autofill).

Alamat	<input type="text" value="Jl. gitta"/>
RT	<input type="text" value="14"/>
RW	<input type="text" value="09"/>
Kota	<input type="text" value="KOTA JAKARTA SELATAN"/>
Kecamatan	<input type="text" value="PASAR MINGGU"/>
Kelurahan	<input type="text" value="PASAR MINGGU"/>
<input type="button" value="Submit"/>	

Pada saat submit, method updateKeluargaSubmit pada controller akan melakukan validasi yang **sama persis** seperti fitur 4 yakni validasi binding result.

[Home](#) / [Update Keluarga](#)

Alamat

RT

Wajib diisi!

RW

```
@RequestMapping(value= "/keluarga/ubah/{nkk}", method = RequestMethod.POST)
public String updateKeluargaSubmit (@PathVariable(value="nkk") String nkk, Model
    @RequestParam(value = "kota") Integer id_kota,
    @RequestParam(value = "kecamatan") Integer id_kecamatan,
    @RequestParam(value = "kelurahan") Integer id_kelurahan)
{
    if(bindingResult.hasErrors()) {
        return "form-update-keluarga";
    }
}
```

Apabila data valid, proses pengubahan nkk hampir sama dengan proses pembuatan nkk pada fitur 4, bedanya apabila terjadi perubahan kota/kecamatan/kelurahan akan dilakukan looping pada seluruh anggota keluarga dan dilakukan pengecekan kembali nik (sama dengan update nik pada fitur 5).

```

if(id_kecamatan != arsipKeluarga.getKelurahan().getKecamatan().getId()){
    KelurahanModel kelurahan = sidukDAO.selectKelurahan(id_kelurahan);
    String kodeKecamatanBaru = kelurahan.getKecamatan().getKode_kecamatan().substring(0, 6);
    String nkkMin= kodeKecamatanBaru + currentDate+ "0001";
    String nkkMax= kodeKecamatanBaru + currentDate + "9999";

    KeluargaModel cekKeluarga = sidukDAO.selectKeluargaTerakhir(nkkMin, nkkMax);

    if(cekKeluarga == null){
        arsipKeluarga.setNomor_kk(nkkMin);
    }else{
        Long nkkBaru = Long.parseLong(cekKeluarga.getNomor_kk());
        String nkkKeluargaBaru = String.valueOf(nkkBaru+1);
        arsipKeluarga.setNomor_kk(nkkKeluargaBaru);
    }
    List<PendudukModel> anggotaKeluarga = sidukDAO.selectAnggotaKeluarga(nkk);
    for(int i = 0; i< anggotaKeluarga.size();i++){
        String nikMin= kodeKecamatanBaru + anggotaKeluarga.get(i).getNik().substring(6, 12) + "0001";
        String nikMax= kodeKecamatanBaru + anggotaKeluarga.get(i).getNik().substring(6, 12) + "9999";
        PendudukModel cekPenduduk = sidukDAO.selectPendudukTerakhir(nikMin, nikMax);
        if(cekPenduduk == null){
            anggotaKeluarga.get(i).setNik(nikMin);
            sidukDAO.updatePenduduk(anggotaKeluarga.get(i));
        }else{
            Long nikBaru = Long.parseLong(cekPenduduk.getNik());
            String nikPendudukBaru = String.valueOf(nikBaru+1);
            anggotaKeluarga.get(i).setNik(nikPendudukBaru);
            sidukDAO.updatePenduduk(anggotaKeluarga.get(i));
        }
    }
}
}else{
    String nkkUpdateDate = arsipKeluarga.getNomor_kk().substring(0, 6) + currentDate + arsipKeluarga.getNomor_kk().substri
    String nkkMin= nkkUpdateDate.substring(0, 12) + "0001";
    String nkkMax= nkkUpdateDate.substring(0, 12) + "9999";
}

```

Query perbarui data keluarga

```

@update("update keluarga set nomor_kk = #{nomor_kk}, alamat= #{alamat}, rt = #{rt}, rw = #{rw}, "
      + "id_kelurahan = #{id_kelurahan}, is_tidak_berlaku = #{is_tidak_berlaku} "
      + "WHERE id=#{id}")
void updateKeluarga(KeluargaModel keluarga);

```

## 7. Fitur 7: Mengubah Status Kematian Penduduk

User dapat mengubah penduduk dengan memasukkan nik penduduk pada field cari NIK yang akan mengembalikan detail penduduk dan menekan tombol Ubah Status Kematian.

### Lihat Data Penduduk - 3101012407110001

NIK	3101012407110001
Nama	Heru Haryanto
Tempat/Tanggal Lahir	Jakarta, 2011-07-24
Jenis Kelamin	Pria
Alamat	Ds. Adisumarmo No. 43
RT/RW	079/025
Kelurahan	PULAU TIDUNG
Kecamatan	KEPULAUAN SERIBU SELATAN
Kota	KABUPATEN KEPULAUAN SERIBU
Golongan Darah	A+
Agama	Islam
Status Perkawinan	Belum Kawin
Pekerjaan	BELUM/TIDAK BEKERJA
Kewarganegaraan	WNI
Status Kematian	Hidup

[Ubah Status Kematian](#)[Ubah Data Penduduk](#)

Saat ditekan tombol Ubah Status kematian, status kematian penduduk akan diubah menjadi wafat, dengan memanggil method setStatusKematian pada controller.

```
@RequestMapping(value= "/penduduk/{nik}", method=RequestMethod.GET)
public String setStatusKematian (Model model,
    @PathVariable(value = "nik") String nik)
{
    if(nik==""){
        String errorArgs= "NIK harus diisi!";
        model.addAttribute("errorNIKArgs", errorArgs);
        return "index";
    }
    if(!nik.matches("\\d*")){
        String errorArgs= "NIK tidak boleh mengandung huruf!";
        model.addAttribute("errorNIKArgs", errorArgs);
        return "index";
    }
}
```

Pada method tersebut akan dilakukan pengecekan terlebih dahulu apakah penduduk ada atau tidak. Pada fitur ini juga dilakukan validasi agar apabila user mengakses dari url dan nik tidak valid seperti mengandung huruf atau null, akan dikembalikan ke halaman index, agar mencari nik melalui field dan memasukkan nik valid. Selain itu juga dilakukan validasi agar nik penduduk yang dicari ada. Berikut adalah validasinya pada method di controller.



```

PendudukModel penduduk = sidukDAO.selectPenduduk(nik);
if(penduduk==null){
    model.addAttribute("nomor",nik);
    return "error/not-found";
}

int wafat = 1;

penduduk.setIs_wafat(wafat);
sidukDAO.updatePenduduk(penduduk);

List<PendudukModel> anggotaKeluarga = sidukDAO.selectAnggotaKeluarga(penduduk.getKeluarga().getNomor_kk());
int jmlWafat=0;

for(int i = 0; i< anggotaKeluarga.size();i++){
    if(anggotaKeluarga.get(i).getIs_wafat() == wafat){
        jmlWafat++;
    }
}

int tidakBerlaku= 1;
if(jmlWafat == anggotaKeluarga.size()){
    penduduk.getKeluarga().setIs_tidak_berlaku(tidakBerlaku);
    sidukDAO.updateKeluarga(penduduk.getKeluarga());
}

model.addAttribute ("nomor", penduduk.getNik());
return "redirect:/penduduk/?nik=" + penduduk.getNik();
}

```

Bila penduduk null maka akan mengeluarkan page berikut.

[Home](#)

Data Nomor = 31010124071100031 tidak ditemukan

[Kembali ke Halaman Utama](#)

jika tidak maka akan dilakukan update penduduk dengan mengubah status is\_wafat menjadi 1, kemudian akan dilakukan pengecekan pada seluruh anggota keluarga penduduk, apabila anggota keluarga penduduk sudah wafat semua, maka status is tidak berlaku keluarga akan diubah menjadi 1. Ini adalah query untuk update kematian penduduk dan keluarga.

```

@update("update penduduk set nik = #{nik}, nama= #{nama}, tempat_lahir = #{tempat_lahir}, tanggal_lahir = #{tanggal_lahir}, "
+ "jenis_kelamin = #{jenis_kelamin}, is_wni = #{is_wni}, is_wafat = #{is_wafat}, "
+ "id_keluarga = #{id_keluarga}, agama = #{agama}, pekerjaan = #{pekerjaan}, "
+ "status_perkawinan = #{status_perkawinan}, status_dalam_keluarga = #{status_dalam_keluarga}, golongan_darah = #{golongan_darah}, "
+ "WHERE id=#{id}")
void updatePenduduk(PendudukModel penduduk);

@update("update keluarga set nomor_kk = #{nomor_kk}, alamat= #{alamat}, rt = #{rt}, rw = #{rw}, "
+ "id_kelurahan = #{id_kelurahan}, is_tidak_berlaku = #{is_tidak_berlaku} "
+ "WHERE id=#{id}")
void updateKeluarga(KeluargaModel keluarga);

```

Lihat Data Keluarga - 3101022010170001

NKK = 3101022010170001

Alamat = Jalan jalan

RT/RW = 02/03

Kelurahan/Desa = PULAU KELAPA

Kecamatan = KEPULAUAN SERIBU UTARA

Kota = KABUPATEN KEPULAUAN SERIBU

Status Berlaku =Tidak Berlaku

Nama Lengkap	NIK	Jenis Kelamin	Tempat Lahir	Tanggal Lahir	Agama	Pekerjaan	Status Perkawinan	Status dalam Keluarga	Kewarganegaraan	Status Hidup
bebas	3101021212170001	0	depok	2017-12-12	islam	Gabut	Belum Kawin	Anak	WNA	Mati

[Ubah Data Keluarga](#)

Pada halaman detail keluarga apabila penduduk telah dinyatakan wafat, maka tombol ubah status kematian akan hilang

[Home](#) / [Data Penduduk](#)

## Lihat Data Penduduk - 3101012407110003

NIK	3101012407110003
Nama	Heru Haryanto
Tempat/Tanggal Lahir	Jakartaa, 2011-07-24
Jenis Kelamin	Pria
Alamat	Jln. Cikapayang No. 594
RT/RW	091/090
Kelurahan	PULAU PARI
Kecamatan	KEPULAUAN SERIBU SELATAN
Kota	KABUPATEN KEPULAUAN SERIBU
Golongan Darah	A+
Agama	Islam
Status Perkawinan	Belum Kawin
Pekerjaan	BELUM/TIDAK BEKERJA
Kewarganegaraan	WNA
Status Kematian	Mati

[Ubah Data Penduduk](#)

8. Fitur 8: Tampilkan Penduduk berdasarkan kota/kabupaten, kecamatan dan kelurahan tertentu  
User dapat mencari penduduk tertentu dengan menekan tombol cari penduduk tertentu pada halaman index/ home.

[Tambah Penduduk](#)[Cari Penduduk Tertentu](#)

Cari NIK

Masukkan NIK

Lihat

Kemudian akan muncul tampilan sebagai berikut.

## Sistem Kependudukan Online Provinsi DKI Jakarta

[Home](#) / [Cari Penduduk](#)

Kota	<div>Pilih Kota</div>
Kecamatan	<div>Pilih Kecamatan</div>
Kelurahan	<div>Pilih Kelurahan</div>

Submit

Dropdown diatas menggunakan dependency dropdown, dimana kecamatan yg muncul pasti hanya kecamatan yang ada pada kota tersebut, dan kelurahan tersebut pasti hanya yang ada pada kecamatan tersebut.

```

@RequestMapping(value= "/penduduk/cari", method = RequestMethod.GET)
public String cariPendudukSubmit (Model model, @RequestParam(value = "kt") Optional<Integer> idKota,
    @RequestParam(value = "kc") Optional<Integer> idKecamatan,
    @RequestParam(value = "kl") Optional<Integer> idKelurahan)
{
    // Cari penduduknya
    if(idKota.isPresent() && idKecamatan.isPresent() && idKelurahan.isPresent()) {
        List <PendudukModel> listPenduduk = sidukDAO.selectListPenduduk(idKelurahan.get());

        PendudukModel pendudukTertua = sidukDAO.selectPendudukTertua(idKelurahan.get());
        PendudukModel pendudukTermuda = sidukDAO.selectPendudukTermuda(idKelurahan.get());

        model.addAttribute("pendudukTertua", pendudukTertua);
        model.addAttribute("pendudukTermuda", pendudukTermuda);
        model.addAttribute("listPenduduk", listPenduduk);

        return "list-penduduk";
    }
    // Return form pencarian
    else {
        List<KotaModel> daftarKota = sidukDAO.selectDaftarKota();
        List<KecamatanModel> daftarKecamatan = sidukDAO.selectDaftarKecamatan();
        List<KelurahanModel> daftarKelurahan = sidukDAO.selectDaftarKelurahan();

        model.addAttribute("daftarKota", daftarKota);
        model.addAttribute("daftarKecamatan", daftarKecamatan);
        model.addAttribute("daftarKelurahan", daftarKelurahan);
        return "form-cari-penduduk";
    }
}

```

Berikut adalah tampilannya menggunakan data table.

Penduduk Tertua

NIK	3101010611520001
Nama	Cahyanto Cemani Gunawan
Tempat/Tanggal Lahir	Jakarta, 1952-11-06

Penduduk Termuda

NIK	3101015212170001
Nama	Aku
Tempat/Tanggal Lahir	Jakartaeeeeee, 2017-12-12

Show

10

entries

Search:

No.	Nama Lengkap	NIK	Jenis Kelamin	Tempat Lahir	Tanggal Lahir	Agama	Pekerjaan	Status Perkawinan	Kewarganegaraan	Lihat Detail
1	Heru Haryanto	3101012407110003	Pria	Jakartaa	2011-07-24	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	WNA	
2	Taswir Rajata	3171020303890001	Pria	Jakarta	1989-03-03	Islam	KARYAWAN HONORER	Kawin	WNI	
3	Mulyono Hutagalung S.Gz	3171021408150001	Pria	Jakarta	2015-08-14	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	WNI	
4	Yuliana Wahyuni	3171026312160001	Wanita	Jakarta	2016-12-23	Islam	BELUM/TIDAK BEKERJA	Belum Kawin	WNI	

User dapat melihat detail penduduk dengan menekan tombol dengan simbol mata disebelah kanan.

#### 9. Validasi untuk semua form post

Dilakukan pada fitur 1,2,3,4,5,6,7. Sudah dijelaskan pada penjelasan fitur, Silakan dicek :)

#### 10. Menampilkan penduduk paling tua dan paling muda

Caranya terdapat pada fitur 8, (lihat method fitur 8), terdapat method select penduduk tertua dan termuda, berikut adalah querynya.

```

@Select("select * "
+ "from penduduk, keluarga "
+ "where keluarga.id_kelurahan= #{id_kelurahan} and penduduk.id_keluarga = keluarga.id "
+ "order by tanggal_lahir "
+ "LIMIT 1 ")
PendudukModel selectPendudukTertua(@Param("id_kelurahan") Integer id_kelurahan);

@Select("select * "
+ "from penduduk, keluarga "
+ "where keluarga.id_kelurahan= #{id_kelurahan} and penduduk.id_keluarga = keluarga.id "
+ "order by tanggal_lahir "
+ "DESC LIMIT 1 ")
PendudukModel selectPendudukTermuda(@Param("id_kelurahan") Integer id_kelurahan);

```

Dan berikut adalah tampilannya, (dapat dilihat saat mengakses fitur 8)

#### Penduduk Tertua

NIK	3172032312520002
Nama	Reksa Baktianto Salahudin
Tempat/Tanggal Lahir	Jakarta, 1952-12-23

#### Penduduk Termuda

NIK	3172031909170002
Nama	Cengkal Waluyo
Tempat/Tanggal Lahir	Surakarta, 2017-09-19

#### 11. Menambahkan error page

Terdapat error page 404.html untuk handle mengakses link yang tidak ada,



Serta page not-found apabila nkk/nik tidak ada pada database, contoh screenshot ada pada fitur-fitur 1-8. Ditambahkan juga 500.html untuk internal server error

## 12. Stress Testing

- Select Penduduk (Fitur 1)

**Thread Properties**

Number of Threads (users): 5000

Ramp-Up Period (in seconds): 50

Loop Count: ☐ Forever 1

---

**View Results in Table**

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename:   Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
4972	21:44:09.018	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4973	21:44:09.028	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4974	21:44:09.038	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4975	21:44:09.048	Thread Group ...	HTTP Request	3	✓	2406	147	3	0
4976	21:44:09.058	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4977	21:44:09.068	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4978	21:44:09.078	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4979	21:44:09.088	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4980	21:44:09.099	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4981	21:44:09.109	Thread Group ...	HTTP Request	4	✓	2406	147	4	0
4982	21:44:09.119	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4983	21:44:09.129	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4984	21:44:09.139	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4985	21:44:09.149	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4986	21:44:09.159	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4987	21:44:09.170	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4988	21:44:09.179	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4989	21:44:09.189	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4990	21:44:09.199	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4991	21:44:09.210	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4992	21:44:09.220	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4993	21:44:09.230	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4994	21:44:09.240	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4995	21:44:09.250	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4996	21:44:09.260	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4997	21:44:09.270	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4998	21:44:09.280	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
4999	21:44:09.290	Thread Group ...	HTTP Request	4	✓	2406	147	4	1
5000	21:44:09.300	Thread Group ...	HTTP Request	5	✓	2406	147	5	1

☒ Scroll automatically? ☐ Child samples? No of Samples 5000 Latest Sample 5 Average 9 Deviation 44

Fitur 1 reliable pada 5000 thread dalam waktu 50 detik, dengan rata rata response time 9ms/thread. Hal ini termasuk cepat.

- Select keluarga (Fitur 2)

**Thread Properties**  
**Number of Threads (users):** 5000  
**Ramp-Up Period (in seconds):** 50  
**Loop Count:** ☐ Forever 1

**View Results in Table**  
 Name: View Results in Table  
 Comments:  
 Write results to file / Read from file  
 Filename:   **Log/Display Only:** ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
4972	21:42:12.578	Thread Group ...	HTTP Request	4	✓	3500	147	4	0
4973	21:42:12.597	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4974	21:42:12.602	Thread Group ...	HTTP Request	4	✓	3500	147	4	0
4975	21:42:12.608	Thread Group ...	HTTP Request	4	✓	3500	147	4	1
4976	21:42:12.628	Thread Group ...	HTTP Request	6	✓	3500	147	6	1
4977	21:42:12.632	Thread Group ...	HTTP Request	7	✓	3500	147	7	2
4978	21:42:12.642	Thread Group ...	HTTP Request	4	✓	3500	147	4	0
4979	21:42:12.648	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4980	21:42:12.659	Thread Group ...	HTTP Request	5	✓	3500	147	5	0
4981	21:42:12.662	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4982	21:42:12.678	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4983	21:42:12.688	Thread Group ...	HTTP Request	4	✓	3500	147	4	1
4984	21:42:12.692	Thread Group ...	HTTP Request	4	✓	3500	147	4	1
4985	21:42:12.708	Thread Group ...	HTTP Request	4	✓	3500	147	4	1
4986	21:42:12.712	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4987	21:42:12.719	Thread Group ...	HTTP Request	4	✓	3500	147	4	1
4988	21:42:12.729	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4989	21:42:12.739	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4990	21:42:12.742	Thread Group ...	HTTP Request	6	✓	3500	147	6	1
4991	21:42:12.749	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4992	21:42:12.759	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4993	21:42:12.772	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4994	21:42:12.779	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4995	21:42:12.789	Thread Group ...	HTTP Request	4	✓	3500	147	4	1
4996	21:42:12.809	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4997	21:42:12.820	Thread Group ...	HTTP Request	5	✓	3500	147	5	1
4998	21:42:12.830	Thread Group ...	HTTP Request	6	✓	3500	147	6	1
4999	21:42:12.842	Thread Group ...	HTTP Request	4	✓	3500	147	4	1
5000	21:42:12.873	Thread Group ...	HTTP Request	4	✓	3500	147	4	0

☒ Scroll automatically? ☐ Child samples? **No of Samples** 5000 **Latest Sample** 4 **Average** 7 **Deviation** 23

Fitur 2 reliable pada 5000 thread dalam waktu 50 detik, dengan rata rata response time 7ms/thread. Hal ini termasuk cepat

- Cari penduduk, fitur 8

**Thread Properties**

Number of Threads (users):

Ramp-Up Period (in seconds):

Loop Count: ☐ Forever

---

**View Results in Table**

Name:

Comments:

Write results to file / Read from file

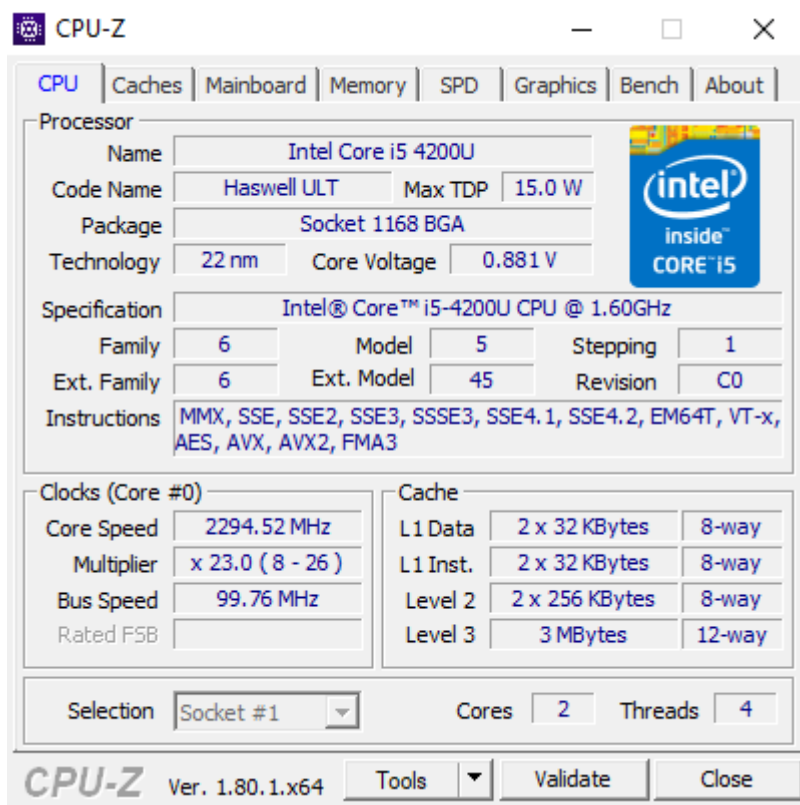
Filename:   Log/Display Only: ☐ Errors ☐ Successes

Sample #	Start Time	Thread Name	Label	Sample Time(...)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
42	22:50:25.768	Thread Group ...	HTTP Request	199	✓	662593	145	185	0
43	22:50:27.196	Thread Group ...	HTTP Request	194	✓	662593	145	180	0
44	22:50:28.630	Thread Group ...	HTTP Request	206	✓	662593	145	193	1
45	22:50:30.054	Thread Group ...	HTTP Request	198	✓	662593	145	183	1
46	22:50:31.488	Thread Group ...	HTTP Request	198	✓	662593	145	184	1
47	22:50:32.911	Thread Group ...	HTTP Request	207	✓	662593	145	192	1
48	22:50:34.338	Thread Group ...	HTTP Request	196	✓	662593	145	179	1
49	22:50:35.775	Thread Group ...	HTTP Request	194	✓	662593	145	180	1
50	22:50:37.204	Thread Group ...	HTTP Request	204	✓	662593	145	186	0
51	22:50:38.626	Thread Group ...	HTTP Request	197	✓	662593	145	181	1
52	22:50:40.061	Thread Group ...	HTTP Request	208	✓	662593	145	196	1
53	22:50:41.490	Thread Group ...	HTTP Request	191	✓	662593	145	176	1
54	22:50:42.918	Thread Group ...	HTTP Request	193	✓	662593	145	179	1
55	22:50:44.341	Thread Group ...	HTTP Request	204	✓	662593	145	190	0
56	22:50:45.775	Thread Group ...	HTTP Request	200	✓	662593	145	186	1
57	22:50:47.205	Thread Group ...	HTTP Request	194	✓	662593	145	179	0
58	22:50:48.630	Thread Group ...	HTTP Request	205	✓	662593	145	191	1
59	22:50:50.062	Thread Group ...	HTTP Request	193	✓	662593	145	178	1
60	22:50:51.485	Thread Group ...	HTTP Request	206	✓	662593	145	194	1
61	22:50:52.919	Thread Group ...	HTTP Request	192	✓	662593	145	178	1
62	22:50:54.343	Thread Group ...	HTTP Request	194	✓	662593	145	180	1
63	22:50:55.777	Thread Group ...	HTTP Request	294	✓	662593	145	278	1
64	22:50:57.204	Thread Group ...	HTTP Request	202	✓	662593	145	188	1
65	22:50:58.632	Thread Group ...	HTTP Request	194	✓	662593	145	179	1
66	22:51:00.061	Thread Group ...	HTTP Request	202	✓	662593	145	188	1
67	22:51:01.491	Thread Group ...	HTTP Request	200	✓	662593	145	185	1
68	22:51:02.920	Thread Group ...	HTTP Request	210	✓	662593	145	197	1
69	22:51:04.348	Thread Group ...	HTTP Request	201	✓	662593	145	186	1
70	22:51:05.776	Thread Group ...	HTTP Request	194	✓	662593	145	180	1

☐ Scroll automatically? ☐ Child samples? No of Samples 70 Latest Sample 194 Average 206 Deviation 23

Fitur 8 reliable pada 70 thread dalam waktu 100 detik, dengan rata rata response time 205ms/thread. Tidak terlalu cepat.

Dengan spec laptop sebagai berikut dan RAM DDR 3L 4Gb 1600 Mhz



The image is a screenshot of the CPU-Z application window. The 'CPU' tab is selected, displaying the following information:

- Processor:**
  - Name: Intel Core i5 4200U
  - Code Name: Haswell ULT
  - Package: Socket 1168 BGA
  - Technology: 22 nm
  - Core Voltage: 0.881 V
  - Max TDP: 15.0 W
- Specification:** Intel® Core™ i5-4200U CPU @ 1.60GHz
  - Family: 6
  - Model: 5
  - Stepping: 1
  - Ext. Family: 6
  - Ext. Model: 45
  - Revision: C0
  - Instructions: MMX, SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2, EM64T, VT-x, AES, AVX, AVX2, FMA3
- Clocks (Core #0):**
  - Core Speed: 2294.52 MHz
  - Multiplier: x 23.0 (8 - 26)
  - Bus Speed: 99.76 MHz
  - Rated FSB: (empty)
- Cache:**
  - L1 Data: 2 x 32 KBytes (8-way)
  - L1 Inst.: 2 x 32 KBytes (8-way)
  - Level 2: 2 x 256 KBytes (8-way)
  - Level 3: 3 MBytes (12-way)
- Selection:** Socket #1 (dropdown)
- Cores:** 2
- Threads:** 4

The bottom of the window shows the CPU-Z logo, version 1.80.1.x64, and buttons for Tools, Validate, and Close.