RestAssured Topics

Section 6

25: Understanding Structure of Complex Nested Json and its  
array notations

{

"dashboard": {

"purchaseAmount": 910,

"website": "rahulshettyacademy.com"

},

"courses": [

{

"title": "Selenium Python",

"price": 50,

"copies": 6

},

{

"title": "Cypress",

"price": 40,

"copies": 4

},

{

"title": "RPA",

"price": 45,

"copies": 10

}

]

}

After converting from json online editor

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | object |  | {2} | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | dashboard |  | {2} | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | purchaseAmount | : | 910 | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | website | : | rahulshettyacademy.com | |
|  |  |  |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | courses |  | [3] | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | 0 |  | {3} | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | title | : | Selenium Python | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | price | : | 50 | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | copies | : | 6 | |
|  |  |  |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | 1 |  | {3} | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | title | : | Cypress | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | price | : | 40 | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | copies | : | 4 | |
|  |  |  |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | 2 |  | {3} | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | title | : | RPA | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | price | : | 45 | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | |  | copies | : | 10 | |

To find number of courses

1.First mock the response, think u mocked the response in payload class with getCoursemethod

JsonPath js= new JsonPath(payload.getCourse());

//Get the purchaseAmount

Js.get(dashboard.purchaseAmount).toString();

//Get the coursecount

Int courseCount=Js.getint(courses);

//Print first course name

System.out.println(StringJs.get(courses[0].title());

//Print all the course titles

For(int i=0;i<courseCount;i++)

{

Js.get(course[“+i+”].title();

}

//Print number of copies present in RPA course

For(int i=0;i<courseCount;i++)

{

If(Js.get(course[“+i+”].title ).equalignorancecase “RPA”)

{

Js.get(course[“+i+”].price();

}

}

//Print Purchase amount is equal to amount of all courses

For(int i=0;i<courseCount;i++)

{

Js.get(course[“+i+”].title();

String price=Js.get(course[“+i+”].price().toString;

String copy=Js.get(course[“+i+”].copy().toString;

Int amount= price\*copy;

Int sum= sum+ amount;

}

System.out.println(sum);

}

Section 7:

**Handling Dynamic Json Payloads with Parameterization**

**Payload stratagies:**

**Daynamically build json payloads with external inputs**

**This is one Type of dynamic palyload, means dynamically sending palylod through method.**

**Create a add place payload in payload class with addBookpayload method**

**Public class payload**

**{**

**Public void addBookpayload()**

**{**

**String payload=“**{

"name":"royjava",

"isbn":"bcd11",

"aisle":"2278",

"author":"Roy"

}

**“**

**Return payload;**

**}**

**Create a java class**

**Public class DynamicJson**

**{**

**@Test**

**Public void addBook()**

**{**

**Given().header(“Content-type”,”application/Json”).body(payload.addbookpayload()).when().post(“**/Library/**Addbook.php”).then().assertThat().stausCode(200);**

**Lecture 35:**  Sending parameters to payload from Test

**@Test**

**Public void addBook()**

**{**

**RestAssured.baseURI=”https://rahulshettyacademy.com”**

**Given().header(“Content-type”,”application/Json”).body(payload.addbookpayload(asdf,12345)).when().post(“**/Library/**Addbook.php”).then().assertThat().stausCode(200);**

**Public class payload**

**{**

**Public void addBookpayload(String isbn,String aisle)**

**{**

**String payload** **{\r\n\r\n\"name\":\"royjava\",\r\n\"isbn\":\"”+isbn+”\",\r\n\"”+aisle+”\":\"2278\",\r\n\"author\":\"Roy\"\r\n}\r\n**

**“**

**Return payload;**

**Lecture 36:**  Understanding TestNg Data provider for parameterization

**Public class payload**

**{**

**Public void addBookpayload(String isbn,String aisle)**

**{**

**String payload** **{\r\n\r\n\"name\":\"royjava\",\r\n\"isbn\":\"”+isbn+”\",\r\n\"”+aisle+”\":\"2278\",\r\n\"author\":\"Roy\"\r\n}\r\n**

**“**

**Return payload;**

**@Test(DataProvider=BooksData)**

**Public void addBook(String isbn, String asile)**

**{**

**RestAssured.baseURI=”https://rahulshettyacademy.com”**

**Given().header(“Content-type”,”application/Json”).body(payload.addbookpayload(isbn,asile)).when().post(“**/Library/**Addbook.php”).then().assertThat().stausCode(200);**

**@DataProvider(name=”BooksData”)**

**Public void getData()**

**{**

**Object[][] data=new Object[][];**

**data[0][0]=”abcd”;**

**data[0][1]=”1234”;**

**data[1][0]=”efgh”;**

**data[1][1]=”5678”;**

**return data;**

**}**

**Or**

**Return new Object[][]{{“abocd”,”1234”},{“efgh”,”5678”},{“ijklm”,”9090”}}**

**So this is how we send the dynamic payload in three ways**

**1.By sending the payload**

**2.sending arguments from Test**

**3.TestNg data Provider**

**Lecture 39:** How to handle with static json payloads

Create a json file

Open the notepad the paste the payload and save as .json format

**@Test()**

**Public void addBook()**

**RestAssured.baseURI=”https://rahulshettyacademy.com”**

**Given().header(“Content-type”,”application/Json”).body(generateStringFromresource(“enter path of the json file)).when().post(“**/Library/**Addbook.php”).then().assertThat().stausCode(200);**

**Public void generateStringFromresource()**

**{**

**Return new String(Files.readAllbytes(Paths.get(path)))**

**}**

**Section 8: Real world Example-Automating Jira API's**

**Jira is test Management tool.**

**Lecture44:** Cookie Authentication API for Jira requests

1. Cookie Based Authentication

**How to get Cookies session id**

**Method:Post**

**BaseURI:** <http://localhost:8080>

**Body:** { "username": "maheshgk287", "password": "Welcome13$jira"}

**Response**

{

    "session": {

        "name": "JSESSIONID",

        "value": "7AF66CBB196B293DBE00737D1B6BFF00"

    },

    "loginInfo": {

        "loginCount": 25,

        "previousLoginTime": "2020-07-22T16:22:34.886+0530"

    }

}

Cookie: JSESSIONID=6E3487971234567896704A9EB4AE501F;

PathParamerter there will be no ? symbol

<http://localhost:8080/rest/api/2/issue/10025>

**Lecture46**:  Defining Path Parameters in Rest Assured code using Add  
Comment API

Public class jiraTest

{

RestAssured.baseURI="http://localhost:8080";

//Login Scenario

SessionFilter session=new SessionFilter();

String response=given().relaxedHTTPSValidation().header("Content-Type","application/json").body("{\r\n" +

"    \"username\": \"RahulShetty\",\r\n" +

"    \"password\": \"XXXX11\"\r\n" +

"}").log().all().filter(session).when().post("/rest/auth/1/session").then().log().all().extract().response().asString();

//Add a comment

//Add comment

**String addCommentResponse = given().pathParam("key", "10101").log().all().header("Content-Type","application/json").body("{\r\n" +**

**"    \"body\": \""+expectedMessage+"\",\r\n" +**

**"    \"visibility\": {\r\n" +**

**"        \"type\": \"role\",\r\n" +**

**"        \"value\": \"Administrators\"\r\n" +**

**"    }\r\n" +**

**"}").filter(session).when().post("/rest/api/2/issue/{key}/comment").then().log().all().assertThat().statusCode(201).extract().response().asString();**

**JsonPath js=new JsonPath(addCommentResponse);**

**String commentId= js.getString("id");**

//Add Attachment

**given().header("X-Atlassian-Token","no-check").filter(session).pathParam("key", "10101")**

**.header("Content-Type","multipart/form-data")**

**.multiPart("file",new File("jira.txt")).when().**

**post("rest/api/2/issue/{key}/attachments").then().log().all().assertThat().statusCode(200);**

//Get Issue

**String issueDetails=given().filter(session).pathParam("key", "10101")**

**.queryParam("fields", "comment")**

**.log().all().when().get("/rest/api/2/issue/{key}").then()**

**.log().all().extract().response().asString();**

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

**JsonPath js1 =new JsonPath(issueDetails);**

**int commentsCount=js1.getInt("fields.comment.comments.size()");**

**for(int i=0;i<commentsCount;i++)**

**{**

**String commentIdIssue =js1.get("fields.comment.comments["+i+"].id").toString();**

**if (commentIdIssue.equalsIgnoreCase(commentId))**

**{**

**String message= js1.get("fields.comment.comments["+i+"].body").toString();**

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

**Assert.assertEquals(message, expectedMessage);**

**}**

**}**

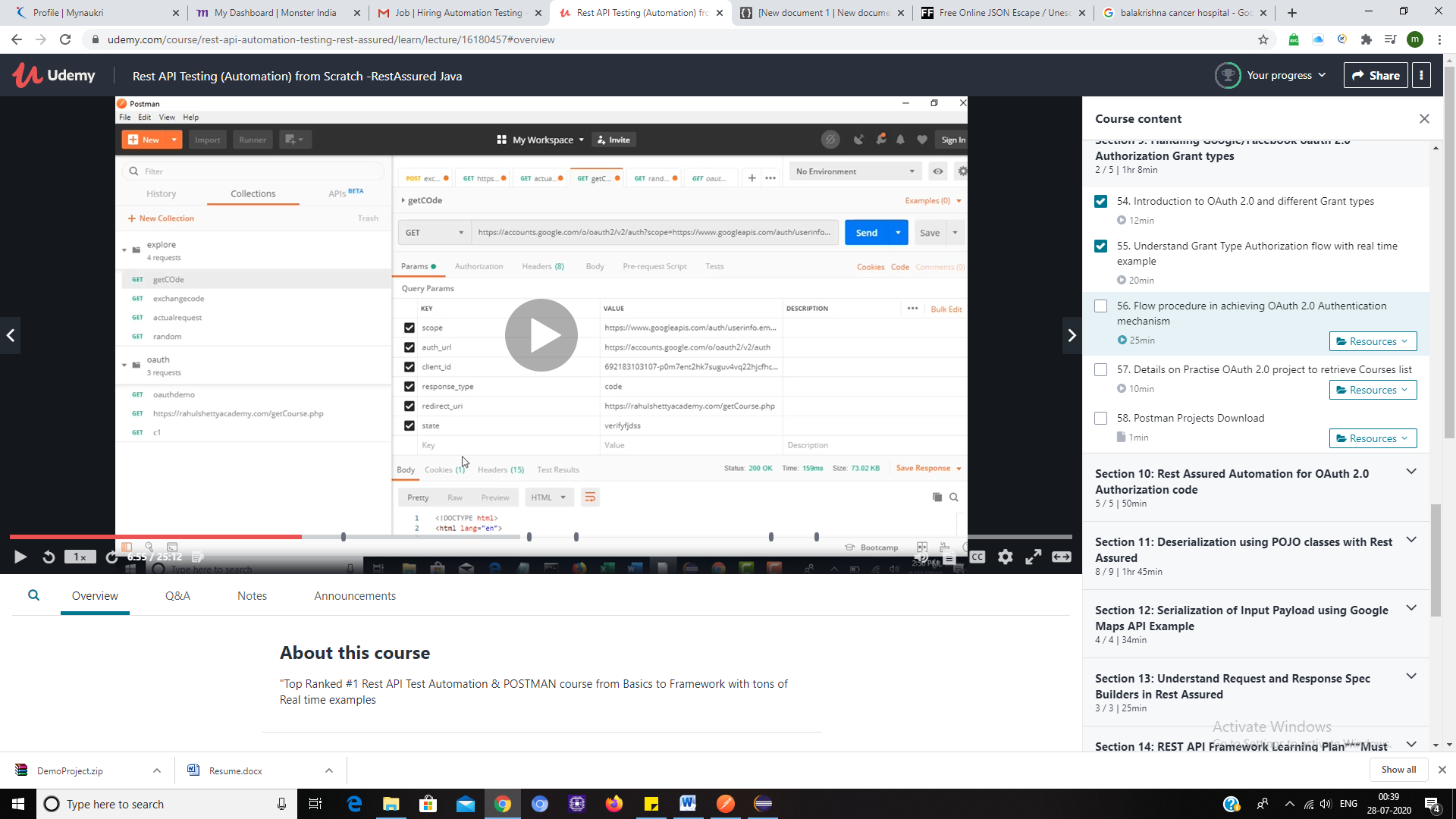
**}**

**}**

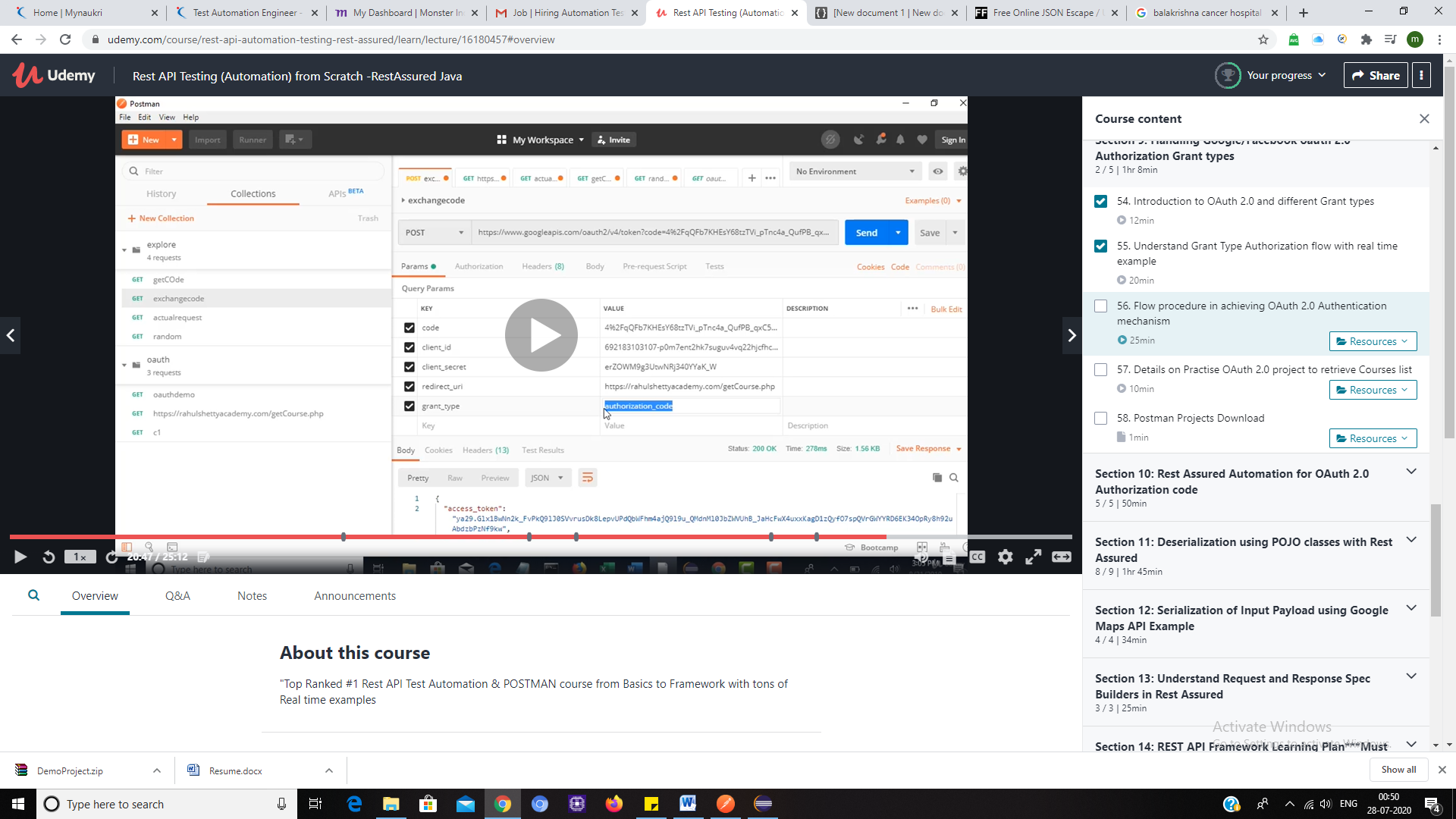
**How to tackle HTTPS validation**

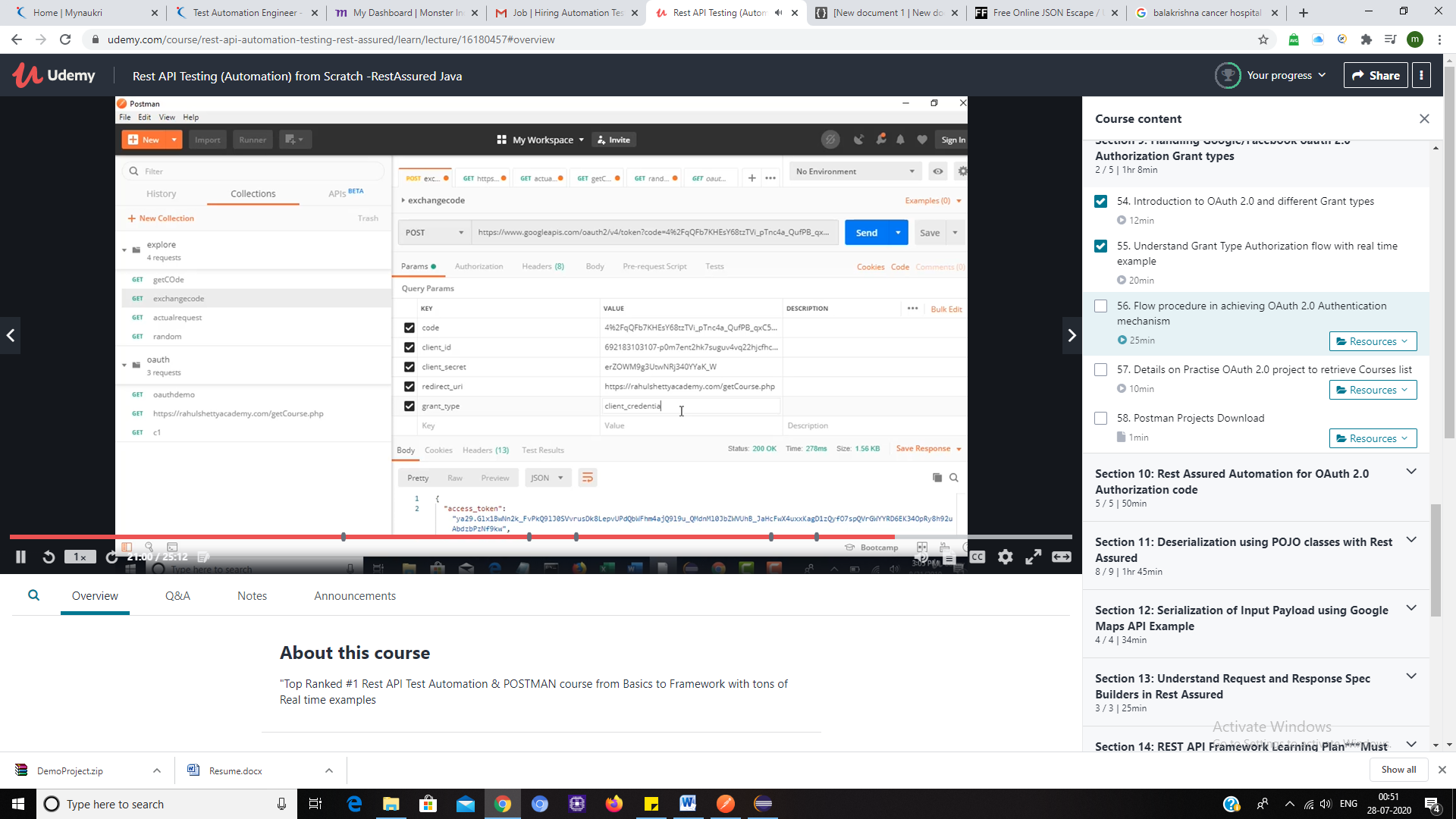
String response=given().relaxedHTTPSValidation()

OAuth to get authoraiztion code



To get access token





**Section 11: Deserialization using POJO classes with Rest  
Assured**