**Rest Assured CRUD Operation**

* In real time Every API should perform CRUD operation before expose
* CRUD is basic operation where we can test the Rest API, whether we can perform READ, Write, Update & delete Operations

**READ/ Get**

* In order to get all the resource from the server, we go for “get” http method

Without BDD

@Test

**public** **void** getAllProjects() {

Response resp = RestAssured.*get*("http://localhost:8084/projects");

String completeBody = resp.prettyPrint();

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

**int** actSC = resp.getStatusCode();

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

String actCT = resp.getContentType();

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

}

With BDD

@Test

**public** **void** getAllProjects() {

*when*()

.get("http://localhost:8084/projects")

.then()

.assertThat().statusCode(200)

.and()

.assertThat().contentType(ContentType.***JSON***)

.log().all();

}

}

**Create / POST**

* In order to create the resource inside the server, we go for “Post” http method

Without BDD

@Test

**public** **void** createProject() {

JSONObject jObject= **new** JSONObject();

jObject.put("createdBy", "deepak");

jObject.put("projectName", "zoho");

jObject.put("status", "Created");

jObject.put("teamSize", 10);

RequestSpecification req = RestAssured.*given*();

req.contentType(ContentType.***JSON***);

req.body(jObject);

Response resp = req.post("http://localhost:8084/addProject");

String completeBody = resp.prettyPrint();

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

**int** actSC = resp.getStatusCode();

Assert.*assertEquals*(201, actSC);

String actCT = resp.getContentType();

Assert.*assertEquals*("application/json", actCT);

}

**With BDD**

@Test

**public** **void** createProject() {

JSONObject jObject= **new** JSONObject();

jObject.put("createdBy", "deepak");

jObject.put("projectName", "zoho-2");

jObject.put("status", "Created");

jObject.put("teamSize", 10);

*given*()

.contentType(ContentType.***JSON***)

.body(jObject)

.when()

.post("http://localhost:8084/addProject")

.then()

.log().all()

.assertThat().statusCode(201)

.assertThat().contentType(ContentType.***JSON***);

}

}

**Update/ PUT**

* In order to create/update the exiting resource inside the server, we go for “Put” http method

**With BDD**

@Test

**public** **void** updateProject() {

JSONObject jObject= **new** JSONObject();

jObject.put("createdBy", "deepak");

jObject.put("projectName", "zoho-22");

jObject.put("status", "Created");

jObject.put("teamSize", 10);

*given*()

.contentType(ContentType.***JSON***)

.body(jObject)

.when()

.put("http://localhost:8084/projects/"+"TY\_PROJ\_402")

.then()

.log().all()

.assertThat().statusCode(201)

.assertThat().contentType(ContentType.***JSON***);

}

**Partial Update/ PATCH**

* In order to partial update the exiting resource inside the server, we go for “Patch” http method

**With BDD**

@Test

**public** **void** updateProject() {

JSONObject jObject= **new** JSONObject();

jObject.put("projectName", "zoho-222");

*given*()

.contentType(ContentType.***JSON***)

.body(jObject)

.when()

.patch("http://localhost:8084/projects/"+"TY\_PROJ\_402")

.then()

.log().all()

.assertThat().statusCode(201)

.assertThat().contentType(ContentType.***JSON***);

}

**Delete / delete()**

**With BDD**

* In order to delete the exiting resource inside the server, we go for “delete” http method

@Test

**public** **void** getAllProjects() {

*when*()

.delete("http://localhost:8084/projects/"+"TY\_PROJ\_402")

.then()

.log().all()

.assertThat().statusCode(204);

}