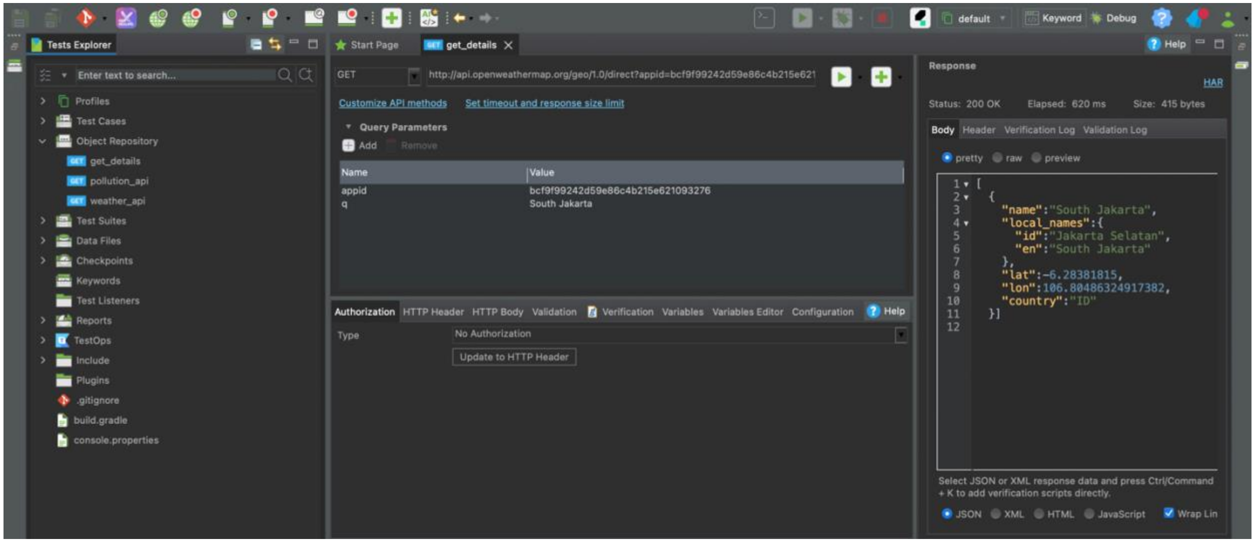
Results

Object Repository/get_details



GET http://api.openweathermap.org/geo/1.0/direct?appid=bcf9f99242d59e86c4b215e621

Name	Value
appid	bcf9f99242d59e86c4b215e621093276
q	South Jakarta

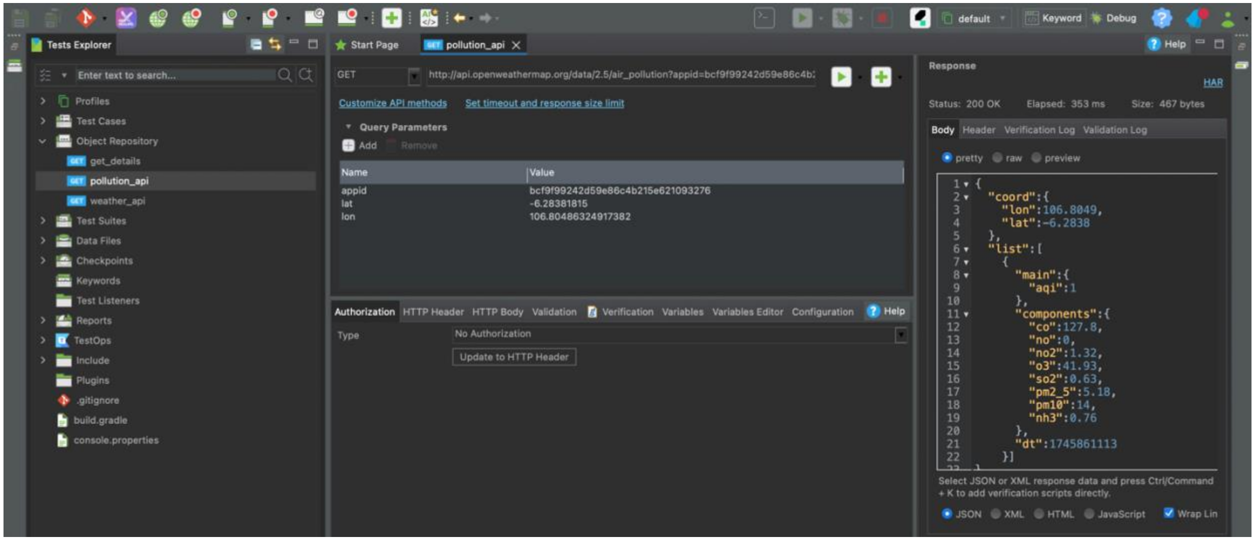
Type: No Authorization

Status: 200 OK Elapsed: 620 ms Size: 415 bytes

```
[
  {
    "name":"South Jakarta",
    "local_names":{
      "id":"Jakarta Selatan",
      "en":"South Jakarta"
    },
    "lat":-6.28381815,
    "lon":106.80486324917382,
    "country":"ID"
  }]
```

Select JSON or XML response data and press Ctrl/Command + K to add verification scripts directly.

Object Repository/pollution_api



GET http://api.openweathermap.org/data/2.5/air_pollution?appid=bcf9f99242d59e86c4b:

Name	Value
appid	bcf9f99242d59e86c4b215e621093276
lat	-6.28381815
lon	106.80486324917382

Type: No Authorization

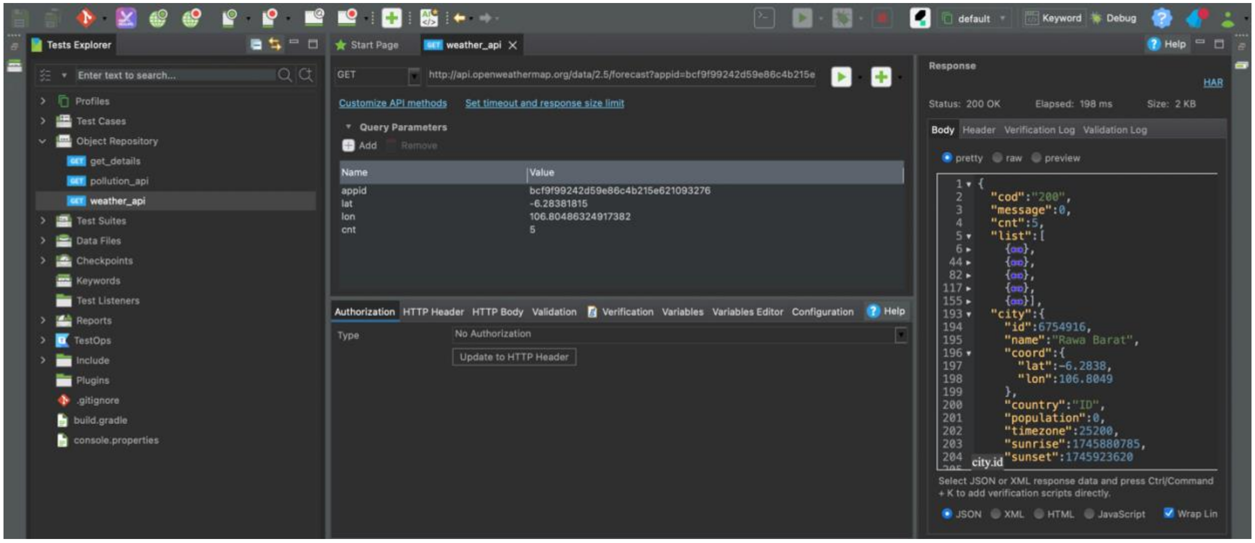
Status: 200 OK Elapsed: 353 ms Size: 467 bytes

```
{
  "coord":{
    "lon":106.8049,
    "lat":-6.2838
  },
  "list":[
    {
      "main":{
        "aqi":1
      },
      "components":{
        "co":127.8,
        "no":0,
        "no2":1.32,
        "o3":41.93,
        "so2":0.63,
        "pm2_5":5.18,
        "pm10":14,
        "nh3":0.76
      },
      "dt":1745861113
    }]
}
```

Select JSON or XML response data and press Ctrl/Command + K to add verification scripts directly.

Object Repository/weather_api

City Name : Rawa Barat, which is located on South Jakarta



GET http://api.openweathermap.org/data/2.5/forecast?appid=bcf9f99242d59e86c4b215e

Query Parameters

Name	Value
appid	bcf9f99242d59e86c4b215e621093276
lat	-6.28381815
lon	106.80486324917382
cnt	5

Type: No Authorization

Response

Status: 200 OK   Elapsed: 198 ms   Size: 2 KB

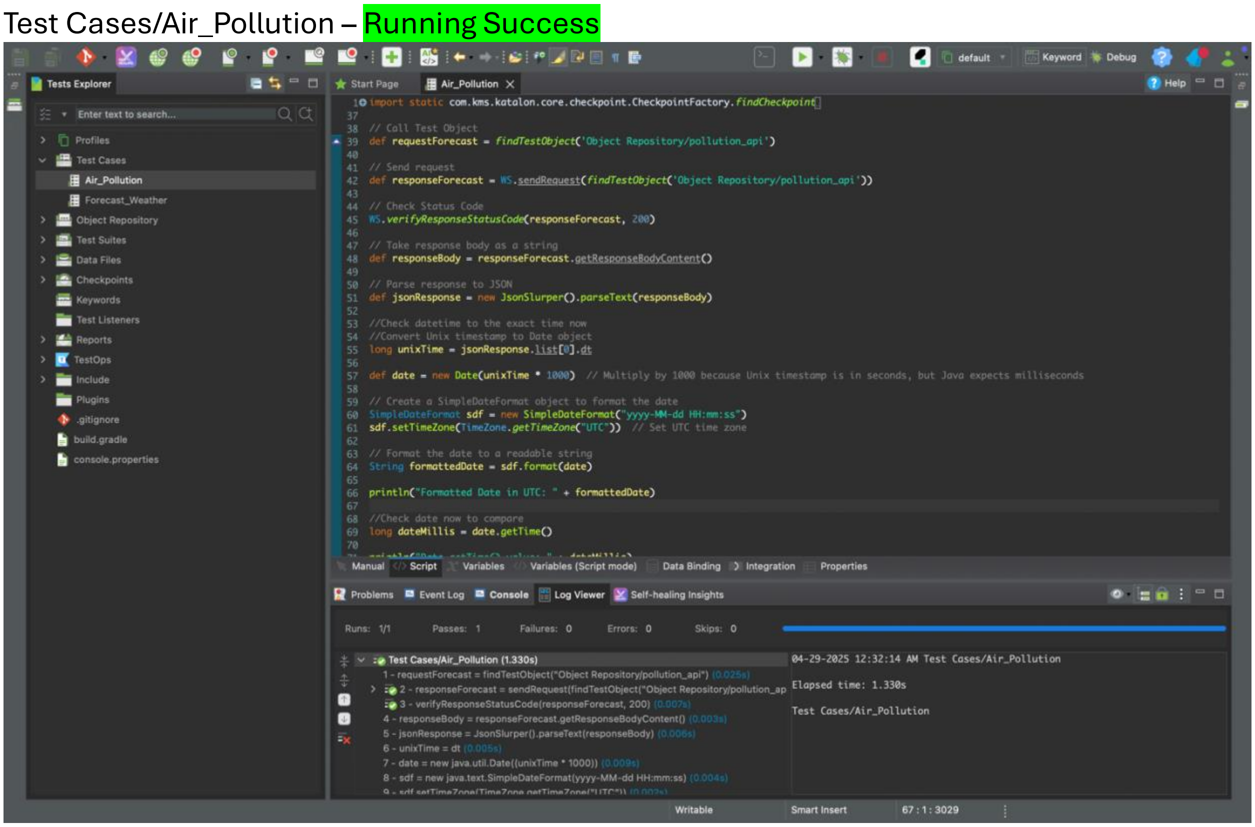
● pretty ○ raw ○ preview

```
{
  "cod":"200",
  "message":0,
  "cnt":5,
  "list":[
    {∞},
    {∞},
    {∞},
    {∞},
    {∞}],
  "city":{
    "id":6754916,
    "name":"Rawa Barat",
    "coord":{
      "lat":-6.2838,
      "lon":106.8049
    },
    "country":"ID",
    "population":0,
    "timezone":25200,
    "sunrise":1745880785,
city.id"sunset":1745923620
```

Select JSON or XML response data and press Ctrl/Command + K to add verification scripts directly.

● JSON ○ XML ○ HTML ○ JavaScript ☑ Wrap Lin

Test Cases/Air_Pollution – Running Success

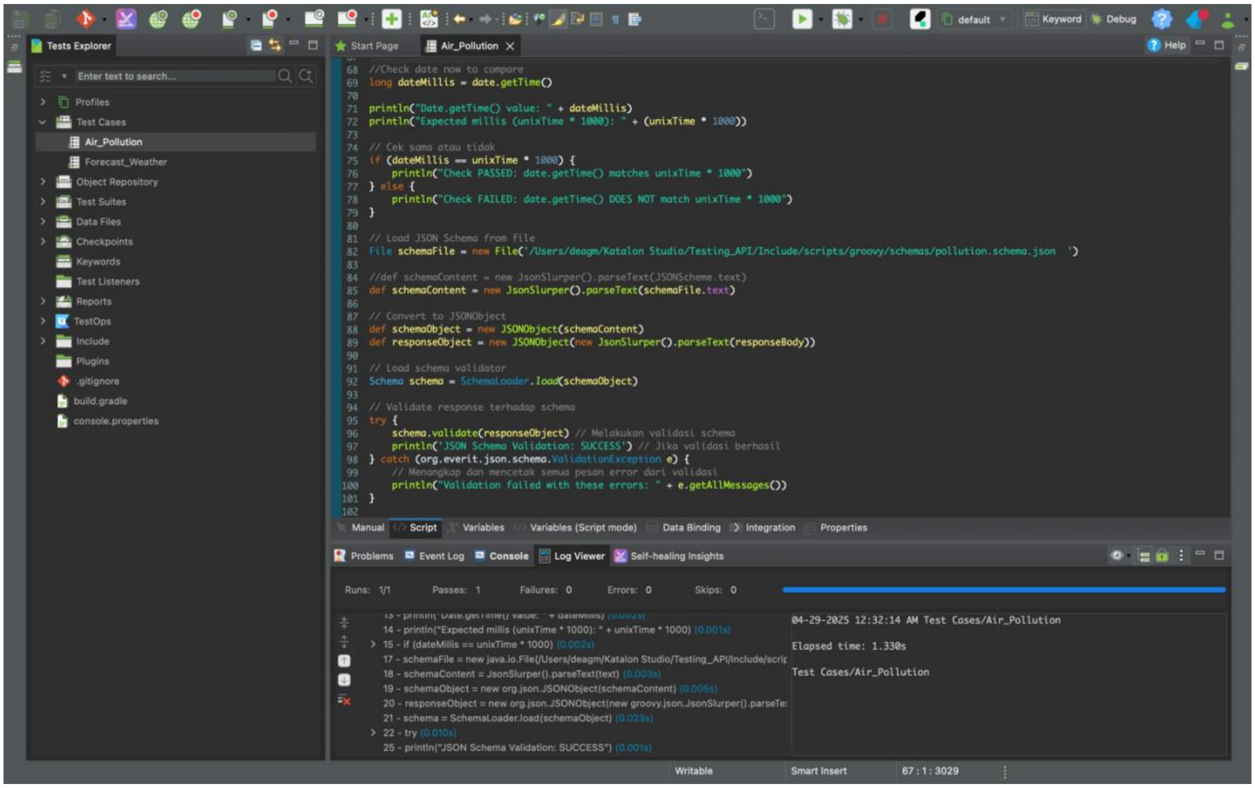


```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

// Call Test Object
def requestForecast = findTestObject('Object Repository/pollution_api')

// Send request
def responseForecast = WS.sendRequest(findTestObject('Object Repository/pollution_api'))

// Check Status Code
WS.verifyResponseStatusCode(responseForecast, 200)

// Take response body as a string
def responseBody = responseForecast.getResponseBodyContent()

// Parse response to JSON
def jsonResponse = new JsonSlurper().parseText(responseBody)

//Check datetime to the exact time now
//Convert Unix timestamp to Date object
long unixTime = jsonResponse.list[0].dt

def date = new Date(unixTime * 1000)  // Multiply by 1000 because Unix timestamp is in seconds, but Java expects milliseconds

// Create a SimpleDateFormat object to format the date
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")
sdf.setTimeZone(TimeZone.getTimeZone("UTC"))  // Set UTC time zone

// Format the date to a readable string
String formattedDate = sdf.format(date)

println("Formatted Date in UTC: " + formattedDate)

//Check date now to compare
long dateMillis = date.getTime()

println("Date.getTime() value: " + dateMillis)
```

Runs: 1/1   Passes: 1   Failures: 0   Errors: 0   Skips: 0

```
Test Cases/Air_Pollution (1.330s)
  1 - requestForecast = findTestObject("Object Repository/pollution_api") (0.025s)
  2 - responseForecast = sendRequest(findTestObject("Object Repository/pollution_ap
  3 - verifyResponseStatusCode(responseForecast, 200) (0.007s)
  4 - responseBody = responseForecast.getResponseBodyContent() (0.003s)
  5 - jsonResponse = JsonSlurper().parseText(responseBody) (0.006s)
  6 - unixTime = dt (0.005s)
  7 - date = new java.util.Date((unixTime * 1000)) (0.009s)
  8 - sdf = new java.text.SimpleDateFormat(yyyy-MM-dd HH:mm:ss) (0.004s)
  9 - sdf.setTimeZone(TimeZone.getTimeZone("UTC")) (0.002s)
```

```
04-29-2025 12:32:14 AM Test Cases/Air_Pollution

Elapsed time: 1.330s

Test Cases/Air_Pollution
```

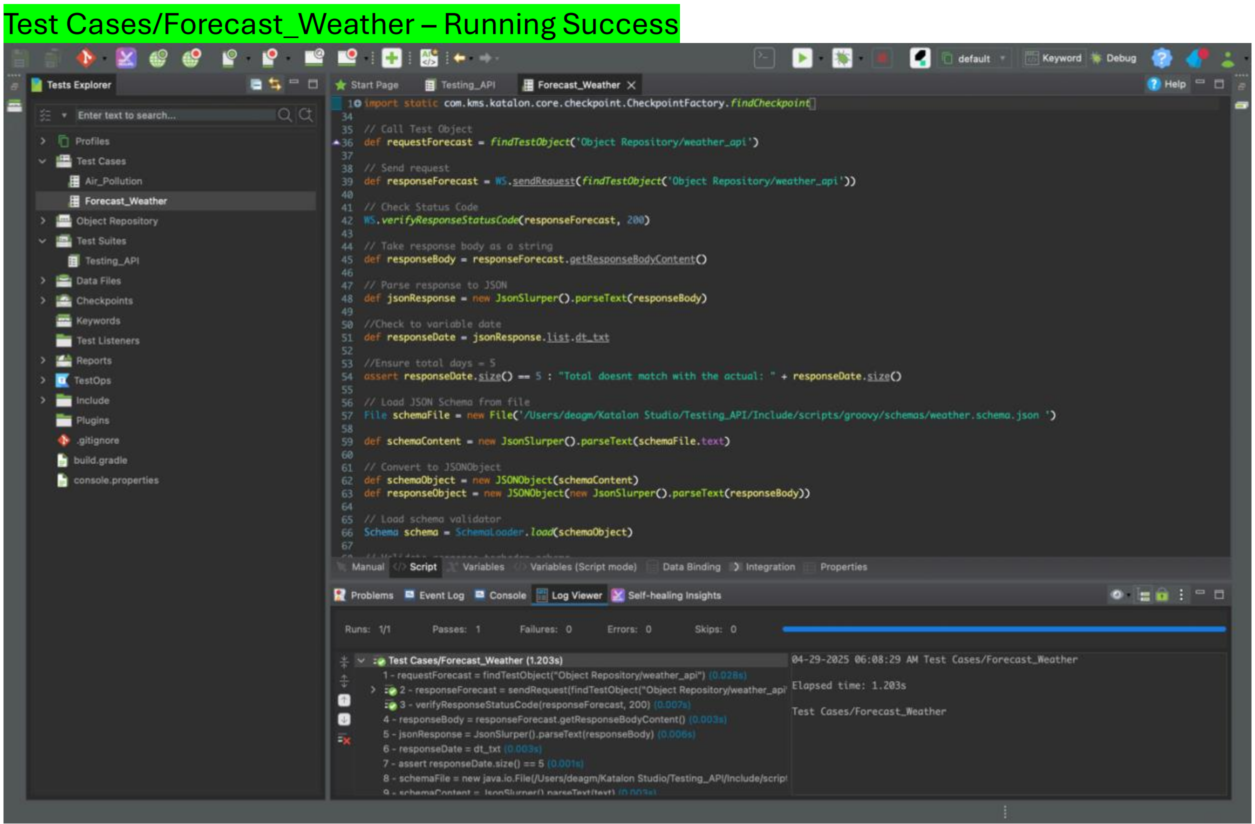


```groovy
//Check date now to compare
long dateMillis = date.getTime()

println("Date.getTime() value: " + dateMillis)
println("Expected millis (unixTime * 1000): " + (unixTime * 1000))

// Cek sama atau tidak
if (dateMillis == unixTime * 1000) {
    println("Check PASSED: date.getTime() matches unixTime * 1000")
} else {
    println("Check FAILED: date.getTime() DOES NOT match unixTime * 1000")
}

// Load JSON Schema from file
File schemaFile = new File('/Users/deagm/Katalon Studio/Testing_API/Include/scripts/groovy/schemas/pollution.schema.json  ')

//def schemaContent = new JsonSlurper().parseText(JSONScheme.text)
def schemaContent = new JsonSlurper().parseText(schemaFile.text)

// Convert to JSONObject
def schemaObject = new JSONObject(schemaContent)
def responseObject = new JSONObject(new JsonSlurper().parseText(responseBody))

// Load schema validator
Schema schema = SchemaLoader.load(schemaObject)

// Validate response terhadap schema
try {
    schema.validate(responseObject) // Melakukan validasi schema
    println('JSON Schema Validation: SUCCESS') // Jika validasi berhasil
} catch (org.everit.json.schema.ValidationException e) {
    // Menangkap dan mencetak semua pesan error dari validasi
    println("Validation failed with these errors: " + e.getAllMessages())
}
```

```
Runs: 1/1        Passes: 1        Failures: 0        Errors: 0        Skips: 0

  13 - println("Date.getTime() value: " + dateMillis) (0.002s)
  14 - println("Expected millis (unixTime * 1000): " + unixTime * 1000) (0.001s)
> 15 - if (dateMillis == unixTime * 1000) (0.002s)
  17 - schemaFile = new java.io.File(/Users/deagm/Katalon Studio/Testing_API/Include/scrip
  18 - schemaContent = JsonSlurper().parseText(text) (0.003s)
  19 - schemaObject = new org.json.JSONObject(schemaContent) (0.005s)
  20 - responseObject = new org.json.JSONObject(new groovy.json.JsonSlurper().parseTe
  21 - schema = SchemaLoader.load(schemaObject) (0.023s)
> 22 - try (0.010s)
  25 - println("JSON Schema Validation: SUCCESS") (0.001s)
```

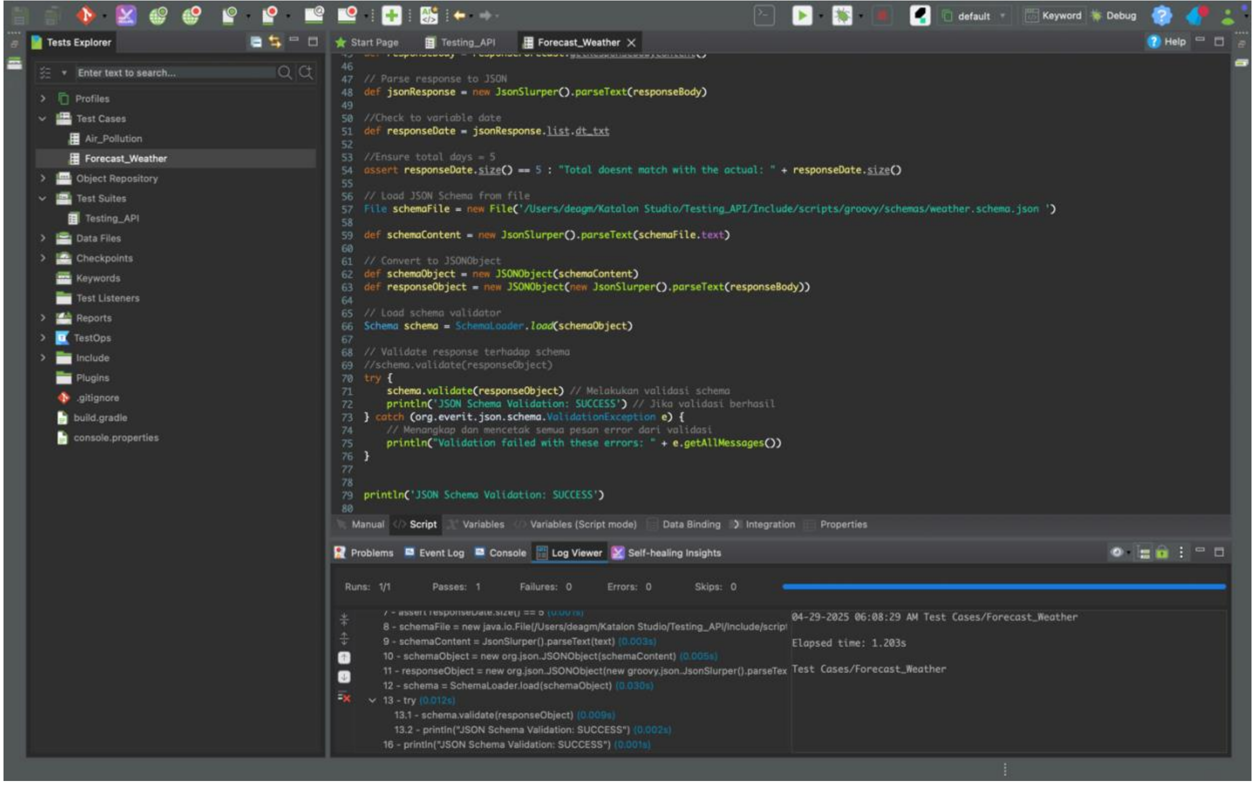
```
04-29-2025 12:32:14 AM Test Cases/Air_Pollution

Elapsed time: 1.330s

Test Cases/Air_Pollution
```

Test Cases/Forecast_Weather – Running Success



```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

// Call Test Object
def requestForecast = findTestObject('Object Repository/weather_api')

// Send request
def responseForecast = WS.sendRequest(findTestObject('Object Repository/weather_api'))

// Check Status Code
WS.verifyResponseStatusCode(responseForecast, 200)

// Take response body as a string
def responseBody = responseForecast.getResponseBodyContent()

// Parse response to JSON
def jsonResponse = new JsonSlurper().parseText(responseBody)

//Check to variable date
def responseDate = jsonResponse.list.dt_txt

//Ensure total days = 5
assert responseDate.size() == 5 : "Total doesnt match with the actual: " + responseDate.size()

// Load JSON Schema from file
File schemaFile = new File('/Users/deagm/Katalon Studio/Testing_API/Include/scripts/groovy/schemas/weather.schema.json ')

def schemaContent = new JsonSlurper().parseText(schemaFile.text)

// Convert to JSONObject
def schemaObject = new JSONObject(schemaContent)
def responseObject = new JSONObject(new JsonSlurper().parseText(responseBody))

// Load schema validator
Schema schema = SchemaLoader.load(schemaObject)
```

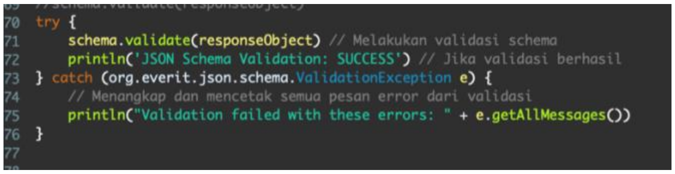
```
Runs: 1/1        Passes: 1        Failures: 0        Errors: 0        Skips: 0

Test Cases/Forecast_Weather (1.203s)
  1 - requestForecast = findTestObject("Object Repository/weather_api") (0.028s)
  2 - responseForecast = sendRequest(findTestObject("Object Repository/weather_api
  3 - verifyResponseStatusCode(responseForecast, 200) (0.007s)
  4 - responseBody = responseForecast.getResponseBodyContent() (0.003s)
  5 - jsonResponse = JsonSlurper().parseText(responseBody) (0.006s)
  6 - responseDate = dt_txt (0.003s)
  7 - assert responseDate.size() == 5 (0.001s)
  8 - schemaFile = new java.io.File(/Users/deagm/Katalon Studio/Testing_API/Include/script
  9 - schemaContent = JsonSlurper().parseText(text) (0.003s)

04-29-2025 06:08:29 AM Test Cases/Forecast_Weather

Elapsed time: 1.203s

Test Cases/Forecast_Weather
```



```
// Parse response to JSON
def jsonResponse = new JsonSlurper().parseText(responseBody)

//Check to variable date
def responseDate = jsonResponse.list.dt_txt

//Ensure total days = 5
assert responseDate.size() == 5 : "Total doesnt match with the actual: " + responseDate.size()

// Load JSON Schema from file
File schemaFile = new File('/Users/deagm/Katalon Studio/Testing_API/Include/scripts/groovy/schemas/weather.schema.json ')

def schemaContent = new JsonSlurper().parseText(schemaFile.text)

// Convert to JSONObject
def schemaObject = new JSONObject(schemaContent)
def responseObject = new JSONObject(new JsonSlurper().parseText(responseBody))

// Load schema validator
Schema schema = SchemaLoader.load(schemaObject)

// Validate response terhadap schema
//schema.validate(responseObject)
try {
    schema.validate(responseObject) // Melakukan validasi schema
    println('JSON Schema Validation: SUCCESS') // Jika validasi berhasil
} catch (org.everit.json.schema.ValidationException e) {
    // Menangkap dan mencetak semua pesan error dari validasi
    println("Validation failed with these errors: " + e.getAllMessages())
}


println('JSON Schema Validation: SUCCESS')
```

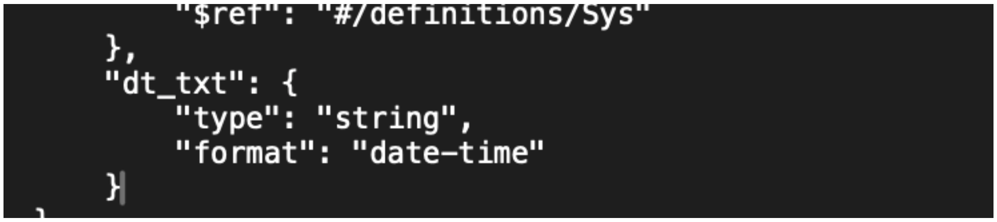
```
Runs: 1/1        Passes: 1        Failures: 0        Errors: 0        Skips: 0

  7 - assert responseDate.size() == 5 (0.001s)
  8 - schemaFile = new java.io.File(/Users/deagm/Katalon Studio/Testing_API/Include/script
  9 - schemaContent = JsonSlurper().parseText(text) (0.003s)
  10 - schemaObject = new org.json.JSONObject(schemaContent) (0.005s)
  11 - responseObject = new org.json.JSONObject(new groovy.json.JsonSlurper().parseTex
  12 - schema = SchemaLoader.load(schemaObject) (0.030s)
  13 - try (0.012s)
    13.1 - schema.validate(responseObject) (0.009s)
    13.2 - println("JSON Schema Validation: SUCCESS") (0.002s)
  16 - println("JSON Schema Validation: SUCCESS") (0.001s)

04-29-2025 06:08:29 AM Test Cases/Forecast_Weather

Elapsed time: 1.203s

Test Cases/Forecast_Weather
```

There's an error



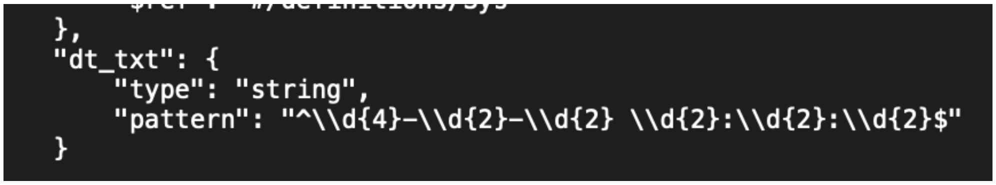
```
//schema.validate(responseObject)
try {
    schema.validate(responseObject) // Melakukan validasi schema
    println('JSON Schema Validation: SUCCESS') // Jika validasi berhasil
} catch (org.everit.json.schema.ValidationException e) {
    // Menangkap dan mencetak semua pesan error dari validasi
    println("Validation failed with these errors: " + e.getAllMessages())
}
```

Then I should fix this first on JSON Schema



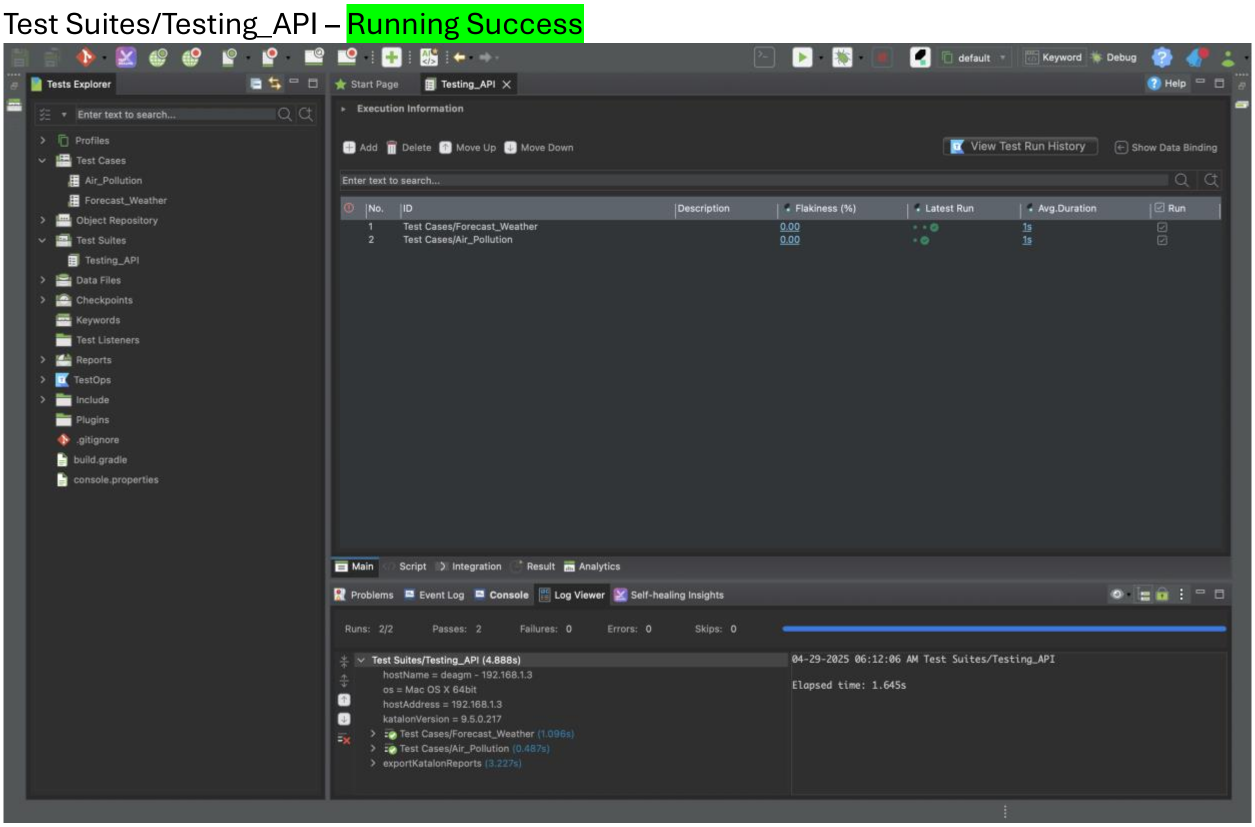
```
            "$ref": "#/definitions/Sys"
        },
        "dt_txt": {
            "type": "string",
            "format": "date-time"
        }
```

JSON Schema Format after I fixed, change the format that suites with the response, cause format date-time combination need the string to follow **standard ISO 8601(2025-04-28T18:00:00Z),** but the result on the response only **yyyy-MM-dd HH:mm:s**, so I changd the validation on schema.



```
        "$ref": "#/definitions/Sys"
    },
    "dt_txt": {
        "type": "string",
        "pattern": "^\\d{4}-\\d{2}-\\d{2} \\d{2}:\\d{2}:\\d{2}$"
    }
```

Test Suites/Testing_API – Running Success



No.	ID	Description	Flakiness (%)	Latest Run	Avg.Duration	Run
1	Test Cases/Forecast_Weather		0.00	● ● ✓	1s	☑
2	Test Cases/Air_Pollution		0.00	● ✓	1s	☑

```
Runs:  2/2      Passes:  2      Failures:  0      Errors:  0      Skips:  0

Test Suites/Testing_API (4.888s)
    hostName = deagm - 192.168.1.3
    os = Mac OS X 64bit
    hostAddress = 192.168.1.3
    katalonVersion = 9.5.0.217
  > Test Cases/Forecast_Weather (1.096s)
  > Test Cases/Air_Pollution (0.487s)
  > exportKatalonReports (3.227s)
```

```
04-29-2025 06:12:06 AM Test Suites/Testing_API

Elapsed time: 1.645s
```