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](#_Toc148736245)

[Database/Makeup User Stories 2](#_Toc148736246)

[User User Stories 2](#_Toc148736247)

[Testing 3](#_Toc148736248)

[Database/Makeup Test Cases 3](#_Toc148736249)

[Test #1 3](#_Toc148736250)

[Test #2 4](#_Toc148736251)

[Test #3 4](#_Toc148736252)

[Test #4 5](#_Toc148736253)

[Test #5 5](#_Toc148736254)

[Test #6 6](#_Toc148736255)

[Test #7 6](#_Toc148736256)

[Test #6 8](#_Toc148736257)

[Test #5 9](#_Toc148736258)

[Test #6 10](#_Toc148736259)

[User Test Cases 10](#_Toc148736260)

[Test Report 11](#_Toc148736261)

[Bugs Report 12](#_Toc148736262)

# 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 user details, so that I can use it in my application.

# 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 |
| 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 | Enter productName = 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 | Enter invalid productName = 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 | Enter company = 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 | Enter productCategory = 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 | 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 | Enter productID=5 and update price field to 59.00 |
| Then | ProductID=5 price successfully updated (ENDLESS NIGHTS EYESHADOW PALETTE) |
| Test Result | Passed Failed |

### 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 | Enter makeupID=24 |
| Then | Product with ID=24 was successfully deleted from database |
| Test Result | Passed Failed |

A screenshot of a computer

Description automatically generated

### Test #11

|  |  |
| --- | --- |
| User Story |  |
| Test Case ID | #11 |
| Description |  |
| When |  |
| Then |  |
| Test Result | Passed Failed |

### Test #12

|  |  |
| --- | --- |
| User Story |  |
| Test Case ID | #12 |
| Description |  |
| When |  |
| Then |  |
| Test Result | Passed Failed |

## User Test Cases

### Test #1

|  |  |
| --- | --- |
| User Story | I want to log in |
| Test Case ID | #1 |
| Description | Verify user successfully logged in |
| When |  |
| Then |  |
| Test Result | Passed Failed |

### Test #2

|  |  |
| --- | --- |
| User Story | I want to log in |
| Test Case ID | #2 |
| Description | Verify user not logged if doesn’t exist in database |
| When |  |
| Then |  |
| Test Result | Passed Failed |

## Test Report

|  |  |  |
| --- | --- | --- |
| **Test Type** | **Number Passed** | **Number Failed** |
| Manual |  |  |
| Automated |  |  |

## 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 |
| My URL route was incorrect, I have added additional “**/**” at the end of the API.  Based on my index.php file the correct route was:    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 |
|  |