Natalia Palej

Software Design With Artificial Intelligence For Cloud Computing  
Year 3 Semester 1

Web Development 3.1

Sprint #1 Project

Student ID: A00279259

Contents

[User Stories 2](#_Toc148746636)

[Database/Makeup User Stories 2](#_Toc148746637)

[User User Stories 2](#_Toc148746638)

[Testing 3](#_Toc148746639)

[Database/Makeup Test Cases 3](#_Toc148746640)

[Test #1 3](#_Toc148746641)

[Test #2 4](#_Toc148746642)

[Test #3 4](#_Toc148746643)

[Test #4 5](#_Toc148746644)

[Test #5 5](#_Toc148746645)

[Test #6 6](#_Toc148746646)

[Test #7 6](#_Toc148746647)

[Test #8 7](#_Toc148746648)

[Test #9 8](#_Toc148746649)

[Test #10 8](#_Toc148746650)

[User Test Cases 9](#_Toc148746651)

[Test #1 9](#_Toc148746652)

[Test #2 9](#_Toc148746653)

[Test Report 10](#_Toc148746654)

[Bugs Report 10](#_Toc148746655)

[Bug #1 10](#_Toc148746656)

[Bug #2 10](#_Toc148746657)

[Bug #3 11](#_Toc148746658)

[Bug #4 12](#_Toc148746659)

[Observations 12](#_Toc148746660)

# User Stories

## Database/Makeup User Stories

1. As a Makeup CAO, I want to connect server and database, so that I can access my inventory.
2. As a Makeup CAO, I want to read data in JSON format for all makeup products, so that I can use it in my application.
3. As a Makeup CAO, I want to search for products based on product name/company/category.
4. As a Makeup CAO, I want to insert new makeup product into database, so that I can expand on my inventory.
5. As a Makeup CAO, I want to update product details in database based on its ID, so that I can keep accurate information.
6. As a Makeup CAO, I want to delete product from database based on its ID, so that I can remove discontinued products.

## User User Stories

1. As a User, I want to log in, so that I can have access to the database.
2. As a Makeup CAO, I want to create new user, so that I can have more clients.
3. As a Makeup CAO, I want to delete existing user, so that I can remove inactive clients.
4. As a Makeup CAO, I want to update existing user details, so that I can keep accurate details.
5. As a Makeup CAO, I want to read users’ details, so that I can use it in my application.
6. As a Makeup CAO, I want to read user details based on their ID, so that I can get their details.

# Testing

## Database/Makeup Test Cases

**PRE-REQUISITES:**

Before conducting each test, it is assumed the below conditions are met:

1. Application server and database is running.
2. Postman REST Client is opened.

When executing each test, it is assumed that:

1. User will enter appropriate request (GET/INSERT/DELETE/UPDATE)
2. After inputting the command, user presses “enter/send”.

### Test #1

|  |  |
| --- | --- |
| User Story | Get API connection |
| Test Case ID | #1 |
| Description | Verify valid URL |
| Given | 1. Application server and database is running 2. Postman REST Client Opened |
| When | 1. Set GET request 2. Enter URL http://localhost/Assessment/Sprint1 3. Press send |
| Then | HTTP response 200 OK |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #2

|  |  |
| --- | --- |
| User Story | Get API connection |
| Test Case ID | #2 |
| Description | Verify invalid URL |
| When | Enter URL http://localhost:/Assessment/MyCompany |
| Then | HTTP response 400 NOT FOUND |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #3

|  |  |
| --- | --- |
| User Story | Read data in JSON format for all makeup products |
| Test Case ID | #3 |
| Description | Verify all products are retrieved |
| When | Enter path to the file http://localhost/Assessment/Sprint1/api/makeup |
| Then | Makeup inventory is displayed |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #4

|  |  |
| --- | --- |
| User Story | Search for product based on product name |
| Test Case ID | #4 |
| Description | Verify one product is retrieved |
| When | http://localhost/Assessment/Sprint1/api/makeup/search/byName/skin |
| Then | Makeup product information displayed |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #5

|  |  |
| --- | --- |
| User Story | Search for product based on product name |
| Test Case ID | #5 |
| Description | Verify error displayed if product doesn’t exist |
| When | http://localhost/Assessment/Sprint1/api/makeup/search/byName/skon |
| Then | Error message displayed |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #6

|  |  |
| --- | --- |
| User Story | Search for product based on company |
| Test Case ID | #6 |
| Description | Verify product(s) are displayed |
| When | http://localhost/Assessment/Sprint1/api/makeup/search/byCompany/benefit |
| Then | List of products where company = Benefit is retrieved |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #7

|  |  |
| --- | --- |
| User Story | Search for product based on category |
| Test Case ID | #7 |
| Description | Verify product(s) are displayed |
| When | http://localhost/Assessment/Sprint1/api/makeup/search/ByCategory/eyes |
| Then | List of products where productCategory = eyes is retrieved |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #8

|  |  |
| --- | --- |
| User Story | Insert new makeup product into database |
| Test Case ID | #8 |
| Description | Verify new product can be added |
| When | Path to file: <http://localhost/Assessment/Sprint1/api/makeup/add>  productName=”NUDE LIQUID LIPSTICK VAULT”, productCategory=”lips”, productDescription=” A 25-piece mini lipstick vault. For all the nudes you could possibly need.”, company=”Jeffree Star”, price=173.25, stock=50, onSale=”no”, discontinued=”no”; |
| Then | Product successfully added to makeupInventory |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #9

|  |  |
| --- | --- |
| User Story | Update product details in database based on its ID |
| Test Case ID | #9 |
| Description | Verify product was successfully updated |
| When | Path to file: <http://localhost/Assessment/Sprint1/api/makeup/update/5>  Update from “price”:”58.99” to “price”:”60.00” |
| Then | productID =5 price successfully updated |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #10

|  |  |
| --- | --- |
| User Story | Delete product from database based on its ID |
| Test Case ID | #10 |
| Description | Verify product was successfully deleted from database |
| When | Path to file: http://localhost/Assessment/Sprint1/api/makeup/delete/24 |
| Then | Product with ID=24 was successfully deleted from database |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

## User Test Cases

### Test #1

|  |  |
| --- | --- |
| User Story | Read users details |
| Test Case ID | #1 |
| Description | Verify users’ details are retrieved |
| When | Enter path to the file http://localhost/Assessment/Sprint1/api/users |
| Then | All users are displayed |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #2

|  |  |
| --- | --- |
| User Story | Read user data based on their ID |
| Test Case ID | #2 |
| Description | Verify userID=4 details are retrieved |
| When | Enter path to the file http://localhost/Assessment/Sprint1/api/users/4 |
| Then | userID=4 details are retrieved |
| Test Result | Passed Failed |

## A screenshot of a computer Description automatically generated

### Test #3

|  |  |
| --- | --- |
| User Story | Update user data based on their ID |
| Test Case ID | #3 |
| Description | Verify userID=2 details are updated |
| When | Enter path to the file <http://localhost/Assessment/Sprint1/api/users/2>  Update from “lastName”:”Black” to “lastName”:”BlackGSX” |
| Then | userID=2 successfully updated |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #4

|  |  |
| --- | --- |
| User Story | Delete user data based on their ID |
| Test Case ID | #4 |
| Description | Verify userID=6 has been deleted successfully |
| When | Enter path to the file http://localhost/Assessment/Sprint1/api/users/6 |
| Then | userID=6 successfully deleted |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #5

|  |  |
| --- | --- |
| User Story | Create new user |
| Test Case ID | #5 |
| Description | Verify new user can be added to database |
| When | Enter path to the file <http://localhost/Assessment/Sprint1/api/users>  “username”:”Emilia”, “password”:”admin7”, “firstName”:”Emilia”, “lastName”:”Quinn”, “address”:”Paradise”, “phoneNo”:”0897451221”, “email”:”EmiliaQuinn@gmail.com” |
| Then | userID=4 details are retrieved |
| Test Result | Passed Failed |

## A screenshot of a computer Description automatically generated

## Test Report

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Type** | **Database** | **Total Tests** | **Number Passed** | **Number Failed** |
| Manual | products | 10 | 10 | 0 |
| users | 5 | 5 | 0 |

## Bugs Report

### Bug #1

|  |
| --- |
| Bug Description |
| A screenshot of a computer  Description automatically generated |
| Resolution |
| In my code I have forgotten to add **‘%** before and after $productName |

### Bug #2

|  |
| --- |
| Bug Description |
|  |
| Resolution |
| URL route was incorrect, I have added additional “**/**” at the end of the API.    I simplified the route and got rid of “makeupInventory”. Final route: |

### Bug #3

|  |
| --- |
| Bug Description |
|  |
| Resolution |
| Instead of returning $makeup**i**nventory I had $makeup**I**nventory which was giving below error:  <div><strong>Message:</strong> Undefined variable $makeupInventory</div> |

### Bug #4

|  |
| --- |
| Bug Description |
|  |
| Resolution |
|  |

# Personal Observations

1. I faced issues with CRUD operations because I missed specifying the correct paths I had set up in the 'index.php' file when entering them in Postman. Eg, the initial paths were '/makeup/**add**' for adding, '/makeup/**update**/<id>' for updating, and '/makeup/**delete**/<id>' for deleting. To make testing and code simpler, I removed these additional inputs. In the final version, the routes are defined as follows, which differs from the screenshots shared during testing. This adjustment was made after running the Database/Makeup Test Cases.

A screenshot of a computer screen

Description automatically generated

1. A close-up of a sign

   Description automatically generatedI have also renamed my table from makeupInventory to products to keep it simple: