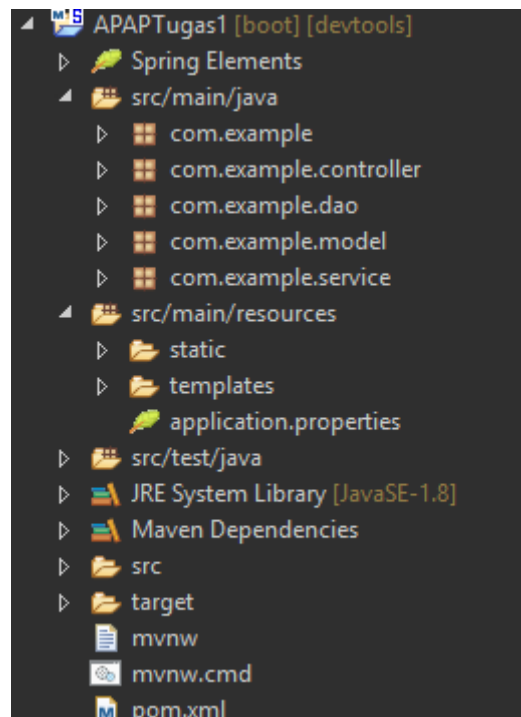


Bintang Glenn J

1506757535

Struktur Project

Pada tugas kali saya menggunakan struktur proyek seperti yang telah saya pelajari pada tutorial-tutorial yang diberikan sebelumnya. Adapun strukturnya adalah sebagai berikut:



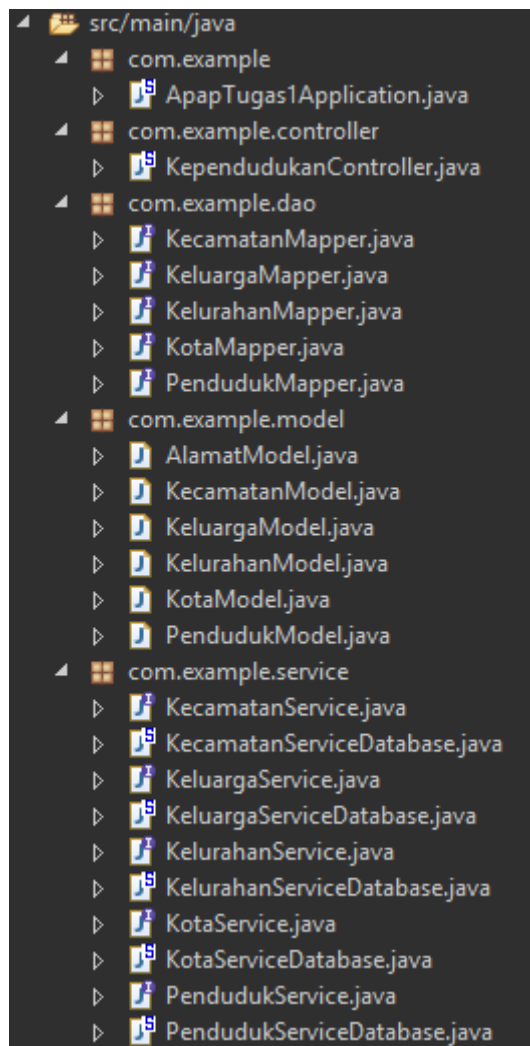
Secara umum saya membaginya menjadi 2 bagian, yaitu bagian *resource* dan bagian kode implementasi. Pada bagian kode implementasi terdiri dari 4 bagian pada *com.example*. Pada *package com.example* sendiri hanya ada file untuk menjalankan *main* untuk menginisiasi aplikasi, seperti yang terlihat berikut:

```
package com.example;

import org.springframework.boot.SpringApplication;

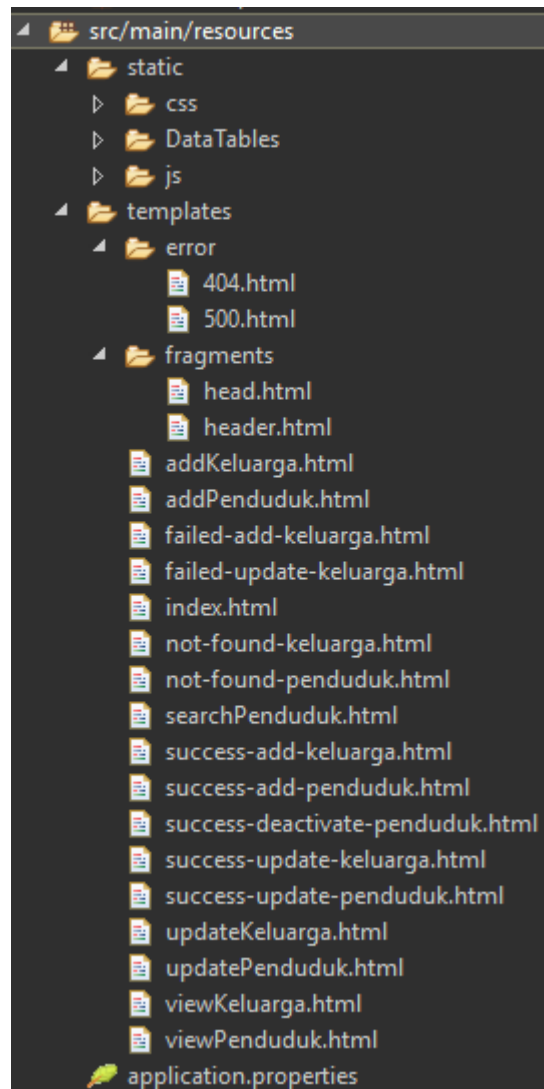
@SpringBootApplication
public class ApapTugas1Application
{
    public static void main (String[] args)
    {
        SpringApplication.run (ApapTugas1Application.class, args);
    }
}
```

Selanjutnya, saya memisahkan *controller*, *mapper*, *model* dan *service* menjadi *package* tersendiri, seperti yang terlihat berikut:



Controller hanya ada satu yang kemudian akan memetakan input url dari pengguna ke *view* yang semestinya. Saya membuat model, *mapper* dan *service* berdasarkan entitas yang diberikan pada kasus untuk mempermudah pembuatan proyek. Saya juga menambahkan satu model baru yaitu *AlamatModel* karena menurut saya alamat cukup sering digunakan dan akan lebih mudah jika dibuatkan model tersendiri. Model digunakan sebagai gambaran dari objek nyata yang berinteraksi. *Mapper* digunakan untuk menghubungkan aplikasi dengan *database* serta *service* digunakan untuk melakukan logika bisnis yang sesuai dengan kebutuhan tiap entitas. Adapun implementasi dari masing-masing bagian dapat dilihat pada proyek karena menurut saya sudah cukup jelas pembagian serta penamaannya.

Untuk *view* dan *resource* lainnya, yang saya buat adalah sebagai berikut:



Saya membuat halaman error dan *fragments* untuk mempermudah membuat tampilan. *View* yang lain dibuat sesuai dengan kebutuhan dengan penamaan yang cukup jelas. Halaman yang berupa respon hasil operasi saya gunakan separator “-” dan yang tidak saya gunakan camel case.

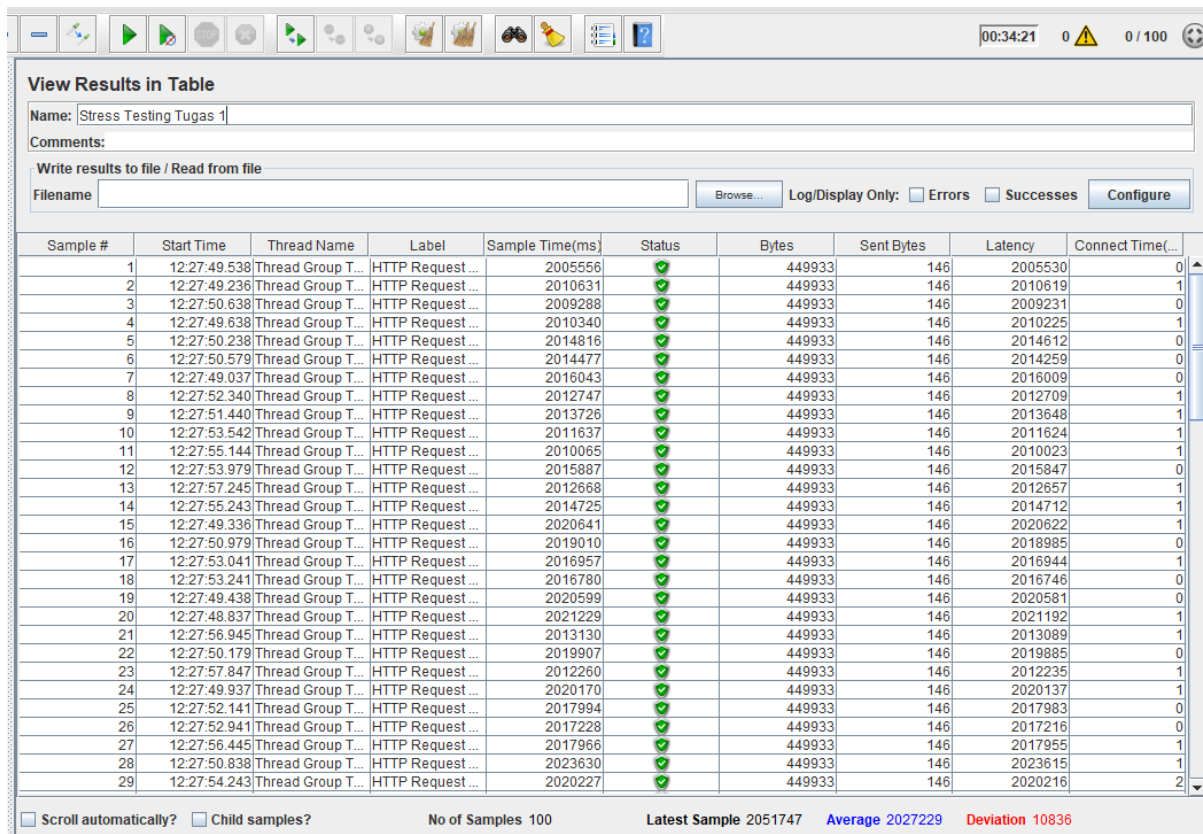
Pengembangan Tugas

Tugas pertama kali saya buat dengan membuat struktur yang sesuai untuk mempermudah pembuatan. Meski demikian, saya pada awalnya hanya membuat satu *mapper* dan *service* saja. Saya kemudian memasukkan Bootstrap dan DataTables pada proyek saya. Saya mengembangkan proyek dengan mengembangkan fitur secara terurut dari fitur 1 hingga 8. Saya mengerjakan fitur terlebih dahulu dan kemudian mengembangkan layout dan tampilannya setelah itu. Setelah fitur dan tampilan dasar selesai, saya kemudian memperbaiki hal-hal yang masih kurang atau terdapat kesalahan. Terakhir, saya memecah *mapper* dan *service* karena ternyata memang sulit untuk membacanya bila *mapper* dan *service* semua entitas dijadikan satu. Hal yang saya maksud di sini berlaku untuk *mapper* dan *service*, dengan kata lain pada awalnya saya membuat satu *mapper* dan satu *service*, bukan menggabungkan

keduanya pada satu file. Setelah semuanya selesai, saya melakukan beberapa percobaan untuk memastikan semua fitur berjalan sesuai dengan yang diminta.

Optimasi Database dan Stress Testing

Basis data awal saya ubah dengan menambahkan primary key untuk atribut id pada setiap tabel. Saya juga menambahkan auto_increment untuk id pada tabel keluarga dan penduduk, karena akan lebih mudah untuk menambahkan kedua hal tersebut pada basis data bila menggunakan auto_increment pada id. Basis data tersebut menjadi acuan saya ketika melakukan tes di tengah-tengah pengembangan, dengan pengurangan jumlah penduduk dan keluarga tentunya. Setelah semua fitur selesai, saya mencoba aplikasi dengan basis data yang berisi seluruh penduduk dan keluarga dengan langsung melakukan *stress testing*. Karena fitur yang menurut saya membutuhkan waktu lebih banyak adalah fitur pencarian, saya menggunakan `http://localhost:8080/penduduk/cari?kt=1&kc=1&kl=1` untuk melakukan pengecekan. Adapun saya menggunakan 100 *thread* dan periode *ramp-up* 10 detik. Hasilnya cukup mengejutkan, seperti berikut:





Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	12:27:49.538	Thread Group T...	HTTP Request ...	2005556	✓	449933	146	2005530	0
2	12:27:49.236	Thread Group T...	HTTP Request ...	2010631	✓	449933	146	2010619	1
3	12:27:50.638	Thread Group T...	HTTP Request ...	2009288	✓	449933	146	2009231	0
4	12:27:49.638	Thread Group T...	HTTP Request ...	2010340	✓	449933	146	2010225	1
5	12:27:50.238	Thread Group T...	HTTP Request ...	2014816	✓	449933	146	2014612	0
6	12:27:50.579	Thread Group T...	HTTP Request ...	2014477	✓	449933	146	2014259	0
7	12:27:49.037	Thread Group T...	HTTP Request ...	2016043	✓	449933	146	2016009	0
8	12:27:52.340	Thread Group T...	HTTP Request ...	2012747	✓	449933	146	2012709	1
9	12:27:51.440	Thread Group T...	HTTP Request ...	2013726	✓	449933	146	2013648	1
10	12:27:53.542	Thread Group T...	HTTP Request ...	2011637	✓	449933	146	2011624	1
11	12:27:55.144	Thread Group T...	HTTP Request ...	2010065	✓	449933	146	2010023	1
12	12:27:53.979	Thread Group T...	HTTP Request ...	2015887	✓	449933	146	2015847	0
13	12:27:57.245	Thread Group T...	HTTP Request ...	2012668	✓	449933	146	2012657	1
14	12:27:55.243	Thread Group T...	HTTP Request ...	2014725	✓	449933	146	2014712	1
15	12:27:49.336	Thread Group T...	HTTP Request ...	2020641	✓	449933	146	2020622	1
16	12:27:50.979	Thread Group T...	HTTP Request ...	2019010	✓	449933	146	2018985	0
17	12:27:53.041	Thread Group T...	HTTP Request ...	2016957	✓	449933	146	2016944	1
18	12:27:53.241	Thread Group T...	HTTP Request ...	2016780	✓	449933	146	2016746	0
19	12:27:49.438	Thread Group T...	HTTP Request ...	2020599	✓	449933	146	2020581	0
20	12:27:48.837	Thread Group T...	HTTP Request ...	2021229	✓	449933	146	2021192	1
21	12:27:56.945	Thread Group T...	HTTP Request ...	2013130	✓	449933	146	2013089	1
22	12:27:50.179	Thread Group T...	HTTP Request ...	2019907	✓	449933	146	2019885	0
23	12:27:57.847	Thread Group T...	HTTP Request ...	2012260	✓	449933	146	2012235	1
24	12:27:49.937	Thread Group T...	HTTP Request ...	2020170	✓	449933	146	2020137	1
25	12:27:52.141	Thread Group T...	HTTP Request ...	2017994	✓	449933	146	2017983	0
26	12:27:52.941	Thread Group T...	HTTP Request ...	2017228	✓	449933	146	2017216	0
27	12:27:56.445	Thread Group T...	HTTP Request ...	2017966	✓	449933	146	2017955	1
28	12:27:50.838	Thread Group T...	HTTP Request ...	2023630	✓	449933	146	2023615	1
29	12:27:54.243	Thread Group T...	HTTP Request ...	2020227	✓	449933	146	2020216	2

☐ Scroll automatically? ☐ Child samples? No of Samples 100 Latest Sample 2051747 Average 2072229 Deviation 10836



Semua *request* baru berhasil mendapatkan respon di atas 2.000.000 ms atau sekitar 33 menit. Waktu yang sangat lama sekali, menunjukkan respon sangat lambat bahkan hampir tidak ada bedanya untuk yang mengakses duluan dan yang belakangan. Saya juga melakukan pengecekan langsung dengan mengakses fitur langsung. Hasilnya, saya baru mendapatkan respon sekitar 10-15 menit setelahnya. Waktu yang cukup lama juga bagi satu respon saja.

Saya kemudian melakukan optimisasi dengan menambahkan index pada atribut-atribut yang sering diakses, seperti id_keluarga pada penduduk. Hasilnya adalah sebagai berikut:


- Tabel kecamatan

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id 	bigint(20)		UNSIGNED	No	None	
2	id_kota 	bigint(20)		UNSIGNED	No	None	
3	kode_kecamatan	char(7)			No	None	
4	nama_kecamatan	varchar(255)			No	None	





- Tabel kelurahan

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id 	bigint(20)		UNSIGNED	No	None	
2	id_kecamatan 	bigint(20)		UNSIGNED	No	None	
3	kode_kelurahan	char(10)			No	None	
4	nama_kelurahan	varchar(255)			No	None	
5	kode_pos	char(5)			No	None	

- Tabel kota

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id 	bigint(20)		UNSIGNED	No	None	
2	kode_kota	char(4)			No	None	
3	nama_kota	varchar(255)			No	None	




- Tabel keluarga

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id  	bigint(20)			No	None	AUTO_INCREMENT
2	nomor_kk 	char(16)			No	None	
3	alamat	varchar(256)			No	None	
4	RT	char(3)			No	None	
5	RW	char(3)			No	None	
6	id_kelurahan 	bigint(20)		UNSIGNED	No	None	
7	is_tidak_berlaku	tinyint(1)			No	0	

- Tabel penduduk

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id 🔑	bigint(20)			No	None	AUTO_INCREMENT
2	nik 🔑	char(16)			No	None	
3	nama	varchar(128)			No	None	
4	tempat_lahir	varchar(128)			No	None	
5	tanggal_lahir	date			No	None	
6	jenis_kelamin	int(1)			No	None	
7	is_wni	tinyint(1)			No	None	
8	id_keluarga 🔑	bigint(20)		UNSIGNED	No	None	
9	agama	varchar(64)			No	None	
10	pekerjaan	varchar(64)			No	None	
11	status_perkawinan	varchar(64)			No	None	
12	status_dalam_keluarga	varchar(64)			No	None	
13	golongan_darah	varchar(32)			No	None	
14	is_wafat	tinyint(1)			No	0	

Saya kemudian melakukan *stress testing* dengan kondisi yang sama dengan sebelumnya. Hasilnya adalah sebagai berikut:


00:00:10 0  0 / 100 

View Results in Table






























Name: Stress Testing Tugas 1 - Optimized

Comments:

Write results to file / Read from file

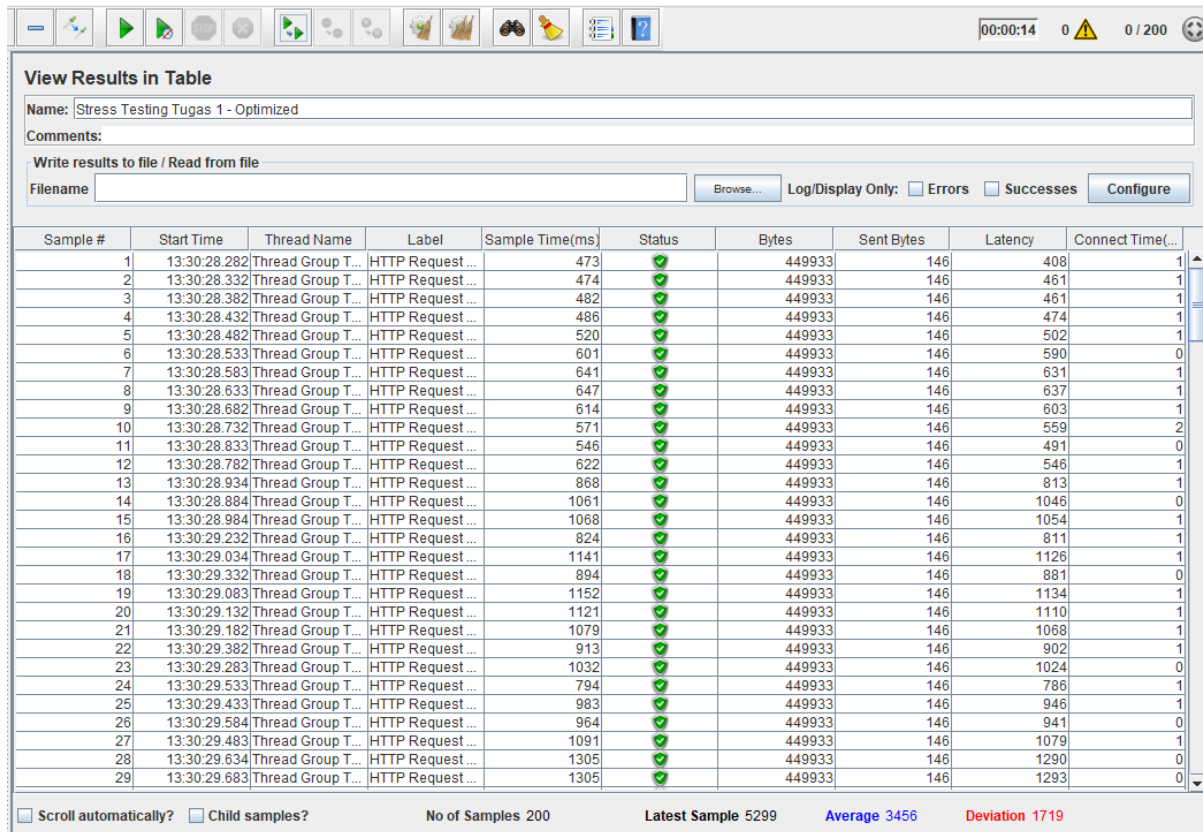
Filename Browse...

Log/Display Only:
☐ Errors
 ☐ Successes
 Configure

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	13:25:46.338	Thread Group T...	HTTP Request ...	324		449933	146	316	1
2	13:25:46.438	Thread Group T...	HTTP Request ...	532		449933	146	519	1
3	13:25:46.538	Thread Group T...	HTTP Request ...	529		449933	146	517	1
4	13:25:46.738	Thread Group T...	HTTP Request ...	412		449933	146	401	1
5	13:25:46.638	Thread Group T...	HTTP Request ...	530		449933	146	520	1
6	13:25:46.838	Thread Group T...	HTTP Request ...	379		449933	146	370	1
7	13:25:46.939	Thread Group T...	HTTP Request ...	361		449933	146	353	1
8	13:25:47.039	Thread Group T...	HTTP Request ...	352		449933	146	344	1
9	13:25:47.139	Thread Group T...	HTTP Request ...	318		449933	146	308	1
10	13:25:47.238	Thread Group T...	HTTP Request ...	308		449933	146	287	1
11	13:25:47.339	Thread Group T...	HTTP Request ...	319		449933	146	310	1
12	13:25:47.439	Thread Group T...	HTTP Request ...	321		449933	146	311	1
13	13:25:47.539	Thread Group T...	HTTP Request ...	442		449933	146	430	1
14	13:25:47.739	Thread Group T...	HTTP Request ...	397		449933	146	387	1
15	13:25:47.639	Thread Group T...	HTTP Request ...	505		449933	146	496	1
16	13:25:47.839	Thread Group T...	HTTP Request ...	398		449933	146	385	1
17	13:25:47.939	Thread Group T...	HTTP Request ...	367		449933	146	359	1
18	13:25:48.041	Thread Group T...	HTTP Request ...	337		449933	146	329	0
19	13:25:48.140	Thread Group T...	HTTP Request ...	312		449933	146	302	1
20	13:25:48.240	Thread Group T...	HTTP Request ...	312		449933	146	305	1
21	13:25:48.341	Thread Group T...	HTTP Request ...	312		449933	146	304	1
22	13:25:48.442	Thread Group T...	HTTP Request ...	304		449933	146	291	1
23	13:25:48.541	Thread Group T...	HTTP Request ...	394		449933	146	385	1
24	13:25:48.642	Thread Group T...	HTTP Request ...	416		449933	146	405	1
25	13:25:48.742	Thread Group T...	HTTP Request ...	334		449933	146	326	0
26	13:25:48.842	Thread Group T...	HTTP Request ...	328		449933	146	319	0
27	13:25:48.942	Thread Group T...	HTTP Request ...	323		449933	146	314	1
28	13:25:49.042	Thread Group T...	HTTP Request ...	307		449933	146	295	2
29	13:25:49.142	Thread Group T...	HTTP Request ...	304		449933	146	293	1

☐ Scroll automatically?
 ☐ Child samples?
 No of Samples 100 Latest Sample 292 Average 370 Deviation 76

Cukup mengejutkan bahwa sekarang rata-rata tidak sampai setengah detik respon sudah didapatkan kembali. Saya kemudian melakukan *stress testing* dengan menaikkan jumlah *thread* menjadi 2 kali lipat yaitu 200 *thread* dengan periode *ramp-up* tetap 10 detik. Hasilnya adalah sebagai berikut:



The screenshot shows the JMeter 'View Results in Table' window. The test name is 'Stress Testing Tugas 1 - Optimized'. The table displays 29 samples of HTTP requests. The status for all samples is 'Success' (green checkmark). The average latency is 3456 ms and the deviation is 1719 ms.

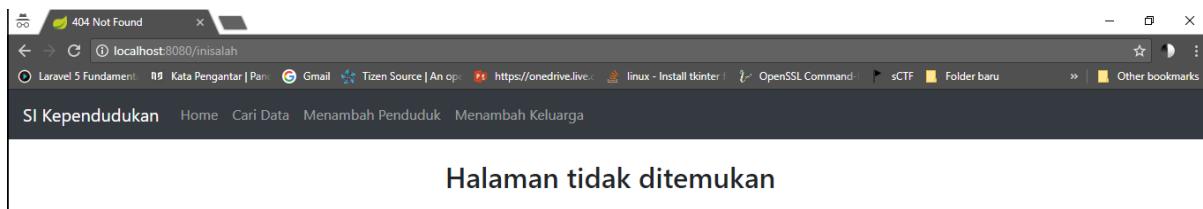
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(...)
1	13:30:28.282	Thread Group T...	HTTP Request ...	473	Success	449933	146	408	1
2	13:30:28.332	Thread Group T...	HTTP Request ...	474	Success	449933	146	461	1
3	13:30:28.382	Thread Group T...	HTTP Request ...	482	Success	449933	146	461	1
4	13:30:28.432	Thread Group T...	HTTP Request ...	486	Success	449933	146	474	1
5	13:30:28.482	Thread Group T...	HTTP Request ...	520	Success	449933	146	502	1
6	13:30:28.533	Thread Group T...	HTTP Request ...	601	Success	449933	146	590	0
7	13:30:28.583	Thread Group T...	HTTP Request ...	641	Success	449933	146	631	1
8	13:30:28.633	Thread Group T...	HTTP Request ...	647	Success	449933	146	637	1
9	13:30:28.682	Thread Group T...	HTTP Request ...	614	Success	449933	146	603	1
10	13:30:28.732	Thread Group T...	HTTP Request ...	571	Success	449933	146	559	2
11	13:30:28.833	Thread Group T...	HTTP Request ...	546	Success	449933	146	491	0
12	13:30:28.782	Thread Group T...	HTTP Request ...	622	Success	449933	146	546	1
13	13:30:28.934	Thread Group T...	HTTP Request ...	868	Success	449933	146	813	1
14	13:30:28.884	Thread Group T...	HTTP Request ...	1061	Success	449933	146	1046	0
15	13:30:28.984	Thread Group T...	HTTP Request ...	1068	Success	449933	146	1054	1
16	13:30:29.232	Thread Group T...	HTTP Request ...	824	Success	449933	146	811	1
17	13:30:29.034	Thread Group T...	HTTP Request ...	1141	Success	449933	146	1126	1
18	13:30:29.332	Thread Group T...	HTTP Request ...	894	Success	449933	146	881	0
19	13:30:29.083	Thread Group T...	HTTP Request ...	1152	Success	449933	146	1134	1
20	13:30:29.132	Thread Group T...	HTTP Request ...	1121	Success	449933	146	1110	1
21	13:30:29.182	Thread Group T...	HTTP Request ...	1079	Success	449933	146	1068	1
22	13:30:29.382	Thread Group T...	HTTP Request ...	913	Success	449933	146	902	1
23	13:30:29.283	Thread Group T...	HTTP Request ...	1032	Success	449933	146	1024	0
24	13:30:29.533	Thread Group T...	HTTP Request ...	794	Success	449933	146	786	1
25	13:30:29.433	Thread Group T...	HTTP Request ...	983	Success	449933	146	946	1
26	13:30:29.584	Thread Group T...	HTTP Request ...	964	Success	449933	146	941	0
27	13:30:29.483	Thread Group T...	HTTP Request ...	1091	Success	449933	146	1079	1
28	13:30:29.634	Thread Group T...	HTTP Request ...	1305	Success	449933	146	1290	0
29	13:30:29.683	Thread Group T...	HTTP Request ...	1305	Success	449933	146	1293	0

Summary statistics at the bottom:
 No of Samples: 200
 Latest Sample: 5299
 Average: 3456
 Deviation: 1719

Hasilnya cukup cepat juga, meskipun kini membutuhkan waktu rata-rata 3 detik untuk mengaksesnya. Namun saya rasa cukup wajar bila mengakses basis data yang cukup besar dengan banyak pengguna yang mengakses bersamaan.

Fitur tambahan

Fitur lain yang saya tambahkan adalah halaman error untuk beberapa jenis error, seperti 400 bad request, 404 not found dan 500 internal server error.



Untuk membuatnya, saya hanya membuat halaman html kemudian diletakkan di dalam folder error pada template. Halamannya sangat sederhana karena hanya menampilkan pesan secara umum, sehingga saya rasa tidak perlu dijelaskan.

Saya juga menambahkan halaman error bila pengguna memasukkan alamat yang tidak ada dalam basis data.

SI Kependudukan Home Cari Data Menambah Penduduk Menambah Keluarga

Gagal!

Alamat:

Kelurahan Sana
Kecamatan Sini
Kota Itu

Tidak ditemukan

Untuk membuatnya saya membuat method addKeluarga mengembalikan null bila seandainya pencarian alamat di basis data tidak mengeluarkan hasil apapun.

```
BigInteger idKelurahan = kelurahanMapper.selectKelurahanByName(alamat.getNamaKelurahan());  
if(idKelurahan == null) {  
    return null;  
}
```

```
@RequestMapping(value = "/keluarga/tambah", method = RequestMethod.POST)  
public String addKeluargaSubmit (Model model, AlamatModel alamat)  
{  
    String nkk = keluargaDAO.addKeluarga (alamat);  
    if(nkk == null) {  
        model.addAttribute("alamat", alamat);  
        return "failed-add-keluarga";  
    } else {  
        model.addAttribute("nkk", nkk);  
        return "success-add-keluarga";  
    }  
}
```

Saya juga menambahkan fitur penduduk termuda dan tertua, seperti berikut:

SI Kependudukan Home Cari Data Menambah Penduduk Menambah Keluarga

Lihat Data Penduduk Berdasarkan di Kota Jakarta Timur, Duren Sawit, Pondok Bambu

Penduduk dengan Usia Termuda		Penduduk dengan Usia Tertua	
NIK	3172075110520001	NIK	3172071909170003
Nama	Laila Safitri	Nama	Hendri Prayoga
Tanggal Lahir	11 October 1952	Tanggal Lahir	19 September 2017

Saya membuat fitur ini dengan pertama-tama mengimplement *interface Comparable<PendudukModel>* pada model penduduk. Saya kemudian melakukan *override* pada method *compareTo* seperti berikut:

```
@Override
public int compareTo(PendudukModel pendudukLain) {
    return getTanggalLahir().compareTo(pendudukLain.getTanggalLahir());
}
```

Beruntung *java.sql.date* sudah *comparable* sehingga saya hanya tinggal memanggil method *compareTo* milik *java.sql.date*. Saya kemudian melakukan sorting pada list berisi penduduk di tempat tersebut saat fitur pencarian dilakukan, seperti berikut:

```
@Override
public List<PendudukModel> selectWargaKelurahan(BigInteger idKelurahan) {
    log.info ("select penduduk in kelurahan with id {}", idKelurahan);
    List<BigInteger> listIdKeluarga = keluargaMapper.selectKeluargaOfKelurahan(idKelurahan);
    List<PendudukModel> pendudukKelurahan = new ArrayList<>();
    for(int i = 0; i < listIdKeluarga.size(); i++) {
        pendudukKelurahan.addAll(pendudukMapper.selectAnggotaKeluarga(listIdKeluarga.get(i)));
    }
    if(pendudukKelurahan.isEmpty()) {
        return null;
    } else {
        Collections.sort(pendudukKelurahan);
        return pendudukKelurahan;
    }
}
```

Terakhir, hanya tinggal ditampilkan pada *view*:

```
<div th:unless="{listKosong}" class="form-inline">
    <div class="col-sm-6">
        <h5 class="h5 text-center">Penduduk dengan Usia Termuda</h5>
        <table class="display table">
            <tr>
                <td>NIK</td>
                <td th:text="{listWargaKelurahan[0].nik}"></td>
            </tr>
            <tr>
                <td>Nama</td>
                <td th:text="{listWargaKelurahan[0].nama}"></td>
            </tr>
            <tr>
                <td>Tanggal Lahir</td>
                <td th:text="{#dates.format(listWargaKelurahan[0].tanggalLahir, 'dd MMMM yyyy')}"></td>
            </tr>
        </table>
    </div>

    <div class="col-sm-6">
        <h5 class="h5 text-center">Penduduk dengan Usia Tertua</h5>
        <table class="display table">
            <tr>
                <td>NIK</td>
                <td th:text="{listWargaKelurahan[lastIndex].nik}"></td>
            </tr>
            <tr>
                <td>Nama</td>
                <td th:text="{listWargaKelurahan[lastIndex].nama}"></td>
            </tr>
            <tr>
                <td>Tanggal Lahir</td>
                <td th:text="{#dates.format(listWargaKelurahan[lastIndex].tanggalLahir, 'dd MMMM yyyy')}"></td>
            </tr>
        </table>
    </div>
</div>
```

Variabel *lastIndex* didapat dengan perhitungan manual *listWargakelurahan.size() - 1*.