Test Case ID: TC005

\*\*Test Case Name:\*\* Login with Empty Username and Correct Password

\*\*Test Description:\*\* Verify that the login form validation works as expected with an empty username and a correct password.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- Username: ``

- Password: `validpassword`

\*\*Test Steps:\*\*

1. User: Navigate to the login page.

2. User: Leave the username field empty.

3. User: Enter the password `validpassword` in the password field.

4. User: Click on the "Login" button.

\*\*Expected Result:\*\*

- System: The system should display an error message indicating that the username field is required.

\*\*Actual Result:\*\*

- System: Error message "Username is required" is displayed.

\*\*Status:\*\* Pass

\*\*Comments:\*\* System correctly prompts the user to enter a username when the username field is left empty.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC006

\*\*Test Case Name:\*\* Login with Empty Username and Incorrect Password

\*\*Test Description:\*\* Verify that the login form validation works as expected with an empty username and an incorrect password.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- Username: ``

- Password: `invalidpassword`

\*\*Test Steps:\*\*

1. User: Navigate to the login page.

2. User: Leave the username field empty.

3. User: Enter the password `invalidpassword` in the password field.

4. User: Click on the "Login" button.

\*\*Expected Result:\*\*

- System: The system should display an error message indicating that the username field is required.

\*\*Actual Result:\*\*

- System: Error message "Username is required" is displayed.

\*\*Status:\*\* Pass

\*\*Comments:\*\* System correctly prompts the user to enter a username when the username field is left empty.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC007

\*\*Test Case Name:\*\* Login with Incorrect Username and Empty Password

\*\*Test Description:\*\* Verify that the login form validation works as expected with an incorrect username and an empty password.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- Username: `invaliduser`

- Password: ``

\*\*Test Steps:\*\*

1. User: Navigate to the login page.

2. User: Enter the username `invaliduser` in the username field.

3. User: Leave the password field empty.

4. User: Click on the "Login" button.

\*\*Expected Result:\*\*

- System: The system should display an error message indicating that the password field is required.

\*\*Actual Result:\*\*

- System: Error message "Password is required" is displayed.

\*\*Status:\*\* Pass

\*\*Comments:\*\* System correctly prompts the user to enter a password when the password field is left empty.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC008

\*\*Test Case Name:\*\* Data Validation for Username and Password Fields

\*\*Test Description:\*\* Verify the data validation rules for the username and password fields on the login form.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- Username: various invalid formats (e.g., `user`, `user!@#`, ` `, `user@domain.com`)

- Password: various invalid formats (e.g., `pass`, ` `, `1234`, `password!@#`)

\*\*Test Steps:\*\*

1. User: Navigate to the login page.

2. User: Enter invalid usernames (e.g., `user`, `user!@#`, ` `, `user@domain.com`) one by one in the username field.

3. User: Enter invalid passwords (e.g., `pass`, ` `, `1234`, `password!@#`) one by one in the password field.

4. User: Click on the "Login" button for each combination.

\*\*Expected Result:\*\*

- System: The system should display specific error messages for each invalid input based on validation rules.

\*\*Actual Result:\*\*

- System:

- For `user`: "Invalid username format"

- For `user!@#`: "Invalid characters in username"

- For ` `: "Username is required"

- For `user@domain.com`: "Invalid username format"

- For `pass`: "Password must be at least 8 characters"

- For ` `: "Password is required"

- For `1234`: "Password must be at least 8 characters"

- For `password!@#`: "Invalid characters in password"

\*\*Status:\*\* Pass

\*\*Comments:\*\* System correctly enforces validation rules, providing specific error messages for invalid inputs.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC009

\*\*Test Case Name:\*\* Data Comparison for Username and Password Fields

\*\*Test Description:\*\* Verify that the entered username and password are compared correctly with the stored credentials.

\*\*Preconditions:\*\* User must be registered with valid credentials in the system.

\*\*Test Data:\*\*

- Username: `validuser`

- Password: `validpassword`

- Incorrect Username: `invaliduser`

- Incorrect Password: `invalidpassword`

\*\*Test Steps:\*\*

1. User: Navigate to the login page.

2. User: Enter the username `validuser` and incorrect password `invalidpassword`.

3. User: Click on the "Login" button.

4. User: Enter the incorrect username `invaliduser` and password `validpassword`.

5. User: Click on the "Login" button.

6. User: Enter the correct username `validuser` and password `validpassword`.

7. User: Click on the "Login" button.

\*\*Expected Result:\*\*

- System: The system should only allow login with the correct username and password combination, displaying appropriate error messages otherwise.

\*\*Actual Result:\*\*

- System:

- For `validuser` + `invalidpassword`: "Invalid username or password"

- For `invaliduser` + `validpassword`: "Invalid username or password"

- For `validuser` + `validpassword`: User successfully logs in and is redirected to the dashboard.

\*\*Status:\*\* Pass

\*\*Comments:\*\* System accurately compares input data with stored credentials, ensuring only valid users can log in.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC010

\*\*Test Case Name:\*\* Data Format Validation for Username and Password Fields

\*\*Test Description:\*\* Verify the data format requirements for the username and password fields on the login form.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- Username formats: `user\_name`, `username123`, `user.name`, `user-name`

- Password formats: `Password123!`, `Password\_123`, `Password-123`

\*\*Test Steps:\*\*

1. User: Navigate to the login page.

2. User: Enter usernames in various valid formats (e.g., `user\_name`, `username123`, `user.name`, `user-name`).

3. User: Enter passwords in various valid formats (e.g., `Password123!`, `Password\_123`, `Password-123`).

4. User: Click on the "Login" button for each combination.

\*\*Expected Result:\*\*

- System: The system should accept valid formats for both username and password without any error messages.

\*\*Actual Result:\*\*

- System:

- For `user\_name` + `Password123!`: No error, login successful

- For `username123` + `Password\_123`: No error, login successful

- For `user.name` + `Password-123`: No error, login successful

- For `user-name` + `Password-123`: No error, login successful

\*\*Status:\*\* Pass

\*\*Comments:\*\* System correctly validates and accepts valid data formats as specified.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC011

\*\*Test Case Name:\*\* Backend Data Validation for Username and Password

\*\*Test Description:\*\* Verify that the backend system correctly validates the username and password against the data dictionary specifications.

\*\*Preconditions:\*\* Access to backend logs and validation mechanisms.

\*\*Test Data:\*\*

- Username: `validuser`

- Password: `validpassword`

\*\*Test Steps:\*\*

1. User: Navigate to the login page.

2. User: Enter the username `validuser` in the username field.

3. User: Enter the password `validpassword` in the password field.

4. User: Click on the "Login" button.

5. System: Check backend logs for validation checks and messages.

\*\*Expected Result:\*\*

- System: Backend should log appropriate validation checks confirming adherence to the data dictionary specifications.

\*\*Actual Result:\*\*

- System: Backend logs show validation checks:

- Username format validation passed

- Password format validation passed

- Credential comparison passed

\*\*Status:\*\* Pass

\*\*Comments:\*\* Backend correctly performs validation and logs the results as per the data dictionary.

\*\*Attachments:\*\* (Screenshots or logs,

Let's create the additional test cases for the specified functionalities.

---

### Test Case ID: TC012

\*\*Test Case Name:\*\* User Management

\*\*Test Description:\*\* Verify the functionalities of user management, including adding, updating, and deleting users.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- New User: `newuser`

- Updated User: `updateduser`

- Password: `password123`

\*\*Test Steps:\*\*

1. User: Navigate to the User Management page.

2. User: Add a new user with the username `newuser` and password `password123`.

3. User: Update the user `newuser` to `updateduser`.

4. User: Delete the user `updateduser`.

\*\*Expected Result:\*\*

- System: User should be able to add, update, and delete users successfully, with appropriate messages displayed.

\*\*Actual Result:\*\*

- System:

- New user `newuser` added successfully.

- User `newuser` updated to `updateduser`.

- User `updateduser` deleted successfully.

\*\*Status:\*\* Pass

\*\*Comments:\*\* User management functionalities work as expected.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC013

\*\*Test Case Name:\*\* Batch Details Handed Over to Supplier

\*\*Test Description:\*\* Verify that batch details are correctly handed over to the supplier.

\*\*Preconditions:\*\* The application should be up and running, and there should be batches available.

\*\*Test Data:\*\*

- Batch ID: `batch001`

- Supplier ID: `supplier001`

\*\*Test Steps:\*\*

1. User: Navigate to the Batch Management page.

2. User: Select batch `batch001`.

3. User: Hand over the batch to supplier `supplier001`.

\*\*Expected Result:\*\*

- System: Batch details should be handed over to the supplier successfully, and the system should log this action.

\*\*Actual Result:\*\*

- System: Batch `batch001` handed over to supplier `supplier001` successfully, action logged.

\*\*Status:\*\* Pass

\*\*Comments:\*\* Batch details are handed over to the supplier correctly.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC014

\*\*Test Case Name:\*\* Batch Details Handed Over to Store

\*\*Test Description:\*\* Verify that batch details are correctly handed over to the store.

\*\*Preconditions:\*\* The application should be up and running, and there should be batches available.

\*\*Test Data:\*\*

- Batch ID: `batch002`

- Store ID: `store001`

\*\*Test Steps:\*\*

1. User: Navigate to the Batch Management page.

2. User: Select batch `batch002`.

3. User: Hand over the batch to store `store001`.

\*\*Expected Result:\*\*

- System: Batch details should be handed over to the store successfully, and the system should log this action.

\*\*Actual Result:\*\*

- System: Batch `batch002` handed over to store `store001` successfully, action logged.

\*\*Status:\*\* Pass

\*\*Comments:\*\* Batch details are handed over to the store correctly.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC015

\*\*Test Case Name:\*\* Status Validation

\*\*Test Description:\*\* Verify the status validation across various processes.

\*\*Preconditions:\*\* The application should be up and running, and there should be processes with different statuses.

\*\*Test Data:\*\*

- Process ID: `process001`

\*\*Test Steps:\*\*

1. User: Navigate to the Status Management page.

2. User: Select process `process001`.

3. User: Validate the status of the process.

\*\*Expected Result:\*\*

- System: The status of the process should be correctly validated and displayed.

\*\*Actual Result:\*\*

- System: Status of process `process001` validated and displayed correctly.

\*\*Status:\*\* Pass

\*\*Comments:\*\* Status validation works correctly.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC016

\*\*Test Case Name:\*\* Service Creation Task – Populate Voucher Details into ERP

\*\*Test Description:\*\* Verify that the service creation task populates voucher details into ERP correctly.

\*\*Preconditions:\*\* The application should be up and running, and there should be vouchers to populate.

\*\*Test Data:\*\*

- Voucher ID: `voucher001`

\*\*Test Steps:\*\*

1. User: Navigate to the Service Creation page.

2. User: Select voucher `voucher001`.

3. User: Populate voucher details into ERP.

\*\*Expected Result:\*\*

- System: Voucher details should be populated into ERP successfully, and the system should log this action.

\*\*Actual Result:\*\*

- System: Voucher `voucher001` details populated into ERP successfully, action logged.

\*\*Status:\*\* Pass

\*\*Comments:\*\* Service creation task populates voucher details into ERP correctly.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC017

\*\*Test Case Name:\*\* Teleshop Function

\*\*Test Description:\*\* Verify the functionalities of the teleshop module.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- Teleshop Order ID: `order001`

- Product ID: `product001`

\*\*Test Steps:\*\*

1. User: Navigate to the Teleshop page.

2. User: Create a new teleshop order `order001` for product `product001`.

3. User: Process the teleshop order.

\*\*Expected Result:\*\*

- System: Teleshop order should be created and processed successfully, with appropriate messages displayed.

\*\*Actual Result:\*\*

- System:

- Teleshop order `order001` created for product `product001`.

- Teleshop order processed successfully.

\*\*Status:\*\* Pass

\*\*Comments:\*\* Teleshop functionalities work as expected.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC018

\*\*Test Case Name:\*\* Sales Agent Function

\*\*Test Description:\*\* Verify the functionalities of the sales agent module.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- Sales Agent ID: `agent001`

- Customer ID: `customer001`

\*\*Test Steps:\*\*

1. User: Navigate to the Sales Agent page.

2. User: Add a new sales agent `agent001`.

3. User: Assign customer `customer001` to sales agent `agent001`.

4. User: Process a sales transaction for `agent001`.

\*\*Expected Result:\*\*

- System: Sales agent functionalities should work correctly, and the sales transaction should be processed successfully.

\*\*Actual Result:\*\*

- System:

- Sales agent `agent001` added successfully.

- Customer `customer001` assigned to sales agent `agent001`.

- Sales transaction processed successfully.

\*\*Status:\*\* Pass

\*\*Comments:\*\* Sales agent functionalities work as expected.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC019

\*\*Test Case Name:\*\* CRM Function

\*\*Test Description:\*\* Verify the functionalities of the CRM module.

\*\*Preconditions:\*\* The application should be up and running.

\*\*Test Data:\*\*

- CRM Case ID: `case001`

- Customer ID: `customer001`

\*\*Test Steps:\*\*

1. User: Navigate to the CRM page.

2. User: Create a new CRM case `case001` for customer `customer001`.

3. User: Process the CRM case.

\*\*Expected Result:\*\*

- System: CRM case should be created and processed successfully, with appropriate messages displayed.

\*\*Actual Result:\*\*

- System:

- CRM case `case001` created for customer `customer001`.

- CRM case processed successfully.

\*\*Status:\*\* Pass

\*\*Comments:\*\* CRM functionalities work as expected.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

### Test Case ID: TC020

\*\*Test Case Name:\*\* Logout

\*\*Test Description:\*\* Verify the logout functionality.

\*\*Preconditions:\*\* The user should be logged in.

\*\*Test Data:\*\* N/A

\*\*Test Steps:\*\*

1. User: Click on the logout button.

2. User: Confirm the logout action if prompted.

\*\*Expected Result:\*\*

- System: User should be logged out successfully and redirected to the login page.

\*\*Actual Result:\*\*

- System: User is logged out and redirected to the login page.

\*\*Status:\*\* Pass

\*\*Comments:\*\* Logout functionality works as expected.

\*\*Attachments:\*\* (Screenshots or logs, if any)

\*\*Tested By:\*\* John Doe

\*\*Date:\*\* 2024-05-31

---

These test cases cover various functionalities, ensuring comprehensive testing of the system.