**Start and End Sort API Automation**

**StationController\_StartandEndSort.feature**

Feature: Station Controller - StartSort

TestCase 1

@StartSort\_01 @StartSort

Scenario Outline: Get the revision id from dock-maintenance-tool-controller

Given The base URI is up and running for dock-maintenance-tool-controller

When User should see the details of dock-maintenance of "<revision>" "<row>"

Then User should see the status code <statusCode>

And The base URI is up and running for StartSort

When User send data for "<facilityId>" and "<overrideFlag>"

Then User should see the status code <statusCode>

And User should see the message "<message>"

Given The base URI is up and running for EndSort

When User send data for "<facilityId>"

Then User should see the status code <statusCode>

Then User should see the message "<message>"

Examples:

|revision|row|statusCode|facilityId|overrideFlag|message|

|7805\_14 |0 |200 |4545 |true |success|

###################################################################################################

#Test Case 2

@StartSort\_02 @StartSort

Scenario Outline: Get the revision id from dock-maintenance-tool-controller for

Given The base URI is up and runnig for dok-maintenance-tool-controller

When User should see the details of dock-maintenance of "<revision>" "<row>"

Then User should see the status code <statusCode>

And The base URI is up and running for StartSort

When User send data for "<facilityId>" and "<overrideFlag>"

Then User should see the status code <statusCode>

And User should see the message "<message>"

And The base URI is up and running for EndSort

When User send data for "<facilityId>"

Then User should see the status code <statusCode>

Then User should see the message "<message>"

Examples:

|revision|row|statusCode|facilityId|overrideFlag|message|

|9728\_4 |0 |200 |0436 |true |success|

##################################################################################################

#Negative Test Scenarios

#Test Case 3

@StartSort\_03 @StartSort

Scenario Outline: Verify user able to start and endsort with invalid data

Given The base URI is up and runnig for dok-maintenance-tool-controller

When User should see the details of dock-maintenance of "<revision>" "<row>"

Given The base URI is up and running for StartSort

When User send data for "<facilityId>" and "<overrideFlag>"

Then User should see the status code <statusCode>

Then verify message invalid "<failureMessage>"

Examples:

|revision|row|facilityId|overrideFlag|failureMessage |statusCode|

|7805\_14 |0 |454 |true |No Station was found with Id: 454 |400 |

|7805\_14 |0 |0000 |true |No Station was found with Id: 0000 |400 |

###################################################################################################

#Negative Test Scenarios

#Test Case 3

@StartSort\_04 @StartSort

Scenario Outline: Verify user able to start and endsort with invalid data

Given The base URI is up and running for StartSort

When User send data for "<facilityId>" and "<overrideFlag>"

Then User should see the status code <statusCode>

Then verify message invalid "<failureMessage>"

Examples:

|facilityId|overrideFlag|statusCode|failureMessage |

|4545 |true |400 |Revision null is not found at facility Id: 4545|

|5531 |true |400 |Revision null is not found at facility Id: 5531|

**StationController\_getActiveSort.feature**

Feature: Station Controller - getActiveSort

#Test Case 1

@getActiveSort\_TS01 @StartSort

Scenario Outline: Get the active sort

Given The base URI is up and running for dock-maintenance-tool-controller

When User should see the details of dock-maintenance of "<revision>" "<row>"

Then User should see the status code <statusCode>

And The base URI is up and running for StartSort

When User send data for "<facilityId>" and "<overrideFlag>"

Then User should see the status code <statusCode>

And User should see the message "<message>"

Given The base URI is up and running for getActiveSort

Then User should see the details as "<revision>" "<facilityId>" "<dockName>"

And User should see the status code <statusCode>

And The base URI is up and running for EndSort

When User send data for "<facilityId>"

Then User should see the status code <statusCode>

Examples:

|revision|row|statusCode|facilityId|overrideFlag|message|dockName |

|7805\_14 |0 |200 |4545 |true |success|SHIPYARDTESTDOCKONE9-11-20 |

**StationController\_getStations.feature**

Feature: Station Controller - getStations

#TestCase 1

@getStations @StartSort

Scenario Outline: Verify station controller getStations

Given The BaseURI is up and running for StationController getStation

When User sees the details of stations as "<facilityId>" and "<facilityName>" for "<row>"

Then User should see the status code <statusCode>

Examples:

|facilityId|facilityName|row|statusCode|

|0436 |TOLE |0 |200 |

|0462 |LAWL |1 |200 |

|0461 |ABCD |2 |200 |

|0405 |ABCD |2 |200 |

|5531 |SITE |2 |200 |

|4545 |SHPY |2 |200 |

|4646 |POST |2 |200 |

|4747 |SPDT |2 |200 |

|0152 |PITT |2 |200 |

|0183 |SSTB |2 |200 |

**Env.properties**

#Get Stations URl(Dev)

stationcontroller.dev.getstations=https://uss-development.app.wtcdev1.paas.fedex.com/stations

#start sort post DEV url

stationcontroller.dev.startSort=https://uss-development.app.wtcdev1.paas.fedex.com/station/start

#End Sort DEV URL

stationcontroller.dev.endSort=https://uss-development.app.wtcdev1.paas.fedex.com/station/end

#dock-maintenance-tool-controller DEV URL

dock-maintenance-tool-controller.dev.returnDockMetaData=https://uss-development.app.wtcdev1.paas.fedex.com/stations/4545/DockMetadata

#dock-mainteance-tool-controller

dock-maintenance-tool-controller-dev=https://uss-development.app.wtcdev1.paas.fedex.com/stations/0436/DockMetadata

#getActiveSort

stationController.dev.getActiveSort=https://uss-development.app.wtcdev1.paas.fedex.com/station/4545/dock

**Resources**:

**StationController.java**

**package** com.fedex.ground.uss.cucumber.ServiceValidation;

**public** **class** StationController {

**public** **static** String startSort(String facilityId,String overrideFlag, String primeTime, String revision,

String unloadTime) {

String startSort= "{ \r\n" +

" \"facilityId\":\""+facilityId+"\",\r\n" +

" \"overrideFlag\":\""+overrideFlag+"\",\r\n" +

" \"primeTime\":\""+primeTime+"\",\r\n" +

" \"revision\":\""+revision+"\",\r\n" +

" \"unloadTime\":\""+unloadTime+"\"\r\n" +

"}";

**return** startSort;

}

**public** **static** String startSortNegative(String facilityId,String overrideFlag,String revision

) {

String startSort= "{ \r\n" +

" \"facilityId\":\""+facilityId+"\",\r\n" +

" \"overrideFlag\":\""+overrideFlag+"\",\r\n" +

" \"revision\":\""+revision+"\",\r\n" +

"}";

**return** startSort;

}

**public** **static** String endSort(String facilityId)

{

String endSort= "{ \r\n" +

" \"facilityId\":\""+facilityId+"\"\r\n" +

"}";

**return** endSort;

}

**public** **static** String getActiveSort(String facilityId)

{

String getActiveSort ="{"

+ "\r\n" +

" \"facilityId\":\""+facilityId+"\"\r\n" +

"}";

**return** getActiveSort;

}

}

**StepDefinitions:**

**package com.fedex.ground.uss.cucumber.stepdefinition;**

**import cucumber.api.java.en.And;**

**import cucumber.api.java.en.Given;**

**import cucumber.api.java.en.Then;**

**import cucumber.api.java.en.When;**

**import io.restassured.RestAssured;**

**import io.restassured.response.Response;**

**import io.restassured.response.ValidatableResponse;**

**import org.slf4j.Logger;**

**import org.slf4j.LoggerFactory;**

**import com.fedex.ground.uss.cucumber.ServiceValidation.PackageController;**

**import com.fedex.ground.uss.cucumber.ServiceValidation.Resources;**

**import com.fedex.ground.uss.cucumber.ServiceValidation.StationController;**

**import static com.fedex.ground.uss.cucumber.utils.ConfigFileReader.ZipToDestAPIProp;**

**import io.restassured.http.ContentType;**

**import static io.restassured.RestAssured.given;**

**import static org.junit.Assert.assertEquals;**

**import java.text.SimpleDateFormat;**

**import java.util.Calendar;**

**import org.json.JSONException;**

**public class USSAPIStepDefinition {**

**Response response;**

**ValidatableResponse json;**

**int zipcode, zipcodeModify;**

**String revision\_JSON;**

**Calendar calender = Calendar.getInstance();**

**SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd.HH:mm:ss");**

**public static Logger log = LoggerFactory.getLogger(USSAPIStepDefinition.class.getName());**

**@Given("^The base URI is up and running for demo URL$")**

**public void the\_base\_URI\_is\_up\_and\_running() {**

**RestAssured.baseURI = ZipToDestAPIProp.getProperty("packagecontroller.lcl.assign");**

**log.info("Base URL is up and Running: " + ZipToDestAPIProp.getProperty("packagecontroller.lcl.assign"));**

**}**

**@Then("^User should see the status code (\\d+)$")**

**public void user\_should\_see\_the\_status\_code(int statuscode) {**

**json = response.then().assertThat().statusCode(statuscode);**

**System.out.println("Status code is:" + statuscode);**

**json.log().all().extract();**

**}**

**@When("^User sends data to assign package \"([^\"]\*)\" \"([^\"]\*)\" \"([^\"]\*)\" \"([^\"]\*)\" \"([^\"]\*)\"$")**

**public void User\_sends\_data\_to\_assign\_package(String chuteid, String packagebarcode, String sitename, String userid,**

**String userProvidedInput) {**

**response = given()**

**.contentType(ContentType.JSON).accept(ContentType.JSON).body(PackageController**

**.createPackageAssign(chuteid, packagebarcode, sitename, userid, userProvidedInput))**

**.when().post(Resources.packagecontroller\_assign);**

**}**

**@And("^Verify User is able to get \"([^\"]\*)\" and \"([^\"]\*)\"$")**

**public void verifySiteName(String siteNameExpected, String doorIDExpected) {**

**String siteNameActual = response.jsonPath().getString("ncScanLabel.currentSiteName");**

**System.out.println("Current SiteName is: " + siteNameExpected);**

**assertEquals(siteNameExpected, siteNameActual);**

**String doorIdActual = response.jsonPath().getString("ncScanLabel.doorId");**

**System.out.println("Current door ID is: " + doorIdActual);**

**assertEquals(doorIDExpected, doorIdActual);**

**}**

**@And("^Verify user able to get invalid barcode \"([^\"]\*)\"$")**

**public void verifyInvaidBarcodeDetail(String scanTypeExpected) {**

**String scanTypeActual = response.jsonPath().getString("scanEngineFeedback.scanType");**

**System.out.println("Scan Type for API is: " + scanTypeActual);**

**assertEquals(scanTypeExpected, scanTypeActual);**

**}**

**@Then("^Verify user able to get success in \"([^\"]\*)\"$")**

**public void verifyScanType(String scanTypeExpected) {**

**String scanTypeActual = response.jsonPath().getString("scanEngineFeedback.scanType");**

**System.out.println("Scan Type for API is: " + scanTypeActual);**

**assertEquals(scanTypeExpected, scanTypeActual);**

**}**

**@And("^Verify User able to get \"([^\"]\*)\" \"([^\"]\*)\" and \"([^\"]\*)\" for valid package barcode$")**

**public void verifyNextLoadPoint(String siteNameExpected, String nextLoadPointExpected, int doorIdExpected) {**

**String siteNameActual = response.jsonPath().getString("ncScanLabel.currentSiteName");**

**System.out.println("Current SiteName is: " + siteNameActual);**

**assertEquals(siteNameExpected, siteNameActual);**

**String nextLoadPointActual = response.jsonPath().getString("ncScanLabel.nextLoadPoint");**

**System.out.println("Next Load Point is: " + nextLoadPointActual);**

**assertEquals(nextLoadPointExpected, nextLoadPointActual);**

**int doorIdActual = response.jsonPath().getInt("ncScanLabel.doorId");**

**System.out.println("Door id is: " + doorIdActual);**

**assertEquals(doorIdExpected, doorIdActual);**

**}**

**@Given("^The BaseURI is up and running for StationController getStation$")**

**public void the\_BaseURI\_is\_up\_and\_running\_for\_StationController\_getStation() {**

**RestAssured.baseURI = ZipToDestAPIProp.getProperty("stationcontroller.dev.getstations");**

**log.info("Base URL is up and Running: " + ZipToDestAPIProp.getProperty("stationcontroller.dev.getstations"));**

**}**

**@When("^User sees the details of stations as \"([^\"]\*)\" and \"([^\"]\*)\" for \"([^\"]\*)\"$")**

**public void user\_sees\_the\_details\_of\_stations\_as\_and(String facilityId, String facilityName, int row)**

**throws JSONException {**

**response = given().contentType(ContentType.JSON).accept(ContentType.JSON).get();**

**for (int i = 0; i < row; i++) {**

**String facilityIdJSON = response.jsonPath().getString("facilityId[" + row + "]");**

**System.out.println("Facility Id is:" + facilityIdJSON);**

**if (facilityId.equals(facilityIdJSON)) {**

**break;**

**}**

**}**

**for (int j = 0; j < row; j++) {**

**String facilityNameJson = response.jsonPath().getString("facilityName[" + row + "]");**

**System.out.println("Facility Name is :" + facilityNameJson);**

**if (facilityName.equals(facilityNameJson)) {**

**break;**

**}**

**}**

**}**

**@Given("^The base URI is up and running for dock-maintenance-tool-controller$")**

**public void the\_base\_URI\_is\_up\_and\_running\_for\_dock\_maintenance\_tool\_controller() {**

**RestAssured.baseURI = ZipToDestAPIProp.getProperty("dock-maintenance-tool-controller.dev.returnDockMetaData");**

**log.info("Base URL is up and Running: "**

**+ ZipToDestAPIProp.getProperty("dock-maintenance-tool-controller.dev.returnDockMetaData"));**

**}**

**@When("^User should see the details of dock-maintenance of \"([^\"]\*)\" \"([^\"]\*)\"$")**

**public void user\_should\_see\_the\_details\_of\_dock\_maintenance\_of(String revision, String row) {**

**response = given().contentType(ContentType.JSON).accept(ContentType.JSON).get();**

**revision\_JSON = response.jsonPath().getString("revision[" + row + "]");**

**System.out.println("Revision Number is :" + revision\_JSON);**

**}**

**@Then("^The base URI is up and running for StartSort$")**

**public void the\_base\_URI\_is\_up\_and\_running\_for\_StartSort() {**

**RestAssured.baseURI = ZipToDestAPIProp.getProperty("stationcontroller.dev.startSort");**

**log.info("Base URL is up and Running: " + ZipToDestAPIProp.getProperty("stationcontroller.dev.startSort"));**

**}**

**@SuppressWarnings("static-access")**

**@When("^User send data for \"([^\"]\*)\" and \"([^\"]\*)\"$")**

**public void user\_send\_data\_for\_and(String facilityId, String overrideFlag) {**

**String primeTime = formatter.format(calender.getTime());**

**System.out.println(primeTime);**

**calender.add(calender.MINUTE, 5);**

**String unloadTime = formatter.format(calender.getTime());**

**System.out.println(unloadTime);**

**response = given().contentType(ContentType.JSON).accept(ContentType.JSON)**

**.body(StationController.startSort(facilityId, overrideFlag, primeTime, revision\_JSON, unloadTime))**

**.when().post();**

**}**

**@Given("^The base URI is up and running for EndSort$")**

**public void the\_base\_URI\_is\_up\_and\_running\_for\_EndSort() {**

**RestAssured.baseURI = ZipToDestAPIProp.getProperty("stationcontroller.dev.endSort");**

**log.info("Base URL is up and Running: " + ZipToDestAPIProp.getProperty("stationcontroller.dev.endSort"));**

**}**

**@When("^User send data for \"([^\"]\*)\"$")**

**public void user\_send\_data\_for(String facilityId) {**

**response = given().contentType(ContentType.JSON).accept(ContentType.JSON)**

**.body(StationController.endSort(facilityId)).when().post();**

**}**

**@And("^User should see the message \"([^\"]\*)\"$")**

**public void user\_should\_see\_the\_message(String message) {**

**json.log().all().extract();**

**String messageExpected = response.jsonPath().getString("message");**

**System.out.println("Sucess message is:" + messageExpected);**

**assertEquals(message, messageExpected);**

**}**

**@Given("^The base URI is up and runnig for dok-maintenance-tool-controller$")**

**public void the\_base\_URI\_is\_up\_and\_runnig\_for\_dok\_maintenance\_tool\_controller() {**

**RestAssured.baseURI = ZipToDestAPIProp.getProperty("dock-maintenance-tool-controller-dev");**

**log.info("Base URL is up and Running: " + ZipToDestAPIProp.getProperty("dock-maintenance-tool-controller-dev"));**

**}**

**@Then("^verify message invalid \"([^\"]\*)\"$")**

**public void verify\_message\_invalid(String failureMessage) {**

**String messageExpected = response.jsonPath().getString("message");**

**System.out.println("failure message is:" + messageExpected);**

**assertEquals(failureMessage, messageExpected);**

**}**

**@Then("^User should see the details as \"([^\"]\*)\" \"([^\"]\*)\" \"([^\"]\*)\"$")**

**public void user\_should\_see\_the\_details\_as(String revision, String facilityId, String dockName) {**

**response = given().contentType(ContentType.JSON).accept(ContentType.JSON).get();**

**String revision\_JSON = response.jsonPath().getString("revision");**

**System.out.println("Revision id is:" + revision\_JSON);**

**assertEquals(revision, revision\_JSON);**

**String facilityId\_JSON = response.jsonPath().getString("facilityId");**

**System.out.println("Facility Id is:" + facilityId\_JSON);**

**assertEquals(facilityId, facilityId\_JSON);**

**String dockName\_JSON = response.jsonPath().getString("dockName");**

**System.out.println("Dock name is :" + dockName\_JSON);**

**assertEquals(dockName, dockName\_JSON);**

**}**

**@Given("^The base URI is up and running for getActiveSort$")**

**public void the\_base\_URI\_is\_up\_and\_running\_for\_getActiveSort() throws Exception {**

**RestAssured.baseURI = ZipToDestAPIProp.getProperty("stationController.dev.getActiveSort");**

**log.info("Base URL is up and Running: " + ZipToDestAPIProp.getProperty("stationController.dev.getActiveSort"));**

**}**

**}**