Tutorial 03 – Model, Service

1. Membuat Model

- Tambahkan *method constructor*, *setter*, dan *getter*.

  Jawab:

  Getter dan setter

```
private String getId() {
    return id;
}
private void setId(String id) {
    this.id = id;
}
private String getTitle() {
    return title;
}
private void setTitle(String title) {
    this.title = title;
}
private String getGenre() {
    return genre;
}
private void setGenre(String genre) {
    this.genre = genre;
}
private long getBudget() {
    return budget;
}
private void setBudget(long budget) {
    this.budget = budget;
}
private Integer getDuration() {
    return duration;
}
private void setDuration(Integer duration) {
    this.duration = duration;
}
```

- Constructor

```
private MovieModel(String title, String genre, long budget, Integer duration, String id) {
    super();
    this.title = title;
    this.genre = genre;
    this.budget = budget;
    this.duration = duration;
    this.id = id;
}
```

2. Membuat Service

- Implementasikan *method* getMovieDetail. *Method* ini menerima id movie dan mengembalikan *object* Movie dengan id tersebut. *Return* null jika tidak ditemukan.

  Jawab:

```
26
27      @Override
28      public MovieModel getMovieDetail(String id) {
29          for (MovieModel movie: archiveMovie) {
30              if (movie.getId().equalsIgnoreCase(id)) {
31                  return movie;
32              }
33          }
34          return null;
35      }
36
```

3. Membuat Controller dan Method Add

- *http://localhost:8080/movie/add?id=1&title=Marvel&genre=Action&budget=50000000&duration=50*

Q1: Apakah hasilnya? Jelaskan apa yang terjadi (lengkapi dengan *screen capture*).

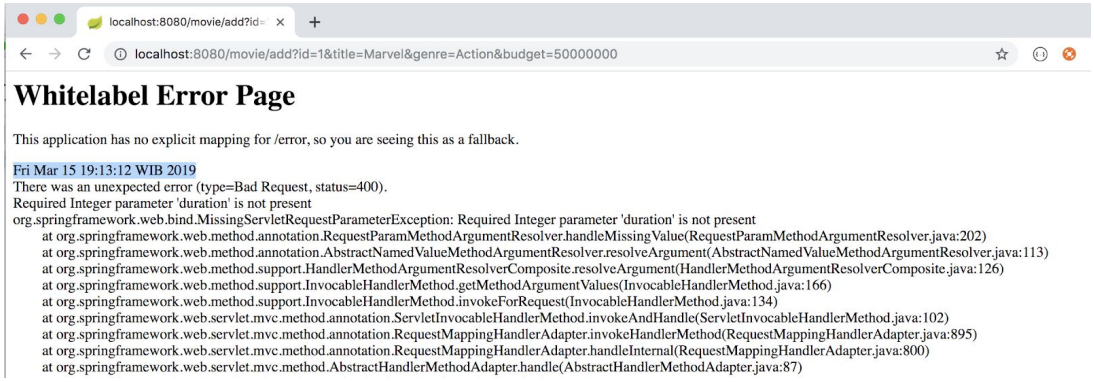
Jawab:
Untuk menambah movie dilakukan pada path "/movie/add", yang menjalankan method Add() dengan melempar parameter (id, title, genre, budget dan duration dimana semua parameter tersebut wajib) . Jika berhasil akan mengembalikan view add.html yang berisi "Data berhasil ditambahkan!"

- *http://localhost:8080/movie/add?id=1&title=Marvel&genre=Action&budget=50000000*

Q2: Apakah hasilnya? Jelaskan apa yang terjadi (lengkapi dengan *screen capture*).

Jawab:

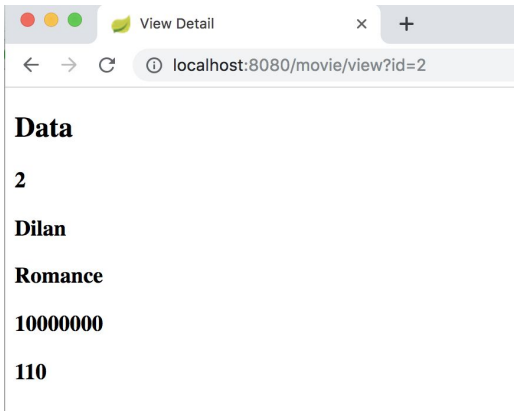
Terjadi error/fallback karena parameter duration tidak disertakan pada link tersebut.



4. Membuat Method View by ID

- *http://localhost:8080/movie/add?id=2&title=Dilan&genre=Romance&budget=10000000&duration=110*

- *http://localhost:8080/movie/view?id=2*

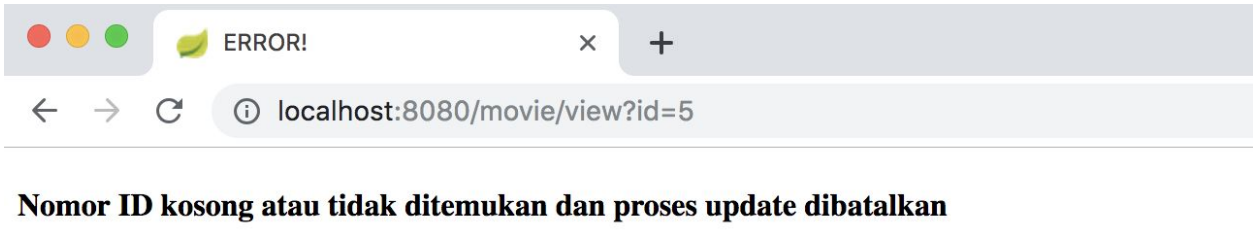


Q3: Apakah data movie tersebut muncul? Jelaskan apa yang terjadi (lengkapi dengan *screen capture*).

Jawab:

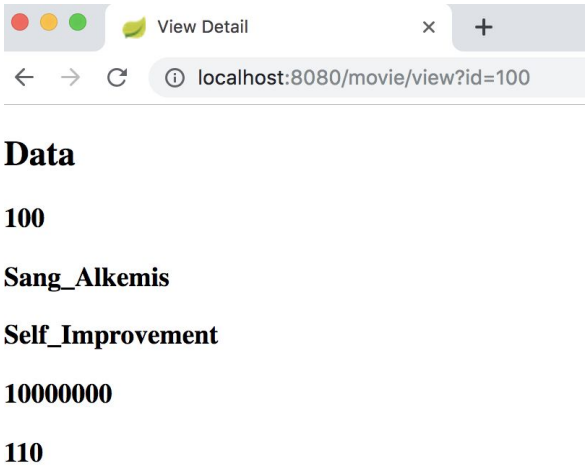
Ya, data movie tersebut muncul jika id movie ditemukan akan mengembalikan halaman view-movie.hmtl. Namun jika id tidak ditemukan akan menampilkan view error.html seperti gambar berikut:

- *http://localhost:8080/movie/view?id=4 (id tidak ditemukan)*



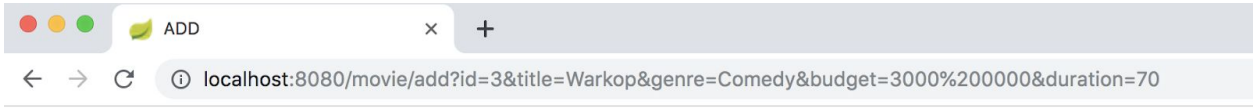
- Tambahkan data movie lainnya dengan id yang berbeda.

View http://localhost:8080/movie/view?id=100



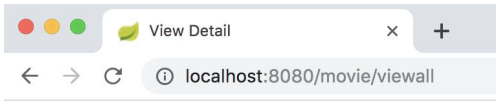
5. Membuat Method View All

- *http://localhost:8080/movie/add?id=3&title=Warkop&genre=Comedy&budget=30000000&duration=70*



Data berhasil ditambahkan!

- *http://localhost:8080/movie/viewall*



Data

100

Sang_Alkemis

Self_Improvement

10000000

110

3

Warkop

Comedy

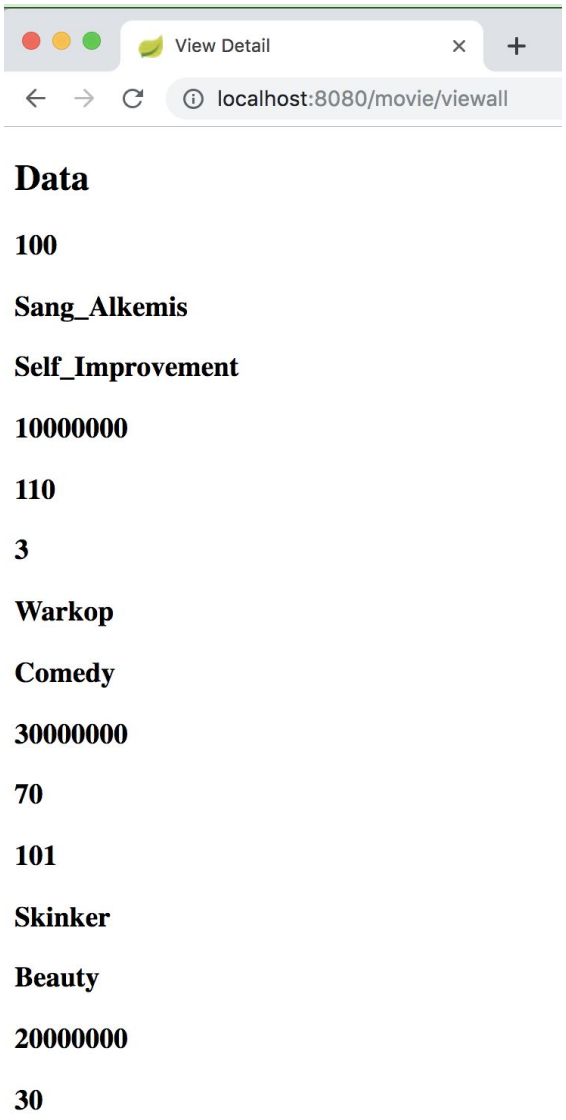
30000000

70

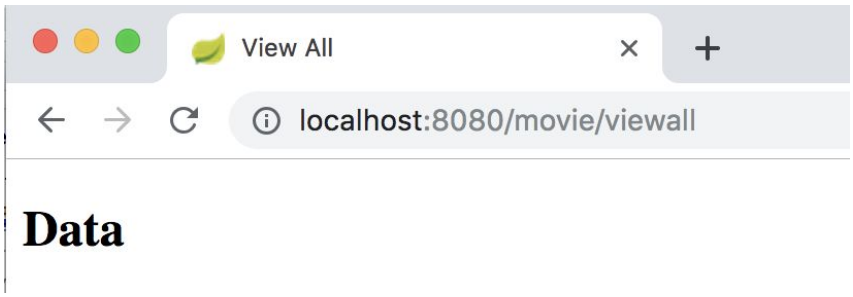
- Q4: Apakah data movie tersebut muncul? Jelaskan apa yang terjadi (lengkapi dengan *screen capture*).
  Jawab:
  Ya, data semua movie muncul dengan mengembalikannya melalui halaman viewall-movie.html

- Tambahkan data movie lainnya dengan id yang berbeda. Lalu akses *http://localhost:8080/movie/viewall*

Data berhasil ditambahkan!



Jika tidak ada movie yang tersedia, akan menampilkan page berikut ini:



Q5: Apakah semua data Movie muncul? Jelaskan apa yang terjadi (lengkapi dengan screen capture).
Jawab:

Ya, data semua movie muncul dengan mengembalikannya melalui halaman viewall-movie.html. Pada halaman view-all.html semua parameter setiap movie akan di-render. Namun, jika tidak terdapat movie akan menampilkan halaman kosong.

Latihan

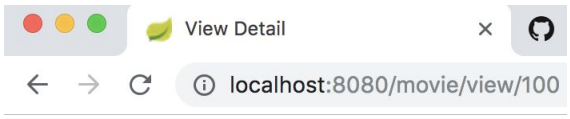
1. *Method* view Movie pada MovieController dengan menggunakan **PathVariable**. Jika nomor id tidak diberikan atau tidak ditemukan kembalikan halaman *error* yang berisi informasi bahwa nomor id kosong atau tidak ditemukan.

   Jawab:

```
@RequestMapping("/movie/view/{id}")
public String viewMovie(@PathVariable Optional<String> id, Model model) {
    if(id.isPresent()) {
        for(MovieModel movie: movieService.getMovieList()) {
            if(movie.getId().equals(id.get())) {
                MovieModel archive = movieService.getMovieDetail(id.get());
                model.addAttribute("movie", archive);
                return "view-movie";
            }
            else {
                return "error";
            }
        }
    }else if(!id.isPresent()) {
        return "error";
    }
    return null;

}
```

   localhost:8080/movie/view/100



Data

100

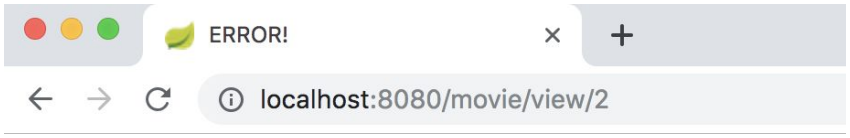
Sang_Alkemis

Self_Improvement

10000000

110

   localhost:8080/movie/view/2

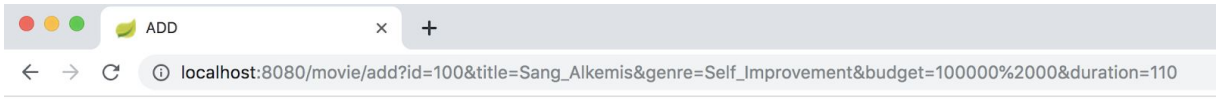


ID yang anda masukkan tidak ditemukan

2. Method untuk update duration dari Movie berdasarkan id.

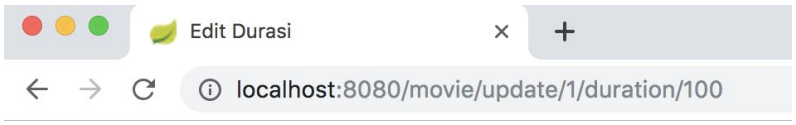
```
@RequestMapping("/movie/update/{id}/duration/{duration}")
public String updateDuration(@PathVariable Optional<String> id, @PathVariable Optional<Integer> duration, Model model) {
    if(id.isPresent()) {
        for(MovieModel movie: movieService.getMovieList()) {
            if(movie.getId().equals(id.get())) {
                String movieTitle = movieService.getMovieDetail(id.get()).getTitle();
                int movieDuration = movieService.getMovieDetail(id.get()).getDuration();
                movieService.getMovieDetail(id.get()).setDuration(duration.get());
                int newMovieDuration = duration.get();
                model.addAttribute("title", movieTitle);
                model.addAttribute("movieDuration", movieDuration);
                model.addAttribute("newMovieDuration", newMovieDuration);
                return "edit-duration";
            }
            else {
                return "error";}
        }
    }
    else if (!id.isPresent()) {
        return "error";}
    return null;
}
```

Add movie id=100



Data berhasil ditambahkan!

Update movie duration id 100: localhost:8080/movie/update/1/duration/100



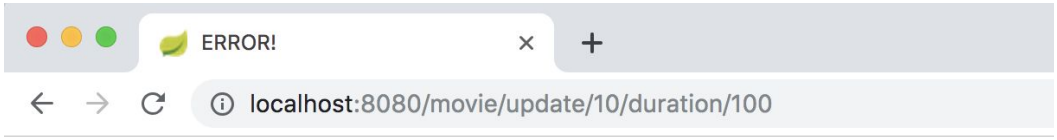
BERHASIL

Marvel

duration is updated to

100

Jika id tidak ditemukan:

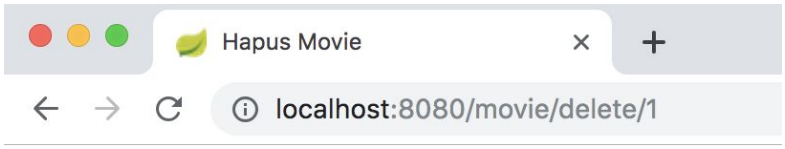


Nomor ID kosong atau tidak ditemukan dan proses update dibatalkan

3. Method untuk delete Movie berdasarkan id.

```
@RequestMapping("/movie/delete/{id}")
public String deleteMovie(@PathVariable Optional<String> id, Model model) {
    if(id.isPresent()) {
        for(MovieModel movie: movieService.getMovieList()) {
            if(movie.getId().equals(id.get())) {
                for (Iterator<MovieModel> iter = movieService.getMovieList().listIterator(); iter.hasNext(); ) {
                    MovieModel a = iter.next();
                    if (a.getId().equalsIgnoreCase(id.get())) {
                        iter.remove();
                    }
                }
                model.addAttribute("id", id.get());
                return "delete";
            }
            else {return "error";}
        }
    }
    else if (!id.isPresent()) {
        return "error";}
    return null;
}
```

localhost:8080/movie/delete/1

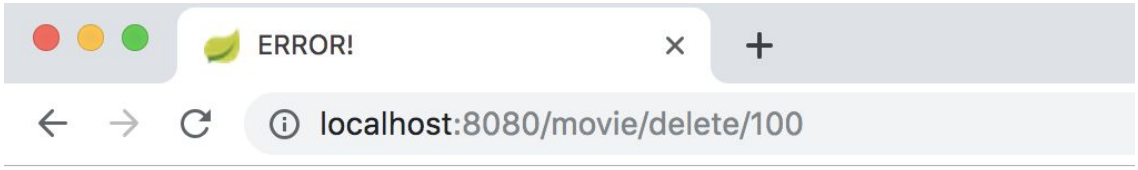


Movie dengan ID:

1

Berhasil dihapus

Jika nomor id tidak diberikan atau tidak ditemukan kembalikan halaman error yang berisi informasi bahwa nomor id kosong atau tidak ditemukan dan proses *delete* dibatalkan.



Nomor ID kosong atau tidak ditemukan dan proses update dibatalkan