

Testing Document

By Eilia Keyhanee

Contents

[Introduction 2](#_Toc518761564)

[Document Purpose 2](#_Toc518761565)

[Types of Testing 2](#_Toc518761566)

[Testing Schedule 3](#_Toc518761567)

[General Unit Testing 3](#_Toc518761568)

[Method 3](#_Toc518761569)

[Features and Functions 3](#_Toc518761570)

[Peer Testing 5](#_Toc518761571)

[Method 5](#_Toc518761572)

[Features: 5](#_Toc518761573)

[Destructive Testing 5](#_Toc518761574)

[Method 5](#_Toc518761575)

[Rules 5](#_Toc518761576)

[User Acceptance 6](#_Toc518761577)

[Method 6](#_Toc518761578)

[Tasks 6](#_Toc518761579)

[Evaluation 6](#_Toc518761580)

# Introduction

## Document Purpose

The following document aims to outline the testing plan and schedule for ‘NargesLogs’; as well as document the results, and changes made to fix the errors and bugs that may arise from these tests. This will be done to ensure that the solution that is sold to the client is polished and fully functional.

The scope of the document is to outline the types of testing that will be conducted, how they will be conducted, and by whom they will be conducted. The OneNote will have attached, the results of all testing, notes made by testers, notes made by the developer and the status of each error. (fixed or not yet fixed)

## Types of Testing

White Box Testing:

* General Unit Testing: Each feature of the program is tested to ensure it completes its task. This will determine that the program is overall, functional. All features will be assessed on a ‘pass/fail’ basis. This test will be conducted by the developer.
* Modular testing: Once an error is produced, some form of a driver will be used to test all the functions that are working together to provide the faulty feature. This test will identify all faulty units within a module.
* Unit Testing: After faulty units have been identified by modular testing, they will be isolated and tested individually so that the issue may be identified, this will most likely be done by a driver. The developer will then attempt to correct the issue.

Black Box Testing:

* System testing:
  + Peer Testing: Once the program is known to be generally functional, 3 testers will be selected to conduct peer testing. These testers will be tasked with attempting to use a list of features within the system. This will determine what aspects of the application are correct under normal circumstances. All errors and bugs that are encountered must be identified and screenshotted by the tester. The developer will then categorise the errors as either “critical”, “high priority”, or “low priority”. Each error will then be addressed in order, with some low priority errors possibly being unattended due to time restraints.
  + Destructive Testing: Once the program is known to be operational under normal circumstances, the program will be tested under extreme circumstances to ensure that the solution can withstand unusual user behaviours. Three testers with a rudimentary understanding of programming will be selected to do this. Using their knowledge of programming, the testers must attempt to crash the application or beak it in some way. Any steps taken that successfully produced a crash or logical error must be recorded and a screenshot of the error must be taken. The errors will then be categorised the same way as peer testing, and then addressed respective to their priority. Errors caused by destructive testing will generally be seen as less crucial.

User Acceptance: Once Testing is complete and all or most issues have been addressed, User Acceptance will be conducted in the workplace of the client, using real world data and will be conducted by Dr Zolfaghari (client), the only known personnel that will be using the application. The user will be asked to complete a series of tasks to ensure all features of the application are working as intended. Once the user is happy with the state of the application, they will be asked to sign a document indicating that they are satisfied with the solution.

# Testing Schedule

* General Unit Testing: will iron out obvious errors and bugs before starting more detailed tests.
* Peer testing: will find errors that were not caught in general unit testing.
  + Modular testing: will test the faulty modules that were found during peer testing and find the source(s) of the error.
  + Unit testing: will test the source of the error and determine were the fault is produced.
* Destructive testing: will find errors and limits that are produced under abnormal circumstances.
  + Modular testing: will test the faulty modules that were found during peer testing and find the source(s) of the error.
  + Unit testing: will test the source of the error and determine were the fault is produced.
* User Acceptance: will ensure that all features are working as expected by the user.

# General Unit Testing

## Method

The test will be conducted in a controlled environment, by Eilia Keyhanee (developer). Each Feature and Function will be assessed on a pass or fail basis, and additional notes will be added when the function fails. The results will be recorded in the OneNote, under the ‘Unit Testing’ section in the ‘Test’ section. When testing is complete, the developer will mark each failed function as fixed or not yet fixed and make notes on their efforts.

## Features and Functions

* Log in:
  + ‘Username’ field
  + ‘Password’ field
  + ‘Login’ button
  + ‘Settings’ button
* Settings:
  + ‘IP’ field
  + ‘Port’ field
  + ‘Cancel’ button
  + ‘Save’ button
* Home:
  + ‘View Patients’ button
  + ‘Add a Patient’ button
  + ‘Logout’ button
* Patients List:
  + Logo button
  + Search function
  + Sorting functions:
    - ID
    - Name
    - Sex
    - Age
    - Visits
    - Latest Condition
  + ‘Back’ button
  + ‘Remove’ button
  + ‘Add’ button
  + Open Patient function
* Patient Details:
  + Profile image
  + Drag and Drop profile image
  + Edit information
  + Save information
  + ‘Back’ button
  + ‘Remove’ button
  + ‘Add’ button
  + Open visit
* Visit Details:
  + ‘Back’ button
  + Drag and drop images
  + Click to add images
  + ‘Add’ button
  + ‘Remove’ button
  + Select and Save images
  + Edit information
  + Save information
  + Edit notes

# Peer Testing

## Method

Three Peer tests will be conducted in a controlled environment by three different testers. The testers will be given a list of features to test and are at liberty to attempt to use these features in any way. If any errors or bugs are encountered, they will be recorded under the section of the feature in which they were found. Screenshots of the crash must be taken; the developer will then address each error using modular and unit testing and make note of the changes made.

## Features:

* Login
* Logout
* Search for patients
* Sort patients
* Add patients
* Remove patients
* View patients
* Add profile images
* Add visits
* Remove visits
* View visits
* Edit notes
* Add visit images
* Remove visit images
* Save visit images

# Destructive Testing

## Method

The tests will take place in a controlled environment and will be conducted by three peers with an understanding of coding. These testers are then at liberty to attempt to cause crashes and errors within the program, being encouraged to use radical and unusual actions within the program to do so. The errors that these testers may cause are then recorded, with details of how the error occurred and a screenshot of the error. The developer will then address each error using modular and unit testing and make note of the changes made.

## Rules

* The tester may not directly access the files of the program in their effort to cause errors.

# User Acceptance

## Method

The testing will occur at Lithgow Valley Medical Practice and will be carried out by Dr Narges Zolfaghari (client), who is currently the only personnel that is expected to use the application. The user will attempt to complete a series of pre-determined tasks and will mark a checklist to determine whether the task was complete. The client has agreed to use real-world data in the form of real images from visits and real patient images. However, mock names and information will be used to protect patient privacy. Once all tasks have been attempted, the user will sign off a document saying that the software behaved as expected.

## Tasks

* Login using client login details
* Add a patient
* Fill out information
* Add a profile image
* Add three visits for the patient
* Add an appropriate number of images for each
* Create a new patient
* Add or do not add a profile image, information, visits etc… with liberty.
* Remove some visits from the first patient
* Access and edit other visits
* Change the profile image of the first patient
* Edit the information of the first patient
* Remove the first patient

# Evaluation

**[This section will be addressed when testing is complete]**