

Pada tutorial ini, mahasiswa belajar bagaimana cara menggunakan Rest API web service, membuat mock server, dan menggunakan service. Web service merupakan software yang dibuat untuk berkomunikasi melalui network. Sedangkan, API adalah library sebagai perantara komunikasi antar-komponen.

Latihan

1. Pada latihan satu, mahasiswa diminta untuk membuat lima API untuk transaksi data flight. Lima API tersebut antara lain : membuat flight baru, update flight, search berdasarkan flight number, search semua flight, menghapus flight.

FlightController antara lain akan seperti berikut :

```
/**
 * FlightController
 */
@RestController
@RequestMapping("/flight")
public class FlightController {
    @Autowired
    private FlightService flightService;

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

    @PutMapping(value = "/update/{flightId}")
    private String updateFlight(@PathVariable(value="flightId") long flightId, @RequestParam("destination") String destination, @RequestParam("origin") String origin, @RequestParam("time") String time) {
        FlightModel flight = flightService.getFlightDetailById(flightId);
        if (flight.equals(null)) {
            return "Couldn't find your flight";
        }
        flight.setDestination(destination);
        flight.setOrigin(origin);
        flight.setTime(time);

        return "flight update success";
    }

    @DeleteMapping(value = "/delete/{flightId}")
    public String deleteFlight(@PathVariable(value="flightId") long flightId) {
        FlightModel flight = flightService.getFlightDetailById(flightId);
        flightService.deleteFlight(flight);
        return "flight has been deleted";
    }

    @GetMapping(value = "/view/{flightNumber}")
    public @ResponseBody FlightModel viewFlight(@PathVariable("flightNumber") String flightNumber) {
        FlightModel flight = flightService.getFlightDetailByFlightNumber(flightNumber);
        return flight;
    }

    @GetMapping(value = "/all")
    public @ResponseBody List<FlightModel> viewAllFlight() {
        List<FlightModel> flight = flightService.getAll();
        return flight;
    }
}
```

Sedangkan, output pada postman akan seperti berikut :

Add flight :

▶ Add a flight

Examples (0)

POST

{{url-tutorial7}}/flight/add

Send

Save

Params

Authorization

Headers (1)

Body

Pre-request Script

Tests

Cookies

Code

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body

Cookies (1)

Headers (3)

Test Results

Status: 200 OK

Time: 485 ms

Size: 385 B

Save

Download

Pretty

Raw

Preview

JSON

1

{

2

"id": 562,

3

"flightNumber": "12121",

4

"origin": "BDO - Bandar Udara Internasional Husein ",

5

"destination": "AMQ - Bandar Udara Internasional Pattimura, Ambon",

6

"time": "1977-07-07",

7

"pilot": {

8

"id": 1,

9

"licenseNumber": "1234",

10

"name": "Coki",

11

"flyHour": 139,

12

"listFlight": []

13

}

14

}

Update :

▶ Update a flight

Examples (0)

PUT

{{url-tutorial7}}/flight/update/1?destination=Surga&origin=Neraka&time=1951-10-10

Send

Save

Params

Authorization

Headers

Body

Pre-request Script

Tests

Cookies

Code

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> destination	Surga			
<input checked="" type="checkbox"/> origin	Neraka			
<input checked="" type="checkbox"/> time	1951-10-10			
Key	Value	Description		

Body

Cookies (1)

Headers (3)

Test Results

Status: 200 OK

Time: 140 ms

Size: 137 B

Save

Download

Pretty

Raw

Preview

Auto

1

flight update success

Search by flight number :

[illegible]

Get all flights :

[CONFLICT] GET Get Airpor POST Add a flight PUT Update a flight GET Get a flight GET Get all flight X + ... APAP

> Get all flight Examples () ▾

GET ▾ {{uri-tutorial7?}}/flight/all Send ▾ Save ▾

Params Authorization Headers Body Pre-request Script Tests Cookies Code

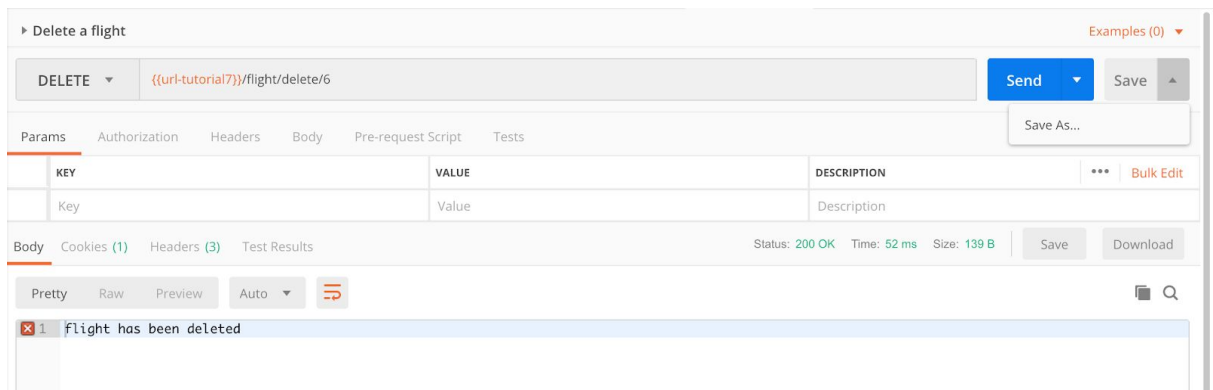
KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (3) Test Results Status: 200 OK Time: 4094 ms Size: 237.84 KB Save Download

Pretty Raw Preview JSON ▾

```
1 [{"id":1,"flightNumber":"1131","origin":"BDO - Bandar Udara Internasional Husein Sastranegara, Bandung","destination":"AMQ - Bandar Udara Internasional Pattimura, Ambon","time":"1970-07-07","pilot":{"id":1,"licenseNumber":"1234","name":"Desmosedic","flyHour":900,"listFlight":[{"id":1,"flightNumber":"1131","origin":"BDO - Bandar Udara Internasional Husein Sastranegara, Bandung","destination":"AMQ - Bandar Udara Internasional Pattimura, Ambon","time":"1970-07-07","pilot":{"id":1,"licenseNumber":"1234","name":"Desmosedic","flyHour":900,"listFlight":[{"id":1,"flightNumber":"1131","origin":"BDO - Bandar Udara Internasional Husein Sastranegara, Bandung","destination":"AMQ - Bandar Udara Internasional Pattimura, Ambon","time":"1970-07-07","pilot":{"id":1,"licenseNumber":"1234","name":"Desmosedic","flyHour":900,"listFlight":[{"id":1,"flightNumber":"1131","origin":"BDO - Bandar Udara Internasional Husein Sastranegara, Bandung","destination":"AMQ - Bandar Udara Internasional Pattimura, Ambon","time":"1970-07-07","pilot":{"id":1,"licenseNumber":"1234","name":"Desmosedic","flyHour":900,"listFlight":[{"id":1,"flightNumber":"1131","origin":"BDO - Bandar Udara Internasional Husein Sastranegara, Bandung","destination":"AMQ - Bandar Udara Internasional Pattimura, Ambon","time":"1970-07-07","pilot":{"id":1,"licenseNumber":"1234","name":"Desmosedic","flyHour":900,"listFlight"}]}]}]}]}]}
```

Delete Flight :



2. Pada latihan 2, mahasiswa diminta untuk menampilkan airport yang berada sesuai dengan kota, menggunakan API yang sudah ada dari service lain, yaitu service Amadeus.

Pertama-tama, tambahkan pada Setting line API key dan URL API untuk mendapatkan list airport

```
package com.apap.tutorial7.rest;

public class Setting {
    public static final String pilotUrl = "https://b5f3a7f1-752e-4217-85a5-5951e6a5b14f.mock.pstmn.io";
    final public static String airportUrl = "https://api.sandbox.amadeus.com/v1.2/airports/autocomplete";
    final public static String airportApiKey = "XGGFKpGY9Dl0oRAAqqQ9dRwn1zqXmTmE";
}
```

Setelah itu, implementasikan ada AirportController :

```
@RestController
@RequestMapping("/airport")
public class AirportController {

    @Autowired
    private RestTemplate restTemplate;

    @Bean
    public RestTemplate restTemplate() {
        return new RestTemplate();
    }

    @GetMapping(value= "/kota/{namaKota}")
    public ResponseEntity<String> getStatus(@PathVariable("namaKota") String query) {
        String path = Setting.airportUrl+"?apikey="+Setting.airportApiKey+"&term="+query+"&country=ID";
        ResponseEntity<String> response = restTemplate.getForEntity(path, String.class);
        return response;
    }
}
```

Sekarang, dicoba dengan GET dari postman :

Get Airports

Examples (0)

GET

{{url-tutorial7}}/airport/kota/jakarta

Send

Save

Params

Authorization

Headers

Body

Pre-request Script

Tests

Cookies

Code

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies (1)

Headers (14)

Test Results

Status: 200 OK

Time: 2438 ms

Size: 914 B

Save

Download

Pretty

Raw

Preview

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

14

[

{

"value": "JKT",

"label": "Jakarta [JKT]"

}

,

{

"value": "CGK",

"label": "Jakarta - Soekarno - Hatta International Airport [CGK]"

}

,

{

"value": "HLP",

"label": "Jakarta - Halim Perdanakusuma Airport [HLP]"

}

]