

TUTORIAL

1. Web Service

- *Web service* memfasilitasi komunikasi antar mesin melalui jaringan. Data yang digunakan adalah XML atau JSON.
- *API (Application Programming Interface)* memfasilitasi komunikasi antar komponen.

2. Service Producer

- *Service Producer* menyediakan layanan untuk memberikan data kepada *client*.
- Menggunakan *RestController*

3. Service Consumer

- *Service consumer* membutuhkan layanan lain untuk mendapatkan data yang dibutuhkan.
- Membuat *mock up* sebagai *service producer*, menggunakan Postman
- *Service consumer* sebagai penerima data dari *producer*, menggunakan *RestTemplate*
- Menambahkan *dependency* di *pom.xml* yaitu *com.fasterxml.jackson.core* untuk melakukan *data bind* ketika kita *post* ke *service* lain

LATIHAN

1. Latihan 1

METHOD 1: Menampilkan Flight

Pada *controller*, tambahkan:

```
@GetMapping("/view/{flightNumber}")
public Optional<FlightModel> getFlight(@PathVariable("flightNumber") String flightNumber) {
    return flightService.getFlightDetailByFlightNumber(flightNumber);
}
```

Pada Postman buat *request* baru sebagai berikut:

The screenshot shows the Postman interface for a new request. The request name is 'Get a Flight'. The method is 'GET' and the URL is '{{url-tutorial7}}flight/view/1133'. There are 'Send', 'Save', and 'Examples (0)' buttons. Below the URL bar, there are tabs for 'Params', 'Authorization', 'Headers', 'Body', 'Pre-request Script', and 'Tests'. The 'Params' tab is active, showing a table with columns 'KEY', 'VALUE', and 'DESCRIPTION'. The table has one row with 'Key' and 'Value' in the first two columns, and 'Description' in the third. There are also 'Cookies' and 'Code' tabs on the right. At the bottom, there is a 'Response' section.

KEY	VALUE	DESCRIPTION
Key	Value	Description

Kemudian klik *save* dan *send*, maka *response* akan dikembalikan

The screenshot shows a Postman interface for a GET request to `{{url-tutorial7}}flight/view/1133`. The status is 200 OK, time is 27 ms, and size is 329 B. The response body is a JSON object:

```
{
  "id": 19,
  "flightNumber": "1133",
  "origin": "UPG - Bandar Udara Internasional Sultan Hasanuddin, Maros",
  "destination": "BDJ - Bandar Udara Internasional Syamsuddin Noor, Banjarbaru",
  "time": "1992-02-01"
}
```

METHOD 2: Menampilkan Seluruh Flight

Mengimplementasikan *method* pada *service* untuk menampilkan seluruh *flight* dalam bentuk *list*:

```
@Override
public List<FlightModel> getAllFlight() {
    return flightDb.findAll();
}
```

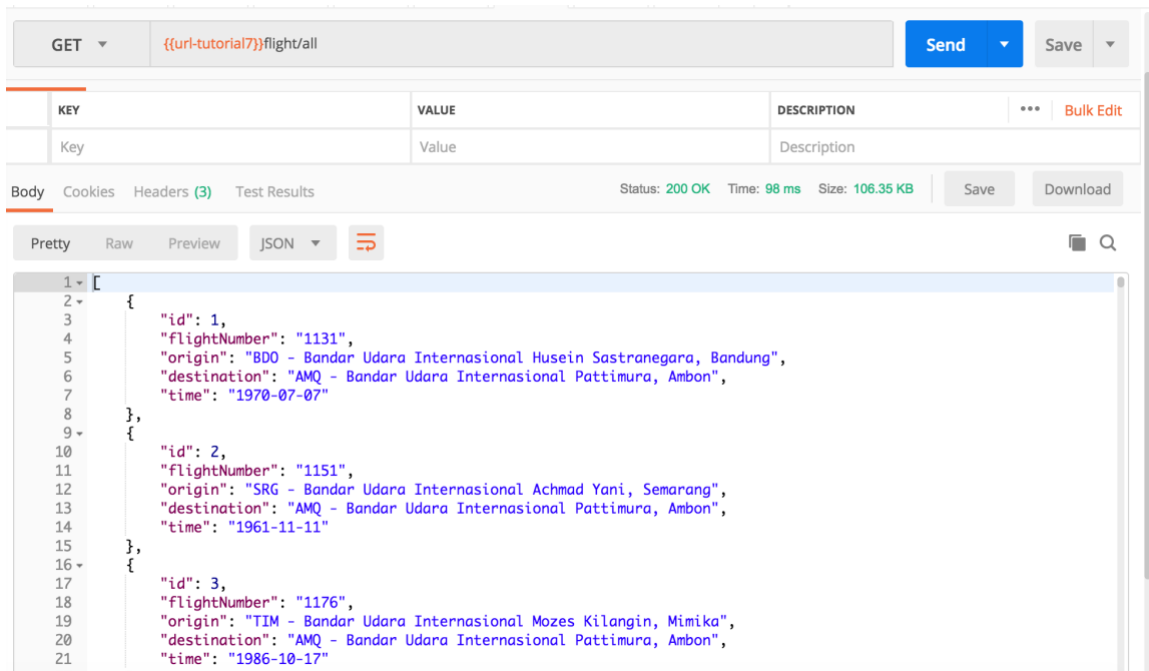
Pada *controller*, tambahkan:

```
@GetMapping("/all")
public List<FlightModel> getListFlight() {
    return flightService.getAllFlight();
}
```

Pada Postman buat *request* baru sebagai berikut:

The screenshot shows a new GET request in Postman to `{{url-tutorial7}}flight/all`. The status is 200 OK, time is 98 ms, and size is 106.35 KB. The response body is a JSON list of flight objects.

Kemudian klik *save* dan *send*, maka *response* akan dikembalikan



METHOD 3: Menghapus Flight

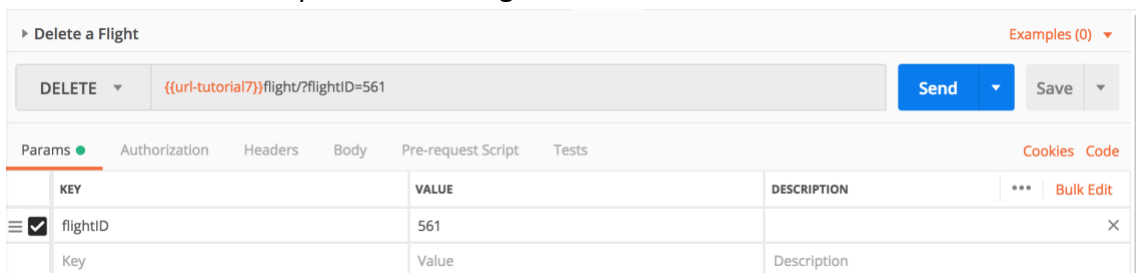
Mengimplementasikan *method* pada *service* untuk menghapus *flight* berdasarkan ID:

```
@Override
public Boolean deleteFlightById(long id) {
    flightDb.deleteById(id);
    return true;
}
```

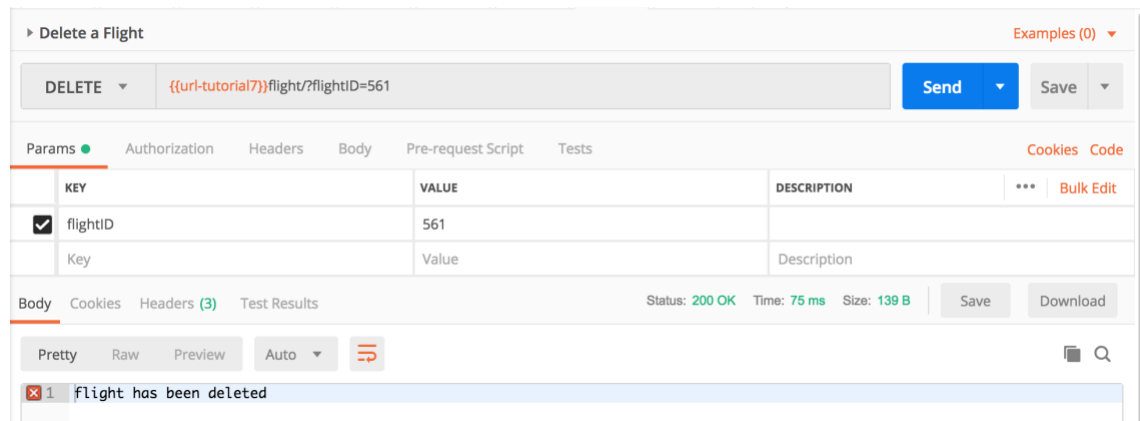
Pada *controller*, tambahkan:

```
@DeleteMapping(value = "/")
public String deleteFlight(@RequestParam("flightID") long flightID) {
    flightService.deleteFlightById(flightID);
    return "flight has been deleted";
}
```

Pada Postman buat *request* baru sebagai berikut:



Kemudian klik *save* dan *send*, maka *response* akan dikembalikan



METHOD 4: Mengubah Flight

Mengimplementasikan *method* pada *service* untuk mengubah *flight* berdasarkan ID:

```
@Override
public void updateFlight(long id, FlightModel flight) {
    FlightModel oldFlight = flightDb.getOne(id);
    oldFlight.setDestination(flight.getDestination());
    oldFlight.setOrigin(flight.getOrigin());
    flightDb.save(flight);
}
```

Pada *controller*, tambahkan:

```
@PostMapping("/update/{flightID}")
public String updateFlight(@PathVariable("flightID") long flightID,
    @RequestParam(value = "destination", required = false) String destination,
    @RequestParam(value = "origin", required = false) String origin) {
    FlightModel flight = flightService.getFlightById(flightID);

    if(flight.equals(null)) {
        return "Couldn't find your flight";
    }

    flight.setDestination(destination);
    flight.setOrigin(origin);
    flightService.updateFlight(flightID, flight);
    return "flight update success";
}
```

Pada Postman buat *request* baru sebagai berikut:

The screenshot shows the Postman interface for a PUT request named "Update a Flight". The URL is `{{url-tutorial7}}flight/update/561?destination=Singapura&origin=Surabaya`. The request body is configured with two parameters: "destination" with value "Singapura" and "origin" with value "Surabaya". The "Send" button is highlighted in blue.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> destination	Singapura	
<input checked="" type="checkbox"/> origin	Surabaya	
Key	Value	Description

Kemudian klik *save* dan *send*, maka *response* akan dikembalikan

The screenshot shows the Postman interface after sending the request. The status is "200 OK", time is "83 ms", and size is "137 B". The response body is displayed in the "Body" tab, showing "1 flight update success".

Status: 200 OK Time: 83 ms Size: 137 B

Body Cookies Headers (3) Test Results

1 flight update success

METHOD 5: Menambahkan Flight

Agar bisa mendapatkan `licenseNumber` dari pilot, maka pada `FlightModel` ubah atribut pilot menjadi sebagai berikut:

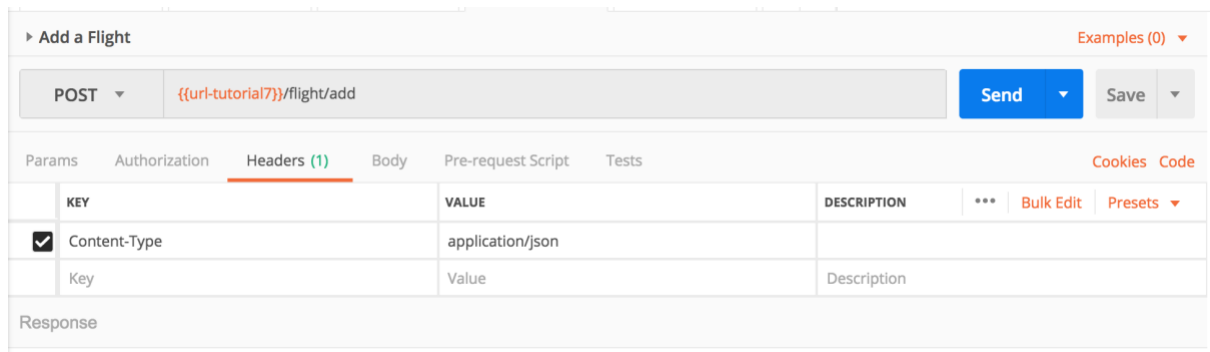
```
@JoinColumn(name = "pilot_licenseNumber", referencedColumnName =
    "license_number", nullable = false)
private String pilotLicenseNumber;
```

dan karena bagian tersebut pada `FlightModel` diubah, untuk sementara hapus atribut `listFlight` pada `PilotModel`.

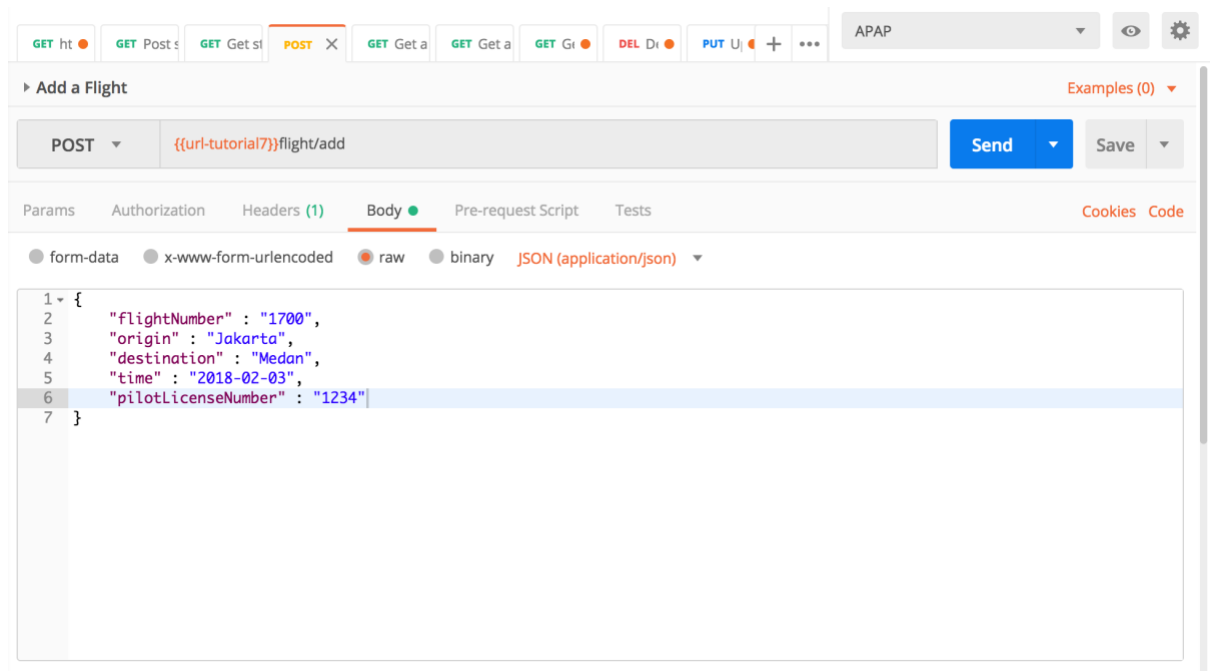
Pada *controller*, tambahkan:

```
@PostMapping("/add")
public FlightModel addFlight(@RequestBody FlightModel flight) {
    return flightService.addFlight(flight);
}
```

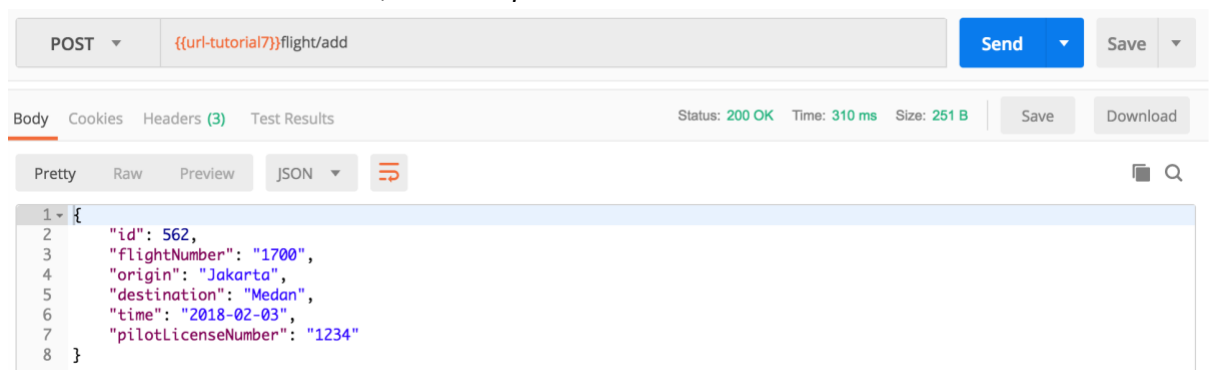
Pada Postman buat *request* baru sebagai berikut:



Lalu pada *body*, tambahkan:



Kemudian klik *save* dan *send*, maka *response* akan dikembalikan



2. Latihan 2

blablabalbla