SE 3XA3: Software Requirements Specification
OpenCameraRefined

Group #211, CAMERACREW
Faisal Jaffer, jaffem1
Dominik Buszowiecki, buszowid
Pedram Yazdinia, yazdinip
Zayed Sheet, sheetz

February 10, 2020

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
02/09/2020	1.0	Initial requirements specification

1 Project Drivers

1.1 The Purpose of the Project

The purpose of OpenCameraRefined is to improve the accessibility of features in the OpenCamera application. By adding the ability to capture images hands-free, and the ability to have filters while maintaining a clean UI we hope to attract more users to use the refined application.

1.2 The Stakeholders

1.2.1 The Client

- Dr. Asghar Bokhari
- TAs

1.2.2 The Customers

- The stakeholders include any user that requires accessible features in their camera application, as well as a clean UI for their day-to-day capturing needs

1.2.3 Other Stakeholders

- Core developer team of OpenCameraRefined
- Other developers collaborating on the open source Open Camera application

1.3 Mandated Constraints

1.4 Naming Conventions and Terminology

1. TFLite: TensorFlowLite is a machine learning framework used to train models to predict smiling faces.
2. Bounding Box: A square box displayed in the camera view around the faces detected
3. OpenCV: A framework that provides image manipulation functions.

1.5 Relevant Facts and Assumptions

User characteristics should go under assumptions.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

2.1.2 Work Partitioning

Event Number	Summary of BUC
1	Create an object detection function in the camera app that can detect a user and gestures
2	Create designs of filters that can be applied to a user/setting
3	Create an editing function that will have a palette of filters that the user can apply
4	Finishing edits to the project

Table 2: Work Partitioning Part 1

Event Number	Event Name	Input	Output
1	Object Detection Creation	Developer Code, TensorFlow	Phone Screen
2	Camera Filter Design	Photoshop	Screen
3	Camera Filter Application	Filter, Developer Code	Phone Screen
4	Camera Final Edits	Developer Code	Phone Screen

Table 3: Work Partitioning Part 2

2.1.3 Individual Product Use Cases

Use Case	Trigger	Pre-Condition	Outcome
User wants to capture an image from far	Smiling face in view of the camera	User has feature turned on	Image is captured
User wants to add a filter	User selects a filter	User has selected a filter	Filter is applied to the camera view

Table 4: Use Cases

2.2 Functional Requirements

VP1. User

BE1.1 The user wants to take a picture using a gesture.

i. The system shall detect the target object.
ii. The system shall "look" for the selected gesture.
iii. The system shall save the taken picture.
iv. The system shall allow the user to zoom in and out.
v. The system shall allow the user to change the resolution.
vi. The system shall allow the user to apply a filter in "real-time".

BE1.2 The user wants to apply a "real time" filter.

i. The system must display a menu of filters
ii. The system must allow the user to select from the menu of filters.
iii. The system must show a preview of a selected filter.
iv. The system must allow the user to close the filters menu.
v. They system must detect and place the filter in the correct context.

BE1.3 The user wants to edit a saved picture.

i. The system must allow the user to simply preview the original picture.
ii. The system must allow the user to change the properties of the picture such as contrast.
iii. The system shall create a copy of the original picture before each edition.
iv. The system shall offer a menu of after effects that can be reversed.

3 Non-functional Requirements

3.1 Look and Feel Requirements

LF1. The overall look and feel will remain unchanged from the original Open Camera application.

LF2. The additional features shall seamlessly integrate with the existing look and feel.

3.2 Usability and Humanity Requirements

UH1. The ease of use of the existing features will remain unchanged.

UH2. The gesture photo feature shall allows a user to take a

3.3 Performance Requirements

PR1. The performance of the existing features must not be affected by the additional features being added.

PR2. Real time camera filters shall be activated immediately after the user selects one.

PR3. Saving a photo that is being edited shall take no more than 5 seconds.

PR4. If a photo is being taken using the gesture feature, the app shall detect the gesture at least 75 percent of the time.

3.4 Operational and Environmental Requirements

OE1. The product shall be operable in any location provided that the camera is functional on the users device.

3.5 Maintainability and Support Requirements

MS1. Maintenance as a result of the additional features shall be kept minimal.

MS2. The hardware and software support for the application shall remain unchanged from the original application.

3.6 Security Requirements

SR1. The application shall be available to download by the general public.

SR2. The app shall not transmit any user data or user photos to a external source unless authorized.

3.7 Cultural Requirements

CP1. This product shall not offend any religious or ethnic group.

3.8 Legal Requirements

LR1. The application shall conform to all applicable laws and regulations.

3.9 Health and Safety Requirements

HS1. The gestures required to take a photo shall be subtle and not endanger those around the user.

4 Project Issues

4.1 Open Issues

Documentation

- The original project git lacks proper documentation.

4.2 Off-the-Shelf Solutions

Smile Capture

- An Android application that only lets you capture an image when a smiling face is in view. This application does not have any UI buttons but only a camera view.

Native Android Camera Application

- This is the native camera application that has a hidden feature in the settings. It lets the user smile to capture a picture.

4.3 New Problems

A new problem that might be arise when a user wants to capture an image without smiling. This might create accessibility issues where the user would have to turn off the feature.

4.4 Tasks

Task	Assignee	Timeline
Import TF + OpenCV and refactor	Software Engineers	Feb 5th
Model Training	Software Engineers	Feb 8th
UI Face Bounding Box Implementation	Software Engineers	Feb 10th
PoC	Software Engineers	Feb 11th
PoC Review	Client	Feb 11th
Live Filter Implementation	Software Engineers	Feb 16th
Unit Testing	Software Engineers	Feb 21st

Table 5: Sub-tasks

4.5 Migration to the New Product

None. The model trained to detect different gestures and faces will have a general understanding and will be exported in terms of weights in the APK deployed to each device. Therefore no special migration needed to export to different devices.

4.6 Risks

None.

4.7 Costs

None.

4.8 User Documentation and Training

None. The new features added will have intuitive user flow and easy to understand icons to help user navigate the application.

4.9 Waiting Room

4.10 Ideas for Solutions

TFLite + OpenCV

- Tensorflow lite is a framework used to make ML predictions on device. It trains the model on a computer with a dedicated GPU and then saves that model into weights and an inference that can be recompiled on the mobile device to make predictions. In our solution this method can be trained on a wide variety of images with smiling faces and various of gestures to make a highly accurate prediction. This model can be imported into the OpenCamera application where the camera view will be fed in as the input image. The output from the model will assign a label with an accuracy percentage. We will accept any prediction that is made over 85%. OpenCV can be used to add live filters into the app. This framework has built in functions that convert a BGR image into gray-scale and allow other image manipulations.

Face Detector API

- FaceDetector API is built into Android 6.0 and later to help find faces in images. Since it is built into the OS, it is highly optimized and is pre-trained on Googles servers for high accuracy. Although it is highly effeicnt at finding faces, it offers APIs that can distinguish between a smiling and a normal face. Also it lacks support for detecting other gestures like a 'thumbs-up'.