**System Requirements Specification**

**of**

**Machine Learning Algorithm for Rhinoplasty (M-LAR)**

**SE 450, Fall, 2020**

**Team Name: Nose Surgery 1**

**Team Members:**

* Chris Graziano
* Victoria Jordan
* Anton Kiselev
* Jacob Preseau
* Chintan Thakrar

|  |  |  |
| --- | --- | --- |
| **Date/Time** | **Author** | **Description** |
| September 17, 3:24 PM | Jacob Preseau | Added Team Name and Team Members |
| September 18, 11:47 PM | Chris Graziano | Added Project Name and Course, Semester, Year; first bullet in Definitions |
| September 18, 11:47 PM | Victoria Jordan | Added System to be Produced |
| September 21, 3:15 PM | Jacob Preseau | Added Assumptions, first bullet in Stakeholders, and second bullet in Definitions |
| September 21, 5:55 PM | Anton Kiselev | Added Use Case Model, DFD Models, and State Chart Model |
| September 21, 7:05 PM | Jacob Preseau | Added second bullet in Stakeholders, Use Case Descriptions |
| September 22, 3:31 PM | Chris Graziano | Added Applicable Standards |
| September 23, 5:44 PM | Anton Kiselev | Updated Use Case Model, model descriptions |
| September 24 | Chris Graziano, Victoria Jordan, Anton Kiselev, Chintan Thakrar | Added Requirements |
| September 25 | Chris Graziano, Victoria Jordan, Anton Kiselev, Jacob Preseau, Chintan Thakrar | Revised Requirements |
| September 25 | Anton Kiselev | Updated models and description |
| October 14, 10:00 PM | Victoria Jordan | Document revisions based on version 1 feedback |

**Table of Contents**

1. **Introduction**
   1. System to be Produced
   2. Applicable Standards
2. **Definition, Acronyms, and Abbreviations**
3. **Product Overview**
   1. Assumptions
   2. Stakeholders
   3. Event Table
   4. Use Case Diagram
   5. Use Case Descriptions
   6. DFD Diagram
   7. DFD Descriptions
4. **Specific Requirements**
   1. Functional Requirements
   2. Interface Requirements
   3. Physical Environment Requirements
   4. Users and Human Factors Requirements
   5. Documentation Requirements
   6. Data Requirements
   7. Resource Requirements
   8. Security Requirements
5. **Supporting Material**
6. **Introduction**
   1. System to be Produced

The system will provide a patient with representative similar faces with different nose types that align with the measurements of their face, therefore allowing the patient to select which nose is preferred for their rhinoplasty procedure. This system also complements https://digitized-rhinoplasty.com/app.

* 1. Applicable Standards

The program will adhere to all U.S. Department of Health and Human Services (HHS) regulations on medical standards.

1. **Definition, Acronyms, and Abbreviations**

* KNN - "*k*-nearest neighbors", a machine learning algorithm that, when given a test point represented by a point on the coordinate plane, will find the *k* points in the data set, also in the coordinate plane, that are nearest to the test point.
* "Patient" and "User" are used as interchangeable terms.

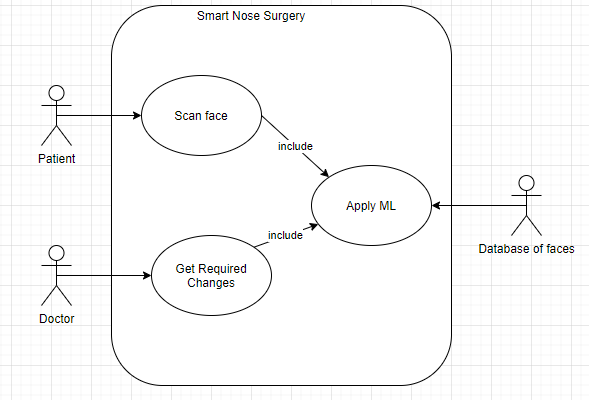
1. **Product Overview**
   1. Assumptions

The product will be delivered through a standalone Python application.

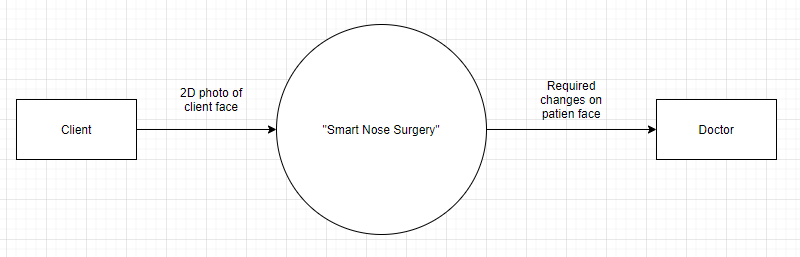
* 1. Stakeholders
* User - The user's interest is in being able to use the product to select their preferred nose for their procedure.
* Doctor - The doctor performing the user's rhinoplasty procedure would benefit from knowing exactly what changes are needed to perform the procedure.
* Cosmetic Investors - Provide cosmetic products or discounts when users complete the rhinoplasty process.
* Model - add face to database.
* Consultant - walk user through the application, can give secondary suggestion
  1. Event Table

|  |  |  |  |
| --- | --- | --- | --- |
| Event Name | External Stimuli | External Responses | Internal data and state |
| Doctor scans patient face | Doctor | None | App transform photo into coordinate file |
| Low quality photo | None | None | App notifies that photo is not appropriate |
| High quality photo | User | None | App transform photo into coordinate file |
|  |  |  |  |

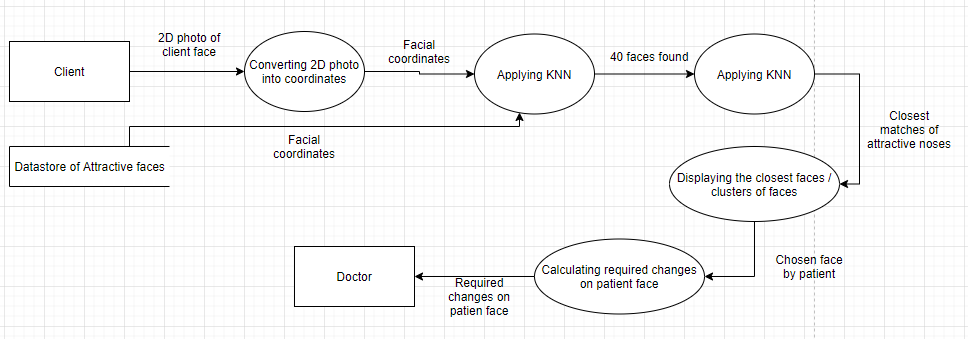
* 1. Use Case Diagram



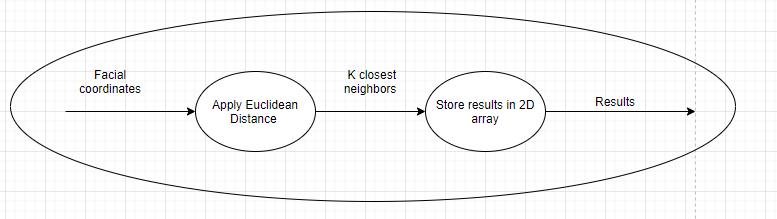
* 1. Use Case Descriptions
* **Scan face** - System asks the patient to provide an image of their face. The patient uploads an image of their face to the system.
* **Apply ML** - Included use case. The system gets images from the database to apply the KNN algorithm.
* **Get Required Changes** - Use case includes “Apply ML” use case. After applying KNN and getting the closest faces, the doctor receives required changes on the patient face from the system.
  1. DFD Diagram
     1. DFD Model level 0



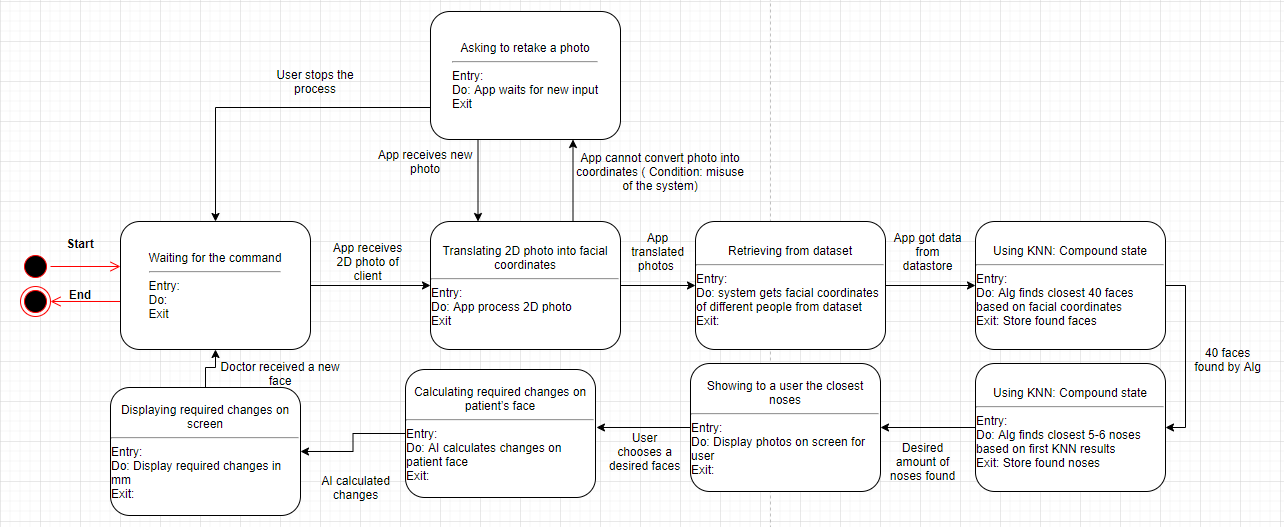
* + 1. DFD Model level 1



* + 1. DFD Model level 1 extended state Applying KNN



* 1. DFD Descriptions
* DFD model level 0 - in this model, we can see three main components: a source of data - Client, main process - “Smart Nose Surgery” and sink for data - doctor.
* DFD model level 1 - in this model we expanded the main process into several smaller processes to get a better idea of what is behind it. First of all the system converts a patient photo into a facial coordinates and retrieves facial coordinates of different people from the data store. After that, the process “Applying KNN” which is described below level 1 model, will do calculations. In the extended state, the model shows that the algorithm, apply Euclidean distance check and store results for future access. After applying KNN for faces, the next process will display the results of KNN with noses and let the client choose the desired face. The last process will calculate the required changes for the client's face and doctor will get the data of it.
  1. State Chart Model



* 1. State Chart Model Description
* State Chart allows seeing what states and conditions will be in the system. This state chart represents the basic cycle of the process.
  + The system waits for the command.
  + After receiving photos, the system will transform it into a file with facial coordinates.
    - Possible error: photo taken incorrectly, system will ask a user to retake photo.
    - User may cancel the process and system will get into initial state
  + After a successful transformation, the system will be obtaining files from the datastore.
  + After that, the system will apply KNN to find 40 closest faces.
  + After that, the system will apply KNN to find 5-6 closest noses.
  + The system will display the result for a user and allow him to choose the desired faces.
  + After the client chooses the desired face, the system will be calculating the required changes and display them to the doctor.
  + After that doctor receives the final results and the system will be ready for a new cycle.

1. **Specific Requirements**
   1. Functional Requirements
      1. The system shall allow a user to scan their face with a camera.
      2. The system shall be able to transform the picture of the user's face into a set of coordinates representing facial landmark measurements.
      3. The system shall contain coordinate files with an array of 67 variables that can be used to uniquely characterize each face in the database and that of the user.
      4. The system shall ask a user to retake the photo in case coordinate detection fails.
      5. The system shall be able to get the files representing faces from the database.
      6. The system shall be able to apply the KNN algorithm on the files in the database.
         1. The system shall be able to apply KNN with the user's face as the test point, using all landmarks other than those relating to the nose.
         2. The system shall be able to save the closest 40 faces from the dataset into an array.
         3. The system shall be able to apply KNN on 40 faces and the user’s face, using only the landmarks relating to the nose.
         4. The system shall save the 5 closest faces into an array.
      7. The system shall be able to display faces stored in the array.
      8. The system shall display to the user 5 different noses.
      9. The system shall allow the user to choose a nose from the 5 presented.
      10. The system shall be able to calculate required changes on the user’s face.
          1. The system shall be able to calculate distances between the nose points of the user face and the chosen face.
          2. The system shall be able to display changes on the user’s photo.
          3. The system shall be able to save the photo with final changes.
      11. The system shall allow the doctor to access the final photo with text explaining changes to be made.
   2. Interface Requirements
      1. The system shall have photo files as an input
      2. The system shall receive one user’s face at a time
   3. Physical Environment Requirements
      1. The system shall operate on any computer operating system (i.e. Linux, Windows) that supports Python 3.
   4. Users and Human Factors Requirements
      1. The systems’ users shall be the patients and the doctor.
      2. The system shall provide instructions for use.
      3. The system shall provide each user a brief description of the output.
      4. The systems’ user shall not require any special accommodations.
      5. The system shall be able to prompt the user to retake their photo in case of coordinate detection failure.
   5. Documentation Requirements
      1. All shareholders shall be provided with a user manual.
      2. The user manual shall specifically describe how to use the system.
      3. The user manual shall specifically describe the purpose of the system.
   6. Data Requirements
      1. The system shall select the initial 40 faces with the KNN machine learning algorithm, using Euclidean distance as the distance metric.
      2. The system shall select the final 5 noses with the KNN machine learning algorithm, using Euclidean distance as the distance metric.
      3. The system shall have a represented user approval rating of at least 80% for the measure of precision of the algorithm.
      4. The system shall obtain data from trials based on whether or not the user successfully found a nose they approved of.
      5. The system shall contain a face library that will exist as a part of the app, and not on a network.
   7. Resource Requirements
      1. The system shall require a skilled team to build.
      2. The system shall be operable by anyone familiar with a computer.
      3. The system shall require minimal resources upon launch.
         1. The system shall not require a skilled personnel to maintain the system.
         2. The system shall run on, and be managed during operation, by the user’s machine.
         3. The system shall be ready on December 1st, 2020.
         4. The system shall require no additional funding
         5. The system shall require no additional hardware.
         6. The system shall utilize software tools: GitHub, ZenHub, Anaconda 3 (and all of its included tools), as well as the cv2 and dlib Python libraries.
   8. Security Requirements
      1. The system shall verify login credentials through the API associated with the user.
      2. The system shall lock users out for 1 minute if the verification process fails 3 times consecutively.