Test Case Development and Bug Reporting

1. **Create test cases for the search feature in an e-commerce application. Consider different scenarios such as searching for a valid product, searching for a non-existent product, and applying filters.**

    - Validate that the search field accepts alphabets numbers or symbols.
    - Validate that after entering search text and clicking on search icon, the search should work.
    - Validate that the search results should be as per the search query.
    - Validate that users should be able to search based on product name, brand name or product specifications.
    - Validate that filter should be present for filtering the search results based on brands, Price, reviews or ratings.
    - Validate that shorting options should be present on search result page.
    - Enter a product name that does not exist in the catalog (e.g., "Alien Coffee Machine") then the UI should suggest similar products (if applicable).
    - Navigate to the search bar then the system should return a "No results found" message.

2. **Write test cases for the user registration process in a mobile app.**
    - Validate all required fields are present on the registration form such as
      Full Name
      Email Address
      Password
      Confirm Password
      Phone Number
    - Validate successful registration with valid inputs
    - Validate error message for missing required fields
    - Validate error message for invalid email format
    - Validate password strength requirements
    - Validate password and confirm password mismatch
    - Validate phone number format and length
    - Validate that duplicate email addresses are not allowed
    - Validate that the confirmations messages should be sent to registered mail after submitting the form.

3. **Develop test cases for adding and removing items from a shopping cart. Consider scenarios like adding multiple items, removing a single item, and checking out with items in the cart.**
    - Validate that adding single items to add to cart after clicking on add to cart button.
    - Validate that adding multiple items to add to cart after clicking on add to cart button for multiple products.

- Validate that after adding items to cart the price should also gets updated as per the items.
- Validate that the single items can get removed after clicking on removing or delete button.
- Validate that all items should be removed after clicking on remove all or delete all button.
- Validate that price should be updated accordingly after removing items from the cart.
- Validate that updating the quantity of the products after adding to cart.
- Validate that price should also gets updates after increasing or decreasing the quantity of the product.
- Validate that adding items to the cart without logging in retains the cart after login.
- Validate that add to cart gets empty after checkout is done.
- Validate that checkout button is enabled only when some products is in cart.

4. Create test cases for updating a user's profile information (e.g., name, email, and phone number) in a web application

- Validate that the user must exist in that web application
- Validate that users must be logged In to update the profile.
- Validate that all the fields should be editable.
- Validate that email should be valid when entering.
- Validate that duplicate emails are not allowed.
- Validate that email confirmation mail should be sent to the updated email.
- Validate that phone number should be of 10 digits.
- Validate that OTP sent to the updated phone number.
- Validate that the "all fields are mandatory" message appears if any field left blank.
- Validate that users can able to update the profile pictures.
- Validate that after updating any changes "save changes or update" buttons should be enabled.
- Validate that after clicking on save changes all the changes should be updated.

5. Write test cases for the password reset feature of a web application.

- Validate that the users should be the part of that web applications
- Validate that the users are able to login after filling login section.
- Validate that if users forget password then forget password option available at the login section page.
- Validate that after clicking on forget password or reset password they ask for the registered email or phone numbers.
- Validate that after giving mail or phone number the one time otp will be sent to that mail or number.
- Validate that after writing OTP its check for valid otp and if its valid then it proceed to the change password section.

- Validate that after entering wrong otp the error message shows that the invalid otp.
- Validate that the otp should be for one time and its valid for limited time only.
- Validate that in case of giving unregistered email address or phone number, the error message appears that wrong email or phone.
- Validate that New password should verify the criteria that it should be of minimum length of 8 and including numbers and special characters.
- Validate that new password and confirm password should be same.
- Validate that if any mismatch in new password and confirm password then error message appears.
- Validate that after clicking on change password and if new and confirm password matches then it shows the message the password changed successfully on web as well on registered mail.
- Validate that the incorrect password appears when login with the old password.
- Validate that the login successfully appears after login with the registered mail and updated password.

6. Create test cases for a notification system that alerts users of important events (e.g., new messages, friend requests)

- Validate that send notification when a new message arrives.
- Validate that to verify notification with message preview.
- Validate that no notification sound when the device is in silent mode.
- Validate that user receives notifications for a new friend request.
- Validate that notifications is removed if the request is accepted/declined.
- Validate that notifications is removed after tapping on notification.
- Validate that as per the setting on the device the notification appears such as pop up.

7. You have been asked to perform exploratory testing on a new feature of a mobile app. Describe your approach, including how you would document any bugs you find during testing.

For exploratory testing a new mobile app feature, my approach would follow these steps:

- **Understand the Feature**: Start by reviewing any available documentation or requirements for the new feature.
- **Define Test Scenarios**: Based on the feature’s intended use, I’d outline high-level test scenarios. This might include typical user flows, edge cases, and negative testing (ensuring the app handles invalid inputs gracefully)
- **Set Up Test Environment**: Ensure the app is installed on various devices (if available) to cover different OS versions, screen sizes, and device capabilities.
- **Execute Testing**: I’d start testing, exploring the feature freely to spot unexpected behavior or usability issues.
- **Document Bugs**: For each bug, I’d log the following details:
  - **Title**: Brief description of the issue.
  - **Steps to Reproduce**: Detailed, step-by-step account of actions leading to the bug.

- **Expected vs. Actual Results**: Description of the intended behavior versus what happened.
- **Screenshots/Videos**: Screenshots or screen recordings to capture the issue visually.

Analyze and Retest: After documenting, I'd revisit the affected areas to see if similar issues appear elsewhere or if fixes impact other parts of the app. Retesting once fixes are applied would ensure the bugs have been resolved effectively.

8. You have received feedback on a bug report you submitted. The feedback states that it lacks sufficient detail. Rewrite the bug report to include more comprehensive information, considering the feedback.

Certainly! Here's an example of how I'd revise the bug report to make it more comprehensive.

Bug Report:

Title: App crashes when attempting to upload a profile picture on Android 13

Environment Details:

- **Device:** Samsung Galaxy S23
- **OS Version:** Android 13
- **App Version:** 2.3.1
- **Network Connection:** Wi-Fi (strong signal)
  Severity: High (prevents user from updating profile)
  Priority: Medium

Description:
When a user attempts to upload a profile picture in the settings, the app crashes. The issue seems to occur specifically when using photos taken with the device's camera. Other image sources (e.g., screenshots, downloaded images) do not appear to trigger the crash.

Steps to Reproduce:

1. Open the app and log in.
2. Go to **Settings** > **Profile** > **Edit Profile**.
3. Tap on **Upload Profile Picture**.
4. Select **Choose from Gallery**.
5. Select an image taken by the device's camera.
   Expected Result:
   The selected image should load, allowing the user to preview and save it as their profile picture.
   Actual Result:
   The app crashes immediately after selecting the photo.
   Frequency: 100% (happens every time under the specified conditions)

Attachments:

- **Crash Log:** [Attached crash log file]
- **Video Recording:** [Attached video showing the crash steps]
- **Screenshots:** [Attached screenshots of each step before the crash]
  Additional Notes:
- This issue only occurs with photos taken by the device's camera.

- I’ve tested the issue on a different device (Pixel 7, Android 13), where the crash did not occur. This may indicate a compatibility issue specific to the Samsung Galaxy S23.

9. You find a bug where the “Submit” button in a form is not working. Describe the steps you would take to verify and reproduce the bug. Include how you would document the results.

To verify and reproduce a bug with a non-functional "Submit" button, I would take the following steps:

Steps to Verify and Reproduce

1. **Open the Form**: Navigate to the form in the app or website where the "Submit" button issue was reported.
2. **Fill Out the Form**: Complete all required fields to ensure that the "Submit" button should be enabled and ready to be clicked.
3. **Attempt to Submit**: Tap or click the "Submit" button to observe any response. Note if nothing happens, if there’s any visible error message, or if the button appears disabled even with all fields filled.
4. **Test with Different Inputs**: Repeat the submission attempt using different types of inputs (e.g., varying lengths or characters in text fields) to see if specific inputs trigger the issue.
5. **Check Console Logs**: If possible, open the developer console (in a browser or using developer tools on a device) to look for any error messages that might appear when clicking "Submit." Note any relevant log details.
6. **Test on Different Browsers/Devices**: If it’s a web app, I would try to replicate the issue across multiple browsers (e.g., Chrome, Safari, Firefox) and devices. For a mobile app, I would try to replicate it on various device models and OS versions.
7. **Change Network Connection**: I would switch between Wi-Fi and mobile data or simulate poor network conditions to check if the issue is network-related.
8. **Clear Cache and Restart App**: Clear the cache, restart the app, and attempt to submit the form again to see if it’s a temporary or cached data issue.

Documenting the Results
I would document the bug report as follows:

Bug Report:
Title: “Submit” button on form is unresponsive

Environment Details:

- **Device:** iPhone 12, Google Pixel 6
- **OS Version:** iOS 16.1, Android 13
- **App Version:** 4.2.0
- **Browser (if web app):** Chrome v92, Safari 15.3
- **Network Connection:** Tested on both Wi-Fi and mobile data
  Severity: High (blocks user from submitting form)
  Priority: High
  Description:
  The "Submit" button on the user registration form is unresponsive, preventing users

from completing registration. Clicking or tapping the button has no effect, and there is no error message or feedback. The issue is consistent across multiple browsers and devices.

Steps to Reproduce:

1. Open the app and navigate to the **Registration Form**.
2. Fill in all required fields (e.g., name, email, password).
3. Tap on the **Submit** button.

   Expected Result:
   The form should submit successfully, and the user should proceed to the confirmation page.

   Actual Result:
   The "Submit" button is unresponsive, and the form does not submit.

10. **Review the following bug report and identify any missing information or areas for improvement:**
    Bug Report Example:
11. **Title: App crashes**
12. **Description: The app crashes when opened.**
13. **Steps to Reproduce: Open the app.**
14. **Expected Result: The app should open without crashing.**
15. **Actual Result: The app crashes.**

11. You have three bug reports:

Bug A: Application crashes on launch (Critical)
Bug B: Misspelled word in the footer (Low)
Bug C: Payment gateway timeout (High)

Prioritize these bugs and justify your reasoning for each prioritization.

Revised Bug Report Example

Title: App crashes on launch on iPhone 14 (iOS 16.1)

Environment Details:

- **Device**: iPhone 14
- **OS Version**: iOS 16.1
- **App Version**: 5.1.0
- **Network Connection**: Wi-Fi, stable connection

Severity: Critical (users cannot access any features)

Description:
The app crashes immediately upon launch, preventing users from accessing any functionality. The issue appears to be specific to iOS 16 devices, as testing on Android and earlier iOS versions did not replicate the crash.

Steps to Reproduce:

1. Ensure device is running iOS 16.1.
2. Open the app by tapping on its icon.

Expected Result:
The app should open to the home screen without crashing.

Actual Result:
The app crashes immediately upon launch, closing itself without displaying an error.

Frequency: 100% (reproduces every time the app is opened)

Attachments:

- **Crash Log**: [Attached crash log from device]
- **Video Recording**: [Attached video showing the crash]

Additional Notes:
The issue seems to occur specifically on iOS 16 devices; Android and earlier iOS versions are unaffected.

Part 2: Bug Prioritization and Justification

1. **Bug A: Application crashes on launch (Critical)**
   - **Priority: Highest**
   - **Justification**: This bug prevents users from accessing the app entirely, making it a critical, show-stopping issue. Since the crash occurs on launch, it affects all users, making it essential to fix as soon as possible to ensure basic app functionality.
2. **Bug C: Payment gateway timeout (High)**
   - **Priority: Medium**
   - **Justification**: This is a high-priority bug as it affects a core feature, payment processing, which directly impacts revenue and user satisfaction. Although it doesn't prevent app access, it disrupts the user journey at a crucial point, which can lead to frustration, abandonment, and loss of business.
3. **Bug B: Misspelled word in the footer (Low)**
   - **Priority: Lowest**
   - **Justification**: Although this bug is valid, it’s a cosmetic issue that doesn’t impact functionality or usability. It can be addressed later, perhaps as part of a routine update or cosmetic review, without affecting user experience significantly.