```
            //=== 1) Response Status ===
//Test for response code
pm.test("Status code should be 200.", () => {
    pm.response.to.have.status (200);
});

            //=== 2) Response Headers ===
//Test if reponse header is present
pm.test("Content-Type header is present.", () => {
    pm.response.to.have.header("Content-Type");
});

//Test if response header has a particular body
pm.test("Content-Type header contains application/json.", () => {
    pm.expect(pm.response.headers.get("Content-
Type")).to.eql("application/json");
});

            //=== 3) Response Cookies ===
//Test if Cookies are present
pm.test("Cookies language is present.", () => {
    pm.expect(pm.cookies.has('language')).to.be.false; //False cos my API has
no cookies.
});

            //=== 4) Response Time ===
//Test Response time to be in a specified range.
pm.test("Response time is less than 50ms", () => {
    pm.expect(pm.response.responseTime).to.be.below(50)
});

            //=== 5) Response Body ====
//Test the data-type of any part of the response body
const jsonData = pm.response.json();

pm.test("Test data-type of response", () => {
    pm.expect(jsonData).to.be.an("object");
    pm.expect(jsonData.id).to.be.a("string");
    pm.expect(jsonData.name).to.be.a("string");
    pm.expect(jsonData.courses).to.be.an("array");
});

//=== Response Body (Asserting Array Properties)
pm.test("Array Properties", () => {
    pm.expect(jsonData.courses).to.include("Java", "Selenium", "HTML", "Seleni
um");
});
```

```
//=== Validating values
pm.test("Value of location field", () => {
    pm.expect(jsonData.id).to.eql("2");
    pm.expect(jsonData.location).to.eql("South Africa");
    pm.expect(jsonData.courses[0]).to.eql("Java");
    pm.expect(jsonData.courses[1]).to.eql("HTML");
});

//=== Json Schema Validation
let schema = {
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "id": {
      "type": "string"
    },
    "name": {
      "type": "string"
    },
    "location": {
      "type": "string"
    },
    "phone": {
      "type": "string"
    },
    "courses": {
      "type": "array",
      "items": [
        {
          "type": "string"
        },
        {
          "type": "string"
        }
      ]
    }
  },
  "required": [
    "id",
    "name",
    "location",
    "phone",
    "courses"
  ]
}

pm.test("Schema is valid.", () => {
    pm.expect(tv4.validate (jsonData, schema)).to.be.true;
});
```