

## File Write Up

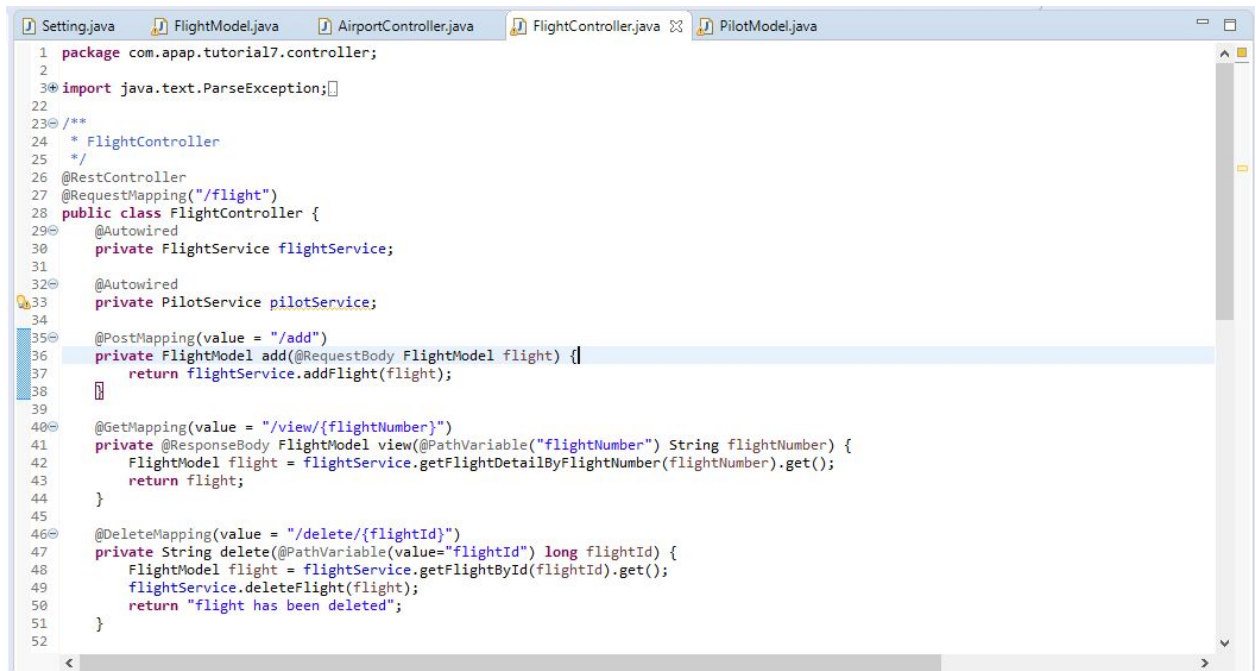
1. Hal yang saya pelajari cukup banyak pada tutorial ini, bagaimana membuat RESTful *Web Service* dengan menggunakan framework Spring Boot serta menggunakan Postman sebagai API development environment. Pertama-tama, membuat RestController yang akan handle request baik untuk Pilot, Flight, dan Airport (latihan). Setelah itu setup *environment* pada Postman yang nantinya akan melakukan *request* ke local server kita. Postman ini juga membantu kita dalam melakukan proses *debugging* dan *testing*.

*Web Service* disini ada dua yaitu Service Producer dan Service Consumer. Service Producer dibuat yang nantinya akan mem-*provide* service untuk *client/consumer*. Sedangkan, Service Consumer yang akan menggunakan service yang dibuat sebelumnya. Service Producer pada tutorial berupa *backend* sistem yang akan handle request dari *client*, dan *request* yang dilakukan *consumer* disini kita lakukan melalui Postman.

### 2. Latihan 1

Pada latihan 1, kita membuat Service Producer yang akan handle *request* flight, meliputi add, update, view dan view all, dan delete flight. Pembuatan service tersebut dimulai dengan membuat RestController untuk Flight sesuai dengan spesifikasi yang ada di soal:

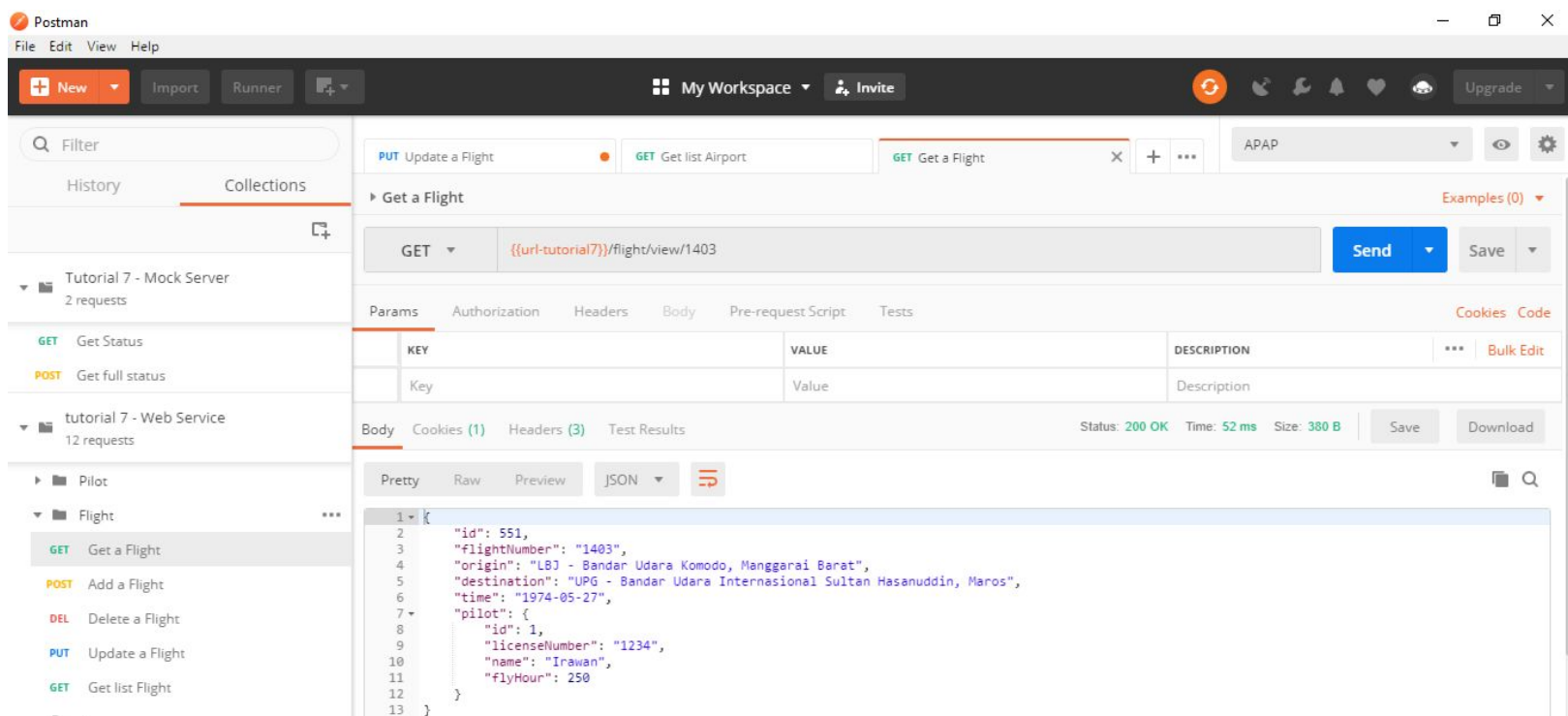
#### Rest Controller



```
1 package com.apap.tutorial17.controller;
2
3 import java.text.ParseException;
4
5 /**
6  * FlightController
7  */
8 @RestController
9 @RequestMapping("/flight")
10 public class FlightController {
11     @Autowired
12     private FlightService flightService;
13
14     @Autowired
15     private PilotService pilotService;
16
17     @PostMapping(value = "/add")
18     private FlightModel add(@RequestBody FlightModel flight) {
19         return flightService.addFlight(flight);
20     }
21
22     @GetMapping(value = "/view/{flightNumber}")
23     private @ResponseBody FlightModel view(@PathVariable("flightNumber") String flightNumber) {
24         FlightModel flight = flightService.getFlightDetailByFlightNumber(flightNumber).get();
25         return flight;
26     }
27
28     @DeleteMapping(value = "/delete/{flightId}")
29     private String delete(@PathVariable(value="flightId") long flightId) {
30         FlightModel flight = flightService.getFlightById(flightId).get();
31         flightService.deleteFlight(flight);
32         return "flight has been deleted";
33     }
34 }
```

```
Setting.java FlightModel.java AirportController.java FlightController.java PilotModel.java
53 @PutMapping(value = "/update/{flightId}")
54 private String updatePilotSubmit(
55     @PathVariable("flightId") long flightId,
56     @RequestParam(value="destination", required=false) String destination,
57     @RequestParam(value="origin", required=false) String origin,
58     @RequestParam(value="time", required=false) String time) throws ParseException { //format tanggal String "dd-MM-yyyy"
59
60     FlightModel flight = flightService.getFlightById(flightId).get();
61     if(flight.equals(null)) return "Couldn't find your flight";
62
63     if(origin.equals("")) {
64         //do nothing
65     }
66     else {
67         flight.setOrigin(origin);
68     }
69     if(destination.equals("")) {
70         //do nothing
71     }
72     else {
73         flight.setDestination(destination);
74     }
75     if(time.equals("")) {
76         // do nothing
77     }
78     else {
79         // convert String into java.sql.Date
80         SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
81         java.util.Date date = sdf.parse(time);
82         java.sql.Date sqlDate = new java.sql.Date(date.getTime());
83         flight.setTime(sqlDate);
84     }
85     flightService.updateFlight(flightId, flight);
86     return "flight update success";
}
```

Serta membuat *request* add, delete, update, view dan view all flight pada Postman dan eksekusi *request* tersebut:



Postman

File Edit View Help

New Import Runner +

My Workspace Invite

Filter

History Collections

Tutorial 7 - Mock Server  
2 requests

GET Get Status  
POST Get full status

tutorial 7 - Web Service  
12 requests

Pilot

Flight

GET Get a Flight  
POST Add a Flight  
DEL Delete a Flight  
PUT Update a Flight  
GET Get list Flight

Airport

GET Get list Airport

PUT Update a Flight GET Get list Airport GET Get a Flight POST Add a Flight

POST {{url-tutorial7}}/flight/add

Send Save

```
1 {
2   "flightNumber": "1000",
3   "origin": "Jakarta",
4   "destination": "Singapore",
5   "time": "2018-10-10",
6   "pilot": {"id": 1, "flyHour": 250, "licenseNumber": "1234", "name": "Irawan"}
7 }
```

Body Cookies (1) Headers (3) Test Results

Status: 200 OK Time: 195 ms Size: 297 B Save Download

Pretty Raw Preview JSON

```
1 {
2   "id": 565,
3   "flightNumber": "1000",
4   "origin": "Jakarta",
5   "destination": "Singapore",
6   "time": "2018-10-10",
7   "pilot": {
```

Postman

File Edit View Help

New Import Runner +

My Workspace Invite

Filter

History Collections

Tutorial 7 - Mock Server  
2 requests

GET Get Status  
POST Get full status

tutorial 7 - Web Service  
12 requests

Pilot

Flight

GET Get a Flight  
POST Add a Flight  
DEL Delete a Flight  
PUT Update a Flight  
GET Get list Flight

Airport

GET Get list Airport

PUT Update a Flight GET Get list Airport GET Get a Flight POST Add a Flight DEL Delete a Flight

DELETE {{url-tutorial7}}/flight/delete/565

Send Save

Params Authorization Headers Body Pre-request Script Tests

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies (1) Headers (3) Test Results

Status: 200 OK Time: 134 ms Size: 139 B Save Download

Pretty Raw Preview Auto

```
1 flight has been deleted
```

Postman

File Edit View Help

New Import Runner

My Workspace Invite

Filter

History Collections

Tutorial 7 - Mock Server  
2 requests

GET Get Status

POST Get full status

tutorial 7 - Web Service  
12 requests

Pilot

Flight

GET Get a Flight

POST Add a Flight

DEL Delete a Flight

PUT Update a Flight

GET Get list Flight

Airport

GET Get list Airport

PUT Update a Flight

GET Get list Airport

GET Get a Flight

POST Add a Flight

DEL Delete a Flight

GET Get list Flight

APAP

Send Save

Params Authorization Headers Body Pre-request Script Tests

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies (1) Headers (3) Test Results

Status: 200 OK Time: 37 ms Size: 144.69 KB Save Download

Pretty Raw Preview JSON

```
1 [
2   {
3     "id": 1,
4     "flightNumber": "1131",
5     "origin": "BDO - Bandar Udara Internasional Husein Sastranegara, Bandung",
6     "destination": "AMQ - Bandar Udara Internasional Pattimura, Ambon",
7     "time": "1970-07-07",
8     "pilot": {
9       "id": 1,
10      "licenseNumber": "1234",
11      "name": "Irawan",
12      "flyHour": 250
13    }
14  },
15  {
16    "id": 2,
17    "flightNumber": "1151",
18    "origin": "SRG - Bandar Udara Internasional Achmad Yani, Semarang",
19    "destination": "AMQ - Bandar Udara Internasional Pattimura, Ambon",
20    "time": "1961-11-10",
21    "pilot": {
22      "id": 1,
```

Postman

File Edit View Help

New Import Runner

My Workspace Invite

Filter

History Collections

Tutorial 7 - Mock Server  
2 requests

GET Get Status

POST Get full status

tutorial 7 - Web Service  
12 requests

Pilot

Flight

GET Get a Flight

POST Add a Flight

DEL Delete a Flight

PUT Update a Flight

GET Get list Flight

Airport

GET Get list Airport

PUT Update a Flight

GET Get list Airport

PUT Update a Flight

GET Get list Airport

GET Get a Flight

POST Add a Flight

DEL Delete a Flight

PUT Update a Flight

APAP

Send Save

Params Authorization Headers Body Pre-request Script Tests

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> destination	Bandung	
<input checked="" type="checkbox"/> origin	Surabaya	
<input checked="" type="checkbox"/> time	01-11-2018	
Key	Value	Description

Body Cookies (1) Headers (3) Test Results

Status: 200 OK Time: 115 ms Size: 137 B Save Download

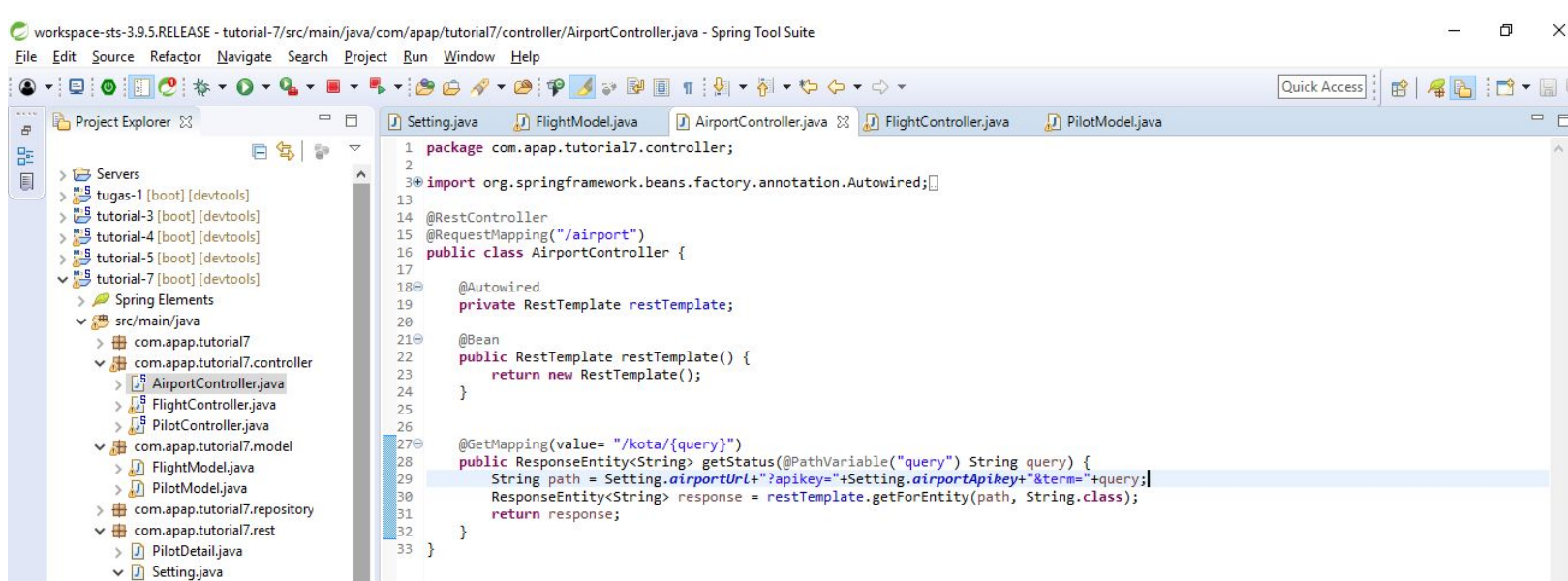
Pretty Raw Preview Auto

```
1 Flight update success
```

### 3. Latihan 2

Pada latihan 2, kita diminta untuk mengimplementasikan *Web Service* dengan menggunakan API dari <https://sandbox.amadeus.com/travel-innovationsandbox/apis/get/airports/autocomplete>. Pertama-tama, kita membuat RestController untuk Airport yang nantinya akan di-request. Setelah itu, kita menambahkan Setting url pada Setting.java berupa url dari sandbox di atas dan API key yang didapat dengan membuat aplikasi yang kita buat di website tersebut.

RestController :



Setting url pada Setting.java :

```
public class Setting {
    final public static String pilotUrl = "https://61560fa4-0952-4c34-b297-d90d4168e442.mock.pstmn.io";
    final public static String airportUrl = "https://api.sandbox.amadeus.com/v1.2/airports/autocomplete";
    final public static String airportApiKey = "xQhKcAILqP4C2PrmknhvzYOE8XAV4rWi";
}
```



Membuat *request* pada Postman untuk melakukan memanggil request Airport tersebut dan eksekusi request Airport dengan spesifikasi yang diminta dari soal sebagai berikut :

