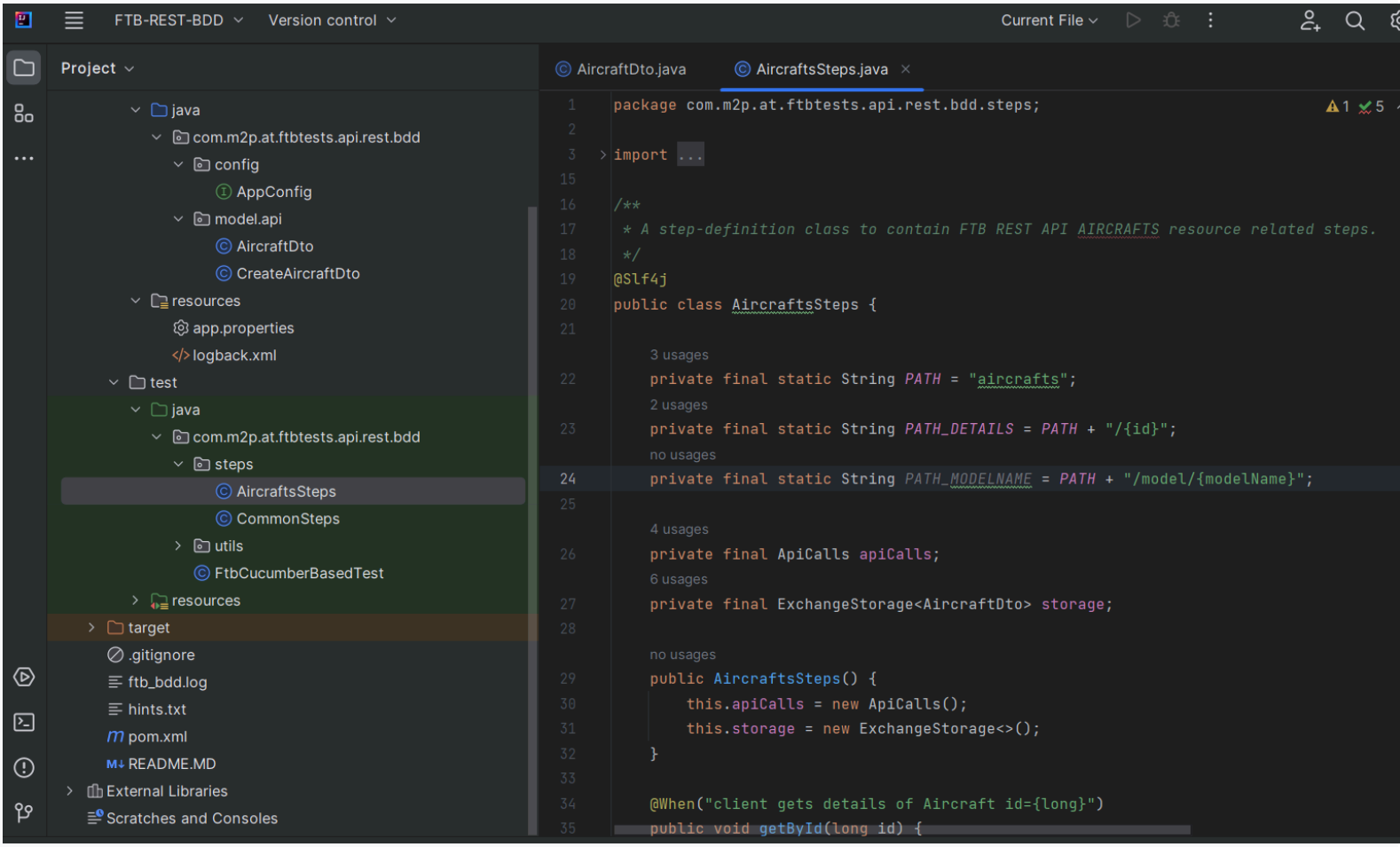
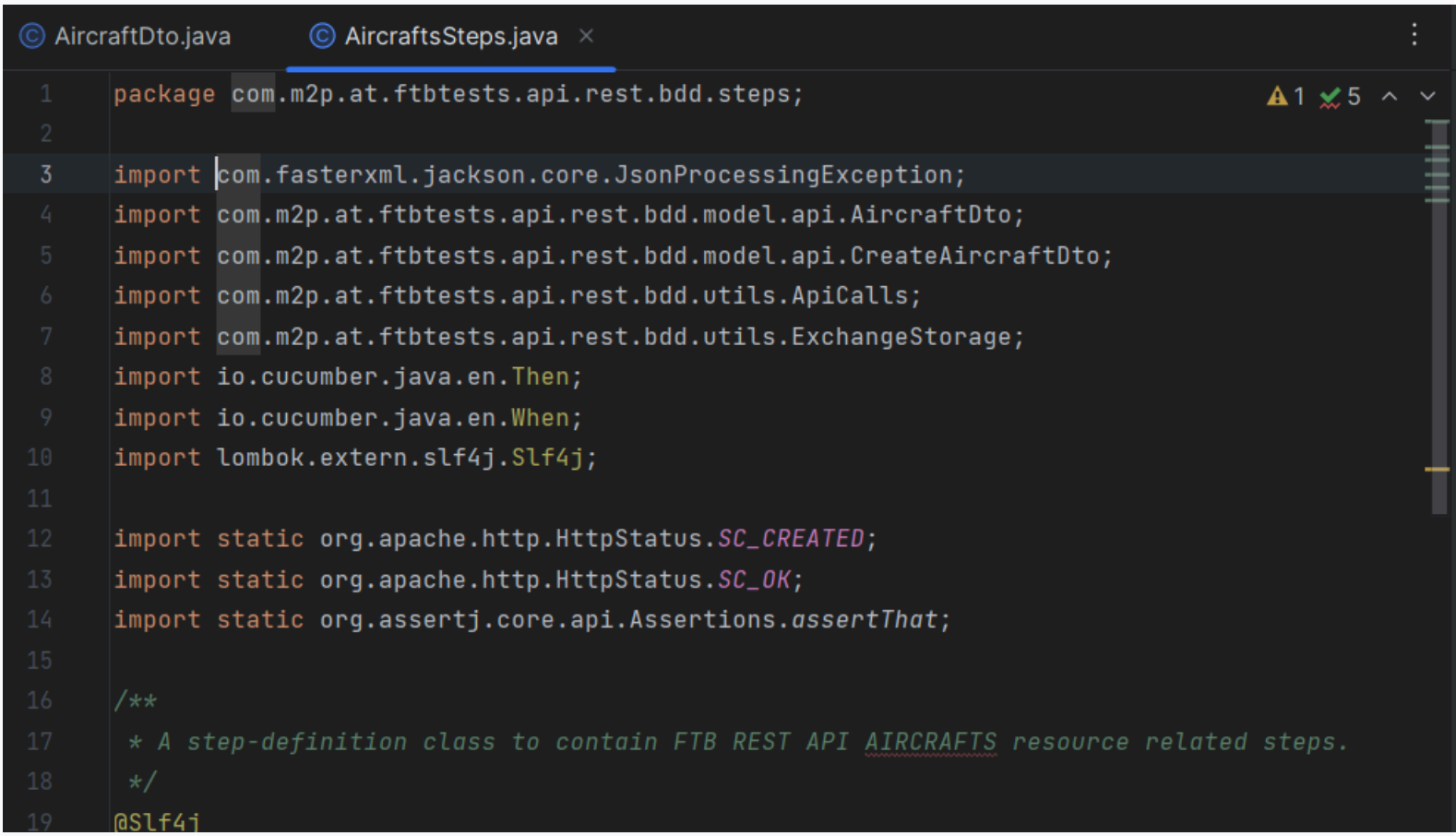
Task: For the project, attached to the LMS system, do the following:

1) Understand the AircraftSteps module

We need to open Intellij Idea



This module imports all necessary information



```java
package com.m2p.at.ftbtests.api.rest.bdd.steps;

import com.fasterxml.jackson.core.JsonProcessingException;
import com.m2p.at.ftbtests.api.rest.bdd.model.api.AircraftDto;
import com.m2p.at.ftbtests.api.rest.bdd.model.api.CreateAircraftDto;
import com.m2p.at.ftbtests.api.rest.bdd.utils.ApiCalls;
import com.m2p.at.ftbtests.api.rest.bdd.utils.ExchangeStorage;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;
import lombok.extern.slf4j.Slf4j;

import static org.apache.http.HttpStatus.SC_CREATED;
import static org.apache.http.HttpStatus.SC_OK;
import static org.assertj.core.api.Assertions.assertThat;

/**
 * A step-definition class to contain FTB REST API AIRCRAFTS resource related steps.
 */
@Slf4j
```

This info will be used later in this module.

Class Declaration: The AircraftsSteps class is declared, which contains step definitions for the API interactions

Public class we can call this class from other places and we have private attributes - we can use them inside the class.

```java
public class AircraftsSteps {

    3 usages
    private final static String PATH = "aircrafts";
    2 usages
    private final static String PATH_DETAILS = PATH + "/{id}";
    no usages
    private final static String PATH_MODELNAME = PATH + "/model/{modelName}";

    4 usages
    private final ApiCalls apiCalls;
    6 usages
    private final ExchangeStorage<AircraftDto> storage;

    no usages
    public AircraftsSteps() {
        this.apiCalls = new ApiCalls();
        this.storage = new ExchangeStorage<>();
    }
```

Constants such as PATH, PATH_DETAILS (additional info about id), and PATH_MODELNAME are declared, which represent the endpoints for different API calls

ApiCalls - is used to make API calls and ExchangeStorage is used to store responses.

Storages are the places where info will be stored

```java
@When("client gets details of Aircraft id={long}")
public void getById(long id) {
    var response = apiCalls.doGet(SC_OK, AircraftDto.class, PATH_DETAILS, String.valueOf(id));
    storage.setLastApiCallSingleItemResponse(response);
}

@When("client gets details of just created Aircraft")
public void getJustCreated() {
    var response = apiCalls
            .doGet(SC_OK, AircraftDto.class, PATH_DETAILS,
                    storage.getLastApiCallSingleItemResponse().getAircraftId());
    storage.setLastApiCallSingleItemResponse(response);
}

@When("client tries to create an Aircraft having manufacturer={string} and model={string} and number of seats={
public void create(String manufacturer, String model, Integer numberOfSeats) throws JsonProcessingException {
    var data = CreateAircraftDto.of()
            .manufacturer(manufacturer)
            .model(model)
            .numberOfSeats(numberOfSeats)
            .build();

    var response = apiCalls.doPost(SC_CREATED, AircraftDto.class, data, PATH);
    storage.setLastApiCallSingleItemResponse(response);
}
```

When :

getById: "client gets details of Aircraft id={long}". It performs a GET request to retrieve details of an aircraft by its ID.

getJustCreated: "client gets details of just created Aircraft". It performs a GET request to retrieve details of the aircraft that was just created.

Create: "client tries to create an Aircraft having manufacturer={string} and model={string} and number of seats={int}". It creates a new aircraft by sending a POST request with manufacturer, model, and number of seats data.

```java
    @Then("aircraft data to be manufacturer={string} and model={string} and number of seats={int}")
    @Then("returned aircraft data to be manufacturer={string} and model={string} and number of seats={int}")
    public void verifySingleAircraftData(String manufacturer, String model, Integer numberOfSeats) {
        var lastResponse = storage.getLastApiCallSingleItemResponse();
        assertThat(lastResponse.getManufacturer()) AbstractStringAssert<capture of ?>
                .as( description: "Seems Aircraft response contained unexpected manufacturer value.") capture of ?
                .isEqualTo(manufacturer);
        assertThat(lastResponse.getModel()) AbstractStringAssert<capture of ?>
                .as( description: "Seems Aircraft response contained unexpected model value.") capture of ?
                .isEqualTo(model);
        assertThat(lastResponse.getNumberOfSeats()) AbstractIntegerAssert<capture of ?>
                .as( description: "Seems Aircraft response contained unexpected number-of-seats value.") capture of ?
                .isEqualTo(numberOfSeats);
    }
}
```

Then :

“client tries to create an Aircraft having manufacturer={string} and model={string} and number of seats={int}”

"returned aircraft data to be manufacturer={string} and model={string} and number of seats={int}"

verifySingleAircraftData: This step definition verifies the data of the single aircraft returned in the response. It compares the manufacturer, model, and number of seats with the expected values provided in the DB

And different responses are shown because of different data:

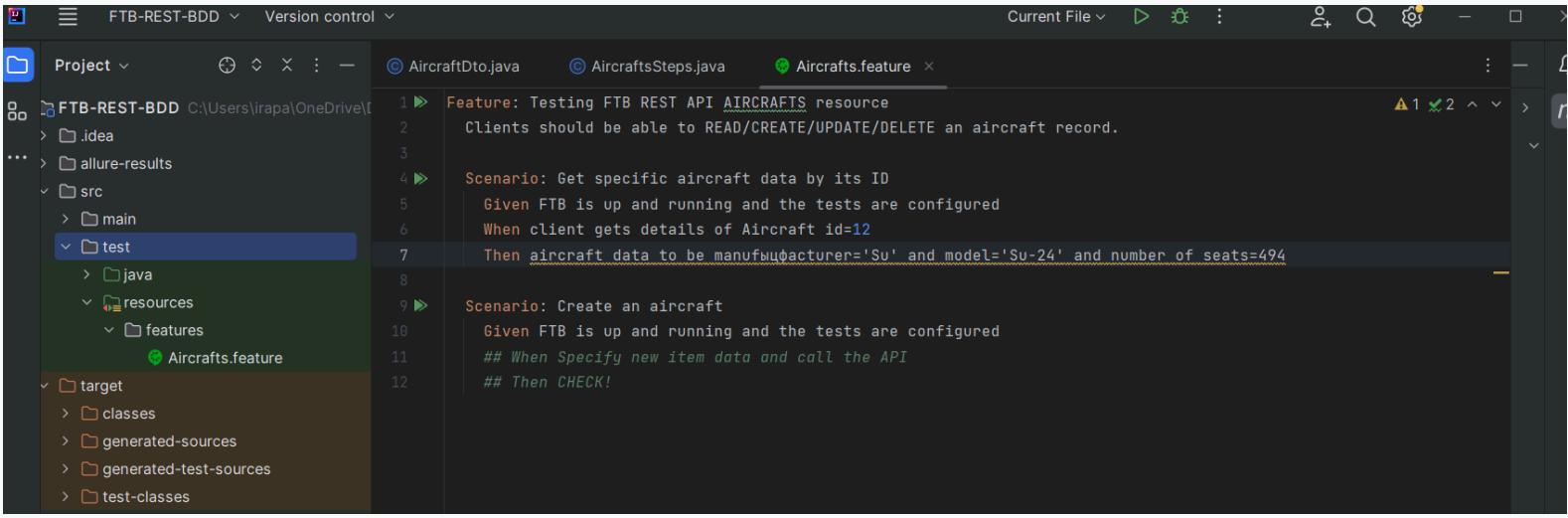
"Seems Aircraft response contained unexpected manufacturer value."

"Seems Aircraft response contained unexpected model value."

"Seems Aircraft response contained unexpected number-of-seats value."

This AircraftSteps module help to test the functionality of an API that manages aircraft resources, including retrieving aircraft details, creating new aircraft, and verifying the data of returned aircraft.

2) Understand the Aircrafts.feature module



Feature is named: Testing FTB REST API AIRCRAFTS resource

Clients should be able to READ/CREATE/UPDATE/DELETE an aircraft record.

We have 2 Scenarios, with standard steps for Gherkin: Scenario, Given, When, Then.

Scenario 1: Get specific aircraft data by its ID:

Given FTB is up and running and the tests are configured

When client gets details of Aircraft id=12

Then aircraft data to be manufacturer='Su' and model='Su-24' and number of seats=494

Scenario 2: Create an aircraft

Given FTB is up and running and the tests are configured

this part is commented out and is not executed:

When Specify new item data and call the API

Then CHECK!

This module defines two scenarios to test the functionality of an API related to aircraft resources: retrieving specific aircraft data by ID and creating an aircraft. In the second scenario steps to specify the creation and checking of the aircraft are commented and will not be executed.

3) Try to write the following test:

- **add an aircraft with number_of_seats = null to your DB**
- **Write a test that checks this aircraft the same way as it is written in the current project.**

.

The planned steps are:

1) Add an aircraft with seats = null (using INSERT to the DB)

let us insert aircraft with number_of_seats=NULL

Aircraft_id is auto_increment

```
INSERT INTO ftb_ipapara.aircraft
( manufacturer, model, number_of_seats)
VALUES( 'Tupolev', 'Tu-214', NULL);
```

We can see it in DB

	aircraft_id	manufacturer	model	number_of_seats
1060	1,136	[NULL]	[NULL]	[NULL]
1061	1,137	[NULL]	[NULL]	[NULL]
1062	1,138	[NULL]	[NULL]	[NULL]
1063	1,139	[NULL]	[NULL]	[NULL]
1064	1,140	[NULL]	[NULL]	[NULL]
1065	1,141	[NULL]	[NULL]	[NULL]
1066	1,142	[NULL]	[NULL]	[NULL]
1067	1,143	[NULL]	[NULL]	[NULL]
1068	1,144	[NULL]	[NULL]	[NULL]
1069	1,145	string	string	1,000
1070	1,146	string	string	1,000
1071	1,147	Boeing	Boeing 201	180
1072	1,148	Boeing	Boeing 307	100
1073	1,149	Boeing	Boeing 201	180
1074	1,150	Boeing	Boeing 201	180
1075	1,151	Boeing	Boeing 201	180
1076	1,152	1231231	2123131	111
1077	1,153			[NULL]
1078	1,154	Tupolev	Tu-214	[NULL]
1079	1,155	Tupolev	Tu-214	[NULL]

2) Get this aircraft by id, using existing step (check that this is possible)

```
@When("client gets details of Aircraft id={long}")
    public void getById(long id) {
        var response = apiCalls.doGet(SC_OK, AircraftDto.class, PATH_DETAILS, String.valueOf(id));
        storage.setLastApiCallSingleItemResponse(response);
    }
```

3) Use the step @Then("returned aircraft data to be manufacturer={string} and model={string} and number of seats={int}"), get an error (Red test)

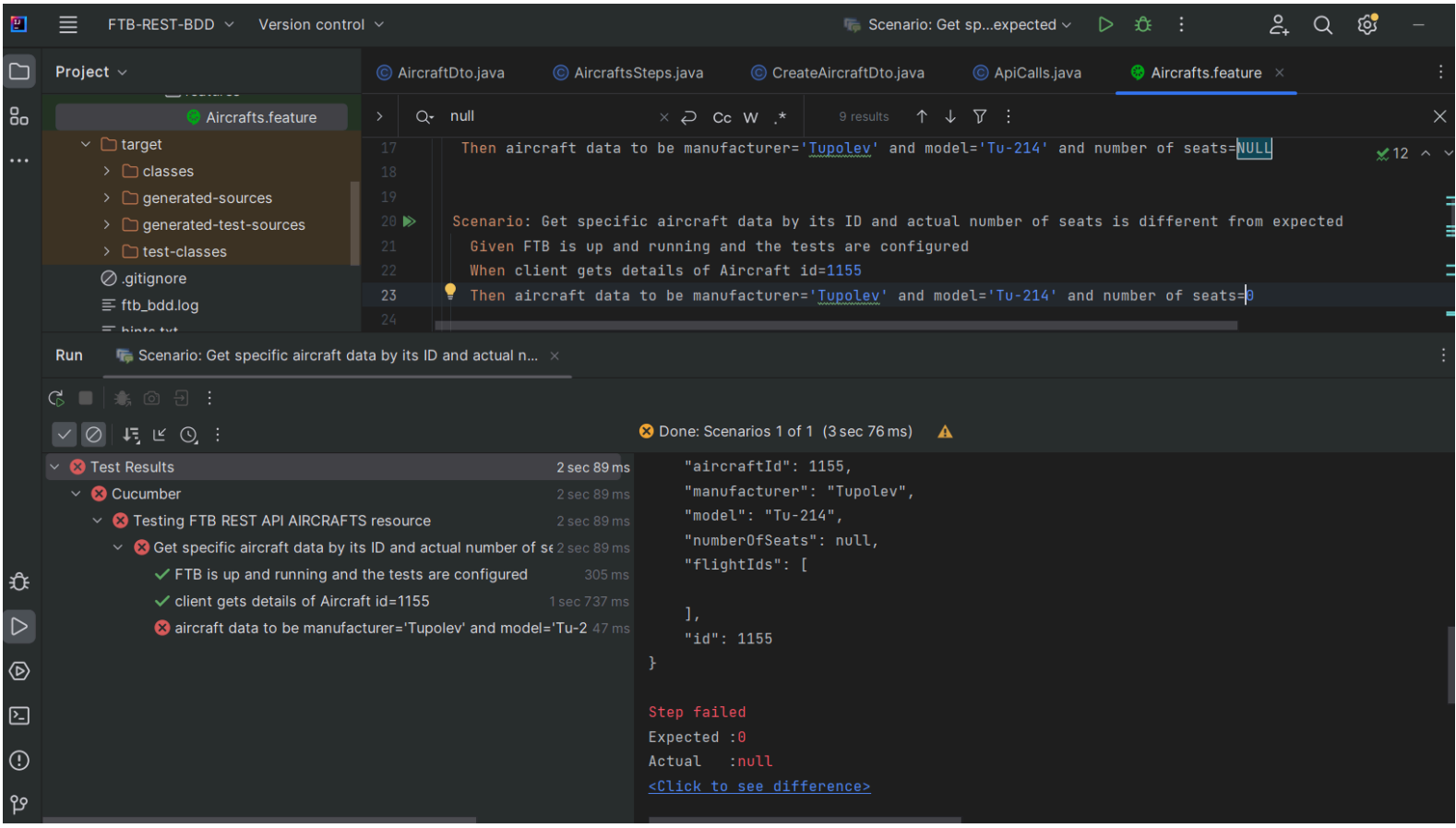
```
@Then("aircraft data to be manufacturer={string} and model={string} and number of seats={int}")
@Then("returned aircraft data to be manufacturer={string} and model={string} and number of seats={int}")

    public void verifySingleAircraftData(String manufacturer, String model, Integer numberOfSeats) {

        var lastResponse = storage.getLastApiCallSingleItemResponse();

        assertThat(lastResponse.getManufacturer())

                .as("Seems Aircraft response contained unexpected manufacturer value.")

                .isEqualTo(manufacturer);
```

```
        assertThat(lastResponse.getModel())
                .as("Seems Aircraft response contained unexpected model value.")
                .isEqualTo(model);
        assertThat(lastResponse.getNumberOfSeats())
                .as("Seems Aircraft response contained unexpected number-of-seats value.")
                .isEqualTo(numberOfSeats);
    }
```

Let us write a scenario .

Let us get an error (Red test) in this test

```
Scenario: Get specific aircraft data by its ID and actual number of seats is different from expected
  Given FTB is up and running and the tests are configured
  When client gets details of Aircraft id=1155
  Then aircraft data to be manufacturer='Tupolev' and model='Tu-214' and number of seats=0
```



**** 4) Try to write a test that creates an aircraft with number_of_seats = NULL**

Your project should be put to the GitHub.

Attach a link to your GitHub project to your homework assignment in Moodle

Scenario: Create an aircraft with number of seats=NULL

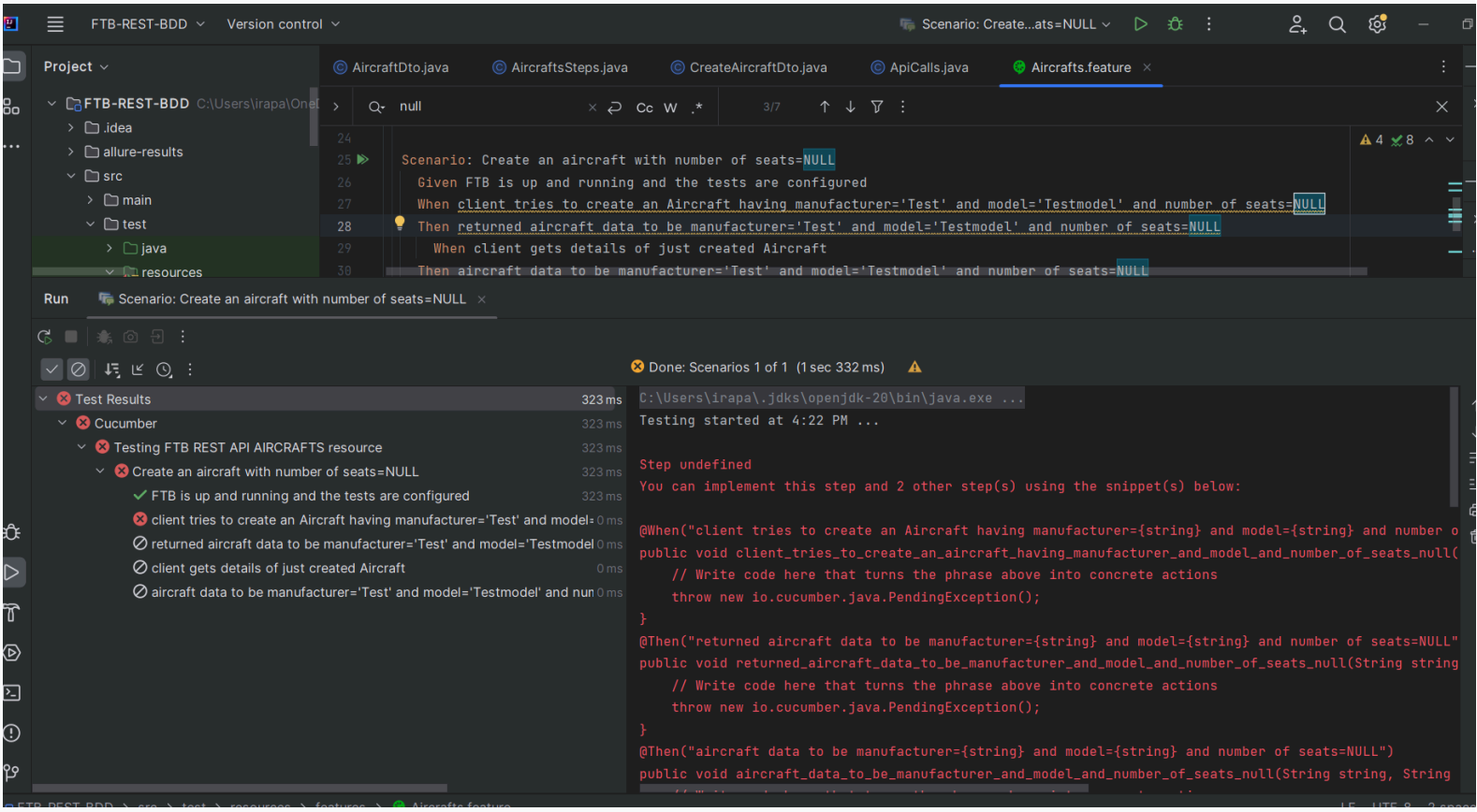
Given FTB is up and running and the tests are configured

When client tries to create an Aircraft having manufacturer='Test' and model='Testmodel' and number of seats=NULL

Then returned aircraft data to be manufacturer='Test' and model='Testmodel' and number of seats=NULL

When client gets details of just created Aircraft

Then aircraft data to be manufacturer='Test' and model='Testmodel' and number of seats=NULL



Smth is wrong. As we can see in message about mistake

It seems to me that ,we need to change integer for NULL in all places in Aircraftsteps.java

1 @When("client tries to create an Aircraft having manufacturer={string} and model={string} and number of seats={int}")

public void create(String manufacturer, String model, Integer numberOfSeats) throws JsonProcessingException {

var data = CreateAircraftDto.of()

.manufacturer(manufacturer)

.model(model)

.numberOfSeats(numberOfSeats)

```
                .build();
        var response = apiCalls.doPost(SC_CREATED, AircraftDto.class, data, PATH);
        storage.setLastApiCallSingleItemResponse(response);
    }and the scenario does not work.


2 @Then("aircraft data to be manufacturer={string} and model={string} and number of seats={int}")
    @Then("returned aircraft data to be manufacturer={string} and model={string} and number of seats={int}")
    public void verifySingleAircraftData(String manufacturer, String model, Integer numberOfSeats) {
        var lastResponse = storage.getLastApiCallSingleItemResponse();
        assertThat(lastResponse.getManufacturer())
                .as("Seems Aircraft response contained unexpected manufacturer value.")
                .isEqualTo(manufacturer);
        assertThat(lastResponse.getModel())
                .as("Seems Aircraft response contained unexpected model value.")
                .isEqualTo(model);
        assertThat(lastResponse.getNumberOfSeats())
                .as("Seems Aircraft response contained unexpected number-of-seats value.")
                .isEqualTo(numberOfSeats);
    }
```

https://github.com/irapapara/3task **27HW-FTB-REST-BDD.zip**