Test plans – GibJohn

Contents

[Understanding of Component Interrelation 1](#_Toc158203589)

[Functional Testing 1](#_Toc158203590)

[Non-Functional Testing 1](#_Toc158203591)

[Front-end Testing 1](#_Toc158203592)

[Incorporating Component Interrelation into the Test Approach 1](#_Toc158203593)

[Types of Testing. 2](#_Toc158203594)

[Security Testing 2](#_Toc158203595)

[Unit Testing 2](#_Toc158203596)

[White Box Testing 2](#_Toc158203597)

[Test approach 2](#_Toc158203598)

# Understanding of Component Interrelation

This is accessing how the solution components interrelate within the GibJohn Tutoring when testing. This can include:

## Functional Testing

Evaluation of the functional aspects can ensure that the component preforms its intended tasks correctly and efficiently. This will also make sure it will be up to standard to the business needs and requirements. This also verifies if data inputting, processing and outputs of the solutions are met with the business needs.

## Non-Functional Testing

Examination of non-functional attributes such as performance, and reliability. Assessing the applications will handle under loads and conditions to ensure optimal system performance.

## Front-end Testing

Validating the user interface and user experience linked with the tested component. This will make sure the design is met with the business requirements. This will confirm that the front-end functionalities align with design specification and provide a user-friendly experience.

# Incorporating Component Interrelation into the Test Approach

To insert the linked components into the test approach we would need to identify the types of tests needed to be incorporate into the test approach. For functional testing, applying white box testing can be the first way of linked components. Non-functional testing is equally essential for evaluating the linked components. This includes performance testing to assess the responsiveness and scalability of the interconnected elements, ensuring they collectively meet the system's non-functional requirements. Integration testing validates how the linked components collaborate and interact within the broader system architecture. It ensures that the flow of data and functionalities among these components is seamless, detecting any issues that may arise when components are integrated. It’s important to have this in early testing process. User interface testing remains integral, especially when assessing how linked components contribute to the front-end functionalities.

# Types of Testing.

Software testing is the process of verifying and evaluating the functions of the Gibjohn application. This type of testing will be used on the application to assure the quality of the software and make sure that the requirements are met correctly. This type of testing is used to identify and fix defects in the application. This be tests such as unit testing, integration testing, system testing, acceptance testing, etc. There will be three testing types that will be performed to assure the quality, security and functionality of the solution is met in the business needs.

## Security Testing

This type of testing it used to uncover vulnerabilities in the software being developed. This is used to prevent any malicious attacks that can happen. Security testing must identify all loopholes to ensure that the program cannot be attacked. There are multiple types of security testing such as:

* Vulnerability Scanning Test
* Security Scanning
* Penetration Scanning
* Risk Assessment
* Penetration Test

An attack we should use on the Gibjohn application is penetration testing and having a risk assessment. The first test that should be done will be penetration testing. This will simulate a malicious hacker externally trying to harm the system. Then after we find the vulnerabilities and develop mitigations, run a risk assessment to make sure the application has no vulnerabilities.

## Unit Testing

This process is where the individual would focus on a certain unit or component of the application. This is used to meet the requirements and needs of the program which allows to determine if it is suitable for user or not. This will be efficient and effective as there are multiple components used in the application which allows the user to easily log down and find out if the application can be used to improve the quality of the application.

## White Box Testing

This testing technique is where the software’s internal structure, design and functionality and security are tested to verify input and output flow to improve the design and to determine if the program is suitable for use. This allows the individual to see how the program works from the inside allowing them to see what happens inside the program when running the tests.

# Test approach

For software testing Verifying and evaluating the applications functions to determine if the solution meets the business requirements. The first type of testing that would be used is Security testing. This is used to mitigate security risks as soon as possible. Preforming a penetration test should be done first as simulating a real malicious attack can help identify any risks that can be brought on the application. This is to ensure that all the security risks can be mitigated. After the penetration testing has been completed, run a risk assessment to fortify the solution. This will mitigate any malicious attacks that may occur.

Unit testing contains an examination of individual units/components within the Gibjohn application. The primary objective is to assess the suitability of each unit for user needs and to ensure that they function efficiently and effectively. By isolating and examining specific components, unit testing verifies that each unit meets its requirements and contributes to the overall quality of the application. The testing team will execute detailed tests, validating input, processing, and output to guarantee the functionality and reliability of each unit.

White box testing requires a full evaluation of the internal algorithm, design, functionality, and security of the software. This testing technique verifies the input and output flow, aiming to improve the overall design and determine the suitability of the program for use. By understanding the internal workings of the application, white box testing enables the testing team to identify potential vulnerabilities, enhance security measures, and optimize the overall performance of the software. This approach ensures a thorough look of the applications inside code to ensure the quality of it.

# Test Template

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Dates they will be tested. | Description of Test | Component that will be tested | Type of Test | Data that will be used | Expected Outcome | Prerequisites and Dependencies |
| 01/01/2024 | Testing the pages with correct information | Register page | White box testing | First Name: John Name: John Doe Email Address: [JohnDoe@gmail.com](mailto:JohnDoe@gmail.com) Password: test!ng1 Confirm Password: test!ng1 | The program should validate each input, add the information to the database, and display a success message indicating the account creation. | The program needs validation for each input to ensure security. Error messages should be implemented for incorrect or improperly formatted inputs. |
| 01/01/2024 | Testing with incorrect email format | Register page | White box testing | First Name: 12356  Name: Not a Nam3 Email Address: Abcdefg Password: password  Confirm password: password | The program should display an error message indicating that the email format is incorrect. | The program must have email validation in place. |
| 01/01/2024 | Testing with mismatched passwords | Register page | White box testing | First Name: John Name: John Doe Email Address: [JohnDoe@gmail.com](mailto:JohnDoe@gmail.com) Password: test!ng1 Confirm Password: test!ng | The program should display an error message indicating that the passwords do not match. | The program must have password confirmation validation. |
| 01/01/2024 | Testing with empty fields | Register page | White box testing |  | The program should prompt the user to fill in all required fields and not proceed with registration. | The program must have validation for empty fields and appropriate error messages. |
| 02/01/2024 | Testing the login functionality with valid credentials | Login page | White box testing |  | The program should log in successfully and direct the user to the homepage. | A registered user account with the provided valid credentials must exist in the database. |
| 02/01/2024 | Testing the login functionality with invalid credentials | Login page | White box testing |  | The program should display an error message indicating invalid credentials. | The provided credentials should not match any existing user account in the database. |