System Test Plan

For

Facial Analyzer for Rhinoplasty Operation

Team members:
Priscilla Carbo, Marvin Cazeau, Jared Curtis, Maree Kelly, Alessandra Oo

Version/Author	Date
Version 1	16 September 2021
Version 2	19 October 2021
Version 3	5 December 2021
Version 4	6 December 2021

Table of Contents

1. Introduction

1.1 Purpose

This document is a test plan for Facial Analyzer for Rhinoplasty Operations System Testing, produced by the development team. It describes the testing strategy and approach to testing the team will use to verify that the application meets the established requirements of the business prior to release.

1.2 Objectives

- Meets the requirements, specifications and the Business rules.
- Supports the intended business functions and achieves the required standards.
- Satisfies the Entrance Criteria for User Acceptance Testing.

2. Functional Scope

The Modules in the scope of testing for the Facial Analyzer for Rhinoplasty Operation System Testing are mentioned in the following documents:

1. The System Requirement Specification Documents: https://github.com/jcurtis664/FacialAnalyzer/blob/main/Deliverables/System%20Requirements%20Specification%20Template.pdf

2. User Manuals: https://github.com/jcurtis664/FacialAnalyzer/tree/main/Deliverables

3. Section 3.1: Testing Strategy

3. Overall Strategy and Approach

3.1 Testing Strategy

Facial Analyzer for Rhinoplasty Operation System Testing will include testing of all functionalities that are in scope (Section 2). System testing activities will include the testing of new functionalities, modified functionalities, screen level validations, work flows, functionality access, testing of internal & external interfaces.

This section will describe how the types will be tested while section 3.3 will describe what will be tested.

3.1.1 Function Testing

Test Objective: Confirm the functionality of the machine learning algorithms, and output of the digitized rhinoplasty programs.

Technique: Execute multiple use cases to confirm that the provided output matches the expected output.

Completion Criteria: All use cases for the KNN algorithm and the CNN have been tested and provide the expected result.

Special Consideration: Access to the digitized rhinoplasty system and related system requirements document.

3.1.2 Database Testing

Test Objective: Confirm the functionality and usability of the database of face scans for CNN algorithm and the database of face ratios for KNN algorithm.

Technique: Upload multiple face scans and JSON files to the database and test these file's usability in the machine learning algorithms.

Completion Criteria: The database functions properly when utilized by all machine learning algorithms and face groupings are generated from the results of the KNN algorithm.

Special Consideration: Access to the database of faces, access to the website 'digitized-rhinoplasty.com', and access to Bellus3D iOS application.

3.1.3 Performance Testing

Test Objective: CNN algorithm can read faces in the database and build an Image Data Generator model that suggests a group of similar faces.

Technique: Execute the CNN algorithm using a test face to confirm that the model suggests a relevant face group.

Completion Criteria: CNN algorithm should output a group of similar looking faces to the user.

Special Consideration: Access to the database of faces and access to the database of face ratios.

3.2 System Testing Entrance Criteria

In order to start system testing, certain requirements must be met for testing readiness. The readiness can be classified by: access to file structure from GitHub (https://github.com/jcurtis664/FacialAnalyzer), access to the internet, access to digitized rhinoplasty web application, access to Apple device with App Store, and access to a computer with Windows 10 or Mac OSX, Blender 2.9, and Python 3.0 or later.

3.3 Testing Types

3.3.1 Usability Testing

User interface attributes, cosmetic presentation and content will be tested for accuracy and general usability. The goal of Usability Testing is to ensure that the User Interface is comfortable to use and provides the user with consistent and appropriate access and navigation through the functions of the application (e.g., access keys, consistent tab order, readable fonts etc.)

The user interface for Facial Analyzer for Rhinoplasty is the Blender software, the command prompt for Windows, terminal for Macs and command line for Linux.

Test objective: The user will run programs and view output in the command prompt/terminal/command line window.

Technique: The user will utilize Blender to load a 3D face with their chosen landmarks and manipulate the 3D model of the patient's nose.

Completion Criteria: The model is successfully uploaded to the Blender software without any errors.

Special consideration: Access to a computer and the database of faces and ratios obtained.

3.3.2 Functional Testing

The objective of this test is to ensure that each element of the component meets the functional requirements of the business as outlined in the:

- Business / Functional Requirements
- Business rules or conditions
- Other functional documents produced during the course of the project i.e. resolution to issues/change requests/feedback

System Requirements Specification, 3.1.1: "The system can be installed on any computer of medical, education or recreational settings. "

System Requirements Specification, 3.1.2: "The user shall have version 3.6 of Python installed"

System Requirements Specification, 3.1.3: "The user shall have version 3.0.9 of openpyxl installed in their Python site-packages"

System Requirements Specification, 3.1.4: "The user shall have at least 2.9 version of Blender installed"

System Requirements Specification, 3.1.5: "The user shall have version 3.0.9 of openpyxl installed in their Blender modules"

3.4 Suspension Criteria and Resumption Requirements

This section will specify the criteria that will be used to suspend all or a portion of the testing activities on the items associated with this test plan.

3.4.1 Suspension Criteria

If the user uploads the wrong format of files, the program will suspend.

If the new points that are generated do not align with the rules, the algorithm does not work therefore the test is suspended. The error could be caused by incorrect data conversion from JSON to XLSX or incorrect arithmetic.

For unit testing, the test values in the array should match the values in the CSV files. If it does not match, the program will terminate.

3.4.2 Resumption Requirements

The program will continue once the user has uploaded the correct files, verify that the points inputted aligned with the rules and the test values in the array matches the value in the CSV files, the program will be able to re-execute.

4. Execution Plan

4.1 Execution Plan

The execution plan will detail the test cases to be executed. The Execution plan will be put together to ensure that all the requirements are covered. The execution plan will be designed to accommodate some changes if necessary, if testing is incomplete on any day. All the test cases of the projects under test in this release are arranged in a logical order depending upon their inter dependency.

The test plan for the system is as follows:

4.1.1 Database Testing (See 3.1.2)

4.1.2 Function Testing (See 3.1.1)

4.1.3 Performance Testing (See 3.1.3)

Requirement (From SRS)	Test Case Identifier	Input	Expected Behavior	Pass/Fail
3.3.1 The system shall render a 3D image of the patient's scanned face.	1.0	Open patient's scan info in Blender	Blender generates an editable 3D image of the chosen patient's face	Pass
3.3.2 The user	1.1	Scan the user's	Outputs an obj	Pass

shall generate a 3D scan of their face using the Bellus3D iOS application.		face on an Apple device	file, jpg file, and mtl file specific to the patient	
3.3.3 The surgeon shall mark points on the patient's face upon upload using the rhinoplasty web application.	1.2	Open patient's scan info on digitizedrhinoplasty.com and manually choose the 50 significant landmarks	Digitized rhinoplasty web application allows the output of a JSON file containing the landmarks	Pass
3.3.4 The system shall use the KNN algorithm to group the most similar faces.	1.3	Run the 'sorter.py' file.	Generates a group of faces with similar ratios for every patient in database	Pass
3.3.5 The system shall display k options for the patient to choose from.	1.4	Run the 'match_faces_cnn.py' file with jpg file of the patient of choice	Return a group of faces that look similar to the patient's	Pass
3.3.6 The system shall generate more suggestions based on the patient's request.	1.5	Edit KNN algorithm to make groups of k patients, re-run 'sorter.py' file	Generate new face groups with k or less patient's	Pass
3.3.7 The system shall send the patient's choice to Blender.	1.6	Selected Patient scan info	Selected scan info sent to Blender for 3D image generation	Pass

5. Traceability Matrix & Defect Tracking

5.1 Traceability Matrix

Requirement	Requirement Dependencies	Test Case	Test Case Dependencies	Severity	Responsibility	Result
3.3.1 The	3.3.5	1.1	1.5	Medium	J. C.	Success

system shall render a 3D image of the patient's scanned face.	3.3.7		1.7			
3.3.2 The user shall generate a 3D scan of their face using the Bellus3D iOS application.	3.3.1 3.3.3	1.2	1.1 1.3	Critical	P.C. M.C. J.C. M.K. A.O.	Success
3.3.3 The surgeon shall mark points on the patient's face upon upload using the rhinoplasty web application.	3.3.1	1.3	1.1	Critical	P.C. M.C. J.C. M.K. A.O.	Success
3.3.4 The system shall use the KNN algorithm to group the most similar faces.	3.3.1 3.3.5	1.4	1.1 1.5	Critical	J. C. M. C.	Success
3.3.5 The system shall display k options for the patient to choose from.	3.3.1 3.3.5 3.3.7	1.5	1.1 1.5 1.7	Critical	J. C.	Success
3.3.6 The system shall generate more suggestions based on the patient’s request.	N/A	1.6	N/A	Low	J. C. M. C.	Success

3.3.7 The system shall send the patient's choice to Blender.	3.3.1 3.3.5	1.7	1.1 1.5	Medium	P.C. M.C. J.C. M.K. A.O.	Success

5.2 Defect Severity Definitions

Critical: A critical severity defect can be classified as a defect that severely or significantly impairs the overall functionality of the software. This class of defect is fixed with a significant amount of effort in the functionality of the system.

Medium: A medium severity defect can be classified as a defect that does not impair the overall functionality of the software, but impairs certain aspects. This class of defect can be fixed with a medium amount of effort.

Low: A low severity defect can be classified as a defect that does not impair the functionality of the software. This class of defect can be fixed with a low amount of effort.

6. Environment

6.1 Environment

- The System Testing environment will be used for System Testing.

In order to conduct the testing the tester needs to have the following installed onto their computer:

- Blender V2.9 or later
- Access to database of faces
- Access to database of face ratios
- Python V3.6 or later
- V3.0.9 of Openpyxl in Python site packages

7. Assumptions

This section lists the assumptions for this project.

- User is on a machine with the necessary software
- User is on a machine that has access to internet
- All files are in the correct file format
- User must use Blender
- Blender must be modified in order to function correctly
- Clients will submit pictures with no facial expressions to render a 3D image
- No face tattoos/piercings
- No glasses/facemasks

8. Risks and Contingencies

Risk #	Risk	Impact	Contingency Plan
1	Unsuitable Suggestion	High	Making sure different scenarios are tested rigorously to insure this happens as infrequently as possible
2	Machine Learning Incompatibility	High	Testing deployment in many scenarios to ensure the ML AI can consistently produce results in the suitable environments