**Deom Blaze Software Test Design**

**Project Name:** Demo Blaze Web App Software Test Design (STD)

**Prepared by:** Ehab Khalil

**Document Created:** 2024-08-03

### Software Test Design (STD) for Demoblaze

#### 1. Introduction

* **Objective:** Creating a plan or strategy to test the entire Demoblaze application, including all of its features and functions. Test design aims to identify software defects early in the development cycle and ensures that the Demoblaze application meets its requirements and functions correctly before it is deployed to production.
* **Scope:** Test design aims to ensure that the testing process is efficient and effective and identify all the bugs, errors, and future mistakes in the software. Functional testing will be the main goal in this project with a focus on automated tests.

#### 2. Approach

##### **2.1 Testing Levels**

The following testing levels will be employed:

1. **Unit Testing**
   * Performed by developers to test individual components and ensure they function correctly.
   * Focus on testing methods and functions within classes.
2. **Integration Testing**
   * Test the interaction between integrated components.
   * Ensure that modules work together as expected.
   * Use of stubs and drivers if necessary.
3. **System Testing**
   * End-to-end testing of the entire application.
   * Verify that the system meets functional and non-functional requirements.
   * Conducted by the QA team.
4. **Acceptance Testing**
   * Performed to ensure the system meets the business requirements.
   * Involves end-users and stakeholders.
   * User Acceptance Testing (UAT) will be conducted before the release.

##### **2.2 Testing Types**

The following types of testing will be conducted:

1. **Functional Testing**
   * Validate that the application functions according to requirements.
   * Includes testing of user authentication, product browsing, cart management, order processing, user profile management, and product search.
2. **Regression Testing**
   * Ensure that new changes do not negatively impact existing functionality.
   * Automated regression tests will be run after each new build.
3. **Performance Testing**
   * Assess the application's responsiveness, stability, and scalability under load.
   * Includes load testing and stress testing.
4. **Usability Testing**
   * Evaluate the user interface and user experience.
   * Ensure the application is intuitive and user-friendly.
5. **Security Testing**
   * Identify vulnerabilities and ensure the application is secure from threats.

**3. Test Identification**

|  |  |  |
| --- | --- | --- |
| Test Case 001 | **Test case name:** ValidSign Up(API) | **Pre-conditions:** Internet connection |
| **Test Objective:** Verify successful sign up with valid user name and password | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, valid password, sign up endpoint | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/signup the password in the request body should be encoded in base64 {valid\_user} = valid user {valid\_password} = encoded valid password with request body {  "username": "{valid\_user}",  "password": "{valid\_password}" } | API expected to return a success response with status code 200 and status: ok |
| 2 | Verify the status code | The status code should be 200. |
| 3 | Verify the status.ok | The response body should include "status": "ok". |
| Test Case 002 | **Test case name:** InvalidSign Up(API) | **Pre-conditions:** Internet connection |
| **Test Objective:** Verify successful error message for messing password | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, sign up endpoint | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/signup the password in the request body should be encoded in base64 {valid\_user} = valid user {valid\_password} = insert an empty string with request body {  "username": "{valid\_user}",  "password": "" } | API expected to return a success response with status code 200 and a message "Please fill out Username and Password." |
| 2 | Verify the status code | The status code should be 200. |
| 3 | Verify the response body | The response body should include "Please fill out Username and Password." |

|  |  |  |
| --- | --- | --- |
| Test Case 003 | **Test case name:** ValidSign Up(UI) | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:** Verify successful sign up with valid user name and password | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, valid password, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "Sign Up" link located in the top right side of the page. | Sign up page pops up |
| 2 | Insert the user name in the first name input | the text will appear in the username name input field |
| 3 | Insert the password in the password input | the text will appear in the password input field |
| 4 | Click on the "sign up" button | Alert message pops up |
| 5 | Verify the alert message | The user should get a welcome message "Sign up successful.". |

|  |  |  |
| --- | --- | --- |
| Test Case 004 | **Test case name:** UISign Up With Username Only | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:** Verify that a user receives an appropriate error message when attempting to sign in with a valid username but without providing a password. | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "sign Up" link located in the top right side of the page. | Sign up page pops up |
| 2 | Insert the user name in the username input | the text will appear in the username name input field |
| 3 | Verify the password input is empty | The password input field should not contain any text |
| 4 | Click on the "sign up" button | Alert message pops up |
| 5 | Verify the alert message | The user should get a message "Please fill out Username and Password." |

|  |  |  |
| --- | --- | --- |
| Test Case 005 | **Test case name:** UI Sign Up With Password Only | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:** Verify that a user receives an appropriate error message when attempting to sign in with a valid password but without providing a username. | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid password, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "sign Up" link located in the top right side of the page. | Sign up page pops up |
| 2 | Verify the username input is empty | The username input field should not contain any text |
| 3 | Insert the password in the password input | the text will appear in the password name input field |
| 4 | Click on the "sign up" button | Alert message pops up |
| 5 | Verify the alert message | The user should get a message "Please fill out Username and Password." |

|  |  |  |
| --- | --- | --- |
| Test Case 006 | **Test case name:** Valid Log In(API) | **Pre-conditions:** Internet connection |
| **Test Objective:** Verify successful login with valid user name and password | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, valid password, login endpoint | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/login the password in the request body should be encoded in base64 {valid\_user} = valid user {valid\_password} = encoded valid password with request body {  "username": "{valid\_user}",  "password": "{valid\_password}" } | API expected to return a success response with status code 200 and status: ok with response body that includes auth\_token |
| 2 | Verify the status code | The status code should be 200. |
| 3 | Verify the status.ok | The response body should include "status": "ok". |
| 4 | Verify the response body | the response body should include the next structure: "Auth\_token: c0R0TExmSW8xNzIzNTQx" the value " c0R0TExmSW8xNzIzNTQx" is an example of auth\_token |

|  |  |  |
| --- | --- | --- |
| Test Case 007 | **Test case name:**  Log In(API) With invalid password | **Pre-conditions:** Internet connection |
| **Test Objective:** Ensure the display of the correct error message when attempting to log in with a valid username and an incorrect password. | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, invalid password, login endpoint | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/login the password in the request body should be encoded in base64 {valid\_user} = valid user with request body {  "username": "{valid\_user}",  "password": "{invalid\_password}" } | API expected to return a success response with status code 200 and status: ok with response body that includes error message |
| 2 | Verify the status code | The status code should be 200. |
| 3 | Verify the status.ok | The response body should include "status": "ok". |
| 4 | Verify the response body | the response body should be: {  "errorMessage": "Wrong password." } |

|  |  |  |
| --- | --- | --- |
| Test Case 008 | **Test case name:**  Log In(API) With Unregistered Username | **Pre-conditions:** Internet connection |
| **Test Objective:** Ensure the display of the correct error message when attempting to log in with an unregistered username and any password. | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** None registered username, password, login endpoint | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/login {user} = None registered username {password} = password with request body {  "username": "{user}",  "password": "{password}" } | API expected to return a success response with status code 200 and status: ok with response body that includes error message |
| 2 | Verify the status code | The status code should be 200. |
| 3 | Verify the status.ok | The response body should include "status": "ok". |
| 4 | Verify the response body | the response body should be: {  "errorMessage": "User does not exist." } |

|  |  |  |
| --- | --- | --- |
| Test Case 009 | **Test case name:** ValidLogin(UI) | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:** Verify successful login with valid username and password | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, valid password, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "Log in" link located in the top right side of the page. | Log in page pops up |
| 2 | Insert the user name in the first name input | the text will appear in the username name input field |
| 3 | Insert the password in the password input | the text will appear in the password input field |
| 4 | Click on the "Log in" button | The browser directs the user to the home page |
| 5 | Verify the welcome message in the top right menu | The user should see a welcome message: "Welcome {username}". {username} = the same user that was inserted in the user input in the login page |

|  |  |  |
| --- | --- | --- |
| Test Case 010 | **Test case name:** Login(UI) With Username Only | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:**  Verify that a user receives an appropriate error message when attempting to log in with a valid username but without providing a password. | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "Log in" link located in the top right side of the page. | Log in page pops up |
| 2 | Insert the user name in the username input | the text will appear in the username name input field |
|  | Verify the password input field is empty | The password input filed should not contain any text |
| 3 | Click on the "Log in" button | Alert pops up with a message. |
| 4 | Verify the alert message | The user should see an alert message: "Please fill out Username and Password." |

|  |  |  |
| --- | --- | --- |
| Test Case 011 | **Test case name:** Login(UI) With Password Only | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:**  Verify that a user receives an appropriate error message when attempting to login with a valid password but without providing a username. | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid password, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "Log in" link located in the top right side of the page. | Log in page pops up |
| 2 | Verify the username input field is empty | The username input filed should not contain any text |
| 3 | Insert the password in the password input | the text will appear in the password input field |
| 4 | Click on the "Log in" button | Alert pops up with a message |
| 5 | Verify the alert message | The user should see an alert message: "Please fill out Username and Password." |

|  |  |  |
| --- | --- | --- |
| Test Case 012 | **Test case name:** Login(UI) With wrong password | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:** Verify that a user receives an appropriate error message when attempting to login with a valid username and invalid password | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, invalid password, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "Log in" link located in the top right side of the page. | Log in page pops up |
| 2 | Insert the user name in the username input | the text will appear in the username name input field |
| 3 | Insert the invalid password in the password input | the text will appear in the password input field |
| 4 | Click on the "Log in" button | Alert pops up with a message |
| 5 | Verify the alert message | The user should see an alert message:  "Wrong password." |

|  |  |  |
| --- | --- | --- |
| Test Case 013 | **Test case name:** Verify Successful Welcome Message Via UI With Login via API | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:** Ensure that a user can log in via API, add the authentication token to the browser's cookies, and verify the welcome message on the web UI. | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, valid password in base64 format, demo blaze home page URL | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/login the password in the request body should be encoded in base64 {valid\_user} = valid user {valid\_password} = encoded valid password with request body {  "username": "{valid\_user}",  "password": "{valid\_password}" } | API expected to return a success response with status code 200 and status: ok with response body that includes auth\_token |
| 2 | Verify the response body | The response body should include and auth token example for response body: "Auth\_token: c0R0TExmSW8xNzIzNTQ4" |
| 3 | Update the page cookie with the token and refresh the page | Browser directs the user to the home page |
| 4 | Verify the welcome message in the top right menu | The user should see a welcome message: "Welcome {username}". {username} = the same user that was inserted in the user input in the login page |

|  |  |  |
| --- | --- | --- |
| Test Case 014 | **Test case name:** Logout(UI) | **Pre-conditions:** Internet connection, Chrome browser, user logged in demo blaze home page |
| **Test Objective:** Verify successful logout | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** valid username, valid password | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "Log out" link located in the top right side of the page. | Browser directs the user to the home page |
| 2 | Verify the "Log in" link appears in the top right menu | The user should see the "Log in" link in the top right menu |
| 3 | Verify the "Sign up" link appears in the top right menu | The user should see the "Sign up" link in the top right menu |

|  |  |  |
| --- | --- | --- |
| Test Case 015 | **Test case name:** About Us video player(UI) | **Pre-conditions:** Internet connection, Chrome browser, demo blaze home page |
| **Test Objective:** Verify the functionality of the main play button and video control play button and full screen | **Test Designed by:** Ehab Khalil |
| **Date:** 06/08/2024 |
| **Data:** None | **Test Priority:** 2 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on "About us" link located in the top right section of the page | About us page pops up |
| 2 | Click on the play link(big play icon) located in the top left of the video player | The user should see the video start plating |
| 3 | Click on the play button(small play icon) located in the bottom left of the video player | The user should see the video stops playing |
| 4 | Click on the full screen button(full screen icon) located in the bottom right of the video player | The user should see the video in full screen mode |
| 5 | Verify that the video state | The video should be paused and not playing |
| 6 | Verify the video screen mode | The video should be in full screen mode |

|  |  |  |
| --- | --- | --- |
| Test Case 016 | **Test case name:**  Send Contact Message(UI) | **Pre-conditions:** Internet connection, Chrome browser |
| **Test Objective:** Confirm that the contact message is sent successfully and appropriate confirmation is received. | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** Contact Email, Contact Name, Contact Message | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on "Contact" link located in the top right menu | Contact page pops up |
| 2 | Insert contact email in the contact email input field | The user should see the text in the input field |
| 3 | Insert contact name in the contact name input field | The user should see the text in the input field |
| 4 | Insert a message in the message input field | The user should see the text in the input field |
| 5 | Click "Send message" button | Alert pops up with a message |
| 6 | Verify the alert message | The alert message should be:  "Thanks for the message!!" |

|  |  |  |
| --- | --- | --- |
| Test Case 017 | **Test case name:**  View Cart With Unregistered User(API) | **Pre-conditions:** Internet connection, Chrome Browser |
| **Test Objective:** Validate that the cart items are correctly retrieved via the API With Unregistered User | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** Cart view endpoint, user cookie | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/viewcart request body {  "cookie": {user cookie},  "flag": false } | API expected to return a success response with status code 200 and status: ok response body should include product list: {  "Items": [{product\_list}] } |
| 2 | Verify the status code | The status code should be 200. |
| 3 | Verify the status state | The status state should be : ok |
| 4 | verify the response body | The response body should be in this structure: {  "Items": [{product\_list}] } |

|  |  |  |
| --- | --- | --- |
| Test Case 018 | **Test case name:**  View Cart With Registered User(API) | **Pre-conditions:** Internet connection, auth token after user log in |
| **Test Objective:**  Validate that the cart items are correctly retrieved via the API With Registered User | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** Cart view endpoint, user cookie | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Send a POST request to: https://api.demoblaze.com/viewcart request body: {  "cookie": auth\_token,  "flag": "true" } | API expected to return a success response with status code 200 and status: ok response body should include product list: {  "Items": [product\_list] } |
| 2 | Verify the status code | The status code should be 200. |
| 3 | Verify the status state | The status state should be : ok |
| 4 | verify the response body | The response body should be in the next structure: {  "Items": [product\_list] } |

|  |  |  |
| --- | --- | --- |
| Test Case 019 | **Test case name:**  Add Product To Cart By Name With Unregistered User(UI) | **Pre-conditions:** Internet connection, Chrome Browser |
| **Test Objective:** Verify Success Message Displayed Upon Clicking "Add to Cart" Button | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** Product name | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the product by the product name | The browser navigates the user to the product page |
| 2 | click on the "Add to cart" button | Alert message pops up |
| 3 | Verify the alert message | The user should get the next message: "Product added" |

|  |  |  |
| --- | --- | --- |
| Test Case 020 | **Test case name:**  Purchase Product By Name With Unregistered User(UI) | **Pre-conditions:** Internet connection, Chrome Browser, user in demo blaze home page |
| **Test Objective:** Verify Success Message Displayed Upon Purchasing a Product by Name with a Unregistered User. | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** Product name, user details(credit card details, name, city, country) | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the product to purchase by name | The browser navigates the user to the product page |
| 2 | click on the "Add to cart" button | Alert message pops up |
| 3 | Accept the Alert by Pressing ENTER | The alert message should disappear. |
| 4 | Click on the cart button located in the top right menu | The browser navigates the user to the cart page |
| 5 | Click on the green button " Place Order" located in the right side of the cart display | Place Order page pops up |
| 6 | Insert the name in the name input field | The user should see the text appear in the field input |
| 7 | Insert the country in the country input field | The user should see the text appear in the field input |
| 8 | Insert the city in the city input field | The user should see the text appear in the field input |
| 9 | Insert the credit card number in the credit card number input field | The user should see the text appear in the field input |
| 10 | Insert the Credit Card Expiry Month in the Credit Card Expiry Month input field | The user should see the text appear in the field input |
| 11 | Insert the Credit Card Expiry year in the Credit Card Expiry year input field | The user should see the text appear in the field input |
| 12 | Click the " Purchase" button | Success purchase page pops up |
| 13 | Verify the alert message | The user should see the message: "Thank you for your purchase!" |

|  |  |  |
| --- | --- | --- |
| Test Case 021 | **Test case name:**  Purchase Product By Name With Registered User(UI) | **Pre-conditions:** Internet connection, Chrome Browser, user logged in demo blaze home page |
| **Test Objective:** Verify Success Message Displayed Upon Purchasing a Product by Name with a Registered User. | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** Product name, user details(credit card details, name, city, country) | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the product to purchase by name | The browser navigates the user to the product page |
| 2 | click on the "Add to cart" button | Alert message pops up |
| 3 | Accept the Alert by Pressing ENTER | The alert message should disappear. |
| 4 | Click on the cart button located in the top right menu | The browser navigates the user to the cart page |
| 5 | Click on the green button " Place Order" located in the right side of the cart display | Place Order page pops up |
| 6 | Insert the name in the name input field | The user should see the text appear in the field input |
| 7 | Insert the country in the country input field | The user should see the text appear in the field input |
| 8 | Insert the city in the city input field | The user should see the text appear in the field input |
| 9 | Insert the credit card number in the credit card number input field | The user should see the text appear in the field input |
| 10 | Insert the Credit Card Expiry Month in the Credit Card Expiry Month input field | The user should see the text appear in the field input |
| 11 | Insert the Credit Card Expiry year in the Credit Card Expiry year input field | The user should see the text appear in the field input |
| 12 | Click the " Purchase" button | Success purchase page pops up |
| 13 | Verify the alert message | The user should see the message: "Thank you for your purchase!" |

|  |  |  |
| --- | --- | --- |
| Test Case 022 | **Test case name:**  Purchase 2 Products By Name With Registered User(UI+API) | **Pre-conditions:** Internet connection, Chrome Browser, user logged in demo blaze home page through API |
| **Test Objective:** Verify Success Message Displayed Upon Purchasing 2 Products by Name with a Registered User. | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** 2 Products name, user details(credit card details, name, city, country) | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the first product to purchase by name | The browser navigates the user to the product page |
| 2 | click on the "Add to cart" button | Alert message pops up |
| 3 | Accept the Alert by Pressing ENTER | The alert message should disappear. |
| 4 | Click on the store logo in the top left corner of the page | The browser navigates the user to the home page |
| 5 | Click on the "Next" button located in the bottom right corner of the page | The browser navigates to the next section, displaying the home page product list. |
| 6 | Click on the second product to purchase by name | The browser navigates the user to the product page |
| 7 | click on the "Add to cart" button | Alert message pops up |
| 8 | Accept the Alert by Pressing ENTER | The alert message should disappear. |
| 9 | Click on the cart button located in the top right menu | The browser navigates the user to the cart page |
| 10 | Click on the green button " Place Order" located in the right side of the cart display | Place Order page pops up |
| 11 | Insert the name in the name input field | The user should see the text appear in the field input |
| 12 | Insert the country in the country input field | The user should see the text appear in the field input |
| 13 | Insert the city in the city input field | The user should see the text appear in the field input |
| 14 | Insert the credit card number in the credit card number input field | The user should see the text appear in the field input |
| 15 | Insert the Credit Card Expiry Month in the Credit Card Expiry Month input field | The user should see the text appear in the field input |
| 16 | Insert the Credit Card Expiry year in the Credit Card Expiry year input field | The user should see the text appear in the field input |
| 17 | Click the " Purchase" button | Success purchase page pops up |
| 18 | Verify the alert message | The user should see the message: "Thank you for your purchase!" |

|  |  |  |
| --- | --- | --- |
| Test Case 023 | **Test case name:**  Send empty contact message(UI) | **Pre-conditions:** Internet connection, Chrome Browser |
| **Test Objective:** Validate that an appropriate error alert message is displayed when attempting to send a contact message with empty fields. | **Test Designed by:** Ehab Khalil |
| **Date:** 21/07/2024 |
| **Data:** None | **Test Priority:** 1 |
|  |  |  |
| **Action** | **Step** | **Expected Result** |
| 1 | Click on the "Contact" link located in the top left menu of the page | The browser directs the user to the contact page |
| 2 | Verify all the input fields are empty | The user should not see text in the: Contact Email Contact Name Message |
| 3 | Click on the "Send message" button | The user should see appropriate error alert message is displayed when attempting to send a contact message with empty fields. |

**4. Pass/Fail Criteria**

**Pass Criteria**

1. **Functional Requirements Met:**
   * The test item meets all specified functional requirements.
   * All test cases for the feature are executed and pass without any critical or high-severity defects.
2. **Performance Criteria Met**
   * The application performs within acceptable limits under various load conditions.
3. **Usability Criteria Met**
   * The application is intuitive and user-friendly.
4. **Security Criteria Met**
   * The application is secure from potential threats and vulnerabilities.
5. **Compatibility Criteria Met**
   * The application functions correctly across different browsers, devices, and operating systems.
6. **No Critical Defects**
   * No critical or high-severity defects are present in the application.

**Fail Criteria**

1. **Functional Requirements Not Met:**
   * The test item fails to meet one or more specified functional requirements.
   * Any critical or high-severity defects are identified in the feature.
2. **Performance Criteria Not Met**
   * The application does not perform within acceptable limits under load conditions.
3. **Usability Criteria Not Met**
   * The application is not intuitive or user-friendly.
4. **Security Criteria Not Met**
   * The application has vulnerabilities or security threats.

**5. Summary**

The Software Test Design (STD) consists of a number of stages which progressively elaborate the design of tests from an initial high-level strategy to detailed test procedures. The design of tests must be driven by the specification of the software. At the highest level, this means that tests will be designed to verify that the software faithfully implements the requirements of the Requirements Specification. At lower levels, tests will be designed to verify that items of software implement all design decisions made in the Architectural Design Specification and Detailed Design Specifications. As with any design process, each stage of the test design process should be subject to informal and formal review. The ease with which tests can be designed is highly dependent on the design of the software. It is important to consider testability as a key (but usually undocumented) requirement for any software development.