# **SD-TA-011 Quality Assurance & Software Testing** **Oleksii Nikitin Assignment**

# Task 1 - Understanding Quality Assurance Principles and Practices

### **A) Outline the importance of involving the principles of quality assurance throughout the various stages of a software development process.**

**Quality Assurance** (QA) is an essential component of the software development process, aimed at improving the quality and reliability of software products by identifying and correcting defects before they become more severe and costly to fix. Involving QA principles throughout the various stages of the Software Development Life Cycle (SDLC) is crucial for several reasons.

##### Preventative Measures

QA is fundamentally preventative. By integrating QA practices from the earliest stages of a project — such as during the requirements gathering and design phases — teams can ensure that quality is built into the software from the beginning. This proactive approach helps in identifying potential issues that might not be obvious until later stages, such as integration or deployment. By addressing these issues early, the development team can avoid the much higher costs associated with fixing problems discovered later in the lifecycle.

##### Continuous Improvement

QA is not a one-time activity but a continuous process that spans the entire lifecycle of software development. During the planning phase, QA involves evaluating the project scope and requirements for clarity and testability. In the design and implementation phases, QA focuses on the correctness of the design and the code quality, including adherence to coding standards and best practices. During testing, QA ensures that the software meets the specified requirements and functions as intended. In the deployment phase, QA involves final validations before the software goes live. Finally, during maintenance, QA continues to play a role in ensuring that updates and patches do not introduce new issues.

##### Risk Management

Effective QA helps manage risk by identifying critical areas of concern early in the development process. Doing so, not only ensures that the final product is of high quality but also helps in predicting potential failure points, which in turn aids in creating robust risk mitigation strategies. This risk management is crucial not only for the software’s functionality but also for its security and compliance with regulatory standards.

##### Stakeholder Confidence

Regular involvement in QA provides transparency and builds confidence among stakeholders, including clients, management, and end-users. It demonstrates a commitment to quality and reliability, which can be crucial for maintaining business reputation and customer trust.

##### Summary

By embedding QA throughout the software development process, organizations can ensure that their products are reliable, meet user expectations, and are delivered on time and within budget. This holistic approach to quality helps in creating software products that are not only functional and efficient but also robust and secure.

### **B) Discuss how organisations can benefit from adopting and implementing robust quality assurance and software testing processes effectively.**

Adopting and implementing robust quality assurance (QA) and software testing processes is highly beneficial for organizations for several key reasons:

##### Reduced Costs

One of the most significant benefits of effective QA is cost reduction. By catching defects early in the software development lifecycle, organizations can avoid the exponentially higher costs associated with fixing bugs found at later stages or after deployment. Studies and industry data often cite that the cost of fixing a bug after release can be 4 to 5 times as much as during the development phase, and up to 100 times more if found post-deployment. This cost includes not only the direct costs of fixing the bugs but also the indirect costs such as customer dissatisfaction, impact on brand reputation, and potential regulatory penalties.

##### Improved Product Quality and Customer Satisfaction

Robust QA processes ensure that the software products meet the required standards and function as expected, which leads to higher customer satisfaction. By consistently releasing reliable and high-quality products, an organization can enhance its market reputation, which is crucial for competitive differentiation and customer retention. Quality products are less likely to receive negative feedback, thereby reducing support costs and increasing customer loyalty.

##### Enhanced Operational Efficiency

Effective QA and testing streamline various development processes by integrating best practices and automation where appropriate. Automated testing, for example, can significantly reduce the time required for repetitive testing scenarios, allowing human testers to focus on more complex and value-added activities. This not only speeds up the development cycles but also improves the accuracy of testing by eliminating human error in repetitive tasks.

##### Compliance and Risk Management

In many industries, software products must meet stringent regulatory standards. Robust QA processes help ensure compliance with these standards by systematically checking for any deviations and documenting the processes for audit trails. This proactive approach to compliance significantly reduces the risk of legal or financial penalties that could arise from non-compliance. Furthermore, thorough testing identifies potential risks and vulnerabilities that could compromise the software’s performance, security, or user data integrity, allowing organizations to mitigate these risks before they cause harm.

##### Facilitates Continuous Improvement

By incorporating feedback from QA processes, organizations can continuously improve their software products. QA generates valuable insights into the performance and usability of software, guiding developers on where improvements are needed. This iterative process of testing and feedback is essential for agile development environments, where responsiveness to change is critical.

##### Summary

Robust QA and software testing are integral to developing high-quality software that meets user expectations and regulatory requirements. These processes lead to reduced development costs, increased customer satisfaction, enhanced operational efficiency, and effective risk management. Organizations that invest in strong QA practices not only safeguard their products but also position themselves as reliable and customer-focused in the marketplace.

### **C) Explain how both poor and robust quality assurance can impact the overall security of software.**

The level of quality assurance (QA) applied during the software development process can have a significant impact on the overall security of the software. The effects of both poor and robust QA can be profound, influencing the software's vulnerability to attacks and its ability to protect user data.

##### Impact of Poor QA on Software Security:

* **Increased Vulnerabilities:** Inadequate QA often leads to software that contains numerous security vulnerabilities. Without comprehensive testing, critical flaws such as SQL injection, cross-site scripting (XSS), and buffer overflows can go unnoticed. These vulnerabilities are easily exploitable by attackers, potentially leading to unauthorized access, data breaches, and other security incidents.
* **Compliance Issues:** Poor QA can result in software that fails to meet security regulations and standards, such as GDPR, HIPAA, or PCI DSS. Non-compliance can expose the organization to legal and financial penalties, not to mention reputational damage.
* **Higher Costs of Remediation:** Addressing security flaws after software deployment is typically much more expensive and complex than fixing issues during development. The costs include not only the technical fixes but also the expenses related to incident response, customer notification, and potential legal actions.
* **Loss of Trust:** Security breaches caused by poor QA can severely damage an organization's reputation, leading to loss of customer trust and loyalty, which can be difficult and costly to rebuild.

##### Impact of Robust QA on Software Security:

* **Early Detection of Security Flaws:** Robust QA processes incorporate security testing (such as penetration testing, security audits, and code reviews) early and throughout the software development lifecycle. This early detection allows teams to address vulnerabilities before the software is deployed, significantly reducing the risk of attacks.
* **Adherence to Best Security Practices:** Strong QA involves the enforcement of security best practices in coding and development. This includes the use of secure coding standards, regular updates to libraries and frameworks, and the application of security patches. By adhering to these practices, software is built with a stronger defence against common security threats.
* **Enhanced Risk Management:** Effective QA enables organizations to identify and assess potential security risks associated with software features and functionality. This risk management ensures that appropriate mitigation strategies are implemented, and that the software is resilient against both known and emerging threats.
* **Increased Customer Confidence:** By consistently releasing secure and reliable software, an organization can boost its reputation as a trustworthy provider. This reliability can translate into increased customer confidence and business growth.

The quality of QA processes directly influences the security of the software. While poor QA can lead to vulnerable, unreliable products that jeopardize user security and organizational reputation, robust QA supports the creation of secure, compliant, and reliable software that upholds the integrity of data and user trust. Investing in thorough QA is essential not only for detecting and resolving functional issues but also for ensuring that software is secure from potential threats.

# Task 2 - Preparing a Software Testing Plan

#### With reference to the provided scenario, prepare a test plan for the application currently in development. The test plan should clearly outline and justify the objectives, scope, and testing methodology being utilised.

### 1. Introduction:

**Overview of the software being tested**

The application in focus is a web-based platform tailored for a community of book enthusiasts. This application will allow users to register, log in, and post short, Twitter-like updates about the books they are reading. Users will be able to follow other users but will not engage in direct messaging.

**Purpose and objectives of testing**

The primary purpose of this testing phase will be to ensure that the application meets all functional and non-functional requirements as specified, operates with stability, handles errors gracefully, and provides a user-friendly experience.

**Test Scope**

Testing will cover all primary functions: user registration, login, posting updates, following/unfollowing users, and profile viewing. Additional test cases derived from the documentation will be included to ensure comprehensive coverage. Each test case will be executed three times using varied scenarios to enhance robustness and detect atypical issues.

### 2. Test Strategy:

**Description of the overall testing approach**

The testing will be executed manually and intended to meet the Customer's Requirements.

**Selection of appropriate testing techniques and methodologies**

The strategy will focus on manual testing techniques to provide detailed insight into the application's behaviour under various conditions.

**Identification of testing tools and resources**

The test environment will utilize Visual Studio Code as the IDE, the most current version of Python, and the PIP module at the time of testing. A virtual environment will also be configured to support testing.

### 3. Test Environment:

**Integrated Development Environment (IDE)**

Visual Studio Code will be used to navigate the application structure, manage the virtual environment, and handle dependencies.

**Python version for Testing**

Python v3.12.2 will be the primary version. While testing across multiple Python versions is recommended, it is outside the current scope but considered for future testing phases.

**Virtual Test Environment:**

Commands for setting up the environment include:

* Navigating to the application directory: **cd bookclub-webapp**
* Creating a virtual environment: **py -m venv .\.venv**
* Activating the virtual environment: **.\.venv\scripts\activate**
* Installing dependencies: **pip install -r requirements.txt**
* Launching the application: **flask run** (The application will be accessible at http://localhost:5000/)

### 4. Test Cases:

**Documentation and Structure**

Test cases will be meticulously documented using a structured template that includes the test case ID, title, description, steps, input data, expected outcomes, and actual results. This format will enhance clarity and facilitate easier understanding and replication of tests.

**Prioritization of Test Cases**

Test cases will be prioritized based on their criticality and potential impact on user experience and system stability, with essential functionalities such as registration and login tested first.

**Handling Test Failures**

In case of test failure, errors will be classified as Critical or Non-Critical, with a recommendation for prioritization in subsequent fixes to aid developers in addressing the most impactful issues first.

**Test Case Template**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| Example | Example | Example | Example | Example | Example | Example |

This structured testing approach will ensure that the application achieves its design and functional objectives, providing a reliable and engaging platform for its users.

# Task 3 - Implementing a Software Testing Plan

#### Based on the test plan produced in Task 2, implement your plan by conducting a robust testing process on the application currently in development. The tests must be appropriately evidenced and supported by using a test log which clearly defines the test data, inputs, expected outputs, actual outputs and test results.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | | | **Expected Results** | | **Actual Results** |
| FR-001-A | Register a new account (Case A). | The user with typical credentials registers a new account on the platform. | 1. Navigate to the registration page by pressing ‘New User? [Click To Register](http://127.0.0.1:5000/register)’ from the Login page.  2. Enter the registration details.  3. Submit the registration form by pressing ‘Register’. | 1. Username: Caoimhe Ni Chathasaigh  2. Email: caoimhe@example.com  3. Password: lyFa$gnh}}Z8@hVMvH8n | | | A new account is successfully created, and confirmation is received. | | [Passed]  As expected. |
| FR-001-B | Register a new account (Case B). | Register with edge-case credentials and a weak password | 1. Navigate to the registration page by pressing ‘New User? [Click To Register](http://127.0.0.1:5000/register)’ from the Login page.  2. Enter the registration details.  3. Submit the registration form by pressing ‘Register’. | 1. Username: Oisín N'Gallagher  2. Email: oisin@example.com  3. Password: 123 | | | The user should not be able to create an account due to a weak password and a warning message should appear. | | [Failed/Critical]  This is a critical error. The password requirements should be implemented in order to secure user accounts. An edge-case test showed that even a one-symbol password is allowed. |
| FR-001-C | Register a new account (Case C). | Register with existing username and email, using credentials from the FR-001-A test case. | 1. Navigate to the registration page by pressing ‘New User? [Click To Register](http://127.0.0.1:5000/register)’ from the Login page.  2. Enter the registration details.  3. Submit the registration form by pressing ‘Register’. | 1. Username: Caoimhe Ni Chathasaigh  2. Email: caoimhe@example.com  3. Password: V]\_=%IfOPz2CA4Sefc}[ | | | The error message is displayed about username and/or email already existing. | | [Passed]  As expected. |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | | **Input Data** | **Expected Results** | | **Actual Results** | |
| FR-002-A | Login to existing accounts (Case A) | The user logs in to an existing account using FR-001-B credentials. | 1. Navigate to the [login page](http://127.0.0.1:5000/login).  2. Enter login credentials.  3. Submit the login form by pressing ‘Sign In’ button. | | 1. Username: Oisín N'Gallagher  2. Password: 123 | The user is successfully logged in and redirected to the home (index) page. | | [Passed]  As expected. | |
| FR-002-B | Login to existing accounts (Case B) | Login with an incorrect password | 1. Navigate to the [login page](http://127.0.0.1:5000/login).  2. Enter login credentials.  3. Submit the login form by pressing the ‘Sign In’ button. | | 1. Username: Oisín N'Gallagher  2. Password: password123 | An error message is displayed: The username or password is incorrect. | | [Passed]  As expected. | |
| FR-002-C | Log in to existing accounts (Case C) | Login attempt with an unregistered username | 1. Navigate to the [login page](http://127.0.0.1:5000/login).  2. Enter login credentials.  3. Submit the login form by pressing the ‘Sign In’ button. | | 1. Username: Oisín N'Gallaghe  2. Password: 123 | An error message is displayed: The username or password is incorrect | | [Passed]  As expected. | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-003-A | Create a post (Case A) | The user creates a new post with max characters. | 1. Navigate to the post creation page (home page).  2. Enter 140 characters of text.  3. Submit the post by pressing the ‘Post’ button. | “Exploring the art of brevity: This post is designed to fit within 140 characters, showcasing the challenge of succinct expression.” | A new post with max characters is successfully created and visible on the user's profile. | [Passed]  As expected. |
| FR-003-B | Create a post (Case B) | The user creates a new post with over maximum characters. | 1. Navigate to the post creation page (home page).  2. Enter 150 characters of text.  3. Submit the post by pressing the ‘Post’ button. | “Quick insights in a flash: This ultra-short post is crafted to fit perfectly within 150 characters, demonstrating concise communication.” | A new post fails to appear on the user's profile after creation due to the 140-character limit. | [Failed/Medium]  The maximum allowed length should be 140. |
| FR-003-C | Create a post (Case A) | Create a post with special characters | 1. Navigate to the post creation page (home page).  2. Enter text with special characters.  3. Submit the post by pressing the ‘Post’ button. | “Amazing read! 📚💖 #inspired” | A new post with special characters is successfully created and visible on the user's profile. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-004-A | View existing posts (Case A) | The user views posts from the current user. | 1. Log in to the Oisín N'Gallagher account.  2. Navigate to the homepage to see the latest posts from the current user. | Target profile Oisín N'Gallagher | The user can see the posts made by himself starting from the recent ones. | [Passed]  As expected. |
| FR-004-B | View existing posts (Case B) | The user views posts from other users. | 1. Log in to the Oisín N'Gallagher account.  2. Navigate to Explorer page | Target the Explorer page | The user can see and interact with posts from other users. | [Passed]  As expected. |
| FR-004-C | View existing posts (Case C) | The user views the posts of a specific user. | 1. Log in to the Oisín N'Gallagher account.  2. Navigate to Explorer page  3. Pick a user from currently posted posts to navigate to their profile | Target profile Caoime Ni Chathasaigh | The user can see and interact with posts from the selected profile. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-005-A | Follow profiles (Case A) | The user follows another profile. | 1. Log in to the user account. 2. Navigate to the profile of another user. 3. Click the 'Follow' button. | Target profile Caoime Ni Chathasaigh | The user is now following the profile, and updates from this profile appear in the user's feed. | [Passed]  As expected. |
| FR-005-B | Follow a profile already followed (Case B) | Attempt to follow a profile that the user is already following. | 1. Log in to the user account. 2. Navigate to a profile that the user already follows. 3. Attempt to click the 'Follow' button (should be disabled or not visible). | Target profile Caoimhe Ni Chathasaigh | The system prevents the action and may display a message indicating the user already follows the profile. | [Failed/Non-Critical] The system didn’t prevent the action, instead, the following procedure was executed a second time, which didn’t cause any error, but still should be processed by the application. |
| FR-005-C | Follow a non-existent profile (Case C) | Attempt to follow a profile that does not exist. | 1. Log in to the user account. 2. Manually enter a URL /user/user123. 3. Attempt to follow. | Target non-existent profile username ‘user123’ | The system displays an error message indicating the profile does not exist. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-006-A | Unfollow a profile successfully (Case A) | A user successfully unfollows another user's profile. | 1. Log in to the user account. 2. Navigate to the profile of a user they are following. 3. Click the 'Unfollow' button. | Target profile Caoimhe Ni Chathasaigh | The user is no longer following the profile and will not see new updates in their feed. | [Passed]  As expected. |
| FR-006-B | Unfollow a profile not followed (Case B) | Attempt to unfollow a profile the user does not follow. | 1. Log in to the user account. 2. Navigate to a profile that the user is not following. 3. Attempt to click the 'Unfollow' button (should be disabled or not visible). | Target profile Caoimhe Ni Chathasaigh | The system does not display an 'Unfollow' button. | [Passed]  As expected. |
| FR-006-C | Unfollow a non-existent profile | Attempt to unfollow a profile that does not exist. | 1. Log in to the user account. 2. Manually enter a URL /unfollow/user123 and attempt to unfollow. | Target non-existent profile username ‘user123’ | The system displays an error message indicating the profile does not exist and the unfollow action won’t be executed | [Passed]  As expected. The Error message appeared and the user was redirected to the Home page. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-007-A | View profiles | The user views the details of a profile of another user. | Log in to the user account.  Navigate directly to another user's profile page via the search function or URL. | Target profile Caoimhe Ni Chathasaigh | Profile details are visible including posts, followers, and following count. | [Passed]  As expected. |
| FR-007-B | View a profile from a post | A user navigates to a profile by clicking on the username ‘Caoimhe Ni Chathasaigh’ from a post. | 1. Visit the Explore page to find posts from Caoimhe Ni Chathasaigh  2. Select and navigate to the [Caoimhe Ni Chathasaigh profile page](http://127.0.0.1:5000/user/Caoimhe%20Ni%20Chathasaigh) by clicking on the username | Target profile Caoimhe Ni Chathasaigh | The system navigates to the profile page of the user linked from the post. | [Passed]  As expected. |
| FR-007-C | Attempt to view a private or non-existent profile | A user tries to view a profile that is set to private or does not exist. | 1. Log in to the user account. 2. Search for a non-existent user ‘user123’ profile and attempt to view it. | Non-existent ‘user123’ profile | The system displays a message that the profile does not exist. | [Passed]  As expected. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | | **Input Data** | **Expected Results** | **Actual Results** |
| FR-008-A | Change username successfully | The user changes their username. | 1. Navigate to account settings.  2. Change the username.  3. Save changes. | | New Username set to ‘Oisín N'Gallagher1’ | The username is successfully updated. | [Passed]  As expected. |
| FR-008-B | Change username to an already taken username | A user attempts to change their username to one that is already in use by another account. | Log in to the user account.  Navigate to the Profile page.  Attempt to change the username to one known to be already taken. | | Set the Username to ‘Caoimhe Ni Chathasaigh’ | The system rejects the change and displays an error message indicating the username is already taken. | [Passed]  As expected. |
| FR-008-C | Change username to an invalid format | A user attempts to change their username to one that does not meet the format requirements (e.g., too long, contains multiple invalid characters). | 1. Log in to the user account. 2. Navigate to the account settings. 3. Attempt to change the username to an invalid format. | | Very long and invalid username format | The system prevents the username change and displays an error message about the invalid format. | [Failed/Critical] The system didn’t prevent the action, it renamed to an invalid format username, which could cause significant errors for other users or significantly slow down the system. The Critical Error should be fixed. |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | | **Expected Results** | **Actual Results** |
| FR-009-A | Customize the ‘about me’ section of your own profile | A user successfully updates the ‘About Me’ section of their profile. | 1. Log in to the user account. 2. Navigate to the profile edit page. 3. Update the ‘About Me’ section and save the changes. | New ‘About Me’ text | | The system saves the changes and displays the updated ‘About Me’ on the user profile. | [Passed]  As expected. |
| FR-009-B | Update the ‘About Me’ section with an excessive character limit | A user attempts to update the ‘About Me’ section with text exceeding the character limit. | 1. Log in to the user account. 2. Navigate to the profile edit page. 3. Enter text exceeding the character limit in the ‘About Me’ section and attempt to save. | Excessively long ‘About Me’ text | | The system does not save the text and shows an error message regarding the character limit. | [Passed]  As expected. |
| FR-009-C | Clear ‘About Me’ section | A user clears all text from the ‘About Me’ section of their profile. | 1. Log in to the user account. 2. Navigate to the profile edit page. 3. Clear the ‘About Me’ text and save changes. | Empty text field | | The system saves the changes and displays an empty ‘About Me’ section on the profile. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-010-A | See when a profile was last active | A user views the last active time displayed on current user's profile. | 1. Navigate to another user’s profile page. |  | The profile page shows the last active time correctly based on the user's last login or activity. | [Passed]  As expected. |
| FR-010-B | Last active time updates on activity | Verify that a user’s last active time updates when they perform an activity (e.g., posting). | 1. Log in to the user account. 2. Post a new message. 3. View the user's profile. | Any message content for the post | The profile's last active time updates to reflect the time of the last post. | [Failed/Critical]  The user’s Last activity time was updated on login, not on the post activity, which could cause privacy concernes. |
| FR-010-C | Verify display of approximate post time | Ensure that the system correctly displays the approximate time a post was made. | 1. Log in to the user account. 2. Navigate to the homepage or feed where posts are displayed. 3. Observe the time indication next to a post. |  | Each post should display an approximate time since posting, such as "a few seconds ago," "5 minutes ago," "2 hours ago," "yesterday," "2 days ago," etc., based on the current time and the post creation timestamp. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-011-A | See how many followers a profile has | The user checks how many followers a profile has. | Navigate to a profile page. |  | Number of followers is visible on the profile. | [Passed]  As expected. |
| FR-011-B | Update follower count after new follow | Verify that the follower count updates immediately after a new user follows the profile. | 1. Loggin with a secondary account and follow a primary user. 2. Refresh or revisit the primary user's profile from a different account to check the updated follower count. |  | The follower count on the primary user’s profile increments by one immediately after the follow. | [Passed]  As expected. |
| FR-011-C | Follower count accuracy with unfollow action | Ensure the follower count decreases accurately when a user unfollows the profile. | Unfollow a user from a secondary account.  Revisit the unfollowed user's profile to check the updated follower count. |  |  | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-012-A | See how many people a profile is following | The user checks how many people a profile is following. | Navigate to another user's profile. |  | The number of people the profile is following is visible. | [Passed]  As expected. |
| FR-012-B | Following count updates after new follow | Verify that the following count updates correctly when a user follows a new profile. | Log in to the user account.  Follow another user.  Check the user’s own profile to see if the following count has been updated. | Caoimhe Ni Chathasaigh of the user to follow | The user's profile page shows an incremented following count. | [Passed]  As expected. |
| FR-012-C | View own following count after unfollowing | A user unfollows another user and checks if their own following count decreases. | Log in to the user account.  Unfollow another user.  View own profile to check updated following count. | Caoimhe Ni Chathasaigh user profile | The following count on the user’s profile decreases by one. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-013 | Customize their own avatar using the Gravatar service | The user customizes their avatar using Gravatar. | 1. Navigate to the Gravatar website [https://gravatar.com/profile](https://gravatar.com/profile/)  2. Navigate to Gravatar’s avatars on the profile page <https://gravatar.com/profile/avatars>  3. Customize the user’s avatar by uploading a custom picture.  4. Register a user in our application with the same email as registered in the Gravatar service | 1. Source Avatar image:  A person standing in a desert with a large brain  Description automatically generated  2. Email associated with Gravatar ‘o.nitin23@ddletbtc.ie’  A screenshot of a login form  Description automatically generated | A new avatar that has been attached through the Gravatar service is successfully displayed on the profile page in the local application. | [Passed]  As expected, the Gravatar service successfully provided the uploaded image from the user’s Avatar page and the application successfully attached it to the user’s profile.  A screenshot of a computer  Description automatically generated  A screenshot of a social media post  Description automatically generated |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-014-A | View a feed of posts from profiles being followed | The user views a feed of posts from profiles they are following. | Navigate to the Home page. |  | The feed displays posts from all profiles the user is following in reverse chronological order. | [Passed]  As expected. |
| FR-014-B | Feed updates after following a new profile | Verify that the feed updates to include posts from a newly followed profile. | Log in to the user account.  Follow a new profile.  Navigate back to the Home page. | The newly followed profile is Oisín N'Gallagher | The feed now includes posts from the newly followed profile. | [Passed]  As expected. |
| FR-014-C | No posts in feed from unfollowed profiles | Ensure that the feed does not show posts from profiles after they have been unfollowed. | Log in to the user account.  Unfollow a profile whose posts were previously visible in the feed.  Navigate back to the main feed page. | The unfollowed profile is Oisín N'Gallagher | The feed no longer shows posts from the unfollowed profile. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| NFR-001 | User profiles on the site can only be changed by the owner of the profile | Verify that the profile owner can successfully modify their profile information. | Log in to the user account.  Navigate to profile settings.  Make changes to profile details and save. | New profile name admin1 | Changes are saved and reflected on the profile. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| NFR-002 | The site shall be efficient and easy to use | Evaluate the site's usability and efficiency. | 1. Navigate through the site's main features.  2. Assess load times and responsiveness.  3. Check for intuitive navigation and user interface.  4. Check the Console Panel on Dev Tools for errors |  | The site is user-friendly, and error-free with quick load times and intuitive navigation. | [Passed]  As expected. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| NFR-003 | Viewing posts on the site shall require registration to the site | The site requires users to register before they can view posts. | 1. Attempt to view posts without logging in.  2. Register a new account.  3. Attempt to view posts after logging in. |  | Posts are not visible without registration. After registration, posts are accessible. | [Passed]  As expected. |

## Additional extra test cases were taken from The Assessment Scenario Documentation but weren't mentioned in the test cases list.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-015 | Posts will be made up of any characters of a length from 1 to 140 | Ensure posts can be created with any length within the specified range. | 1. Attempt to create posts of various lengths within and outside the specified range. | Post content of various lengths | Post-creation is successful for lengths from 1 to 140 characters and fails for lengths beyond 140. | [Failed/Non-Critical]  The actual maximum allowed length is 160. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-016 | Each and every feed shall support pagination to the degree of ten posts being displayed per page | Verify the pagination supports displaying exactly ten posts per page. | 1. Populate the feed with more than ten posts. 2. Navigate through pages to verify the count per page. |  | Each page should display exactly ten posts until there are fewer than ten remaining. | [Failed/Non-Critical]  Each page shows only 9 posts. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | **Description** | **Test Steps** | **Input Data** | **Expected Results** | **Actual Results** |
| FR-017 | The user should not be able to click ‘newer posts’ or ‘older posts’ if there are none of the desired left to view | Ensure the navigation buttons for newer and older posts are disabled when appropriate. | 1. Navigate to the oldest and newest posts in the feed.  2. Check the state of the navigation buttons. |  | The ‘Older Posts’ buttons are disabled when there are no more posts to navigate to. | [Passed]  As expected. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Title** | | **Description** | | **Test Steps** | **Input Data** | | **Expected Results** | | **Actual Results** | | |
| FR-018 | Only users registered to the site should be able to view posts on the site. | | Confirm that viewing posts requires user registration. | | 1. Attempt to view posts without being logged in. |  | | Unregistered users cannot view posts, and a prompt to log in or register is displayed; registered users can view posts. | | [Passed]  As expected. | | |
| **Test Case ID** | **Test Title** | **Description** | | **Test Steps** | | | **Input Data** | | **Expected Results** | | **Actual Results** |
| FR-019-A | Users should not be able to **follow** themselves | Ensure users cannot follow their own profiles. | | 1. Login.  2. Enter a custom URL to execute a command of following themselves. | | | /follow/admin1 | | A related Error message should appear emphasising the impossibility of the action | | [Passed]  As expected. |
| FR-019-B | Users should not be able to **Unfollow** themselves | Ensure users cannot unfollow their own profiles. | | 1. Login.  2. Enter a custom URL to execute a command of following themselves. | | | /unfollow/admin1 | | A related Error message should appear emphasising the impossibility of the action | | [Passed]  As expected. |

# Task 4 - Analysing and Reviewing Testing Processes & Results

#### Produce a report containing a critical review of the testing process carried out in Task 3 including an analysis of the results produced by the application compared to the initial design specification. The report should also include a justification of the validity of the tests and identification of any potential issues.

## Report on Software Testing and Critical Review

This report provides a comprehensive analysis of the software testing carried out as per the test cases defined in the initial design specification. It critically reviews the testing process, assesses the validity of the tests, and identifies potential issues with the software application. Each functional requirement (FR) and non-functional requirement (NFR) was tested, and the results were documented to compare expected outcomes with actual results. This assessment aims to ensure that the application meets all defined requirements and adheres to quality standards.

### Analysis of Test Results

The testing covered a broad spectrum of functionalities including user registration, login processes, content creation, profile management, and system security measures. The following sections discuss the outcomes and implications of these tests:

##### User Registration and Login (FR-001 to FR-002)

* Not all related to user registration (FR-001-x) scenarios have passed, as a major password validation issue was revealed in the FR-001-B case, which proves the effectiveness of the tests.
* User login tests (FR-002) were successful across standard and erroneous credential entries, demonstrating robust authentication processes.

##### Content Creation and Management (FR-003 to FR-004)

* FR-003 highlighted a critical issue where the application allows posts to exceed the specified character limit of 140 characters, resulting in a failure. This suggests a lack of enforcement in input validation, which could lead to data inconsistency and usability issues.
* Viewing and interacting with posts (FR-004) functioned as expected, ensuring that the user experience in content access and interaction was intact.

##### Profile Interaction (FR-005 to FR-007)

* Tests for following and unfollowing profiles (FR-005 and FR-006) unveiled a non-critical failure where the system did not prevent multiple follows by the same user, potentially leading to duplicated entries and confusion.
* Viewing profile details (FR-007) operated correctly, which is crucial for maintaining an engaging user experience.

##### System Security and Profile Management (FR-008 to FR-010)

* Changing usernames (FR-008) revealed a critical error where the system accepted invalid username formats. This poses significant risks, including potential security vulnerabilities and system instability.
* The last active time update (FR-010) failed to correctly reflect user activity, which could mislead other users about the activity status and potentially affect user engagement.

##### Advanced Functionalities and Usability (FR-011 to FR-014, NFR-001 to NFR-003)

* Follower and following counts (FR-011 and FR-012) were accurate, enhancing the social connectivity features of the application.
* The pagination feature (FR-016) failed by displaying only nine posts per page instead of ten. This indicates a fault in pagination logic, affecting user experience and interface consistency.
* Non-functional requirements testing (NFR-001 to NFR-003) passed, affirming that the site is user-friendly and meets essential usability standards.

### Justification of Test Validity

The validity of the tests performed is crucial for ensuring that the application not only functions as intended but also meets all user expectations and compliance requirements. Here’s a detailed justification of the test validity based on the scope, methodology, and execution:

##### Scope and Coverage

The tests were designed to cover all primary functionalities of the application as outlined in the initial design specifications. This included user interactions such as registration, login, content management, profile interactions, and system security features. The broad coverage ensures that no aspect of the application is left unchecked, thereby reducing the risk of undiscovered faults.

##### Alignment with Requirements

Each test case was directly aligned with a specific functional or non-functional requirement. This alignment ensures that every feature is tested against its intended purpose and behaviour as described in the specification document. By mapping test cases to specific requirements, directly can be assessable whether each requirement is fulfilled.

##### Methodology

The methodology employed for testing involved both positive and negative test scenarios. This comprehensive approach is vital for not only validating the normal operation conditions but also for identifying how the application behaves under erroneous inputs or unexpected user actions. The inclusion of edge cases, such as using weak passwords or duplicate usernames, further validates the robustness of the system.

##### Repeatability and Consistency

The tests were designed to be repeatable, ensuring that the application behaves consistently over multiple test cycles and conditions. This repeatability is essential for detecting intermittent issues and for verifying that fixes for bugs remain effective over time.

##### Documentation and Traceability

Each test result was meticulously documented, with clear delineation between the expected and actual outcomes. This documentation provides traceability and aids in the rapid identification of discrepancies between the system’s operation and the specifications. It also facilitates ongoing maintenance and debugging processes.

### Identification of Potential Issues

Through the analysis of the test results, several potential issues have been identified that could impact the application’s performance, security, and user experience:

**Security Risks**

* **Enforce Strong Password Policies:** The failure of test FR-001-B where a weak password was accepted is critical and poses significant security risks. Implementing strict password requirements will prevent the creation of vulnerable user accounts and reduce the risk of unauthorized access.
* **Email Verification During Registration:** This functionality was not implemented which raises potential security concerns. Implementation of MFA registration is essential for any web application.
* **Input Validation Flaws:** The system’s acceptance of posts beyond the specified character limit and usernames in invalid formats exposes it to potential security threats such as buffer overflow attacks or injection flaws.
* **Authentication Gaps:** The critical failure in updating the 'last active time' based on user actions rather than just login could mislead other users and impact features dependent on activity timestamps.

**Usability and Functional Concerns**

* **Pagination Logic Error:** Incorrect pagination, which shows only nine posts instead of ten per page, could frustrate users and disrupt the browsing experience, reflecting poorly on the application's reliability and attention to detail.
* **Profile Following Mechanism:** Allowing users to follow a profile multiple times could lead to duplicated content and unnecessary data processing, which wastes resources and complicates user interactions.

**System Stability and Data Integrity**

* **Exceeding Character Limits:** Allowing entries that exceed defined character limits could cause database integrity issues, potentially leading to truncation errors or inconsistencies in data handling.
* **Handling of Non-Existent or Private Profiles:** The system effectively handles errors related to non-existent profiles, but the logic surrounding interactions with such profiles needs to be robust to avoid potential breaches or data leaks.

**Performance Issues**

* **Efficiency and Load Handling:** While not explicitly failed in the tests, the efficiency and load handling capabilities of the system under high traffic conditions remain untested, which could be crucial for scalability and performance during peak usage.

By addressing these potential issues, the application can be significantly improved to ensure a secure, efficient, and user-friendly experience. This detailed examination and subsequent resolution of identified issues will enhance the overall quality and reliability of the software.

### Recommendations Based on Test Results

Based on the detailed testing and analysis of the application, several recommendations are proposed to address the identified issues. These are prioritized to focus on critical and high-impact improvements that should be implemented urgently, along with general enhancements that could further refine the application.

##### High Priority Fixes (Implement ASAP)

1. **Enforce Input Validation Strictly**

* **Implementation of Password Strength Requirements:** Modify the registration logic to enforce strong password policies. This should include checks for minimum password length, complexity (inclusion of numbers, symbols, upper- and lower-case letters), and possibly checks against common password lists.
* **Character Limit Enforcement:** Implement strict validation on the character limits for posts and usernames. Ensure that inputs exceeding the set limits are rejected, and clear error messages are provided to the user.
* **Username Format Validation:** Improve the validation rules for username changes to reject any formats that do not comply with defined standards (e.g., length, allowed characters). This will prevent potential security vulnerabilities and data inconsistencies.

1. **Email Verification During Registration**

* **Account Verification:** Implement an email confirmation step as part of the registration process. Once users register, send an automated email with a verification link to confirm that the email address provided belongs to them. This step helps prevent the creation of fraudulent accounts and ensures that users can recover their accounts or receive notifications securely.

1. **Rectify Pagination Logic**

* **Correct Pagination Display:** Fix the logic that handles pagination to ensure that exactly ten posts are displayed per page unless fewer posts are available. This correction will improve the user experience and align functionality with user expectations.

1. **Secure Profile Following Mechanism**

* **Prevent Duplicate Follow Actions:** Modify the system to check if a user already follows another profile before allowing a new follow action to be executed. This prevents data duplication and enhances system efficiency.

1. **Update Activity Timestamp Logic**

* **Accurate Activity Tracking:** Adjust the system to update the 'last active time' based on actual user activities (such as posting and commenting) rather than just on login. This enhancement will provide more accurate information to users and can improve engagement metrics.

1. **Limitation of Login Attempts**

* **Brute Force Attack Mitigation:** Implement a limit on the number of unsuccessful login attempts. After a specified number of failed attempts (e.g., 5), the account should be temporarily locked, and a notification should be sent to the user’s registered email address. This measure helps prevent brute force attacks and secures user accounts against unauthorized access attempts.

1. **Password Recovery Functionality**

* **Secure Password Reset Mechanism:** Develop a robust password recovery process that includes multi-factor authentication (MFA) to verify the user's identity before allowing a password reset. This could involve sending a verification code to the user's registered email or phone number. Ensuring a secure password recovery process is essential to maintain the integrity and security of user accounts.
* **Security Guidelines:** Provide users with security guidelines and best practices for creating strong passwords and maintaining account security.

1. **Encryption of Sensitive Data**

* **Data Protection:** Ensure that all sensitive data, including passwords and personal user information, is encrypted using strong encryption algorithms both at rest and in transit. This is vital to protect user data from being compromised in the event of a data breach.

##### Medium Priority Enhancements

1. **Review and Enhance Error Handling**

* **Comprehensive Error Messages:** Enhance error handling across the system to provide more informative and user-friendly feedback, especially for operations like follow/unfollow actions and profile interactions.
* **Graceful Handling of Non-Existent Profiles:** Implement more robust checks and balances when attempting to interact with non-existent or private profiles, ensuring that the system gracefully handles such cases with appropriate feedback to the user.

1. **Optimize System Performance and Scalability**

* Load Testing: Conduct thorough load testing to assess the system's performance under high-traffic conditions. This will help identify potential bottlenecks and areas for optimization to ensure scalability and smooth operation during peak usage.

1. **Security Audits and Updates**

* **Proactive Security Assessments:** Schedule regular security audits and conduct penetration testing to identify and address potential vulnerabilities in the application. This proactive approach is crucial for maintaining a secure environment as it helps in detecting flaws that could be exploited by attackers.
* **Update and Patch Management:** Implement a rigorous update and patch management protocol to ensure all components of the system are up-to-date with the latest security patches and updates.

1. **Session Management**

* **Secure Session Handling:** Implement automatic session expiration and provide users with the ability to log out remotely from all devices. This helps prevent unauthorized access if a user's credentials are compromised or if they forget to log out from a public or shared device.

##### Long-Term Strategic Improvements

1. **Continuous User Experience (UX) Enhancement**

* **Iterative UX Reviews:** Regularly review and enhance the user interface and experience based on user feedback and usability testing. Focus on making the system more intuitive and user-friendly.
* **Accessibility Improvements:** Ensure the application is accessible to all users, including those with disabilities, by adhering to web accessibility standards.
* **User-Friendly Security Features:** Design security features such as captcha, MFA, and password policies to be as user-friendly as possible to avoid negatively impacting the user experience.

1. **Data Management and Integrity**

* **Data Cleanup Mechanisms:** Develop mechanisms for periodic data cleanup and integrity checks to prevent data corruption and maintain the quality of user-generated content.
* **Data Minimization:** Ensure that the application only collects the minimal amount of data necessary for its functionality. Regularly review data retention policies and implement strict access controls.
* **User Consent:** Incorporate clear consent mechanisms for data collection and sharing, especially in compliance with regulations such as GDPR.
* **Incident Response Plan:** Develop and regularly update an incident response plan to ensure rapid and effective action in the event of a security breach.

1. **API Security**

* **Secure API Endpoints:** If the application exposes APIs, ensure they are secured with appropriate authentication and authorization checks. Regularly audit API access and usage to detect potential abuse or vulnerabilities.
* **Rate Limiting:** Apply rate limiting to APIs to prevent abuse and mitigate denial-of-service attacks.

By addressing these recommendations in the order of priority, the application can significantly improve in terms of security, functionality, and user satisfaction. These improvements will not only resolve the current issues but also enhance the overall robustness and reliability of the software.