A5 DD-1 D_04 Design Document for P-Corrector

Group#06

Madi Uristen: %marks

Shokhan Aidarov: %marks

Nurzhigit Shalkarbay: %marks

Akylbek Ordabayev: % marks

Version	Date	Author	Change
0.1	10/08/04	SM	Initial Document

Table of Contents

1 Introduction

P-corrector is designed for users of Android devices. This application is created to handle photos and videos, overlay effects and filters, and sending them to instagram.

PURPOSE

Become the best editor of pictures and videos in PlayMarket

This document will only serve as an introduction to the concepts of modular design.

DEFINITIONS, ACRONYMS, ABBREVIATIONS

Term	Description
Canvas Page	Any processed image or video is laid out in instagram
Feed	Feed is a data format used for providing users with frequently updated news content.
DB	Database that is used in the project.

DESIGN GOALS

1. Reliability

   Our project needs to be reliable because it is supposed to be used by a certain amount of users, who are going to be provided by the highest quality application.

   Response Time. The project has to be made the way it is going to respond very quickly, as fast as possible.

   The application must handle the videos and photos so quickly, as much as possible

2. Extensibility

   Extensibility is one of the most important things when we deal with project development. From the first steps, when the application is going to be released, we do not know the reaction of audience. Based on users' reaction the project is supposed to be changed or not. That is why the project has to be extensible, so a new features could be added further.

3. Energy Saving

   One of the priorities that has to be in focus is energy saving. The project has to be build the way it is going to use the mobile phone’s performance, battery and internet traffic properly.

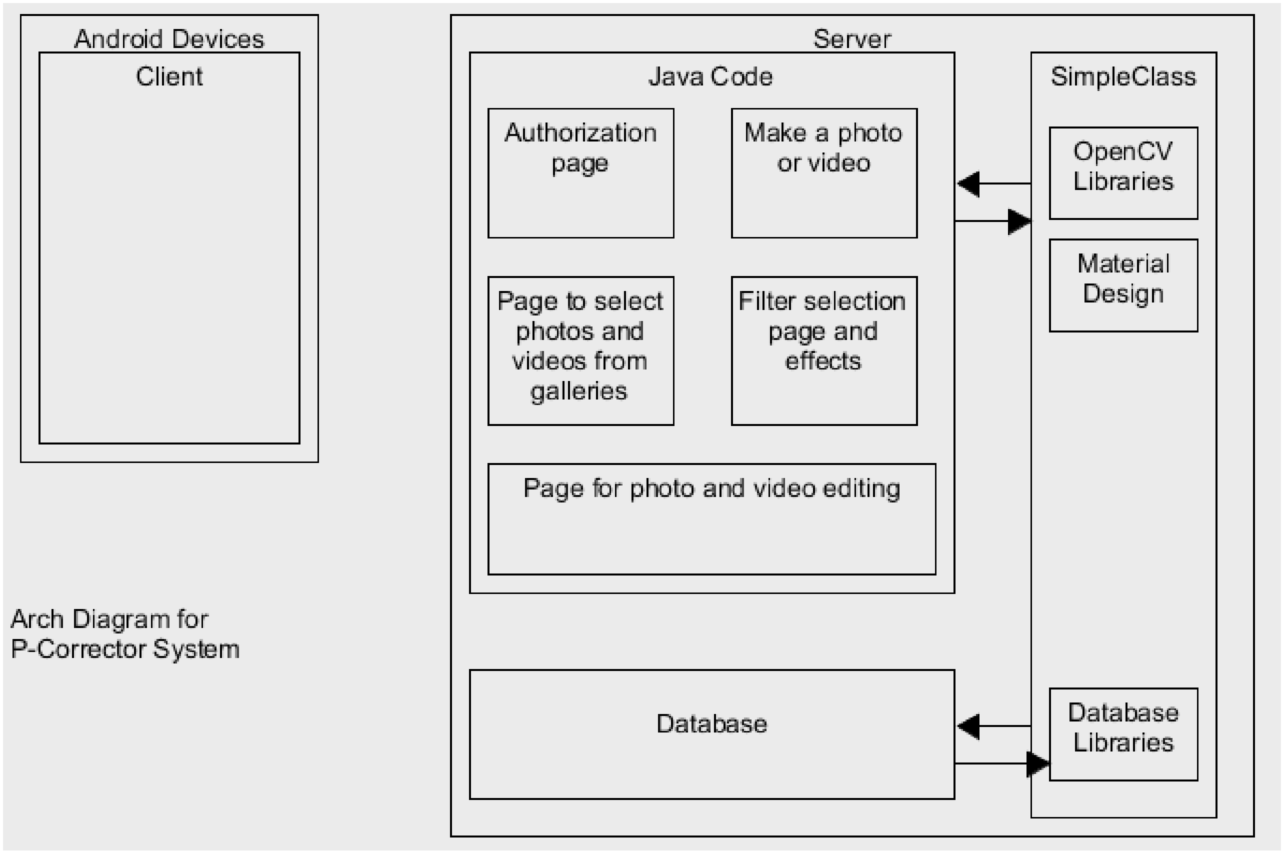
2 References

(If any)

3 Decomposition Description

MODULE

DECOMPOSITION


Android Devices
Client
Server
Java Code
Authorization page
Make a photo or video
Page to select photos and videos from galleries
Filter selection page and effects
Page for photo and video editing
SimpleClass
OpenCV Libraries
Material Design
Database
Database Libraries
Arch Diagram for P-Corrector System

THE PROJECT HAS AN INHERENT CLIENT-SERVER ARCHITECTURE, BUT THERE WILL BE LITTLE OR NO CLIENT CODE. THEREFORE, THE ARCHITECTURE OF THE SERVER IS THE MORE IMPORTANT ASPECT OF THE ARCHITECTURE. THIS DESIGN HAS TWO LAYERS. THE FIRST CONTAINS THE JAVA CODE AND MODULES. THE SECOND CONTAINS ONLY THE LIBRARIES MODULE THROUGH THE DATABASE LIBRARIES, DATABASE LIBRARIES, MATERIAL DESIGN LIBRARIES AND OPENCV LIBRARY.

3.1.1 Java code is consisted with 5 different pages. Each page takes charge of specific events and graphical process.

3.1.2 The libraries modules contains the classes and functions needed by the Java Code

<Repeat for as many subsystems that you have>

DATA DECOMPOSITION

3.1.3 OPENCV filters

```
package improctry2;
import org.opencv.core.Core;
import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.highgui.Highgui;
import org.opencv.imgproc.Imgproc;


public class ImProcTry2 {
   public static void main( String[] args )
   {
   try {
      int kernelSize = 9;
      System.loadLibrary( Core.NATIVE_LIBRARY_NAME );
      Mat source = Highgui.imread("grayscale.jpg",
      Highgui.CV_LOAD_IMAGE_GRAYSCALE);
      Mat destination = new Mat(source.rows(),source.cols(),source.type());
      Mat kernel = new Mat(kernelSize,kernelSize, CvType.CV_32F){
      {
         put(0,0,-1);
         put(0,1,0);
         put(0,2,1);

         put(1,0-2);
         put(1,1,0);
         put(1,2,2);

         put(2,0,-1);
         put(2,1,0);
         put(2,2,1);
      }
      };
      Imgproc.filter2D(source, destination, -1, kernel);
      Highgui.imwrite("output.jpg", destination);
      } catch (Exception e) {
         System.out.println("Error: " + e.getMessage());
      }
   }
}
```

3.1.4 Gabor filter in OpenCV

```
private Mat enhanceImg(Mat myImg){

      // prepare the output matrix for filters
    Mat gabor1 = new Mat (myImg.width(), myImg.height(), CvType.CV_8UC1);
    Mat gabor2 = new Mat (myImg.width(), myImg.height(), CvType.CV_8UC1);
    Mat gabor3 = new Mat (myImg.width(), myImg.height(), CvType.CV_8UC1);
    Mat gabor4 = new Mat (myImg.width(), myImg.height(), CvType.CV_8UC1);
    Mat enhanced = new Mat (myImg.width(), myImg.height(), CvType.CV_8UC1);

      //predefine parameters for Gabor kernel
    Size kSize = new Size(31,31);

    double theta1 = 0;
    double theta2 = 45;
    double theta3 = 90;
    double theta4 = 135;

    double lambda = 30;
    double sigma = 24;
    double gamma = 1;
    double psi =  0;

       // the filters kernel
    Mat kernel1 = Imgproc.getGaborKernel(kSize, sigma, theta1, lambda, gamma, psi,
CvType.CV_32F);
    Mat kernel2 = Imgproc.getGaborKernel(kSize, sigma, theta2, lambda, gamma, psi,
CvType.CV_32F);
    Mat kernel3 = Imgproc.getGaborKernel(kSize, sigma, theta3, lambda, gamma, psi,
CvType.CV_32F);
    Mat kernel4 = Imgproc.getGaborKernel(kSize, sigma, theta4, lambda, gamma, psi,
CvType.CV_32F);

      // apply filters on my image. The result is stored in gabor1...4
    Imgproc.filter2D(myImg, gabor1, -1, kernel1);
    Imgproc.filter2D(myImg, gabor2, -1, kernel2);
    Imgproc.filter2D(myImg, gabor3, -1, kernel3);
    Imgproc.filter2D(myImg, gabor4, -1, kernel4);

    //enhanced = gabor1+gabor2+gabor3+gabor4 - something like that

    return enhanced;
}
```

3.1.5 OPENCV filters

```
import org.opencv.core.Core;

import org.opencv.core.CvType;

import org.opencv.core.Mat;

import org.opencv.core.Size;
```

```java
import org.opencv.highgui.Highgui;
import org.opencv.imgproc.Imgproc;


public class Main {
   public static void main( String[] args ){

      try {
         System.loadLibrary( Core.NATIVE_LIBRARY_NAME );

         Mat source = Highgui.imread("digital_image_processing.jpg",
         Highgui.CV_LOAD_IMAGE_COLOR);

         Mat destination = new Mat(source.rows(),source.cols(),source.type());
         Imgproc.GaussianBlur(source, destination,new Size(45,45), 0);
         Highgui.imwrite("Gaussian45.jpg", destination);

      } catch (Exception e) {
         System.out.println("Error: " + e.getMessage());
      }
   }
}
```

3.1.6 <State/System 2> Description

4 Dependency Description

INTERMODULE DEPENDENCIES

INTERPROCESS DEPENDENCIES

DATA DEPENDENCIES

5 Interface Description

MODULE INTERFACE

5.1.1 <Module 1> Interface

5.1.2 <Module 2> Interface

PROCESS INTERFACE

5.1.3 <Process 1> Interface

5.1.4 <Process 2> Interface

6 Detailed Design

NOT REQUIRED <Java Docs to be used instead>

7 Design Rationale

DESIGN ISSUES

<ISSUE 1>

7.1.1 Description

7.1.2 Factors affecting Issue

7.1.3 Alternatives and their pros and cons

7.1.4 Resolution of Issue

<ISSUE 1>

7.1.5 Description

7.1.6 Factors affecting Issue

7.1.7 Alternatives and their pros and cons

7.1.8 Resolution of Issue

8 Traceability

No	Use Case/ Non-functional Description	Subsystem/Module/classes that handles it
1		
2		

FEEL FREE TO ADD APPENDICES AS NEEDED. UPDATE TOC BEFORE SUBMITTING