**Chapter 1**

**INTRODUCTION**

* 1. **Camera Manipulation**

The camera can imitate the image of the environment. It is basically used for production, socialization, business and other purposes. Many applications are using camera to perform processes easily by taking photos, videos and recordings. Camera can now perform complicated tasks, thus, I.T developers are frequently putting art into it using camera manipulation to create effects such as photo filters, face detection, face swapping and much more.

Since IT developers are increasing, the number of technologies and applications becomes greater. Due to competition, privacy and other purposes, some processes of creating the technologies and applications are hidden. One of the most amazing and interesting applications that is trending in the market nowadays is the face swapping. However, none of the source codes of any existing face swapping applications such as Snapchat, Snow, Hong Chou Face Swap, Magic Mirror and others is available for other developers to refer. Furthermore, it has been found out that only snapchat and snow can swap two or more person’s faces, while others, including the hong chou face swap and magic mirror can only swap two faces.

The proponents became interested about the beauty of camera manipulation particularly the face swapping, face detection and image processing. Therefore, the proponents proposed to develop “Android Face Swapping Application” that enables users to randomly swap more than two person’s faces in real-time face detection using android phone’s front and back camera. Faces should be detected before being swapped with another. To detect a face, camera must capture any facial landmarks such as the eyes, nose, and lips, which are all located in the convex hull of the face.

Lastly, the study was developed to share its documentation & source code of the algorithm of android face swapping application. The future I.T developers can now access the source code of the algorithm of face swapping feature. Also, the people can now enjoy swapping two or more faces other than Snapchat and Snow.

* 1. **Project Context**

The Android Face Swapping Application is developed to apply camera manipulation to an android application. The proponents developed it to publicly share the source code of the algorithm of face swapping feature, detect faces, swap more than two faces, and contribute enjoyment and entertainment to the people around the society. However, developing the android face swapping application has requirements to make it successful. The tools such as android programming language, android studio, emulator, android phone/tablet, computer/laptop, internet connection (for researching purposes) and related studies (written in chapter 2) especially the OpenCv Libraries were used during the development.

The proponents used an *Android Programming Language*to set instructions to the application that enables it to run on the phone/tablet*.  Android studio* was used to build and compile all codes/syntaxes (android programming language) that served as the instructions that a device can interpret and execute.

The proponents also used *Genymotion Virtual Emulator* to temporary build/create the application during the debugging and testing phase. The genymotion virtual emulator is ten (10) times faster than the built-in emulator, it has an API (Application Programming Interface) level 18.

The proponents also used a phone/tablet with an android version of 4.3 (JellyBean) at a minimum, or higher because the application has a slow performance if it will be installed on a phone with a lower android version.

The proponents also used a laptop/computer to create the whole application, research everything about the project and create the documentation. Using an internet connection, the proponents was able to research all the data and requirements about the study.

Lastly, the proponents referred from related studies about the ways and techniques on how the camera manipulation is applied especially the face swapping, face detection and image processing. The *Open Source Computer Vision*or OpenCV libraries were the guidelines in using the algorithms, sample codes, logics and detections used for the application’s development.

* 1. **Purpose and Description**

The purpose of the study is to develop an application that addresses the scarcity of resources for those I.T developers who were searching for the documentation and source code of the algorithm of face swapping feature, face detection and other camera manipulation. Since there are only two face swapping applications that can swap two or more person’s faces such as snapchat and snow, the proponents intended to develop an android face swapping application that can swap two or more faces.

Real time face detection and camera manipulation is one of the new trends of the current generation. As the proponents explored the new software creation specifically about camera manipulation, the proponents became curious on how the software works. Thus, the proponents wants to have a deeper understanding on the logic behind camera manipulation and face detection, develop and share it to local and international future new developers/researchers.

The **Android Face Swapping Application** will be a significant endeavor to the following:

* + 1. **Future I.T Developers**

The study will not only provide the source code of the algorithm of face swapping feature but also give an idea on how to detect faces, save captured image to the gallery and other camera manipulation that are developed in the application. The documentation will also guide them on what specific requirements, approaches, and methodologies will be used when developing an android application.

* + 1. **Camera Application Enthusiasts**

Android Face Swapping Application will let people be aligned with the new trends in the society. Also, it will contribute enjoyment and entertainment to people through detecting and swapping two or more faces.

* 1. **Objectives of the Study**

**General Objective:**

The main objective of the study is to learn the process of designing a face swapping application without referring to the source code of existing face swapping applications (such as Snapchat and Snow) which is not shared publicly. The proponents aimed to explore the logic behind camera manipulation and face swapping application which swaps more than two persons faces to publicly share its documentation and source code of the algorithms on how it is being developed.

**Specific Objectives:**

* To gather specific data and resources related to the android face swap.
* To design graphical user interface and functional module of android face swapping application based on the gathered data.
* To identify specific features and requirements for the application.
* To identify the best methodology and approach to use for the application’s development.
* To identify significant application processes.
* To refer from OpenCV libraries for the algorithms, sample codes, logics and face detections.
* To identify significant camera application functions such as saving image in a storage.
* To combine the identified algorithms, processes and camera functions, tailoring the best possible components for the application.
* To provide a document properly implementing the face swapping process to be referred to by future developers.
* To develop an application with the use of android studio.
  1. **Scope**
* Android face swapping application’s APK (Android Package Kit) is installable in android phones.
* Android face swapping application can take photos using smart phone’s camera.
* Android face swapping application can randomly swap two or more person’s faces.
* Android face swapping application can detect face, capture image and display picture.
* Android face swapping application can save captured photos to the phone’s gallery.

**1.6 Limitations**

* Android face swapping application cannot connect online.
* Android face swapping application is not yet available at play store.
* Android face swapping application cannot connect to any social media such as Facebook, Twitter, Instagram and others.
* Android face swapping application will only efficiently run on updated android version of phone with at least version 4.3 (JellyBean) at a minimum, or higher. This version can handle and is necessary for face detection.
  1. **Definition of Terms**
* **Algorithm–** a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.
* **Android App-** is a software application running on the Android platform. Because the Android platform is built for mobile devices, a typical Android app is designed for a smartphone or a tablet PC running on the Android OS.
* **App-** a self-contained program or piece of software designed to fulfill a particular purpose; an application, especially as downloaded by a user to a mobile device.
* **Android Programming Interface –** a set of routine definitions, protocols, and tools for creating software and applications
* **Programming Language-** is a formal computer language designed to communicate instructions to a machine, particularly a computer. Programming languages can be used to create programs to control the behavior of a machine or to express algorithms.
* **APK (**Android Package Kit**)-** is the package file format used by the Android operating system for distribution and installation of mobile apps and middleware.
* **Build–** the construction of something that has an observable and tangible result.
* **Camera manipulation** -is where you use the camera to create unique effects such as cut scenes, flashbacks and much more.
* **Compile–** the act of translating another program written in a high-level language into machine language so that it can be executed
* **Emulator–** an emulation of a particular computer system. It operates based on the computer architecture and functions of a real or hypothetical computer and the implementations may involve specialized hardware, software, or a combination of both.
* **Real-time face detection**– a computer technology being used in a variety of applications that detects human faces while using the camera
* **Run–** the process by which a computer or a virtual machine performs the instructions of a computer program
* **Installation-** (or setup) of a computer program (including device drivers and plugins), is the act of making the program ready for execution.
* **Gallery-** a large room for the exhibition of pictures; also a collection of pictures.

**Chapter 2**

**REVIEW AND RELATED SYSTEMS**

* 1. **Face Detection, Extraction, and Swapping on Mobile Devices**

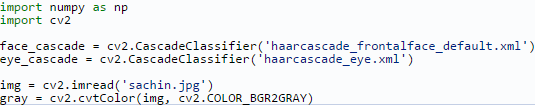
The study of **Hong Chou Face Swap**, developed by Andrew Chou from the Department of Computer Science Stanford University, focuses on identifying the methods that is essential in the process of face swapping which are the face detection then the swapping of the detected faces. The study is similar to what the proponents emphasized in developing the study of the Android Face Swapping Application. What the proponents got from the study of Hong Chou is the logic on how to swap faces which is to detect the faces present before swapping it. However, the only difference between the study Hong Chou face swap and android face swapping application is the way on how to swap faces; the android face swapping application swaps faces in real-time live camera feed face detection, while Hong Chou face swap only swaps faces present in an image. The concept of Hong Chou face swap is to take first a photo before the faces would swap. The proponents did not get any of hong chou face swap’s algorithms since their method of swapping faces is not through live camera feed.

Another study that helped the proponents in facial detection is the **Open Source Computer Vision** which is a computer vision library that started by Intel. The cross-platform library sets its focus on real-time image processing and includes patent-free implementations of the latest computer vision algorithms. In 2008, Willow Garage took over support and OpenCV 2.3.1 now comes with a programming interface to Android. It now comes with the very new *FaceRecognizer* class for face recognition that helped the proponents in experimenting with face recognition. It showed the proponents how to perform face recognition with *FaceRecognizer* in OpenCV and gave an introduction into algorithms behind. According to David Hubel and Torsten Wiesel, human’s brain has specialized nerve cell responding to specific local features of a scene, such as lines, edges, angles, or movement. Since \the proponents don’t see the world as scattered pieces, our visual cortex must somehow combine the different sources of information into useful patterns. Automatic face recognition is all about extracting meaningful features from an image, putting it into a useful representation and performing some kind of classification.

The main algorithm that the proponents used was the Haar feature-based cascade classifiers. The Haar cascades are an effective object detection method proposed by Paul Viola and Michael Jones in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.  Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, haar features shown in below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

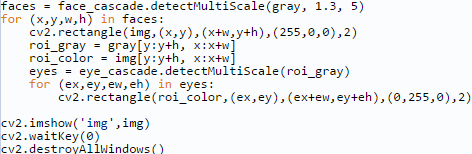
*Figure 1 Concept of Cascade of Classifiers*

OpenCV already contains many pre-trained classifiers for face, eyes, smile etc. First we need to load the required XML classifiers. Then load our input image (or video) in gray scale mode. Now we find the faces in the image. Here is the code for the algorithm:



*Figure 2 Haar Algorithm*

If faces are found, it returns the positions of detected faces as Rect(x,y,w,h). Once we get these locations, we can create a ROI for the face and apply eye detection on this ROI (since eyes are always on the face). Here is the code for the algorithm:



*Figure 2.1 Haar Algorithm*

Real-Time Face Swapping in Video Sequences: **Magic Mirror** written by Nuri Murat Arar and team has guided the proponents on what to do after detecting and swapping the faces of the users. The magic mirror study includes Face Alignment which states that input frames coming from video camera also differ and size and orientation because user may not stand in front of the camera at constant distance and position. In order to perform directly swapping of input and target faces, an alignment step is needed to bring various sized and positioned face images into a common coordinate system with a particular size. With the concept of magic mirror, the proponents has analyzed on how to align the faces that are swapped. The proponents planned to locate special facial landmarks such as the eyes, nose, and lips which are all located in the convex hull of the face. The faces will now be aligned in terms of the position of the convex hull by finding common facial coordinates.

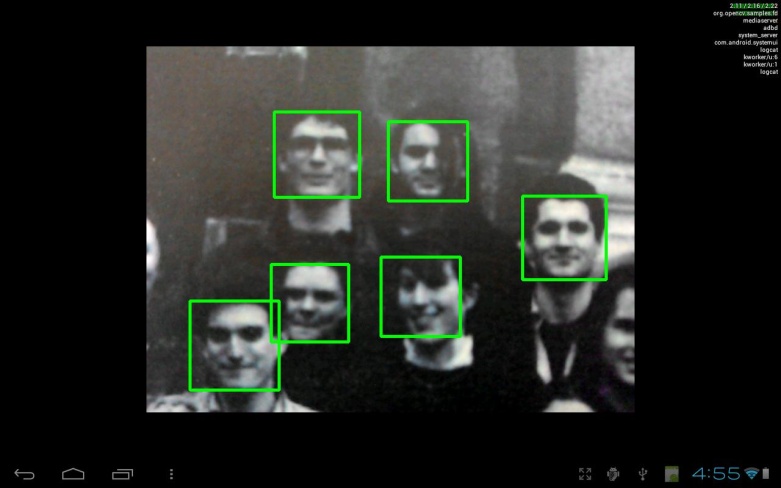
**Word Lens** developed by Octavio Good in 2010, an augmented reality translation application from Quest Visual, uses built-in cameras on smartphones and similar devices to quickly scan and identify foreign text (such as that found in a sign or a menu), and then translate and display the words in another language on the device’s display. The words were displayed in the original context on the original background, and the translation was performed in real-time without connection to the internet. For example, using the viewfinder of a camera to show a shop sign on a smartphone’s display would result in a real-time image of the shop sign being displayed, but the words shown on the sign would be the translated words instead other original foreign words. Face swapping also uses augmented reality in processing live camera feed input and analyzes the users’ faces. The analyzed faces will then be swapped from their original positions to another person’s face location.

* 1. **Technical Background**

In developing the Android Face Swapping Application, the proponents will use different tools in order to make the development successful.

**Android programming language** is needed to create the whole application. A [formal](https://en.wikipedia.org/wiki/Formal_language) [computer language](https://en.wikipedia.org/wiki/Computer_language) or [constructed language](https://en.wikipedia.org/wiki/Constructed_language) designed to send [instructions](https://en.wikipedia.org/wiki/Machine_instruction) to a [machine](https://en.wikipedia.org/wiki/Machine), particularly a [computer](https://en.wikipedia.org/wiki/Computer), cellphone, and other devices. Programming languages can be used to create [programs](https://en.wikipedia.org/wiki/Program_(machine)) to control the behavior of a machine or to express [algorithms](https://en.wikipedia.org/wiki/Algorithm).

Using the **Open Source Computer Vision** or **OpenCV** will guide the proponents with the algorithms, sample codes, logics and detections that will be used as a reference for developing the Android Face Swapping Application.



*Figure 3 OpenCV Face Detection Test Image*

Through **Android Studio** application, all the codes/syntaxes (programming language) will be compiled and created as an application. Android studio will make codes functional. Android studio will detect whenever the codes are incorrect which may cause an interruption to the development. All the codes of the different features of face swapping application such as face detection, face swapping, and image processing will be created inside the android studio based on each computer’s architectures, functions and its implementation.

Using Genymotion emulator with an API (Application Programming Interface) level 18, the created application will be tested and run. The created application will also be temporarily installed into the emulator. Android Programming Interface (API) is called *Face Detector Listener*.

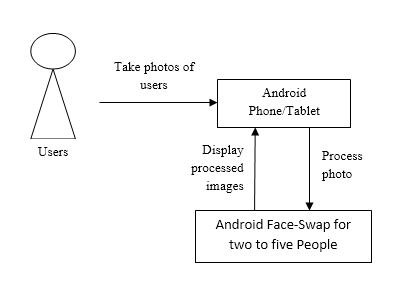
After the development, the application will be officially installed and operated in an android phone with the android version of 4.3 (called Jelly Bean) at minimum, or higher. The android phone is a high-tech smartphone that runs on the android mobile operating system (OS) – which has been designed primarily for touchscreen mobile devices such as smartphones and tablets.

Lastly, the proponents can assure that the Android Face Swapping Application will be fully functional if all the stated requirements are satisfied.

**Chapter 3**

**RESEARCH METHODOLOGY**

**3.1 The Project Concepts**

****

View image

*Figure 4 Project Concept Diagram*

Figure 4 illustrates how the users interact with the application. Every time the users open the application, it automatically detects and captures a person’s faces and swaps it randomly. Lastly, after taking a photo, it would automatically save in the phone’s/tablet’s storage.

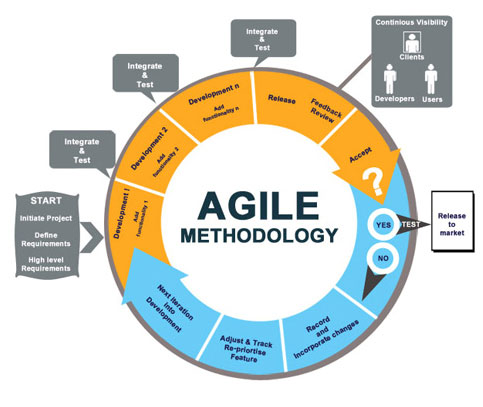
**3.2 System Analysis and Design**

While planning the android face swap application, the proponents has detected several possible problems than may be encountered in the designing and development process. One of the problems that have been discovered is about how to set the number of faces that will be detected by the application. The number of faces that the proponents planned to swap is at most five. With the current face swapping applications that are widely used right now, existing applications have only limited the face-swapping feature to two faces only. The most basic way to swap more than two faces is to capture a picture first which consists of more than two faces. The image will now be converted into a bitmap then the application will detect the faces present on the image. The app will now swap faces randomly after detecting the faces. The problem only is that the face swapping process is not in real-time. Based on the proponent’s research, faces are detected at high frame rates. So the proponents decided to set the minimum of android version to 4.3.x, which is the Android Jellybean (API level 18). Through the minimum android version, the proponents can easily achieve in detecting face in high frame rates. The proponents found a solution for the multiple face detection, which is the Open Source Computer Vision or OpenCV. OpenCV is used with a strong focus on real-time applications; has a number of samples, like the image manipulation and face detection, which greatly helped the proponents in detecting multiple faces.

The proponents used the Object-Oriented (OO) approach. Through Object-Oriented approach, proponents can work with reusable code components which preserve the ability to focus in unique application area. The approach allows making use of numerous existing code resources within both the Java language and the Android platform itself. The OO approach is often used by developers working as teams, who need to be able to use one another’s code without having to understand the implantation details within any particular section. The idea of an application being made up of code excerpts, each of which has clearly defined behaviors, is the one key to understand, whether you’re working with classes and interfaces from the proponent’s own application, or from the Android platform, or from the Java language itself.

The application requires android phone/tablet with at least android version four (4), the android phone’s camera, and Open Computer Vision or OpenCV libraries, which helped the proponents in achieving some of the modules like the face detection and swapping, for its development. Thus, the application will be successful and functional if the requirements are satisfied. The application will be installed in the phone/tablet with the use of the application’s installer.

**3.3 Development Model**



*Figure 5 Agile Development Methodology diagram*

Since the android face swapping study is about mobile application development, the development team decided to use the agile development model. The proponents defined agile development as the ability to move quickly and easily; relating to a method of project management that is characterized by the division of tasks into short phases of work and regular re-evaluation and adjustment of plans. So the proponents decided to use the agile development model to easily break down functional requirements which will eventually be the foundation of the proponent’s application development schedule. Also, mobile applications, such as the face swapping application, need to be frequently revised to meet end-user expectations and to support frequent mobile application deployments and updates. The agile methodology provides users, developers, and testers with a continuous feedback loop which means the users, developers and testers provide feedback every step of the way to ensure the success of the project.

**3.3.1 Planning**

From the beginning, the plan was about creating a team. Every members were assigned to particular position that defines each member’s roles when conducting a study and creating an application project. The positions that have been assigned to each member were the following: project leader, researcher and developer.

Before the team started to study about camera manipulation, the team planned regarding the specific camera effects that must be researched and developed first. Among all the effects that has been researched, the team chose the face detection, face swapping, and image processing as the main features of the application; therefore, the team decided “Android Face Swapping Application” to be the title of the study. After defining all the features of the application, all scopes and limitations has been enumerated to finalize the list of tasks that should be done for the rest of the development time.

**3.3.2 Requirements Gathering**

In finding all the requirements that must be used for developing the android face swapping application, there are some references that helped the proponents gather all requirements such as PDFs and Related Systems. The PDF is entitled **Hong Chou Face Swap**, published by the author— Mr. Andrew Chou, from the Department of Computer Science Stanford University.More details about related systems are showed in chapter 2.1 (Review and Related Systems).

The scopes and limitations are also the basis for gathering the requirements. The proponents researched each scope over the internet. Thus, through the scopes, and references above, proponents discovered the following requirements: android programming language, android studio, Genymotion virtual emulator, android phone with an android version 4.3 at a minimum or higher, and OpenCV’s sample codes. Some details of these requirements are showed in chapter 1.2 (Project Context).

**3.3.3 Analysis**

The team identified the overall direction that the project will take through.

The android face swapping application is free for all users, as shown above, who are interested with the current trend which is the face manipulation. The users will install the Android Application Package or the APK file in the android phones to use it. The application will need to use the camera of the phone and take a photo of the users. The face that will be detected by the camera will be processed. Then the captured image will now be displayed to the user as an edited image.

Captured Image

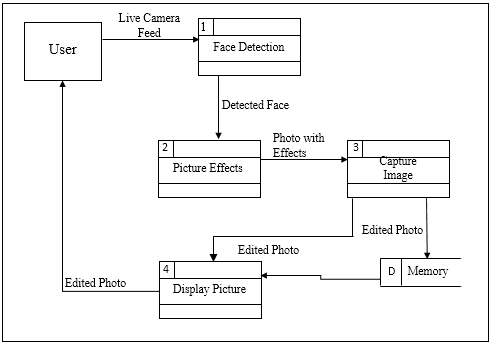
**User**

Raw Image through Camera

*Figure 6 Context Diagram*

**3.3.4 Design**

The Android Face Swapping Application is focused on the user only, who will solely interact with the application. The app has four processes; (1) Face Detection, (2) Picture Effects, (3) Capture Image, and (4) Display Picture. The capturing of image is focused on detecting the face of the users while using the camera of the phone. The face detection module can be seen here. The second process, where the users will choose desired effects, is focused on photo manipulation like face swapping through real time camera feed. The third process simply captures the image of the users with the desired effects already applied. The last one is where the application displays the result of the photo with the applied effects to the user. And the storage (Memory) that lets the user decides to whether or not save the photo on the memory or delete it.

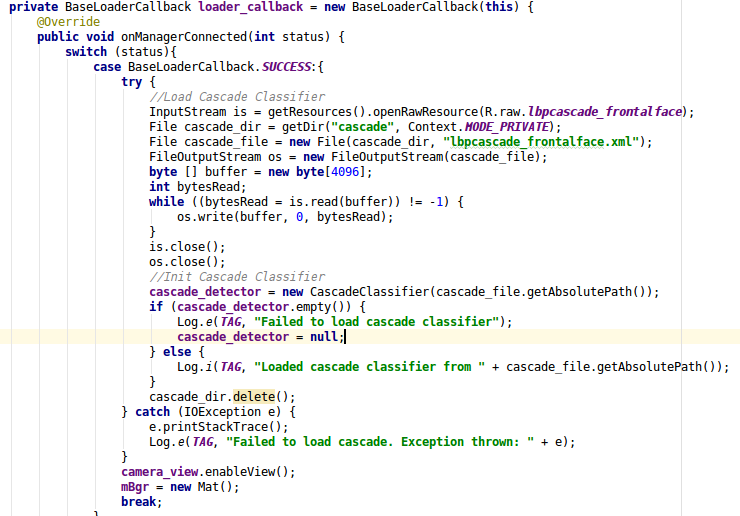


*Figure 7 Data Flow Diagram*

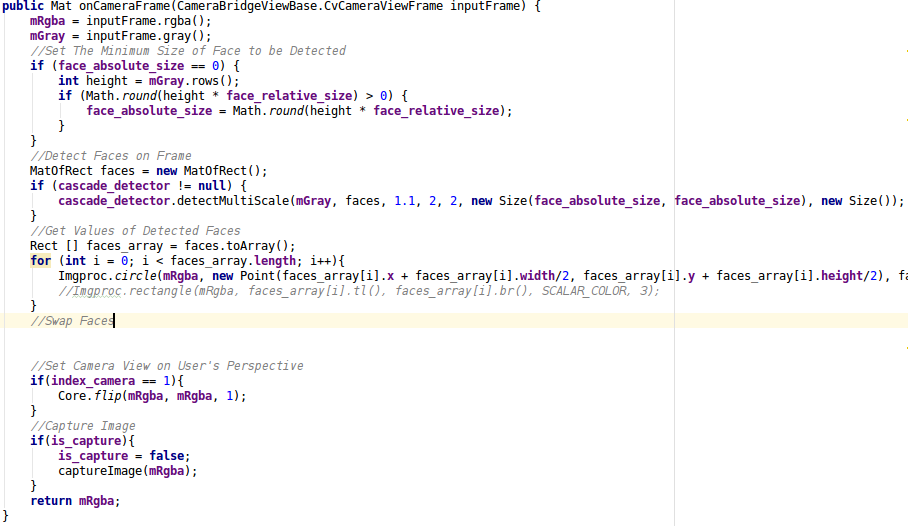
After defining the project’s architecture, the proponents started to break down the processes into modules and created a timeline which can be seen in Chapter 3.6.

**3.3.5 Development**

During the development phase, the proponents have explained on how the developers implemented and coded the defined scopes of the project.

First, the developers have analyzed and tried to use the gathered tools, namely the android studio and OpenCV, which is related to our project. After studying the tools, proponents have followed the development timeline which consists of the modules defined by the team. The proponents first started developing the camera module by importing the OpenCV libraries as a module of the project and selected the OpenCV libraries as a dependency in the project structure of the application. The developers then created a folder namely the *jnilib* which contains the OpenCV native libraries. In the *activity\_main.xml*, we have created a *JavaCameraView* that is from the OpenCV library and not from the android studio. This lets the developers to create a camera view that can be integrated with the other OpenCV libraries such as the face detection. Once you are finished setting up the *JavaCameraView* of OpenCv, the next thing to do is to add the lbpcascade\_frontalface.xml, which is an Haar Cascade in order to detect faces in a frame or image, in the res folder of the Android Project. After adding it, on *BaseLoaderCallback* setup the following syntax to initialize the Cascade we added before. The following figure will show you how initialize it.

Once done initializing, on method onCameraFrame, set the minimum absolute size of the image to be detected to avoid misdetection, then with help of the Haar Cascade we will be able to detect multiple faces in the frame. The following figure will show you how.



After detecting the faces, the developers began to develop the face swapping sub-module. The logic behind the said module is to swap the positions of the landmarks and creating a mesh on the detected faces with the use of the camera. The testing comes after developing each module.

**3.3.6 Testing and Debugging**

During the testing and debugging phase, the proponents have manually tested the android face swap application for its functionality, usability and performance.

One of the tests that the proponents have performed is the **system usability testing**. The proponents verified if the application is achieving the goals by testing out the defined modules, which are the interface and camera modules, by installing the application in android phones and tablets with versions 4.3.x (Jellybean, API level 18) and up with the use of Android Application Package (APK). The testing ensures that the user’s experience is consistent across all devices with those specifications. But before the usability testing, the proponents have undergone the **installation testing** to verify that the installation process goes smoothly with the user have to face any troubles.

**3.3.7 Deployment**

During the deployment phase, the proponents have deployed the android application in android devices which have an API level higher than or equal to 18 by installing the APK file.

**3.4 Development Approach**

The proponents used the bottom-up planning in developing the Android Face Swapping Application. In bottom-up planning, proponents give the project the deeper understanding because proponents have a handful of developers and researchers involved; each with proponent’s own role and responsibilities. Members will work side-by-side and have input during each stage of the process.

**3.5 Software Development Tools**

The tool that proponents decided to use in developing the Android Face Swapping Application is the Android Development Studio, which is written in Java, and the Open Source Computer Vision or OpenCV for Android, which is written in C++, which also aided the proponents in face detection. The proponents used the tools because both are compatible with each other. Developing the application with the use of the android development studio and OpenCv is easier because it is known worldwide by programmers and researchers and there are sample codes released and explained by developers.

Using the Open Source Computer Vision or OpenCV sample, will guide the proponents with the algorithms, sample codes, logics, and detections that will be going to use as a reference for developing the Android Face Swapping Application.

With the help of **Android Studio** application, all the codes/syntaxes (programming language) will be compiled and created as an application. Android studio will make codes functional. Android studio will detect whenever the codes are incorrect which may cause an interruption to the development. All the codes of the different features of face swapping application such as face detection, face swapping, and image processing will be created inside the android studio based on each computer’s architectures, functions and its implementation.

**3.6 Timeline (Gantt chart)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Module** | **Sub Module 1** | **Sub Module 2** | **Sub Module 3** | **Start Date** | **End Date** | **May** | | | **June** | | | | **July** | | | | **August** | | | |
| 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4/5 |
| Planning |  |  |  | 5/10 | 11/26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Requirement Gathering |  |  |  | 6/22 | 7/3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Analysis |  |  |  | 5/10 | 11/29 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Design | Context Diagram |  |  | 7/18 | 7/20 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Data Flow Diagram |  |  | 7/20 | 7/22 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Interface | Menu |  | 7/26 | 7/29 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Shutter Button |  | 7/30 | 8/2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Development | Camera | Detection | Face | 8/3 | 8/11 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Landmarks | 8/12 | 8/26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Swapping |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing | Camera | Detection | Face | 8/3 | 8/11 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Landmarks | 8/12 | 8/26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Swapping |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Deployment |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Module** | **Sub Module 1** | **Sub Module 2** | **Sub Module 3** | **Start Date** | **End Date** | **September** | | | | **October** | | | | **November** | | | | **December** | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4/5 | 1 | 2 | 3 | 4 |
| Planning |  |  |  | 5/10 | 11/26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Requirement Gathering |  |  |  | 6/22 | 11/26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Analysis |  |  |  | 7/4 | 11/29 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Design | Context Diagram |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Data Flow Diagram |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Interface | Menu |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Shutter Button |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Development | Camera | Detection | Face |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Landmarks |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Swapping |  | 9/4 | 11/26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing | Camera | Detection | Face |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Landmarks |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Face Swapping |  | 9/4 | 11/26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Deployment |  |  |  | 11/27 | 11/29 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

*Table 1 Timeline (Gantt chart)*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Module** | **Sub Module 1** | **Sub Module 2** | **Sub Module 3** | **Start Date** | **End Date** | **January** | | | |
| 1 | 2 | 3 | 4 |
| Planning |  |  |  | 5/10 | 11/26 |  |  |  |  |
| Requirement Gathering |  |  |  | 6/22 | 11/26 |  |  |  |  |
| Analysis |  |  |  | 7/4 | 11/29 |  |  |  |  |
| Design | Context Diagram |  |  |  |  |  |  |  |  |
| Data Flow Diagram |  |  |  |  |  |  |  |  |
| Interface | Menu |  |  |  |  |  |  |  |
| Shutter Button |  |  |  |  |  |  |  |
| Development | Camera | Detection | Face |  |  |  |  |  |  |
| Face Landmarks |  |  |  |  |  |  |
| Face Swapping |  | 9/4 | 11/26 |  |  |  |  |
| Testing | Camera | Detection | Face |  |  |  |  |  |  |
| Face Landmarks |  |  |  |  |  |  |
| Face Swapping |  | 9/4 | 11/26 |  |  |  |  |
| Deployment |  |  |  | 11/27 | 11/29 |  |  |  |  |

*Table 1 Timeline (Gantt chart)*

**3.7 Project Teams and Responsibilities**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Appointment** | **Responsibilities** | |
| Villegas, Reynaldo | Project Leader | Documentation | Ch1.5 Scope  Ch1.6 Limitations  Ch2.2 Technical Background  Ch3.6 Timeline  Ch3.7 Project Teams and Responsibilities  Ch3.8 Budget Cost Management Plan |
| Socaoco, Danna Rose | Researcher, Tester | Documentation, Testing | Ch1.3 Purpose and Description  Ch1.4 Objectives  Ch1.6 Definition of Terms |
| Aying, Joneil | Researcher, Analyst | Documentation | Ch1.1 Introduction  Ch1.2 Project Context  Ch2.1 Related Systems  Ch3.1 The Project Concepts  Ch3.3 Development Model  Ch5.1 Summary of Findings  Ch5.2 Conclusion  Ch5.3 Recommendation |
| Bonggot, Brilly Neil | Developer, Tester | System Development & Design, Testing | Ch3.9 Verification, Validation, and Testing Plans  Ch4.1 Validation and Testing  Design: Shutter Button  Development: Face Landmarks Detection  Development: Face Swapping |
| Lai, Jerimy | Researcher, Developer | Documentation, System Development & Design | Ch3.2 System Analysis and Design  Ch3.3 Development Model  Ch4.2 Interpretation/Discussion of  Results  Design: Main menu  Development: Face Detection  Development: Face Landmarks Detection  Development: Face Swapping |

*Table 2 Project Teams and Responsibilities*

**3.8 Budget Cost Management Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Name** | **Price** | | |
| Hardware | Android Phone | P 14,990.00  200.00  505.00 | | |
| Documents | Hard Binding |
| Printing |
|  |  | Days | Hours |  |
| Salary (P42 per hour; 9 working hours per day) | Villegas | 174 | 1,566 | 65,772.00 |
| Socaoco | 107 | 963 | 40,446.00 |
| Aying | 96 | 864 | 36,288.00 |
| Bonggot | 158 | 1,422 | 59,724.00 |
| Lai | 136 | 1,224 | 52,248.00 |
| **Total:** | |  | | **P 270,173** |

*Table 3 Budget Cost Management Plan*

**3.9 Verification, Validation, and Testing Plans**

The proponents planned and performed numerous testing to ensure if the created application coincides the scopes and limitations, and if the application works properly. The testing comes after developing a single module created and defined by the proponents. The first module that the proponents have defined is the interface which focuses on the splash menu and the shutter button. The menu and shutter button is tested in different android phones with an android version 4.3.x and up in order for us to ensure that the design is uniform across those certain phones. The shutter button comes with the camera view. The proponents had testing to verify if the camera view still works proper even if the phone has low battery. The second module that proponents have defined is the camera module which consists of all the functionalities of the face swapping application. The camera module has four sub-modules which are the face detection, face landmarks detection, and face swapping. The proponents tested the face detection first. The module has been tested with the intent to verify the following: if it actually detects a face, if it can detect multiple faces altogether, if it can detect faces while using the camera, if it can detect a face despite of the phone’s orientation whether if it’s in landscape or portrait, and required position of the camera to detect a face. The second sub-module is the face landmarks detection which specifically detects the features of the face like the eyes, eyebrows, nose, and lips. The proponents tested the face landmarks detection by verifying the following: if it can detect the landmarks of a face, if it can detect landmarks on multiple faces altogether, on how far the camera is needed in order to detect the landmarks, and if it can detect the landmarks despite of the camera’s orientation. The third sub-module that proponents have tested is the face swapping. The proponents tested the face swapping module to verify the following: if it can swap two faces based on the faces that have been detected, and if it can swap more than three faces, and if it can swap despite of the camera’s orientation. The proponents validated the face swapping application by creating and achieving what the scope and limitations asked for.