Thryve QA Case Study

Task 1 & 2

Ngumbah Michael Nyika

Task 1

General Characteristics

Language Used: Java

Java Testing Framework: JUnit 4.8.1

Java SDK: Oracle OpenJDK 22.0.1

Project Build Tool: Maven 4.0.0

Test Runner Plugin Type: Surefire

Test Runner Plugin Version: 3.2.5

Integrated Development Environment (IDE): IntelliJ (JetBrains)

Web Client Library Type: OKHttp

Web Client Library Version: 4.12.0

* Haskell solution also attempted : ghc 9.4.8, stack 2.9.3, VS-Code IDE editor

Task 1

Artifacts

Delivery Format: Zip

Name: Thryve_UploadData_TestProject
Size: 12Kb

Hardware

Macbook Pro 16-Inch
Apple Silicon

Test

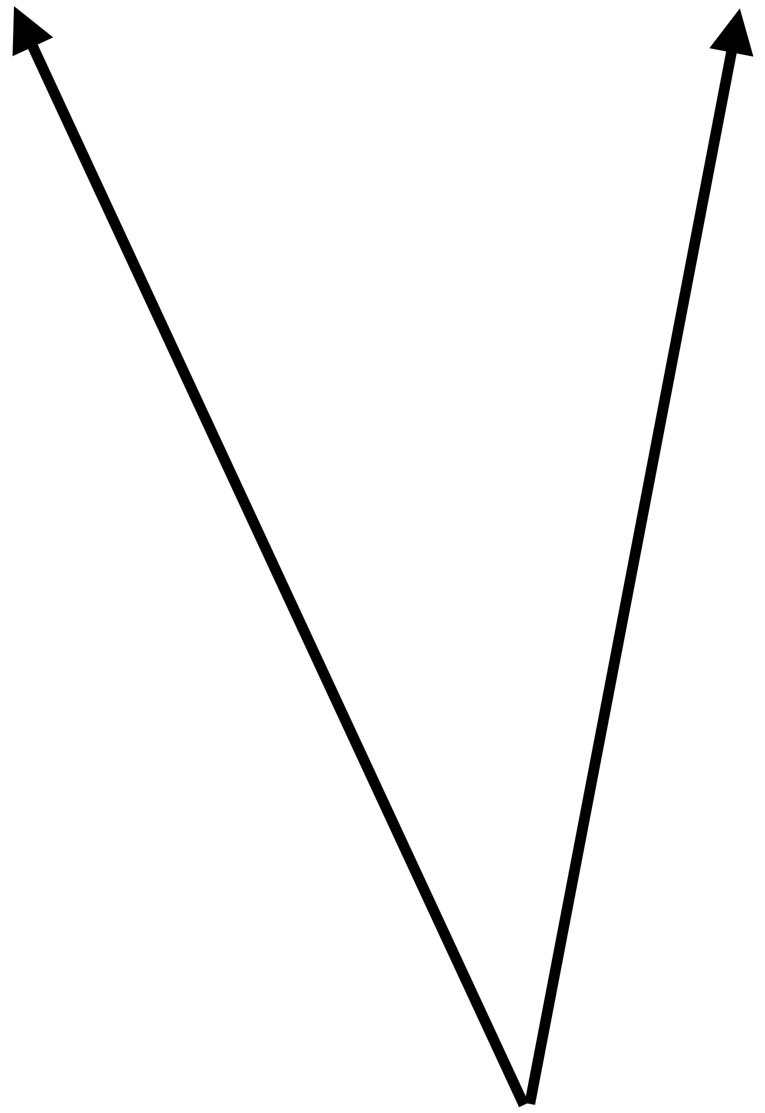
Test Class: PositiveUploadtest
Test Name: check_Uploaded_HealthDataSample_Corresponds_ExactlyTo_Downloaded_Sample

Task 1

Test Intention

```
"{\"height\": "175",\"weight\": “73.2”,\”birthdate\": \"1974-09-14\",\"gender\": \"female\"}";
```

Example of User Health Data Upload JSON Content

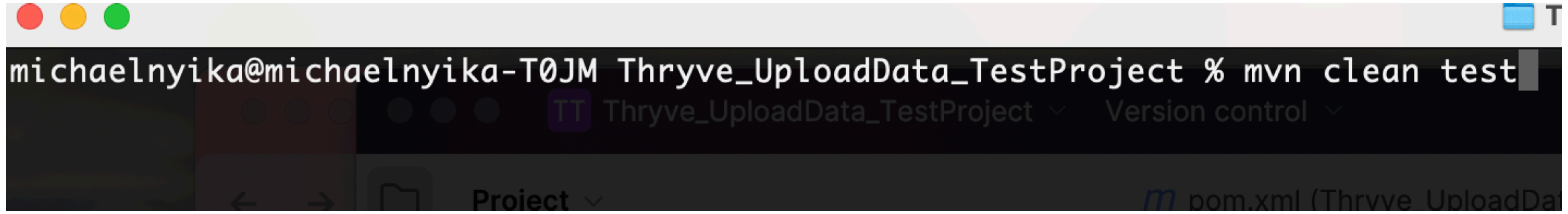


Check that Health Data **Uploaded** with **these** values, corresponds to the Height and Weight values in the **Downloaded** content, f**or that specific Thryve User**

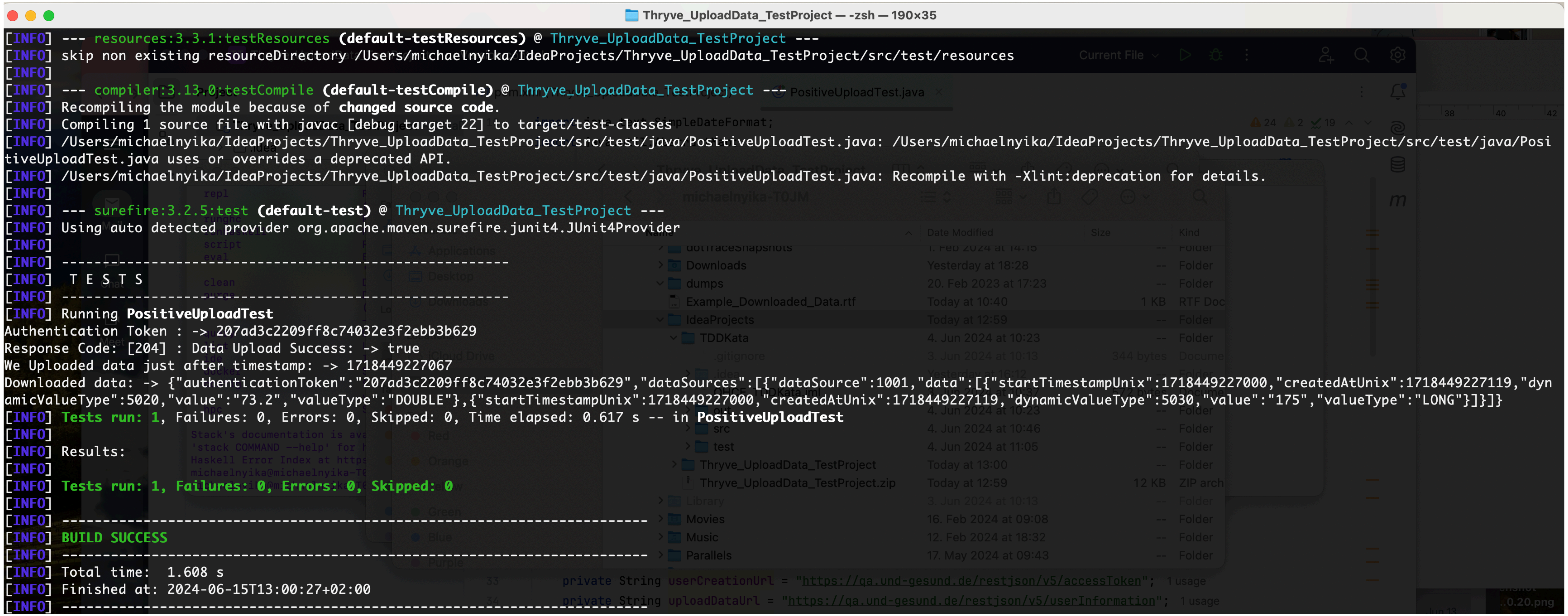
*<u>Simple, static numerical values were used for height and weight</u>:
One way to improve the test is to generate values within a numerical range that are valid (for both height and weight parameters);
It does not make the test any more valuable functionally, but it varies the output interestingly enough to ward off bugs at the edge of the valid ranges, for example,
a weight of *0kg or 1kg is technically feasible*, but **not -1kg**

Task 1

How to run the Test



```
michaelnyika@michaelnyika-T0JM Thryve_UploadData_TestProject % mvn clean test
```



```
[INFO] --- resources:3.3.1:testResources (default-testResources) @ Thryve_UploadData_TestProject ---
[INFO] skip non existing resourceDirectory /Users/michaelnyika/IdeaProjects/Thryve_UploadData_TestProject/src/test/resources
[INFO]
[INFO] --- compiler:3.13.0:testCompile (default-testCompile) @ Thryve_UploadData_TestProject ---
[INFO] Recompiling the module because of changed source code.
[INFO] Compiling 1 source file with javac [debug target 22] to target/test-classes
[INFO] /Users/michaelnyika/IdeaProjects/Thryve_UploadData_TestProject/src/test/java/PositiveUploadTest.java: /Users/michaelnyika/IdeaProjects/Thryve_UploadData_TestProject/src/test/java/Posi
tiveUploadTest.java uses or overrides a deprecated API.
[INFO] /Users/michaelnyika/IdeaProjects/Thryve_UploadData_TestProject/src/test/java/PositiveUploadTest.java: Recompile with -Xlint:deprecation for details.
[INFO]
[INFO] --- surefire:3.2.5:test (default-test) @ Thryve_UploadData_TestProject ---
[INFO] Using auto detected provider org.apache.maven.surefire.junit4.JUnit4Provider
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running PositiveUploadTest
Authentication Token : -> 207ad3c2209ff8c74032e3f2ebb3b629
Response Code: [204] : Data Upload Success: -> true
We Uploaded data just after timestamp: -> 1718449227067
Downloaded data: -> {"authenticationToken":"207ad3c2209ff8c74032e3f2ebb3b629","dataSources":[{"dataSource":1001,"data":[{"startTimestampUnix":1718449227000,"createdAtUnix":1718449227119,"dyn
amicValueType":5020,"value":"73.2","valueType":"DOUBLE"},{"startTimestampUnix":1718449227000,"createdAtUnix":1718449227119,"dynamicValueType":5030,"value":"175","valueType":"LONG"}]}]}
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.617 s -- in PositiveUploadTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.608 s
[INFO] Finished at: 2024-06-15T13:00:27+02:00
[INFO] ------------------------------------------------------------------------
```

Task 2

Functional Cases Brainstorm

Task 2

Areas of Coverage (1)

[Required Params] Authentication Credentials: Secured versus Unsecured

[Required Params] Authentication Credentials: Partly Missing Credentials

[Optional Params] Authentication Credentials: Single, Correct Size Values, Correct Content-Type

[Optional Params] Single Param: Correct Size Values, Invalid Content-Type (eg: invalid chars)

[Optional Params] Single Param: Incorrect Size Values (too short, too long)

[Optional Params] Single Param: No content

[Optional Params] Multiple Params: Correct Size Values, Correct Content-Type *

[Optional Params] Multiple Params: Correct Size Values, Invalid Content-Type (eg: invalid chars) *

[Optional Params] Multiple Params: Incorrect Size Values (too short, too long) *

[Optional Params] Multiple Params: No content *

*All Optional parameters can/should be manipulated in a consistent manner, across the board; however, ***exploratory testing*** would expand that technique to include ***irregular*** patterns **combinations**: for example, one parameter has fewer characters than normal, another has more characters but invalid symbols, another is missing, etc.

Task 2

Areas of Coverage (2)

[Required & Optional] Operational Error Code: Check correct HTTP Response codes (500, 403, 401, 301, etc) correctness

[Optional] Valid Standard Language Values work with correct HTTP Response codes (eg: *en*, *de*, *dk***)**

[Optional] Invalid Language Values work with correct HTTP Response codes (eg: *zz*, *qq*, *mm***)**

For example: language Option in Query Parameter for Thryve User Generation

[Required] Upload: JSON: Proper format, all values filled with correct values within boundary ranges

[Required] Upload: JSON: Missing values

[Required] Upload: JSON: Missing entire key-value JSON objects

[Required] Upload: JSON: Empty lists (JSONArrays)

[Required] Upload: Missing Content / No Content at all

[Required] Upload: JSON: Very Large *valid* payloads (gigabytes)

[Required] Upload: JSON: Successful upload, check Response Code (204 No-Content code, rather than 200)

[Required] Upload: Duplicate Required fields with valid values, for example, multiple authenticationToken fields supplied when uploading simple data

Task 2

Areas of Coverage (3) : Focus on Time/Date Fields

[Optional Params] Test Using a Date representing Epoch (Jan 1 1970), though one cannot retrieve records older than 31 days

[Optional Params] Retrieve records exactly 31 days old, slightly older (literally by seconds or minutes) or slightly under the limit

[Optional Params] Retrieve records for Leap Years

[Optional Params] Retrieve records when request has duplicated fields (for example, multiple startTimestampUnix field values)

[Optional Params] Missing half of a pair of dates combination, required to make a valid query (for example, not specifying endTimestampUnix even though startTimestampUnix was specified

[Outlier Monkey TestCase] Send a valid upload request with valid payload and credentials, but add headers not accounted for in the API Requirements (for example, Cache-Control, Keep-Alive, Cross-Origin-Embedder-Policy, etc)

Deprecated   Connection-Management   Security

Task 2

Non Functional Cases Brainstorm

Task 2

Non-Functional Cases Brainstorm

100,000+ Thryve users created simultaneously: Check Response Times

100,000+ Thryve users created simultaneously: Check all responses are HTTP 200 (valid credentials presented)

100,000+ Thryve users created simultaneously: At least 50% invalid credentials; Check HTTP Responses

10,000+ Thryve users created simultaneously and consistently every second for 1 minute

10,000+ Thryve users created simultaneously: Vary the load distribution of User creation requests pseudo-randomly

100+ Thryve users created with Gigabytes of Payload Data (JSON), sent intermittently

Error Performance Handling: 10,000+ Thryve users attempted creation with bad payloads, bad authentication credentials

! Thank You for the Challenge !

Ngumbah Michael Nyika
June 2024