



Document Change History

| **Version Number** | **Date** | **Contributor** | **Description** |
| --- | --- | --- | --- |
| V1.0 |  |  | What changes (additions and deletions) were made for this version? |
|  |  |  |  |
|  |  |  |  |

**\*\* Note to Document Author –** Red and blue text (with the exception of the title and document name above) in this document is directed at the template user to describe processes, build standards and help build the document from the template. All such red and blue text should be removed before submitting any formal documentation, including both draft and/or final, deliverables. **\*\*\*\***

**Table of Contents**

**1** **INTRODUCTION 3**

1.1 Scope 3

*1.1.1* *In Scope 3*

*1.1.2* *Out of Scope 3*

1.2 Quality Objective 3

*1.2.1* *Primary Objective 3*

*1.2.2* *Secondary Objective 4*

1.3 Roles and Responsibilities 4

*1.3.1* *Developer 4*

*1.3.2* *Adopter 4*

*1.3.3* *Testing Process Management Team 4*

1.4 Assumptions for Test Execution 5

1.5 Constraints for Test Execution 5

1.6 Definitions 6

**2** **TEST METHODOLOGY 6**

2.1 Purpose 6

*2.1.1* *Overview 6*

*2.1.2* *Usability Testing 6*

*2.1.3* *Unit Testing (Multiple) 7*

*2.1.4* *Iteration/Regression Testing 7*

*2.1.5* *Final release Testing 7*

*2.1.6* *Testing completeness Criteria 8*

2.2 Test Levels 8

*2.2.1* *Build Tests 8*

2.2.1.1 Level 1 - Build Acceptance Tests 8

2.2.1.2 Level 2 - Smoke Tests 8

2.2.1.3 Level 2a - Bug Regression Testing 8

*2.2.2* *Milestone Tests 9*

2.2.2.1 Level 3 - Critical Path Tests 9

*2.2.3* *Release Tests 9*

2.2.3.1 Level 4 - Standard Tests 9

2.2.3.2 Level 5 - Suggested Test 9

2.3 Bug Regression 9

2.4 Bug Triage 9

2.5 Suspension Criteria and Resumption Requirements 10

2.6 Test Completeness 10

*2.6.1* *Standard Conditions: 10*

*2.6.2* *Bug Reporting & Triage Conditions: 10*

**3** **TEST DELIVERABLES 11**

3.1 Deliverables Matrix 11

3.2 Documents 12

*3.2.1* *Test Approach Document 12*

*3.2.2* *Test Plan 12*

*3.2.3* *Test Schedule 13*

*3.2.4* *Test Specifications 13*

*3.2.5* *Requirements Traceability Matrix 13*

3.3 Defect Tracking & Debugging 13

*3.3.1* *Testing Workflow 13*

*3.3.2* *Defect reporting using G FORGE 14*

3.4 Reports 16

*3.4.1* *Testing status reports 16*

*3.4.2* *Phase Completion Reports 16*

*3.4.3* *Test Final Report - Sign-Off 16*

3.5 Responsibility Matrix 16

**4** **RESOURCE & ENVIRONMENT NEEDS 16**

4.1 Testing Tools 16

4.2 Test Environment 17

*4.2.1* *Hardware 17*

*4.2.2* *Software 17*

4.3 Bug Severity and Priority Definition 17

*4.3.1* *Severity List 17*

*4.3.2* *Priority List 18*

4.4 Bug Reporting 18

**5** **TERMS/ACRONYMS 18**

# **Introduction**

This test approach document describes the appropriate strategies, process, workflows and methodologies used to plan, organize, execute and manage testing of our Eye witness website project. In this documentation we will introduce the scope of the testing plan then we will explain the roles and the responsibilities of our website. Along with that we will go over the methodologies for the testing of our website. Then we will go over how we will and have corrected our code internally for any bugs and errors. Then we will talk little about this documentation and how we gathered other test based documentation from our testing. Finally in our last section we will go over all the tools and environment we used for conducting our tests.

## **Scope**

In this documentation we will be conducting a few different types of tests on our website. But it will be limited to certain features since the entirety of the project is yet to complete. But we will be testing among our friends and family at this point. So of all the functional features we will be testing on a few of the features at this point of time.

### In Scope

The Eye-Witness Website *Test Plan* defines the unit, integration, system, regression, and Client Acceptance testing approach. The test scope includes the following:

* We will be testing the following functional requirements.
  + - * Admin login
      * Admin will be able to edit witness
      * Witness login
      * Adding suspect photos in the admin
      * Filter suspect based on what witness asked
* We will be testing everything on the web-interface
* End-to-end testing and testing of interfaces of all systems that interact with the <system name>.

### Out of Scope

The following are considered out of scope for caBIG <workspace name> <system name> systemTest Plan and testing scope:

* We will not be testing the following functional requirements.
  + - * Admin assigning lineup to witness
      * Witness submitting the lineup back.
* Testing of Business SOPs, disaster recovery and Business Continuity Plan.

## **Quality Objective**

### Primary Objective

Our primary objective for this testing plan is to test all the functional requirements that we have been able to implement so far. But among all other metrics we want to make sure our security metrics are tested thoroughly since this application is being created for storing a lot of sensitive information from the government and it is really important that this application has the meret to save it. Besides that we also want to test the functionality of the application since this will be used to save all witnessed data too and if one code is wrong or there is a minor bug or error in the system then we cannot let anyone use it. Due to this much at stake our goal with this testing plan is to make sure all the functional requirements are working properly with no chances of errors whatsoever.

### Secondary Objective

Secondary objective of this testing plan is to test if t

## **Roles and Responsibilities**

Roles and responsibilities for the purpose of the testing phase are as follows and are subject to change if required or agreed upon by all the team members. This will allow each of our team members to work efficiently on what we are supposed to and that will eventually help us create a better version of our project.

### Developer

Team responsible for creating eyewitness applications for law enforcement agencies. Team includes Aman Patel, Will Craddock and Khyati Chaudhari and team is fully responsible for the following :

(a) Develop the system/application

(b) Develop Use cases and requirements in collaboration with Adopters

(c) Conduct Unit, system, regression and integration testing

(d) Support user acceptance testing

(e) Creating the documentation for System Requirements specification

(f) Creating the documentation for System Design Document

(g) Creating the documentation for test plan

### Adopter

Team is creating this application to help law enforcement and bystanders better help each other make the community safe by using our system to help witnesses of the crime to identify the criminal and help speed up the investigation. Adopters of our project will be law enforcement, bystanders and our friends and family. And following are the responsibilities of adopters.:

(a) Contribute to Use case, requirement development through review

(b) Contribute to develop and execution of the development test scripts through review

(c) Conduct Full User Acceptance, regression, and end-to-end testing; this includes identifying testing scenarios, building the test scripts, executing scripts and reporting test results

### Testing Process Management Team

At the beginning of this project we all thought that we all will be the part of testing plans, but due to the time constraint we all won't be able to participate in the testing plan but we decided to create a team for testing plan and other member/s will continue to work on the project:

(a) Monitor and manage testing integrity and Support testing activities

(b) Coordinate activities across testing data

Add more as appropriate to testing scope

## **Assumptions for Test Execution**

Below are some minimum assumptions (in black) that have been completed with some examples (in red).

* For User Acceptance testing, the Developer team has completed unit, system and integration testing and met all the Requirements (including quality requirements) based on Requirement Traceability Matrix.
* User Acceptance testing will be conducted by End-users
* Test results will be reported on a daily basis using Gforge. Failed scripts and defect list from Gforge with evidence will be sent to the Developer directly.
* Use cases have been developed by Adopters for User Acceptance testing. Use cases are approved by the test lead.
* Test scripts are developed and approved.
* Test Team will support and provide appropriate guidance to Adopters and Developers to conduct testing
* Major dependencies should be reported immediately after the testing kickoff meeting.

## **Constraints for Test Execution**

Below are some minimum assumptions (in black) followed by example constraints (red). Any example may be used if deemed appropriate for the particular project. New constraints may also be added that are reasoned to be suitable to the project.

* Adopters should clearly understand test procedures and record a defect or enhancement. Testing Process Management Team will schedule a teleconference with Developers and Adopters to train and address any testing related issues.
* Developers will receive a consolidated list of requests for test environment set up, user accounts set up, data set (actual and mock data), defect list, etc. through **GForge** after the initial Adopter testing kick off meeting.
* Developer will support ongoing testing activities based on priorities
* Test scripts must be approved by Test Lead prior test execution
* Test scripts, test environment and dependencies should be addressed during testing kickoff meeting in presence of a SME and request list should be submitted within 3 days of the kickoff meeting
* The Developer cannot execute the User Acceptance and End to End test scripts. After debugging, the developer can conduct their internal test, but no results from that test can be recorded / reported.
* Adopters are responsible to identify dependencies between test scripts and submit clear request to set up test environment

## **Definitions**

Bugs: Any error or defect that causes the software/application or hardware to malfunction. That is also included in the requirements and does not meet the required workflow, process or function point.

Enhancement:

1) Any alteration or modification to the existing system for better workflow and process.

2) An error or defect that causes the software/application or hardware to malfunction.

Where 1) and 2) are NOT included in the requirements can be categorized as an enhancement.

Enhancement can be added as a new requirement after an appropriate Change Management process.

# **Test Methodology**

## 2.1 Purpose

### **2.1.1** **Overview.**

The purpose of the Test Plan is to achieve the following:

* Provide a testing guideline that will cover all functional and nonfunctional requirements as specified by the client
* Identify testing risks and solutions to those risks.
* Provide guidelines for different severity of bugs and documentation of those bugs
* Provide test completion guidelines and suspension Criteria

### **2.1.2** **Usability Testing**

Usability testing will be done by each team member based on the part of the project assigned. We will also ensure here that everything is working to the specifications provided by the client.

Testing will be carried out by testers and non-testers. The non – testers will be provided by the client . After testing findings will be used to make changes to the program while still keeping in line with the functional and non-functional requirements provided by the client.

### **2.1.3** **Unit Testing (Multiple)**

Unit testing will cover the following areas:

* Login Criteria
* Registration Criteria
* File upload/download
* Database querying/upload
* UI

### **2.1.4** **Iteration/Regression Testing**

Iteration/Regression testing will follow the following guidelines

* Tests included

o Functionality

o Performance

* 2 cycles of testing
* Meeting will be held at the end of each iteration to address bugs and changes to the program
* Bug triages will also be held depending on the severity of bugs found
* All low-level bugs must be addressed before entering into the next phase
* All testing must have documentation(Including setup and installation instructions)
* All bugs must be documented through Gforge

### 2.1.5 Final release Testing

Final release testing will consist of the following:

* Testing will be done by the team members and by the testers provided by the client
* Feedback will be documented, and changes will be made based on bug severity
* The end goal is to have a usable product based on the requirements set forth by the client
* Unless High level bugs have not yet been resolved Final release testing will focus on medium to low level bugs

### 2.1.6 Testing completeness Criteria

Testing completeness Criteria will be based on the following:

* Program is stable enough to be used successfully
* Functional and Non-functional requirements have been successfully implemented and tested
* Bugs are non existent at all levels

## 2.2 Test Levels

### 2.2.1 Build Tests

#### 2.2.1.1 Level 1 - Build Acceptance Tests

Build Acceptance Tests should take less than an hour to complete. The purpose of Build acceptance testing is to make sure the project is working at the very basic levels. This testing will cover that the project can be installed and run successfully. We also want to determine if further testing is possible.If the test fails it remains untested and is sent back to be tested again.

#### 2.2.1.2 Level 2 - Smoke Tests

Smoke Tests should take less than two hours. This testing will cover the basic functionality of the project at higher levels. All basic functionalities will be quickly covered through thi testing. We also determine if further testing is possible. If the test fails it remains untested and is back to be tested again.

#### 2.2.1.3 Level 2a - Bug Regression Testing

Once level one and two testing is complete each area of the application will be tested. Each bug present will be given a different level of priority based on the severity of the bug. When level two testing is complete all bugs will be taken care of based on the severity of the bug. It should take less than an hour to regress each bug based on the severity of each bug

### **2.2.2** **Milestone Tests**

#### 2.2.2.1 Level 3 - Critical Path Tests

This testing will cover the basic functionality of the witness and police areas of the project. They will be tested one to two times per milestone and must pass at the end of each cycle. It will also be tested one to two times during the iteration and Final release cycles.

### **2.2.3** **Release Tests**

#### 2.2.3.1 Level 4 - Standard Tests

These tests will cover program installation and GUI testing. These tests will be run once per testing cycle. Installation testing will cover Django installation, virtual environment installation and any third party installs that will be required such as bootstrap.

#### 2.2.3.2 Level 5 - Suggested Test

Suggested test cases that we have thought about implementing but will not have time too include load testing.

## 2.3 Bug Regression

Bug regression will follow the following criteria:

* Bugs will be classified at levels low, medium and high
* When a bug passes regression, it will be labeled as fixed
* If a bug doesn’t pass regression, it will be labeled as not fixed and depending on the severity of the bug a meeting might be called
* Each member will be responsible for bugs in their area of the program included fixing and documentation

## 2.4 Bug Triage

Bug triages will be scheduled throughout the multiple phases of the project to discuss bugs and improvements to the project. Scheduling these triages will be the responsibility of the team lead but each team member will be responsible for reports on bugs and general improvements for their section of the project. The amount of triages held will depend on the severity of the problem or the quality of life of the improvement.

All members of the project will be a part of these triage meetings. The team lead will be responsible for determining the order and resolution in which these bugs are taken care of depending on the report from each individual. From there each individual will be responsible for fixing their own area as determined in these meetings. Each team member will also be responsible for documenting the resolution of each bug using the Gforge bug tracker system. The team lead will be responsible for following up on this documentation as well.

## 2.5 Suspension Criteria and Resumption Requirements

- Testing will be suspended on the project when the smoke test, critical path test or Standard Testing test case bugs are discovered after the second iteration.

A bug report will be filed by the team lead. After fixing the bug, the drop criteria will be followed, and the testing will continue.

## 2.6 Test Completeness

Testing will be considered complete when the following conditions have been met:

### 2.6.1 Standard Conditions:

* Functional and nonfunctional requirements have been implemented and tested
* All code is compiled successfully.
* Test Cases in each area of the project have passed
* High and Medium bugs have been resolved and closed.
* Each test area has been looked at by the team lead and double checked by the other team members

### 2.6.2 Bug Reporting & Triage Conditions:

Bug Reporting and Triage Conditions are as follows:

* Bug find rate has a decreasing trend or no new bugs are found over multiple test iterations
* Level of bug severity is mainly low or medium
* No High level bugs found after multiple tests.
* No bugs found after multiple tests over a pre-determined amount of time and after a decreasing trend

# **Test Deliverables**

**3.1** **Deliverables Matrix**

| **Deliverable** |
| --- |
| **Documents** |
| Test Approach |
| à Test Plan |
| à Test Schedule |
| à Test Specifications |
| **Test Case / Bug Write-Ups** |
| Test Cases / Results |
| Test Coverage Reports |
| GForge Bug tracker for bug reporting |
| **Reports** |
| Test results report |
| Test Final Report - Sign-Off |

After the document is complete each team member will look over it before signing off.

**3.2 Documentation**

### 3.2.1 Test Plan

The test plan will consist of the following:

* The test plan will consist of breaking the project down into multiple areas
  + Cop Functionality
  + Sign up
  + Login
  + Case Creation
  + Witness Creation
  + Photo upload
  + Witness Functionality
  + Login
  + Description input
  + Photo download and display
  + Witness query upload
  + Timer and Rating system
  + Database
  + Security
* Risks
  + Scheduling conflicts
  + Hardware errors
  + Software errors
  + Personal Health
* Risk Management
  + In regards to health issues and scheduling conflicts different forms of communication can be used for meetings
  + Multiple devices can be used in the event of a hardware or software issue on a particular device
  + Bug triages will be based on the severity of the bugs
  + Development drop acceptance will be based on 2.6 of the test plan

.

### 3.2.2 Test Specifications

Test specifications will consist of the following:

* Officer should be able to upload photos to a case and set descriptions of that photo
* Officer should be able register a witness
* Officer should be able to assign a witness to a specific case file
* Witness should be able to pull different photos from the database using predefined selections
* Witness should be able to then save those photos back into the database as a specific line up
* The time it takes for the witness to make their decision is recorded
* The confidence level of the witness is recorded through a scale
* Testing will be based on the criteria set in Part 2

### 3.2.3 Requirements Traceability Matrix

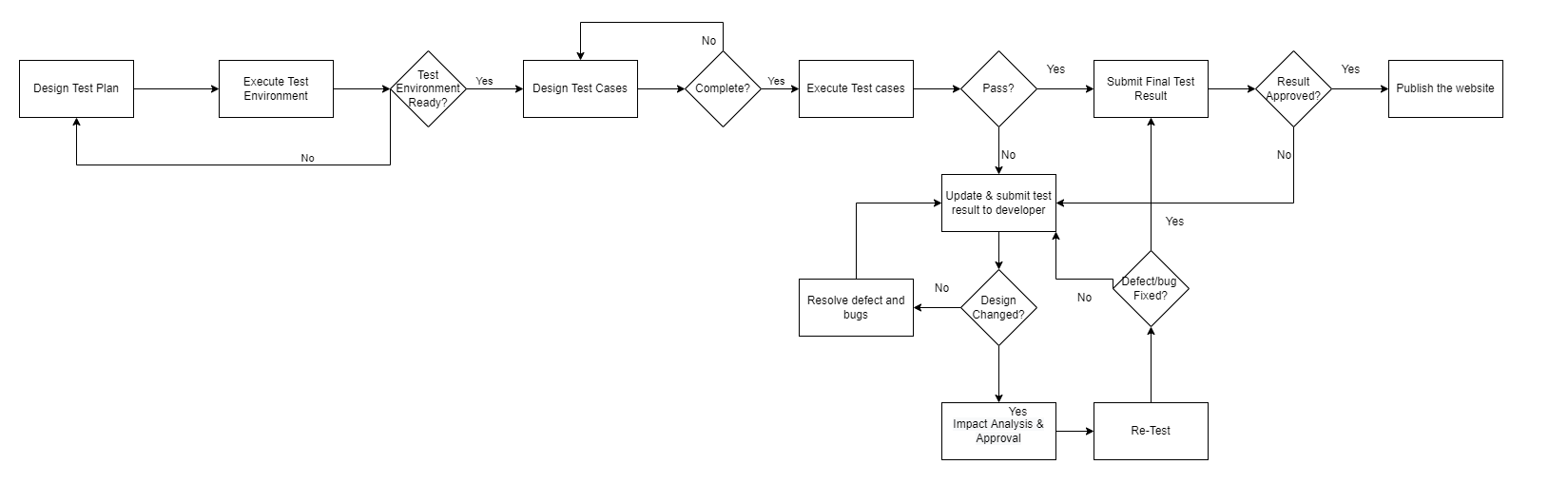
During this section we will talk about the testing of the use cases we introduced in the design document last month. We will discuss how we tested and if the test passed or failed. There are two test cases that were not tested at all. Reason behind that was we were not able to implement that part of the project yet. Two use cases tha we were not able to complete were those use cases were Deleting suspects and Deleting witnesses. We were not able to implement this because of the time constraint.

* Use Case ID: UC 001
* Requirement: Adding suspect in database
* Status: Pass
* Design Document: Design document version 1
* Technical Specification: Django python, sqlite3
* System Component: Database and front end html webpage
* Test case number: 001
* Tested in: Windows laptop
* Implemented: Aman Patel
* Verification: None
* Use Case ID: UC 002
* Requirement: Deleting suspects
* Status: Not tested
* Design Document: Design document version 1
* Technical Specification: Django python, sqlite3
* System Component: Database and front end html webpage
* Test case number: None
* Tested in: None
* Implemented: None
* Verification: None
* Use Case ID: UC 003
* Requirement: Case assign
* Status: Pass
* Design Document: Design document version 2
* Technical Specification: Django python, sqlite3
* System Component: Database and front end html webpage
* Test case number: 002
* Tested in: Windows
* Implemented: Aman Patel
* Verification: None
* Use Case ID: UC 004
* Requirement: Delete witness
* Status: Not tested
* Design Document: Design document version 1
* Technical Specification: Django python, sqlite3
* System Component: Database and front end html webpage
* Test case number: None
* Tested in: None
* Implemented: None
* Verification: None
* Use Case ID: UC 005
* Requirement: Assigning lineup to a witness
* Status: Pass
* Design Document: Design document version 1
* Technical Specification: Django python, sqlite3
* System Component: Database and front end html webpage
* Test case number: 003
* Tested in: Windows
* Implemented: Team
* Verification: None
* Use Case ID: UC 006
* Requirement: Identifying suspects
* Status: Fail
* Design Document: Design document version 1
* Technical Specification: Django python, sqlite3
* System Component: Database and front end html webpage
* Test case number: 004
* Tested in: Windows
* Implemented: Team
* Verification: None

## 3.3 Defect Tracking & Debugging

### 3.3.1 Testing Workflow

The below workflow illustrates the testing workflow process for Developers and Adopters for User Acceptance and End to End testing.



### 3.3.2 Defect reporting using G FORGE

All bugs and defects will be documented using G forge.

Bugs will be given a priority of High, medium and low.

* High level bugs must be addressed ASAP resolved within 2 days
* Medium level bugs must be addressed and resolved within 3 days
* Low level bugs must be addressed and resolved within 4 days

## 3.4 Reports

### 3.4.1 Testing status reports

Reports will follow the following criteria:

* Each member will file a report based on their area of the project
* Reports will be done weekly

### 3.4.2 Phase Completion Reports

The Phase Completion Reports will include the following:

* Number of Test cases
* Number of bugs found
* Number of bugs fixed/not fixed
* Number of passes and fails
* Risk Discussion
* Testing Progress

### 3.4.3 Test Final Report - Sign-Off

A Final Test Report will be signed off on by all three team members. This report will detail the complete testing progress and the current stability of the project and whether it is ready or not to be given to the client.

## 3.5 Responsibility Matrix

The matrix will be broken down into the following three sections. The following isn’t set in stone as each team member will most likely carry out testing on multiple parts to ensure efficiency.

| Witness | Will Craddock |
| --- | --- |
| Officer | Aman Patel |
| Database | Khyati Chaudhari |

# **Resource & Environment Needs**

## **Testing Tools**

Our project will be operated by a locally hosted Django server. Django provides a very good tool for testing called “test client”. The test client is a Python class that acts like a dummy web browser. Test client does not require the webserver to be running because it avoids the overhead of HTTP and deals directly with the Django framework, which accelerates the test time. To retrieve the pages, it only requires the path of the URL only, not the whole domain. The following test can be done with the test client:

* Simulate get and post requests on a URL and observe the response – everything from low-level HTTP to page content.
* Able to see the chain of redirects and URLs and the status code at each step.
* To establish that the correct template is being rendered and that the template is passed the correct context data.
* To use an in-browser framework to test rendered HTML and behavior of web pages functionality.

## **Test Environment**

### Hardware

Include the minimum hardware requirements that will be used to test the Application.

Testing will have access control to one or more application/database servers separate from any used by non-test members of the project team. following list of hardware should be consider a minimum:

* Windows 95 or older.
* Keyboard.
* Mouse.
* External Storage device(Hard Drive or flash drive)(optional)

### Software

In addition to the application and any other customer specified software, the following list of software should be considered a minimum:

* Django
* Python webframe
* Pip

## **Bug Severity and Priority Definition**

Bug Severity and Priority fields are both very important for categorizing bugs and prioritizing if and when the bugs will be fixed. The bug Severity and Priority levels will be defined as outlined in the following tables below. Testing will assign a severity level to all bugs. The Test Lead will be responsible to see that a correct severity level is assigned to each bug. the developer and the program manager are responsible for prioritizing the bug level, and fixing the bug.

### Severity List

The tester entering a bug into GForge is also responsible for entering the bug Severity.

| **Severity ID** | **Severity Level** | **Severity Description** |
| --- | --- | --- |
| 1 | High | The website has been completely shut off, and no further action can be taken |
| 2 | Medium | Some system component unusable due to failure or incorrect functionality such as systematic omissions of spaces, text going beyond the screen bounds, untranslated text, etc. |
| 3 | Low | Incorrect functionality of component or process such as buttons are slightly overlapping with each other. Although, they are still clickable separately |

### Priority List

| **Priority ID** | **Priority Level** | **Priority Description** |
| --- | --- | --- |
| 1 | High | The bug must be fixed immediately; the product cannot released until this issue is resolved. |
| 2 | Medium | These are important problems that should be fixed as soon as possible mainly during the normal course of development and testing. |
| 3 | Low | The problem should be fixed within the time available. There is definitely an issue, but it doesn’t have to be fixed to match the “exit” criteria. |

## **Bug Reporting**

Testing team recognizes that the bug reporting process is a critical communication tool within the testing process. Without effective communication of bug information and other issues, the development and release process will be negatively impacted.The Test Lead will be responsible for managing the bug reporting process. Testing’s standard bug reporting tools and processes will be used throughout the developing and implementing process, as well as after releasing the website.

# **Terms/Acronyms**

The below terms are used as examples, please add/remove any terms relevant to the document.

| **TERM/ACRONYM** | **DEFINITION** |
| --- | --- |
| API | Application Program Interface |
| BAH | Booz Allen Hamilton |
| BCP | Business Continuity Plan |
| CAT | Client Acceptance Testing |
| End-to End Testing | Tests user scenarios and various path conditions by verifying that the system runs and performs tasks accurately with the same set of data from beginning to end, as intended. |
| ER;ES | Electronic Records; Electronic Signatures |
| N/A | Not Applicable |
| NCI | National Cancer Institute |
| QA | Quality Assurance |
| RTM | Requirements Traceability Matrix |
| SME | Subject Matter Expert |
| SOP | Standard Operating Procedure |
| TBPT | Tissue Bank and Pathology Tools |
| TBD | To Be Determined |
| TSR | Test Summary Report |