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

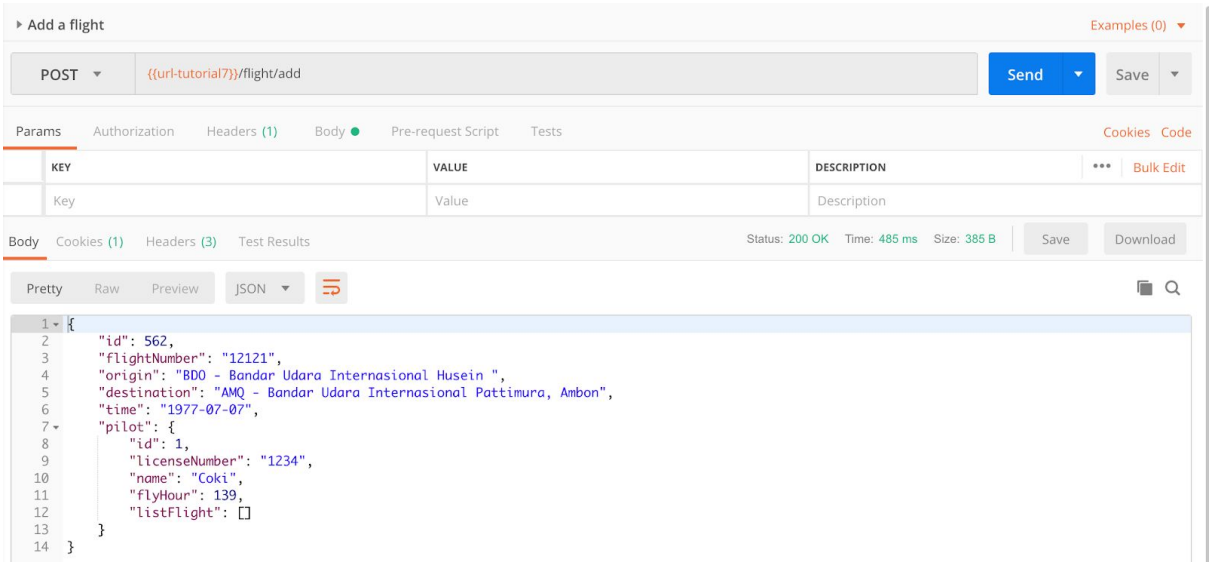
1. Pada latihan satu, mahasiswa diminta untuk membuat lima API untuk transaksi data flight. Lima API tersebut antara lain : membuat flight baru, update flight, search berdasarkan flight number, search semua flight, menghapus flight.

   FlightController antara lain akan seperti berikut :

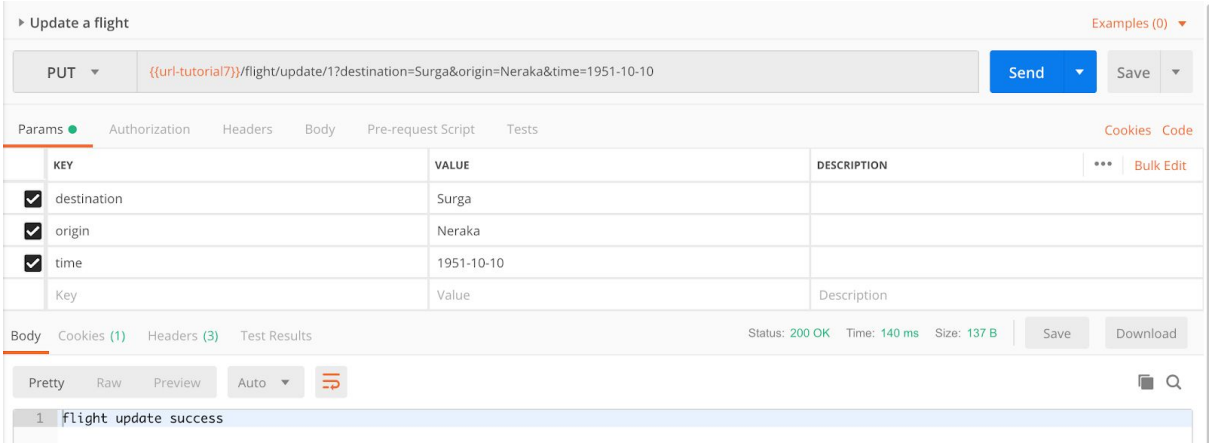
```java
/**
 * FlightController
 */
@RestController
@RequestMapping("/flight")
public class FlightController {
    @Autowired
    private FlightService flightService;

    @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") D
        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 :

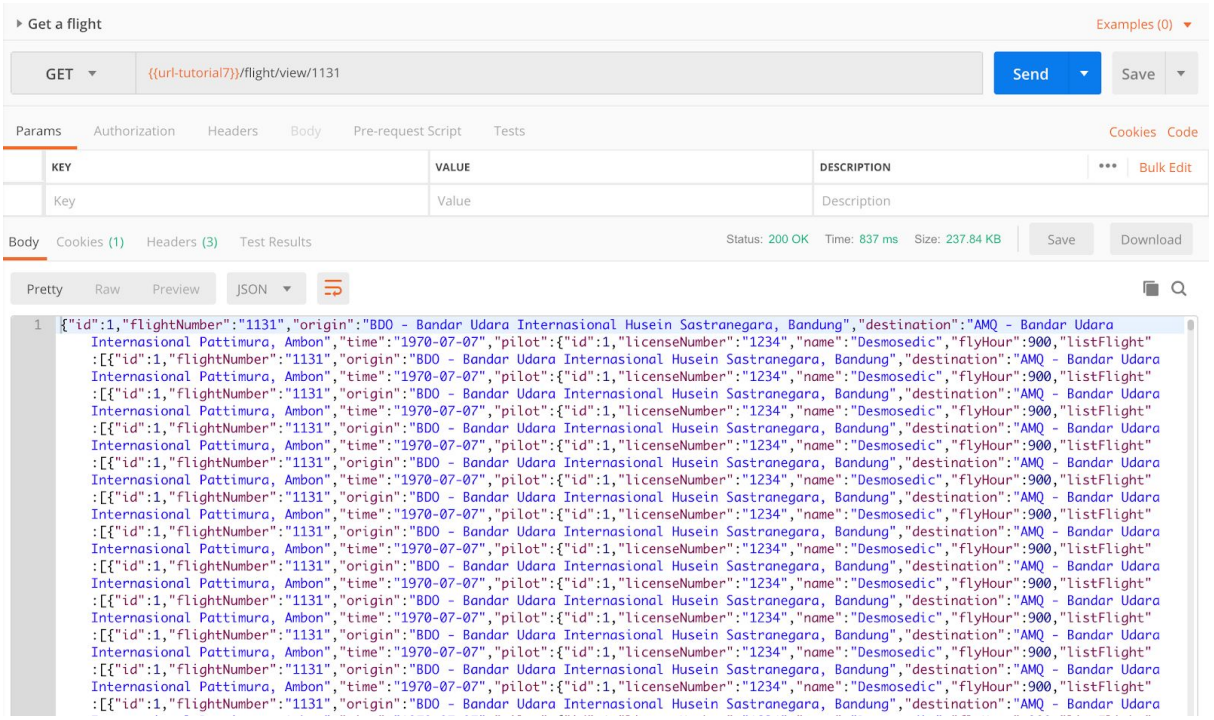
Add flight :



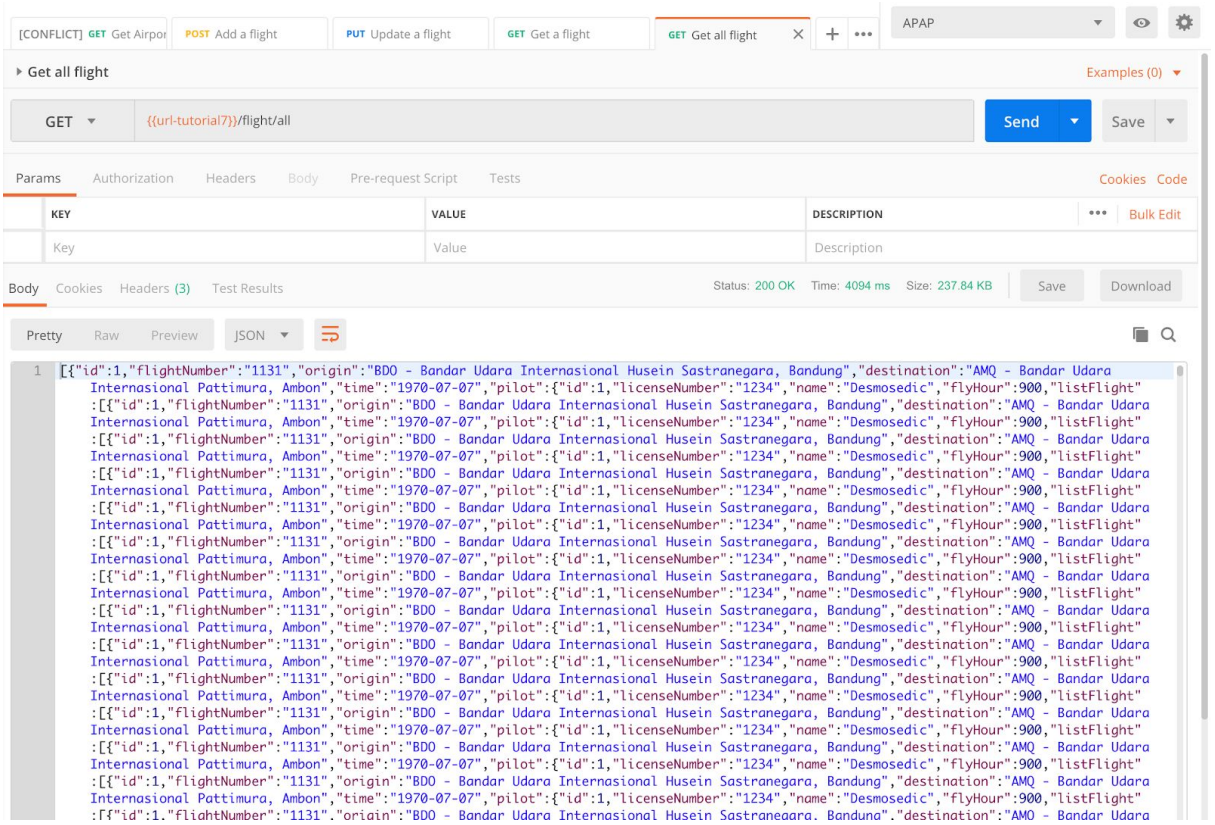
Update :



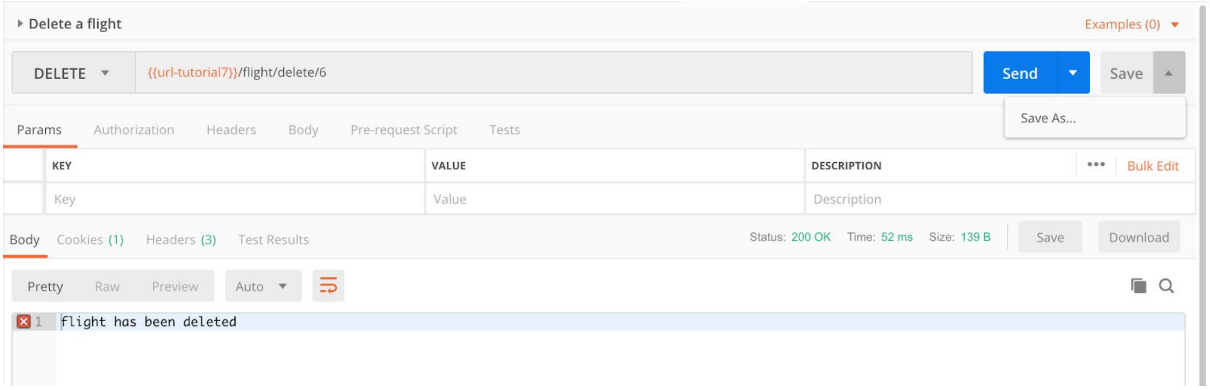
Search by flight number :



Get all flights :



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.

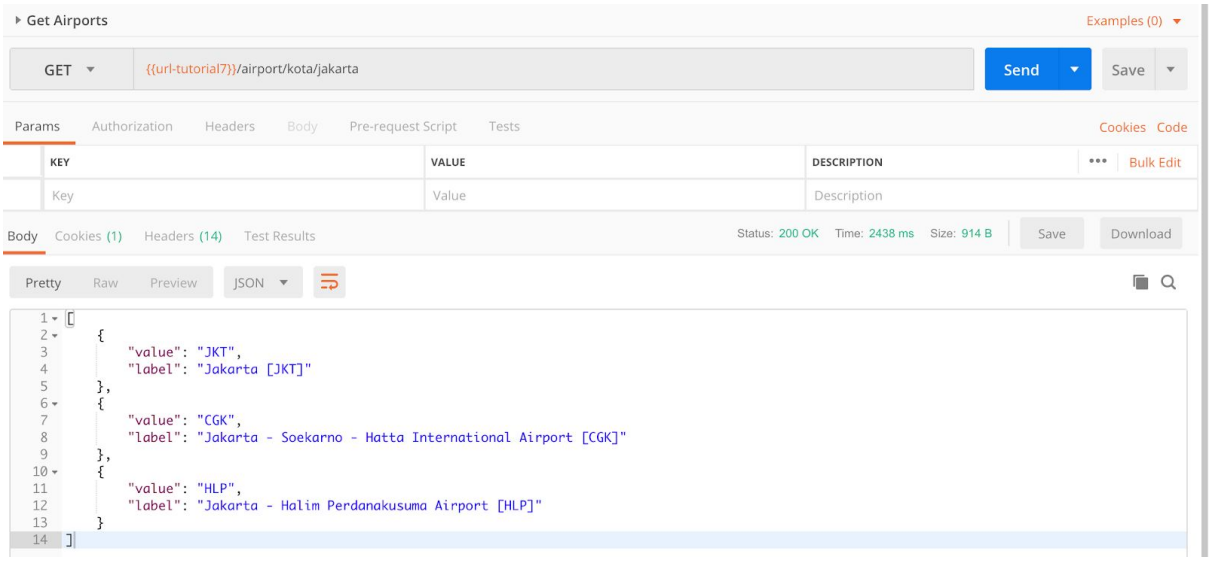
   Pertama-tama, tambahkan pada Setting line API key dan URL API untuk mendapatkan list airport

```java
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 = "XGGFKpGY9DlooRAAqqQ9dRwn1zqXmTmE";
}
```

   Setelah itu, implementasikan ada AirportController :

```java
@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
Bulk Edit
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
[
{
"value": "JKT",
"label": "Jakarta [JKT]"
},
{
"value": "CGK",
"label": "Jakarta - Soekarno - Hatta International Airport [CGK]"
},
{
"value": "HLP",
"label": "Jakarta - Halim Perdanakusuma Airport [HLP]"
}
]