Assignment-1

Course Name: Software Testing and Quality Assurance

Course ID: CSE 4495

Section: A

Group Name: XXX

Group Members:

XXXX(XXXX)

XXXX (XXXX)

Submitted on: XXXX XXXX

* **Abstract Specification Analysis [Member :** XXXX**]**

**API 1: (/student):**

|  |  |  |
| --- | --- | --- |
| **#Choices** | **Rationale** | **#Specification** |
| 1 | For this API, the number of testable choice is 1 since it does not any more testable choice. | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Request Type** | **Body** | **URL** | **Assertions** | **Description** |
| GET | This request does not have a body | http://127.0.0.1:5000/student | pm.**test**("Response time is less than 100ms", **function** () {  pm.expect(pm.response.responseTime).to.be.below(100);  });  pm.**test**("Status code is 200", **function** () {  pm.response.to.have.status(200);  }); | Our goal for this test was to check whether all student info is available, and to check if the API response time is below 100ms with a status code of 200;  For this test, the test result was successfully executed |

Test case check for API 1:

**API 2: (/student/<id>):**

|  |  |  |  |
| --- | --- | --- | --- |
| **Choice 1: “**ID” | **http requests** | **Output** | **Decision** |
| 1 | http://127.0.0.1:5000/student/1 | "courses\_passed" | valid ID |
| 0 > | http://127.0.0.1:5000/student/-1 | "error": "Student ID -1 does not exist" | negative ID doesn’t exist |
| 0 | http://127.0.0.1:5000/student/0 | "error": "Student ID 0 does not exist" | ID 0 doesn’t exist |
| 9 | http://127.0.0.1:5000/student/9 | "courses\_passed" | max valid ID |
| > 9 | http://127.0.0.1:5000/student/10 | "error": "Student ID 10 does not exist | greater than max ID |
| float | http://127.0.0.1:5000/student/1.1 | "error": "Student ID 1.1 is not formatted correctly." | ID isn’t in float datatype |
| C | http://127.0.0.1:5000/student/C | "error": "Student ID C is not formatted correctly." | character is invalid |
| Total #Specifications: 6 | | |  |

Our goal was to check the information from *valid ID.* From our findings, we noticed that the API doesn’t hold any ID with value more than 9, and the datatype of the ID is always in integer data type.

A demo for some test cases is provided below:

|  |  |  |
| --- | --- | --- |
| **Valid** | **Doesn’t Exist** | **Formatted Error** |
|  |  |  |

**API 3: (/create):**

|  |  |  |
| --- | --- | --- |
| **Choice 1:** Name | **Choice 2:** Personnummer | **Choice 3:** CoursesPassed |
| Empty | [yy: 00-99][mm: 01-12][dd: 01-28]-[0000-9999] | [A-Z][00000-10000] |
| Special Characters | [yy: 00-99][mm: 00][dd: --]-[0000-9999] | [A-Z][000-999] |
| Numeric Values | [yy: 00-99][mm: >12][dd: --]-[0000-9999] | [A-Z][0000-9999] |
| String [A-Z a-z] | [yy: 00-99][mm: 01-12][dd: 00]-[0000-9999] |  |
|  | [yy: 00-99][mm: 01-12][dd: >31]-[0000-9999] |  |
|  | [yy: 00-99][mm: 01-12][dd: >31]-[0000-9999] |  |
|  | [yy: 00-99][mm: 02][dd: >28]-[0000-9999] |  |
| #Total Specification: 4 \* 7 \* 3 = 84 | | |

After deducing specifications:

|  |  |  |
| --- | --- | --- |
| **Choice 1:** Name | **Choice 2:** Personnummer | **Choice 3:** CoursesPassed |
| Empty | dont care (-) | **-** |
| Special Characters | **-** | **-** |
| Numeric Values | **-** | **-** |
| String [A-Z a-z] | [yy: 00-99][mm: 01-12][dd: 01-28]-[0000-9999] | [A-Z][0000-9999] |
| **-** | [yy: 00-99][mm: 00][dd: --]-[0000-9999] | **-** |
| **-** | [yy: 00-99][mm: >12][dd: --]-[0000-9999] | **-** |
| **-** | [yy: 00-99][mm: 01-12][dd: 00]-[0000-9999] | **-** |
| **-** | [yy: 00-99][mm: 01-12][dd: >31]-[0000-9999] | **-** |
| **-** | [yy: 00-99][mm: 01-12][dd: >31]-[0000-9999] | **-** |
| **-** | [yy: 00-99][mm: 02][dd: >28]-[0000-9999] | **-** |
| **-** | - | [A-Z][00000-10000] |
| **-** | - | [A-Z][000-999] |
| **#Specification:** 3 + 2 + 6 + (1 \* 1 \* 1) = 12 | | |

A demo for some test cases is provided below:

|  |  |  |
| --- | --- | --- |
| **Valid**  (but should’ve been invalid) |  | Names shouldn’t have contained special chars/numeric values; February month cannot have 30 days |
| **Invalid Course Format** |  | courses have to contain 3 characters followed by 4 digits |
| **Corrected** |  | Passed |

**Repeat for all API**

* **Chosen Test cases for testing (Postman test report)[Member :** XXXX**]**
* **API 1: (/student)**

**TC1: Specification :XXXXXXX : Show\_student\_list ( Passed /Failed)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Request Type** | **Body** | **URL** | **Assertions** | **Description   (Goal/Purpose)** |
| GET | This request do not have a body | http://127.0.0.1:5000/student | pm.**test**("Status code is 200", **function** () {      pm.response.to.have.status(200);  });  pm.**test**("Response time is less than 50ms", **function** () {      pm.expect(pm.response.responseTime).to.be.below(50);  }); | Goal was to get all students information from the system.  2 assertions tested if the status code is 200 and response time is below 50ms and both passed. |

**API 2: (/student/<id>)**

|  |  |
| --- | --- |
| **Choice 1**  **(<Some choice>)** | **Choice 2**  **(<Some other choice>)** |
| V1 (Error)  V2 (! p3true )  V3 ( p3true )  V4 ( p3true )  V5 (Single) | P1 (Single)  P2(Error)  P3 (IF p3true)  P4 (Single)  P5 (Single) |
| Specifications = (5x5)=**25** | |
| Specifications after error/single/if,else = (2(E) + 4(S) + 2(TRUE) + 1(FALSE) )= **9** | |

**Repeat for all API**

* **Chosen Test cases for testing (Postman test report)[Member :** XXXX**]**
* **API 1: (/student)**

**TC1: Specification :XXXXXXX : Show\_student\_list ( Passed /Failed)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Request Type** | **Body** | **URL** | **Assertions** | **Description   (Goal/Purpose)** |
| GET | This request do not have a body | http://127.0.0.1:5000/student | pm.**test**("Status code is 200", **function** () {      pm.response.to.have.status(200);  });  pm.**test**("Response time is less than 50ms", **function** () {      pm.expect(pm.response.responseTime).to.be.below(50);  }); | Goal was to get all students information from the system.  2 assertions tested if the status code is 200 and response time is below 50ms and both passed. |

**Repeat for all APIs and All test cases**