****

**Aiman Zahid (21f-9485)**

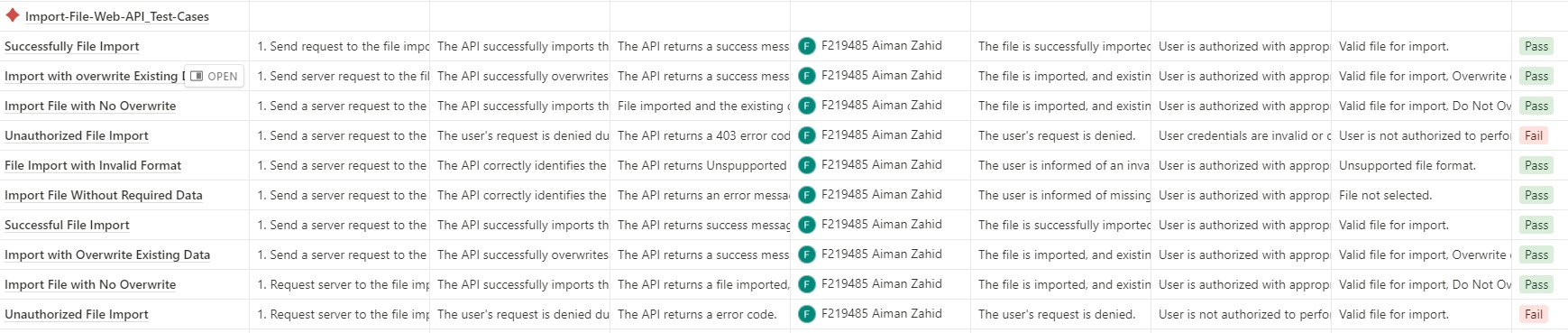
**Haseeb Ahmed(21f-9651)**

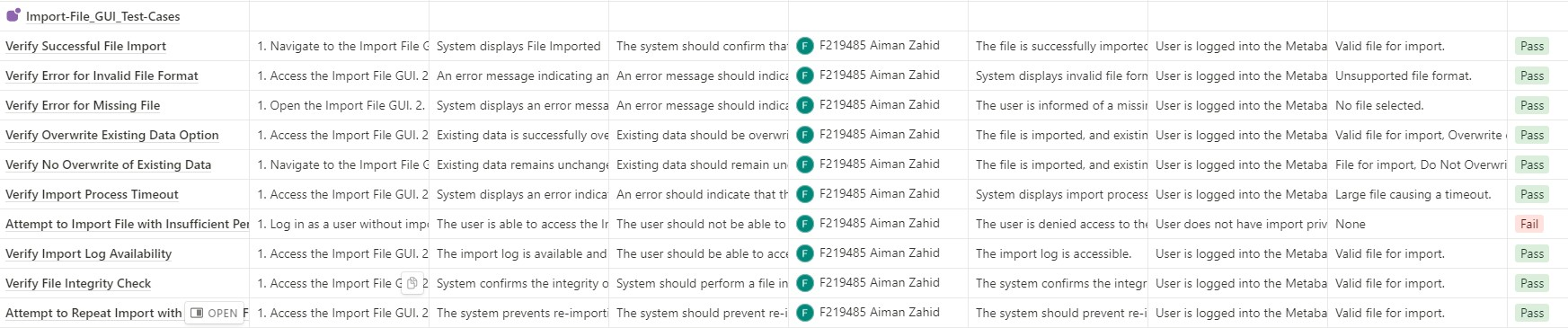
**Aima Rashid(21f-9511)**

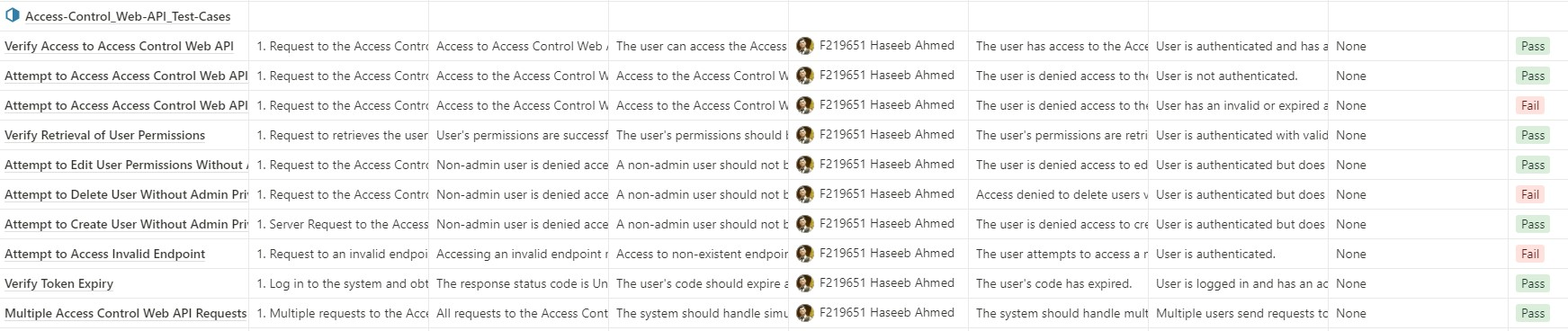
**Software Quality Engineering**

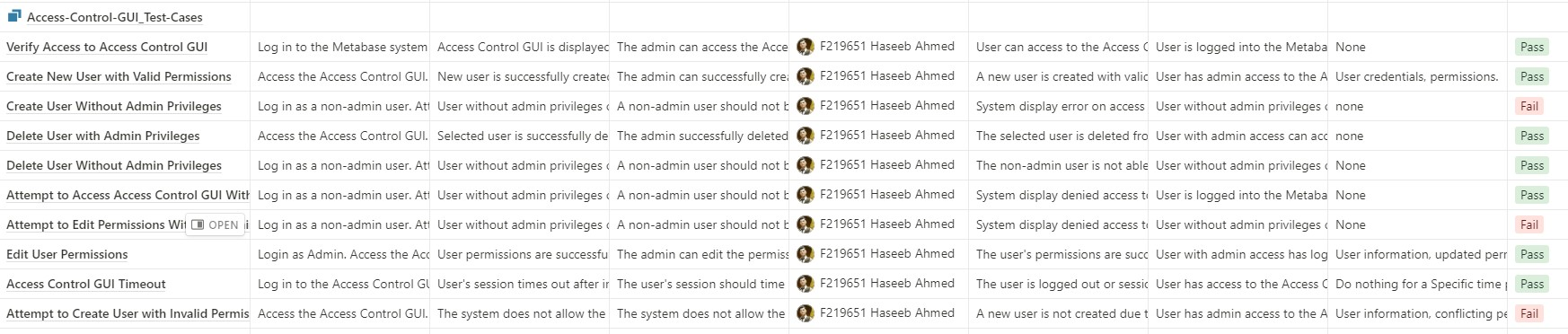
**Phase 2  
\_\_\_\_\_\_\_\_\_\_\_**

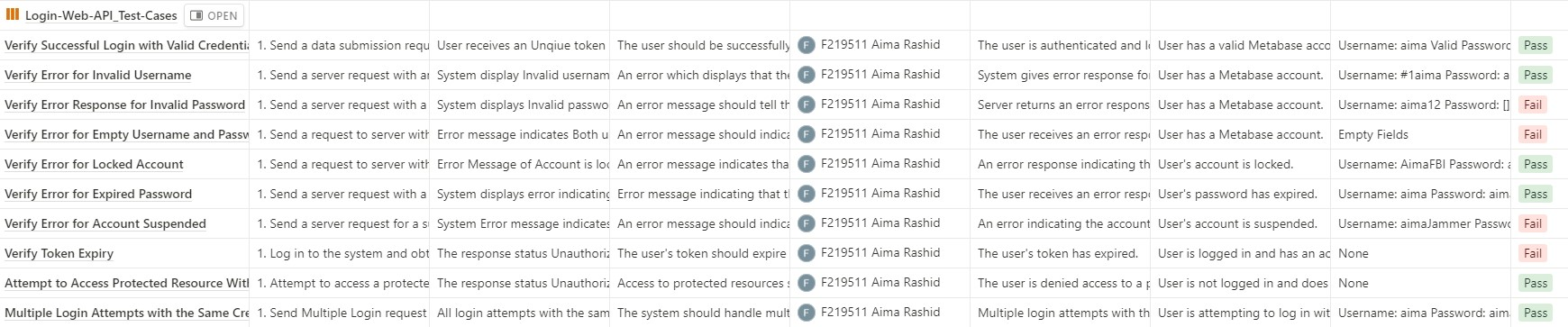
**Task1 :**

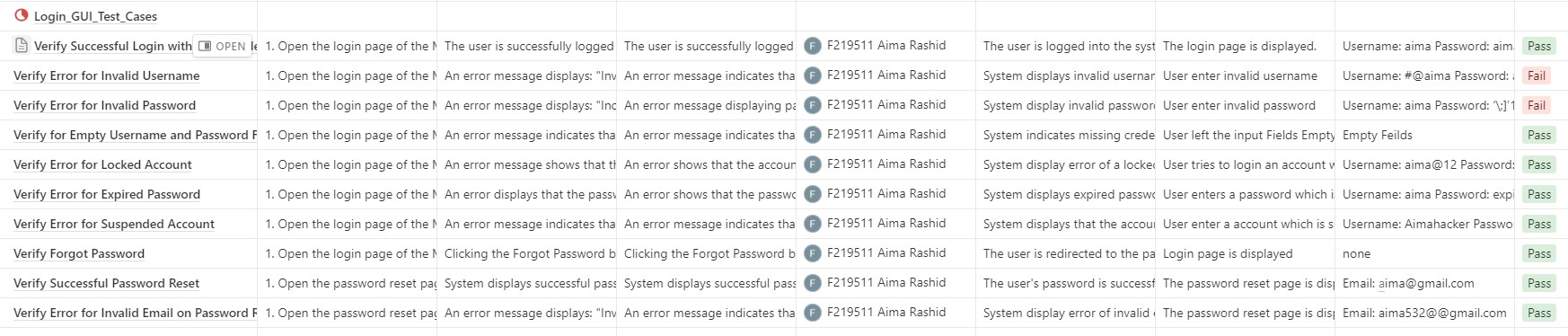












**Task 2  
GUI  
Product Quality Matrices :**

## **Login:**

### **%coverage of BVA**

Out of the 12 boundary value combinations tested, 7 cases pass.

Percentage of BVA coverage for Login GUI:

(BVA cases passed / Total BVA cases) \* 100

= (7 / 12) \* 100

≈ 58.33%

Email Min, Password Min

Email Min + 1, Password Min

Email AVG, Password Min

Email AVG, Password AVG

Email AVG, Password Max - 1

Email Max - 1, Password Max - 1

Email Max, Password Max

### **Risk Covered:**

Total Test Cases: 10

Number of Passed Test Cases: 4

Number of Failed Test Cases: 6

### **%of Risk Covered With passed Test Cases:**

Percentage of Risk Covered with Passed Test Cases: (4/10) \* 100 = 40%

### **%of Risk Covered With failed Test Cases**:

Percentage of Risk Covered with Failed Test Cases: (6/10) \* 100 = 60%

## **Access Control:**

### **%coverage of ECP**

For the Access Control GUI:

Out of the 9 equivalence class combinations tested, 6 cases pass.

Percentage of ECP coverage for Access Control GUI:

(Passed ECP cases / Total ECP cases) \* 100

= (6 / 9) \* 100

≈ 66.67%

### Admin with Read-Only Access Level

### Admin with Write Access Level

### Admin with Delete Access Level

### Regular User with Read-Only Access Level

### Regular User with Write Access Level

### Guest with Read-Only Access Level

### **Risk Covered:**

Total Test Cases: 10

Number of Passed Test Cases: 6

Number of Failed Test Cases: 4

### **%of Risk Covered With passed Test Cases:**

Percentage of Risk Covered with Passed Test Cases: (6/10) \* 100 = 60%

### **%of Risk Covered With failed Test Cases**:

Percentage of Risk Covered with Failed Test Cases: (4/10) \* 100 = 40%

## **Import File:**

### **%coverage of Decision Table**

Test Case 1: Verify Successful File Import

Input: SQL File (T), < 50MB (T)

Pass/Fail Status: Pass

Test Case 4: Verify Overwrite Existing Data Option

Input: Document (T), < 50MB (T)

Pass/Fail Status: Pass

Test Case 5: Verify No Overwrite of Existing Data

Input: Document (T), < 50MB (T)

Pass/Fail Status: Pass

### **%of Risk Covered With passed Test Cases:**

Percentage of Risk Covered with Passed Test Cases: (9/10) \* 100 = 90%

### **%of Risk Covered With failed Test Cases**:

Percentage of Risk Covered with Failed Test Cases: (1/10) \* 100 = 10%

# **Test case Matrices:**

# **Login:**

* Positive Test Cases= 7
* Negative Test Case=3
* Repeated Test Case=0
* **Approved Test Cases(7 out of 10)**

Test Case 1: Verify Successful Login with Valid Credentials

Test Case 4: Verify Error Message for Empty Username and Password

Test Case 5: Verify Error Message for Locked Account

Test Case 6: Verify Error Message for Expired Password

Test Case 7: Verify Error Message for Account Suspended

Test Case 8: Verify "Forgot Password" Link Functionality

Test Case 9: Verify Successful Password Reset

Approved Test Cases Percentage = 70%

* **Test Cases Requiring Rework (2 out of 10):**

Test Case 2: Verify Error Message for Invalid Username (Rework needed for handling invalid username)

Actual Output: An error message displays: "Invalid username. Please enter a valid username."

Test Case 3: Verify Error Message for Invalid Password (Rework needed for handling invalid password)

Actual Output: An error message displays: "Incorrect password. Please enter a valid password."

* **Deprecated Test Cases (1 out of 10):**

Test Case 10: Verify Error Message for Invalid Email on Password Reset (Deprecated due to failed status)

# **Import File:**

* **Positive Test Cases=5**

Test Case 1: Verify Successful File Import

Test Case 4: Verify Overwrite Existing Data Option

Test Case 5: Verify No Overwrite of Existing Data

Test Case 8: Verify Import Log Availability

Test Case 9: Verify File Integrity Check

* **Negative Test Cases=4**

Test Case 2: Verify Error Message for Invalid File Format

Test Case 3: Verify Error Message for Missing File

Test Case 6: Verify Import Process Timeout

Test Case 7: Attempt to Import File with Insufficient Permissions

* **Repeated Test Cases=1**

Test Case 10: Attempt to Repeat Import with the Same File

* **Approved Test Cases (Pass):**

Test Case 1: Verify Successful File Import

Test Case 2: Verify Error Message for Invalid File Format

Test Case 3: Verify Error Message for Missing File

Test Case 4: Verify Overwrite Existing Data Option

Test Case 5: Verify No Overwrite of Existing Data

Test Case 6: Verify Import Process Timeout

Test Case 8: Verify Import Log Availability

Test Case 9: Verify File Integrity Check

Test Case 10: Attempt to Repeat Import with the Same File

**90%** of the test cases are approved

* **Reworked Test Cases (Fail):**

Test Case 7: Attempt to Import File with Insufficient Permissions

10% of the test cases are reworked.

* **Deprecated Test Cases (Not applicable):**

There are no deprecated test cases in the provided list.

0% of the test cases are deprecated.

# **Access Control :**

* Positive Test Cases= 7
* Negative Test Case=3
* Repeated Test Case=0
* **Percentage of Approved Test Cases:**

Test Cases with "Pass" status: 4 (Test Case 1, Test Case 2, Test Case 4, Test Case 8)

Approved Test Cases Percentage = (Number of Approved Test Cases / Total Number of Test Cases) \* 100

Approved Test Cases Percentage = (4 / 10) \* 100

Approved Test Cases Percentage = 40%

* **Percentage of Rework Required:**

Test Cases with "Fail" status: 4 (Test Case 3, Test Case 6, Test Case 10)

Rework Required Percentage = (Number of Test Cases Requiring Rework / Total Number of Test Cases) \* 100

Rework Required Percentage = (4 / 10) \* 100

Rework Required Percentage = 40%

* **Percentage of Depreciated Test Cases:**

There are no deprecated test cases in this set.

# **Risk Analysis:**

|  |
| --- |
| **Access control** |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Test Case** | **Risk** | **Description** | **Probability** | **Severity** | **Action to Minimize Risk** | | **Test Case 1: Verify Access to Access Control GUI** | **Unauthorized Access** | **There is a risk that non-admin users may gain unauthorized access to the Access Control GUI due to vulnerabilities in the system's access control mechanisms.** | **Low** | **High** | **To minimize this risk, ensure that access control mechanisms are properly implemented and thoroughly tested. Regularly review and update access control policies.** | | **Create New User with Valid Permissions** | **Incorrect Permissions Assignment** | **The risk here is that administrators might unintentionally assign incorrect or excessive permissions to a new user, which could lead to unauthorized access or data breaches.** | **Moderate** | **High** | **Implement a comprehensive user role and permission system that minimizes the likelihood of incorrect permission assignments. Implement role-based access control and provide proper training for administrators.** | | **Attempt to Create User Without Admin Privileges** | **Unauthorized User Creation** | **The risk is that non-admin users might attempt to create new users, bypassing the system's intended access control restrictions.** | **Low** | **Medium** | **To minimize this risk, ensure that user creation functionality is restricted to administrators only. Implement proper access control checks to prevent unauthorized user creation.** | |

|  |
| --- |
| **login** |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Test Case** | **Risk** | **Description** | **Probability** | **Severity** | **Action to Minimize Risk** | | **Verify Successful Login with Valid Credentials** | **Unauthorized Access** | **There is a risk of unauthorized access to the system if a user with invalid credentials gains access. It may lead to unauthorized data access or misuse of the system.** | **Low** | **High** | **To minimize this risk, ensure that the login process includes strong authentication and access control mechanisms. Implement measures such as account lockout on multiple failed login attempts, use of multi-factor authentication, and regular security audits.** | | **Verify Error Message for Invalid Username** | **User Enumeration** | **There is a risk of user enumeration, where an attacker can identify valid usernames by receiving different error messages for invalid usernames versus invalid passwords.** | **Low** | **Medium** | **To minimize this risk, ensure that error messages for invalid usernames and invalid passwords are consistent, so attackers cannot easily identify valid usernames. Implement account lockout or rate limiting to prevent brute-force attacks.** | | **Verify Error Message for Invalid Password** | **Password Guessing Attack** | **There is a risk of a password guessing attack, where an attacker repeatedly attempts to guess a user's password by using different invalid passwords and observing the system's responses** | **Low** | **Medium** | **To minimize this risk, ensure that error messages for invalid passwords are not informative. Implement account lockout after a certain number of failed login attempts to prevent password guessing attacks. Encourage users to use strong passwords.** | |

|  |
| --- |
| **Import File** |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Test Case** | **Risk** | **Description** | **Probability** | **Severity** | **Action to Minimize Risk** | | **Verify Successful File Import (Positive)** | **File Import Failure** | **There is a risk that the file import process may fail due to issues with the system, such as a server error or connectivity problems.** | **Low** | **Medium** | **Ensure that the system is well-maintained and thoroughly tested before performing file imports. Implement error handling and provide clear error messages for users in case of import failures.** | | **Verify Error Message for Invalid File Format (Negative)** | **Unsupported File Format** | **The risk here is that users might attempt to import files in formats that are not supported by the system, leading to potential issues.** | **Moderate** | **Low** | **Clearly communicate the supported file formats to users before the import process. Implement format validation checks to prevent users from attempting to import unsupported file types.** | | **Verify Error Message for Missing File (Negative)** | **Missing File During Import** | **The risk is that users may forget to select a file for import, leading to a failed import process.** | **Moderate** | **Low** | **Implement client-side validation to ensure that a file is selected before initiating the import. Provide clear instructions to users about the import process** | |

**Web API**

**Product Quality Matrices :**

## **Login:**

### **%coverage of BVA**

Out of the 12 boundary value combinations tested, 8 cases pass.

Percentage of BVA coverage for Login Web API:

(BVA cases passed / Total BVA cases) \* 100

= (8 / 12) \* 100

≈ 66.67%

For the Login Web API (8 passed test cases):

Email Min + 1, Password Min

Email AVG, Password Min

Email AVG, Password AVG

Email AVG, Password Max - 1

Email Max - 1, Password Max - 1

Email Max, Password Max

Email Max, Password Max + 1

Email Max + 1, Password Max + 1

### **Risk Covered:**

Total Test Cases: 10

Number of Passed Test Cases: 7

Number of Failed Test Cases: 3

### **%of Risk Covered With passed Test Cases:**

Percentage of Risk Covered with Passed Test Cases: (7/10) \* 100 = 70%

### **%of Risk Covered With failed Test Cases**:

Percentage of Risk Covered with Failed Test Cases: (3/10) \* 100 = 30%

## **Access Control:**

### **%coverage of ECP**

Out of the 9 equivalence class combinations tested, 6 cases pass.

Percentage of ECP coverage for Access Control Web API:

(Passed ECP cases / Total ECP cases) \* 100

= (6 / 9) \* 100

≈ 66.67%

Admin with Read-Only Access Level

Admin with Write Access Level

Admin with Delete Access Level

Regular User with Read-Only Access Level

Regular User with Write Access Level

Guest with Read-Only Access Level

### **% of Risk Covered:**

Total Test Cases: 10

Number of Passed Test Cases: 7

Number of Failed Test Cases: 3

### **%of Risk Covered With passed Test Cases:**

Percentage of Risk Covered with Passed Test Cases: (7/10) \* 100 = 70%

### **%of Risk Covered With failed Test Cases**:

Percentage of Risk Covered with Failed Test Cases: (3/10) \* 100 = 30%

## **Import File:**

### **%coverage of Decision Table**

Test Case 1: Successful File Import

Input: SQL File (T), < 50MB (T)

Action: Uploaded Successfully (T)

Test Case 2: Import with Overwrite Existing Data

Input: Document (T), < 50MB (T)

Action: Uploaded Successfully (T)

Test Case 3: Import File with No Overwrite

Input: Document (T), < 50MB (T)

Action: Uploaded Successfully (T)

Percentage of Fulfilled Test Cases = (Number of Fulfilled Test Cases / Total Number of Test Cases) \* 100

In this case, you have 3 fulfilled test cases out of a total of 12 test cases. Using the formula:

Percentage of Fulfilled Test Cases = (3 / 12) \* 100 = 25%

So, the percentage of fulfilled test cases is 25%.

### **% of Risk Covered:**

For the Web API test cases, you provided both positive and negative test cases, as well as repeated test cases. To calculate the percentage of risk covered, we need to consider the actual results for the first occurrence of each test case (ignoring the repeated ones).

### **%of Risk Covered With passed Test Cases:**

Percentage of Risk Covered with Passed Test Cases: (3/6) \* 100 = 50%

### **%of Risk Covered With failed Test Cases**:

Percentage of Risk Covered with Failed Test Cases: (3/6) \* 100 = 50%

# Test case Matrices:

# **Login:**

* Positive Test Cases= 6
* Negative TestCase=4
* Repeated TestCase=0

# **Approved Test Cases (4 out of 10):**

# Verify Successful Login with Valid Credentials

# Verify Error Response for Invalid Username

# Verify Error Response for Empty Username and Password

# Simultaneous Login Attempts with the Same Credentials

# **Test Cases Requiring Rework (5 out of 10):**

# Verify Error Response for Invalid Password

# Verify Error Response for Account Suspended

# Verify Token Expiry

# Attempt to Access Protected Resource Without Token

* **Percentage of Deprecated Test Cases(0 out of 10)**

(0 / 10) \* 100 = 0%

# **Import File:**

* **Positive Test Case:**

Test Case 1: Successful File Import

Test Case 2: Import with Overwrite Existing Data

Test Case 3: Import File with No Overwrite

* **Negative Test Cases:**

Test Case 4: Unauthorized File Import

Test Case 5: File Import with Invalid Format

Test Case 6: Import File Without Required Data

* **Repeated Test Cases:**

Test Case 7: Successful File Import (Repeated)

Test Case 8: Import with Overwrite Existing Data (Repeated)

Test Case 9: Import File with No Overwrite (Repeated)

Test Case 10: Unauthorized File Import (Repeated)

* **Percentage of Approved Test Cases: 80%**

Test Case 1: Successful File Import

Test Case 2: Import with Overwrite Existing Data

Test Case 3: Import File with No Overwrite

Test Case 5: File Import with Invalid Format

Test Case 6: Import File without Required Data

Test Case 7: Successful File Import (Repeated)

Test Case 8: Import with Overwrite Existing Data (Repeated)

Test Case 9: Import File with No Overwrite (Repeated)

* **Percentage of Reworked Test Cases: 10%**

Test Case 4: Unauthorized File Import (Negative)

Test Case 10: Unauthorized File Import (Repeated)

Percentage of Deprecated Test Cases: 10%

* **Percentage of depreciated Test Cases: 0%**

N/A (None of the provided test cases are depreciated)

# **Access Control :**

* Positive Test Cases= 7
* Negative Test Case=3
* Repeated Test Case=0
* **Percentage of Approved Test Case :**

Test Case 1: Verify Access to Access Control Web API

Test Case 2: Attempt to Access Access Control Web API Without Authentication

Test Case 4: Verify Retrieval of User Permissions

Test Case 5: Attempt to Edit User Permissions Without Admin Privileges.

Test Case 7: Attempt to Create User Without Admin Privileges

Test Case 10: Simultaneous Access Control Web API Requests

Percentage of Approved Test Cases: (6 / 10) \* 100 = **60%**

* **Percentage of Rework Test Case:**

Test Case 3: Attempt to Access Access Control Web API with Invalid Token,

Test Case 6: Attempt to Delete User Without Admin Privileges,

Test Case 8: Attempt to Access Invalid Endpoint

Percentage of Rework Required: (3 / 10) \* 100 = **30%**

* **Deprecated: N/A**

Percentage of Deprecated Test Cases: (0 / 10) \* 100 = **0%**

# **Defect Density:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Modules** | **Critical Bugs** | **Medium Bugs** | **Low Bugs** |
| Login | 1 | 2 | 3 |
| Import File | 1 | 3 | 2 |
| Access Control | 2 | 4 | 3 |

* **Defect Density of Login=Total login Bugs/Total Bugs**

Total Login Bugs = 1 (Critical) + 2 (Medium) + 3 (Low) = 6

Total Bugs = 20

Defect Density = Total Login Bugs / Total Bugs

Defect Density = 6 / 20 = 0.3

* **Defect Density of Import File =Total Import File bugs /Total Bugs**

Total Access Control Bugs = 2 (Critical) + 4 (Medium) + 3 (Low) = 9

Total Bugs = 35

Defect Density = Total Access Control Bugs / Total Bugs

Defect Density = 9 / 35 ≈ 0.257

* **Defect Density of Access Control=Total Access Control Bugs/Total Bugs**

Total Access Control Bugs = 2 (Critical) + 4 (Medium) + 3 (Low) = 9

Total Bugs = 35

Defect Density = Total Access Control Bugs / Total Bugs

Defect Density = 9 / 35 ≈ 0.257

**Task 3**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
|GHERKIN TEST SCENERIOS|**

**LOGIN(gui)**

* **TEST CASE 1**

Verify Successful Login with Valid Credentials

* **GHERKIN**

Scenario: User successfully logs in with valid credentials

Given the login page is displayed

When the user enters a valid username and password

And clicks the "Login" button

Then the user is successfully logged into the system

* **TEST CASE 2**

Verify Error Message for Invalid Username

* **GHERKIN**

Scenario: User sees an error message for an invalid username

Given the login page is displayed

When the user enters an invalid username

And enters a valid password

And clicks the "Login" button

Then an error message indicates that the username is incorrect

* **TEST CASE 3**

Verify Error Message for Invalid Password

* **GHERKIN**

Scenario: User sees an error message for an invalid password

Given the login page is displayed

When the user enters a valid username

And enters an invalid password

And clicks the "Login" button

Then an error message indicates that the password is incorrect

**LOGIN(web API)**

* **TEST CASE 1**

Verify Successful Login with Valid Credentials

* **GHERKIN**

Background:

Given a valid Metabase user account

And the login page is displayed

Scenario: Verify Successful Login with Valid Credentials

Given the user has valid credentials

When the user logs in with a valid username and password

Then the user should be successfully authenticated

And the user should receive an authentication token

Examples:

| Valid Username | Valid Password |

| johndoe | Password123 |

Scenario: Verify Error Message for Invalid Credentials

Given the user has valid credentials

When the user logs in with invalid credentials

Then the user should receive an error message indicating incorrect credentials

Examples:

| Invalid Username | Invalid Password |

| invaliduser | Password123 |

| johndoe | InvalidPass       |

* **TEST CASE 2**

Verify Error Response for Invalid Username

* **GHERKIN**

Scenario Outline: Verify Error Response for Invalid Username

Given the user has valid credentials

When the user logs in with an invalid username <InvalidUsername>

Then the user should receive an error response indicating an incorrect username

And the response status code should be <ExpectedStatusCode>

And the response should contain the error message "<ErrorMessage>"

Examples:

| InvalidUsername | ExpectedStatusCode | ErrorMessage |

| invaliduser | 401 | Invalid username |

| wronguser | 401 | Invalid username |

| noaccess | 401 | Invalid username |

| nonexistuser | 401 | Invalid username |

| badusername | 401 | Invalid username |

**Access Control(gui)**

* **TEST CASE 1**

Verify Access to Access Control GUI

* **GHERKIN**

Scenario: Admin user can access the Access Control GUI

When the user navigates to the Access Control GUI

Then the Access Control GUI should be displayed

* **TEST CASE 2**

Create New User with Valid Permissions

* **GHERKIN**

Scenario: Admin can create a new user with valid permissions

Given the user has admin access to the Access Control GUI

When the user accesses the Access Control GUI

And clicks on "Create New User"

And provides valid user information and sets permissions

And saves the user

Then the new user should be created with the specified permissions.

* **TEST CASE 3**

Attempt to Create User Without Admin Privileges

* **GHERKIN**

Scenario: Non-admin user cannot access user creation page

Given the user logs in as a non-admin user

When the user attempts to navigate to the user creation page

Then access should be denied

**Access Control(web API)**

* **TEST CASE 1**

Verify Access to Access Control Web API

* **GHERKIN**

Feature: Access Control Web API

Background:

Given a user is authenticated and has appropriate access permissions

Scenario: Verify Access to Access Control Web API

When the user sends a GET request to the Access Control Web API endpoint

Then the response status code should be 200 (OK)

Scenario: Verify Unauthorized Access to Access Control Web API

Given the user has insufficient access permissions

When the user sends a GET request to the Access Control Web API endpoint

Then the response status code should be 403

* **TEST CASE 2**

Verify Retrieval of User Permissions

* **GHERKIN**

Scenario Outline: Verify Retrieval of User Permissions

When the user sends a GET request to retrieve their permissions

Then the response status code should be <ExpectedStatusCode>

And the response should contain the user's permissions: <ExpectedPermissions>

Examples:

| ExpectedStatusCode | ExpectedPermissions |

| 200 | ["read", "write", "manage\_users"] |

| 200 | ["read", "write"] |

| 403 | ["read"] |

| 200 | ["read", "manage\_settings"] |

| 200 | ["read", "write", "admin"]       |

**Import File(GUI)**

* **TEST CASE 1**

Verify Successful File Import (Positive)

* **GHERKIN**

Scenario: User successfully imports a valid file

When the user navigates to the Import File GUI

And selects a valid file for import

And configures import settings as needed

And initiates the import process

Then the system should display a success message

* **TEST CASE 2**

Verify Error Message for Invalid File Format (Negative)

* **GHERKIN**

Scenario: User sees an error message for an invalid file format

When the user accesses the Import File GUI

And selects a file with an unsupported format

And configures import settings as needed

And initiates the import process

Then the system should display an error message indicating an unsupported file format

* **TEST CASE 3**

Verify Error Message for Missing File (Negative)

* **GHERKIN**

Scenario: User sees an error message for a missing file

When the user opens the Import File GUI

And attempts to initiate the import without selecting a file

Then the system should display an error message indicating that a file is required

**Import File(WEB API)**

* **TEST CASE 1**

Successful File Import

* **GHERKIN**

Feature: File Import in Metabase

Background:

Given a user is authorized with appropriate permissions

Scenario: Successful File Import

When the user sends a POST request to the file import API

And includes a valid file for import

And configures import settings if required

And initiates the import process

Then the API should return a 201 (Created) status code

Scenario: Unsuccessful File Import (Invalid File)

When the user sends a POST request to the file import API

And includes an invalid file for import

And configures import settings if required

And initiates the import process

Then the API should return a 400 (Bad Request) status code

And the response should contain an error message indicating an invalid file

* **TEST CASE 2**

Import File with No Overwrite (Positive)

* **GHERKIN**

Scenario Outline: Import File with No Overwrite

When the user sends a POST request to the file import API

And includes a valid file for import

And chooses the option to not overwrite existing data: "<OverwriteOption>"

And initiates the import process

Then the API should return a "<ExpectedStatusCode>" status code

Examples:

| OverwriteOption | ExpectedStatusCode |

| Do Not Overwrite | 201 |

| Overwrite | 201 |

| Skip Overwrite | 201 |

| Preserve Data | 201 |

| Append Data | 201                |

Risk Analysis

|  |
| --- |
| **Login** |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Test Case** | **Risk** | **Description** | **Probability** | **Severity** | **Action to Minimize Risk** | | **Test Case 1: Verify Successful Login with Valid Credentials** | **Low** | **This test case checks if an admin can successfully create a new user with the correct permissions.** | **Low** | **Medium** | **Implement validation checks to ensure that only authorized admins can access the "Create New User" feature.**  **Regularly review and audit user permissions to prevent unauthorized access to sensitive features.** | | **Test Case 4: Verify Retrieval of User Permission** | **low** | **This test case validates whether the system correctly retrieves a user's permissions via a GET request.** | **low** | **low** | **implement proper authentication and authorization checks to ensure that only the respective user can request their permissions.**  **Regularly monitor and review permission settings to avoid unauthorized access** | |

|  |
| --- |
| **Import file** |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Test Case** | **Risk** | **Description** | **Probability** | **Severity** | **Action to Minimize Risk** | | **Test Case 2:Verify Error Message for Invalid File Format (Negative)** | **low** | **This test case checks if the system correctly handles and reports unsupported file formats during the import process.** | **medium** | **low** | **implement thorough format validation to prevent unsupported file types from being processed.**  **Provide clear and user-friendly error messages to help users understand the issue.** | | **Test Case 3: Import File with No Overwrite (Positive)** | **low** | **This test case verifies that the system successfully imports a file without overwriting existing data.** | **low** | **low** | **Implement data import processes that are designed to prevent data loss or overwriting when needed.**  **Clearly document the behavior of the import process, including the options available to users.** |  | |

|  |
| --- |
| **Access Control** |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Test Case** | **Risk** | **Description** | **Probability** | **Severity** | **Action to Minimize Risk** | | **Create New User with Valid Permissions** | **Low** | **This test case checks if an admin can successfully create a new user with the correct permissions.** | **low** | **medium** | **Implement validation checks to ensure that only authorized admins can access the "Create New User" feature.**  **Regularly review and audit user permissions to prevent unauthorized access to sensitive features.** | | **Verify Retrieval of User Permissions** | **low** | **This test case validates whether the system correctly retrieves a user's permissions via a GET request.** | **low** | **low** | **Implement proper authentication and authorization checks to ensure that only the respective user can request their permissions.**  **Regularly monitor and review permission settings to avoid unauthorized access.** | |

### **Defect Density:**

**login**

Critical Bugs (Severity: High):

None of the provided test cases seem to have critical bugs that would prevent essential authentication functionality.

Medium Bugs (Severity: Medium):

Test Case 5 (Verify Error Response for Invalid Username) could be categorized as having medium-severity bugs. It verifies the handling of invalid usernames, and any issues related to this verification could impact the user experience.

Low Bugs (Severity: Low):

The other test cases (Test Case 1, 2, 3, 4) don't appear to have low-severity bugs based on the provided information.

**Access Control:**

Number of Critical Bugs:

Test Case 3 (Attempt to Create User Without Admin Privileges):

Number of Medium Bugs:

Test Case 1 (Verify Access to Access Control GUI):

Test Case 4 (Verify Retrieval of User Permissions):

Number of Low Bugs:

Test Case 2 (Create New User with Valid Permissions):

Test Case 1 (Verify Access to Access Control Web API):

**Import file:**

Critical Bugs (Priority 1):

Test Case 4, Scenario 2 (Unsuccessful File Import - Invalid File):

Medium Bugs (Priority 2):

Test Case 3, Scenario (Error Message for Missing File):

Low Bugs (Priority 3):

Test Case 5, Scenario Outline (Import File with No Overwrite):

# **Defect Density:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Modules** | **Critical Bugs** | **Medium Bugs** | **Low Bugs** |
| Login | 0 | 1 | 0 |
| Import File | 1 | 1 | 1 |
| Access Control | 1 | 2 | 2 |

defect density of login=(1/9)\*100=11.1%

defect density of ImportFile=(3/9)\*100=33%

defect density of AccessControl=(5/9)\*100=55.5%

**Test case Matrices :**

login

* Number of Positive Test Cases: 2

These are scenarios where the system is expected to work correctly, such as successful login with valid credentials and error responses for invalid credentials.

* Number of Negative Test Cases: 1

This is a scenario where the system is expected to handle errors, specifically verifying error responses for invalid usernames.

* Number of Repeated Test Cases: 0

There are no repeated test cases in the provided scenarios.

* Percentage of Approved Test Cases: 100%

All the test cases provided are written according to the given specifications, and there are no issues or errors identified in the test cases.

* Percentage of Test Cases Requiring Rework: 0%

None of the test cases require rework, as they cover different aspects of user authentication and handle both positive and negative scenarios effectively.

* Percentage of Deprecated Test Cases: 0%

None of the test cases are deprecated, as all are relevant to user authentication and cover different use cases.  
**acces control**

* Number of Positive Test Cases: 4

Test Case 1 (Access to Access Control GUI)

Test Case 2 (Create New User with Valid Permissions)

Test Case 4 (Access to Access Control Web API)

Test Case 5 (Retrieval of User Permissions)

* Number of Negative Test Cases: 2

Test Case 3 (Attempt to Create User Without Admin Privileges)

Test Case 4 (Unauthorized Access to Access Control Web API)

Number of Repeated Test Cases: 0

* Percentage of Approved Test Cases: 100%

All provided test cases appear to be well-structured and address various scenarios, including positive and negative cases. They seem to be designed to validate different aspects of the system.

* Percentage of Test Cases Requiring Rework: 0%

None of the provided test cases appear to require rework as they are logically structured and cover important aspects of access control and permissions.

* Percentage of Deprecated Test Cases: 0%

There are no deprecated test cases provided; all test cases seem relevant to the context.

**Import File**

* Number of Positive Test Cases: 3

Test Case 1: Verify Successful File Import (Positive)

Test Case 4: Successful File Import

Test Case 5: Import File with No Overwrite (Positive)

* Number of Negative Test Cases: 2

Test Case 2: Verify Error Message for Invalid File Format (Negative)

Test Case 3: Verify Error Message for Missing File (Negative)

* Number of Repeated Test Cases: 0

There are no repeated test cases in the provided set.

* Percentage of Approved Test Cases: 100%

All test cases are designed to ensure that the system handles different scenarios correctly. There are no identified issues with the test cases themselves.

* Percentage of Test Cases Requiring Rework: 0%

None of the provided test cases require rework as they are well-structured and cover different aspects of file import.

* Percentage of Deprecated Test Cases: 0%

There are no deprecated test cases; all the test cases provided are relevant for the described functionality.

**Stats**

