APPENDIX I

SOURCE CODE

```
import 'package:flutter/material.dart'; void main() => runApp(MyApp()); class
MyApp extends StatelessWidget
{ // This widget is the root of your application.@override
Widget build(BuildContext context) {return MaterialApp(
title: 'Car Damage Detector',theme: ThemeData( primarySwatch: Colors.blue,
),

home: MyHomePage(title: 'Car Damage Detector'), );

}

}

class MyHomePage extends StatefulWidget

{ MyHomePage({Key key, this.title}) : super(key: key);final String title;
@override _

MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage>

{
int _counter = 0;

void _incrementCounter()

{
```

```
setState(()

{ _ counter++;
})

}

@override

Widget build(BuildContext context)

{

return Scaffold

( appBar: AppBar

( title: Text(widget.title), ),body:
Center( child:
Column( mainAxisAlignment: MainAxisAlignment.center, children:

<Widget>[ Text( 'You have pushed the button this many times:', ),Text(
'$_counter', style: Theme.of(context).textTheme.display1, ),
], ), ),

floatingActionButton:

FloatingActionButton

( onPressed: _incrementCounter,tooltip: 'Detect Car Damage'
, child:

Icon(Icons.camera_alt), )
, // This trailing comma makes auto-formatting nicer for build methods. );

}

}


// Class for the car damage detectionclass CarDamageDetector
{

// Constructor CarDamageDetector()
{}’

// Method to detect

car damage void detectCarDamage()

{

}

}
```

PYTHON:

```
// Import librariesimport cv2
import numpy as np

// Initialize global variables MIN_MATCH_COUNT = 10
// Create SURF detector detector = cv2.xfeatures2d.SURF_create()

// Load images
```

```
img1 = cv2.imread('car_before.jpg', 0)img2 = cv2.imread('car_after.jpg', 0)
// Find the keypoints and descriptors kp1

, des1 = detector.detectAndCompute(img1, None) kp2,des2 =
detector.detectAndCompute(img2, None)
// Create BFMatcher

object bf = cv2.BFMatcher(cv2.NORM_L2, crossCheck= True)



// Match descriptors

matches = bf.match(des1, des2)

// Calculate matches

good_matches = [] for m in matches:

if m.distance < 0.7 * n.max_distance: good_matches.append(m)

// Check if there are enough matches

if len(good_matches) > MIN_MATCH_COUNT:src_pts =
np.float32([kp1[m.queryIdx].pt
for m in good_matches]).reshape(-1,1,2)
dst_pts = np.float32([kp2[m.trainIdx]

.pt for m in good_matches]).reshape(-1,1,2) M,
mask = cv2.

findHomography(src_pts, dst_pts, cv2.RANSAC, 5.0)matches_mask =
mask.ravel().
```

```
tolist()

h,w,d = img1.shape

pts = np.float32([[[0,0]], [[0,h-1]], [[w-1,h-1]], [[w-1,0]]]).

reshape(-1,1,2)

dst = cv2.perspectiveTransform(pts, M)
```

APPLICATION CODE:

```
import android.app.Activity; import android.content.Intent;
import android.graphics.Bitmap;

import android.graphics.BitmapFactory; import
android.graphics.drawable.Drawable;import android.os.Bundle;
import android.view.View; import android.widget.Button;
import android.widget.ImageView; import android.widget.TextView; import
androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity; import
com.google.android.gms.tasks.OnSuccessListener;import
com.google.firebase.ml.vision.FirebaseVision;
import com.google.firebase.ml.vision.common.FirebaseVisionImage;import
com.google.firebase.ml.vision.label.FirebaseVisionLabel;
import com.google.firebase.ml.vision.label.FirebaseVisionLabelDetector;
import java.util.List;
public class CarDamageDetector extends AppCompatActivity

{


private static final int REQUEST_CAPTURE_IMAGE = 1;private Button
```

```
btnCapture;
private ImageView imgCaptured;private TextView
txtResult; @Override
protected void onCreate(Bundle savedInstanceState)

{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_car_damage_detector);btnCapture =
findViewById(R.id.btnCapture);
imgCaptured = findViewById(R.id.imgCaptured);txtResult =
findViewById(R.id.txtResult); btnCapture.
setOnClickListener(new View.OnClickListener()

{

@Override

public void onClick(View v)

{

//open camera

Intent intent = new
Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
startActivityForResult(intent, REQUEST_CAPTURE_IMAGE);
}

}

);

}
```

```
@Override

protected void onActivityResult(int requestCode,int resultCode,
@Nullable Intent data

)

{ super.onActivityResult(requestCode, resultCode, data);

if (requestCode == REQUEST_CAPTURE_IMAGE && resultCode ==
Activity.RESULT_OK)

{

//process image

Bundle extras = data.getExtras();Bitmap bitmap = (Bitmap) extras.get("data");
//set image

imgCaptured.setImageBitmap(bitmap);

//get image labels

FirebaseVisionImage image = FirebaseVisionImage.fromBitmap(bitmap);


FirebaseVisionLabelDetector detector =
FirebaseVision.getInstance().getVisionLabelDetector();
detector.detectInImage(image)
.addOnSuccessListener(new OnSuccessListener
<List<FirebaseVisionLabel>>()

{

@Override


for (FirebaseVisionLabel label : firebaseVisionLabels)

\{

public void onSuccess(List<FirebaseVisionLabel>firebaseVisionLabels)
{

String result = "";

|| label.getLabel().contains("crack")

|| label.getLabel(). contains("damage"))
{

result = "Car is Damaged";

Drawable drawable = getResources().getDrawable(R.drawable.warning);
imgCaptured.setImageDrawable(drawable);

if (label.getLabel().contains("scratch")

|| label.getLabel().contains("dent")

break;

}
```

```
 else

{

result = "Car is in Good Condition";

Bitmap bitmap = BitmapFactory.decodeResource(getResources(),
R.drawable.success);imgCaptured.setImageBitmap(bitmap);
}

}

txtResult.setText(result);

}

}

);

}

}
}
```

DART:

```
import 'dart:math';
void main()
 {
 int damage = 0;
int totalDamage = 0;
 // Check the left side
 if (checkLeftSide()
)
 {
 print("Left side of car has some damage");
damage = damage + 1;
 } // Check the right side



 if (checkRightSide())
 {
 print("Right side of car has some damage");
damage = damage + 1;
 }
 // Check the front
 if (checkFront())
{
 print("Front of car has some damage");
damage = damage + 1;
 }
 // Check the rear
 if (checkRear())
 {
 print("Rear of car has some damage");
damage = damage + 1;
 }
 totalDamage = totalDamage + damage;
// Check the windows
 if (checkWindows())
 {
 print("Windows of car are damaged")

; damage = damage + 1;
}
```

```
totalDamage = totalDamage + damage;
 // Check the wheels
if (checkWheels())
 {
print("Wheels of car are damaged");
 damage = damage + 1;
}
totalDamage = totalDamage + damage;
// Check the headlights
if (checkHeadlights())
 {
print("Headlights of car are damaged");
 damage = damage + 1;
 }
 totalDamage = totalDamage + damage;
 // Check the paint
 if (checkPaint())
 {


 print("Paint of car is damaged");
damage = damage + 1;
 }
 totalDamage = totalDamage + damage;
 // Check the interior

 if (checkInterior())
 {
print("Interior of car is damaged");
damage = damage + 1;
 }
 totalDamage = totalDamage + damage;
// Check the engine
 if (checkEngine())
 {
 print("Engine of car is damaged");
damage = damage + 1;
 }
 totalDamage = totalDamage + damage;
// Print out the total damage
print("Total damage of car is $totalDamage");
}
// Checks the left side of the ca
r bool checkLeftSide()
{
 // Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
 // If the number is greater than 5, there is damage if (randomNumber > 5)
 {
return true;
 }
 return false;
 }
 // Checks the right side of the car
 bool checkRightSide()
 {
 // Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
 // If the number is greater than 5, there is damage
if (randomNumber > 5)
{
return true;
 }
 return false;
 }
 // Checks the front of the car
 bool checkFront()
 {
// Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
// If the number is greater than 5, there is damage
if (randomNumber > 5)
 {
 return true;
 }
 return false;
 }
 // Checks the rear of the car
 bool checkRear()
{
// Generate a random number between 0 and 10
 int randomNumber = Random().nextInt(10);
 // If the number is greater than 5, there is damage
if (randomNumber > 5)
 {
 return true;
 }
 return false;
 }
 // Checks the windows of the car
bool checkWindows()
 {
 // Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
 // If the number is greater than 7, there is damage
if (randomNumber > 7)
 {
 return true;
 }
 return false;
 }
 // Checks the wheels of the car
bool checkWheels()
{
// Generate a random number between 0 and 10
 int randomNumber = Random().nextInt(10);
 // If the number is greater than 8, there is damage
if (randomNumber > 8)
{
 return true;
 }
 return false;
 }

 // Checks the headlights of the car
bool checkHeadlights()
 {
 // Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
 // If the number is greater than 6, there is damage
if (randomNumber > 6)
 {
return true;
 }
 return false;
 }
//
 Checks the paint of the car
 bool checkPaint()
 {
 // Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
// If the number is greater than 7, there is damage
```

```
if (randomNumber > 7)
{
 return true;
 }
 return false;
 }

 // Checks the interior of the car
bool checkInterior()
 {
// Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
// If the number is greater than 6, there is damage
if (randomNumber > 6)
 {
 return true;
 }
 return false;
 }
 // Checks the engine of the car
 bool checkEngine()
 {

//
 Generate a random number between 0 and 10
int randomNumber = Random().nextInt(10);
// If the number is greater than 8, there is damage
if (randomNumber > 8)
 {
return true;
 }
 return false;
 }
```