

Acknowledgement

We are pleased to present “**Currency Detection**” project and take this opportunity to express our profound gratitude to all those people who helped us in completion of this project.

We thank our college for providing us with excellent facilities that helped us to complete and present this project. We would also like to thank the staff members and lab assistants for permitting us to use computers in the lab as and when required.

We express our deepest gratitude towards our project guide for his/her valuable and timely advice during the various phases in our project. We would also like to thank him/her for providing us with all proper facilities and support as the project co-coordinator. We would like to thank him/her for support, patience and faith in our capabilities and for giving us flexibility in terms of working and reporting schedules.

We would like to thank all our friends for their smiles and friendship making the college life enjoyable and memorable and family members who always stood beside us and provided the utmost important moral support. Finally, we would like to thank everyone who has helped us directly or indirectly in our project.

TABLE OF CONTENT

Sr. No.	Contents
1	<u>PROJECT OVERVIEW</u>
1.1	Introduction
1.2	Scope and Objective
1.3	Modules and its Description
1.4	Existing System & Proposed System
2	<u>PROJECT ANALYSIS</u>
2.1	Gantt Chart
3	<u>PROJECT LIFECYCLE</u>
3.1	Project Lifecycle Details
4	<u>PROJECT DESIGN</u>
4.1	E-R Diagram
4.2	Use Case Diagram
4.3	Sequence Diagram
4.4	Activity Diagram
4.5	Data Flow Diagram
4.6	System Architecture
5	<u>SNAPSHOTS</u>
5.1	Project Snapshots
6	<u>PROJECT IMPLEMENTATION</u>
6.1	Project Implementation Technology
6.2	Feasibility Report
7	<u>CODING</u>
7.1	Project Coding
8	<u>TESTING</u>
8.1	Testing
8.2	Levels of Testing
8.3	Test Cases
9	<u>ADVANTAGES & LIMITATIONS</u>
9.1	Advantages
9.2	Limitations
9.3	Features

10	<u>CONCLUSION</u>
10.1	Project Conclusion
11	<u>BIBLIOGRAPHY</u>
11.1	Website Links

PROJECT OVERVIEW

Introduction

Main problems faced by people with visual disabilities is the inability to recognize the paper currencies due to the similarity of paper texture and size between the different categories. These people face a lot of difficulty in their monetary transactions. This application can help visually impaired to recognize money. In this application blind people can speak and give command to open camera and camera will click the picture of note and tell the user by speech medium how much rupee note it is. This system uses speech to text to convert command given by blind people, Speech recognition is the interdisciplinary subfield of computational linguistics that develops methodologies and technologies that enables the recognition and translation of spoken language into text. For the result purpose this system has text to speech concept which helps to read the value of note and then it converts the text value into speech. Android allows to convert your text into voice. Not only you can convert it but it also allows you to speak text in variety of different languages.

Scope and Objective

It became too difficult for blind and visually impaired people to recognize money. Individuals with low vision need contrast colours and large fonts to identify currency note denomination. Individuals with low vision need contrast colours and large fonts to identify currency note denomination. This application will help them to recognize the value of the currency. This system uses speech to text to convert command given by blind people, Speech recognition is the interdisciplinary subfield of computational linguistics that develops methodologies and technologies that enables the recognition and translation of spoken language into text. For the result purpose this system has text to speech concept which helps to read the value of note and then it converts the text value into speech. Android allows to convert your text into voice.

Modules and their Description

The system comprises of 1 major module with their sub-modules as follows:

1. User:

- **Currency Detection:** Azure custom vision API using Machine learning classification technique used to detect currency based on images or paper using mobile camera.
- **Dial a Contact:** User can also give command to dial a call.

Existing System & Proposed System

❖ **Problem with current scenario**

- It become too difficult for blind and visually impaired people to recognize money.
- Blind people take help of other while recognizing money, and also while buying things.
- Individuals with low vision need contrast colours and large fonts to identity currency note denomination.

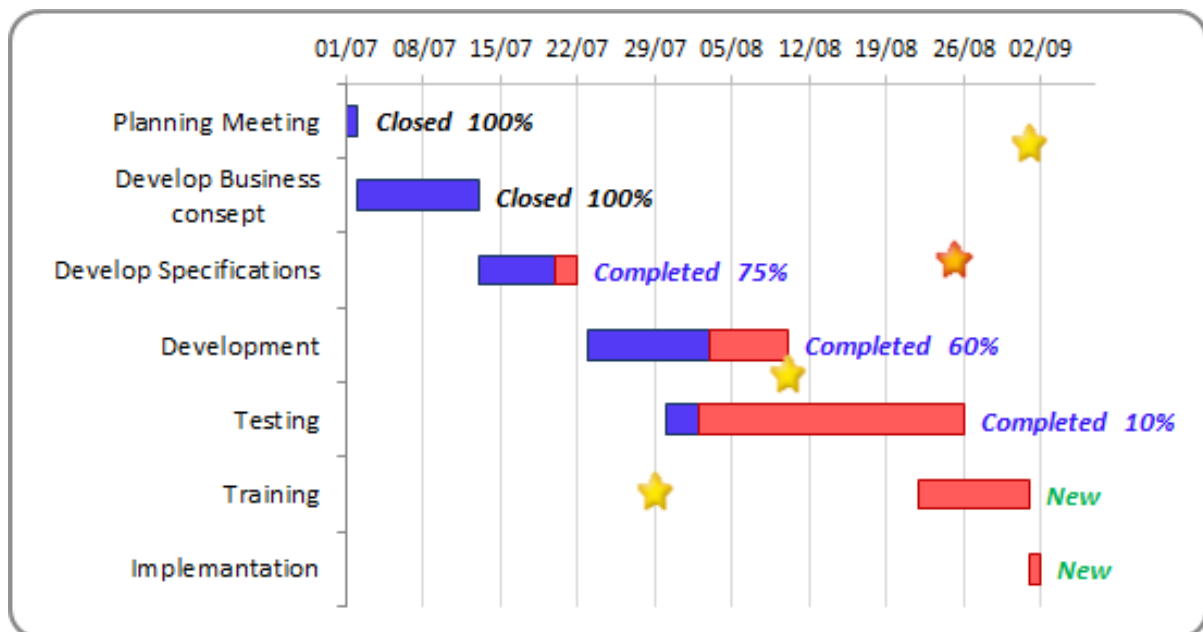
Drawbacks of the existing system

- Maintenance of the system is very difficult.
- There is a possibility for getting inaccurate results.
- User friendliness is very less.
- It consumes more time for processing the task.

PROPOSED SYSTEM

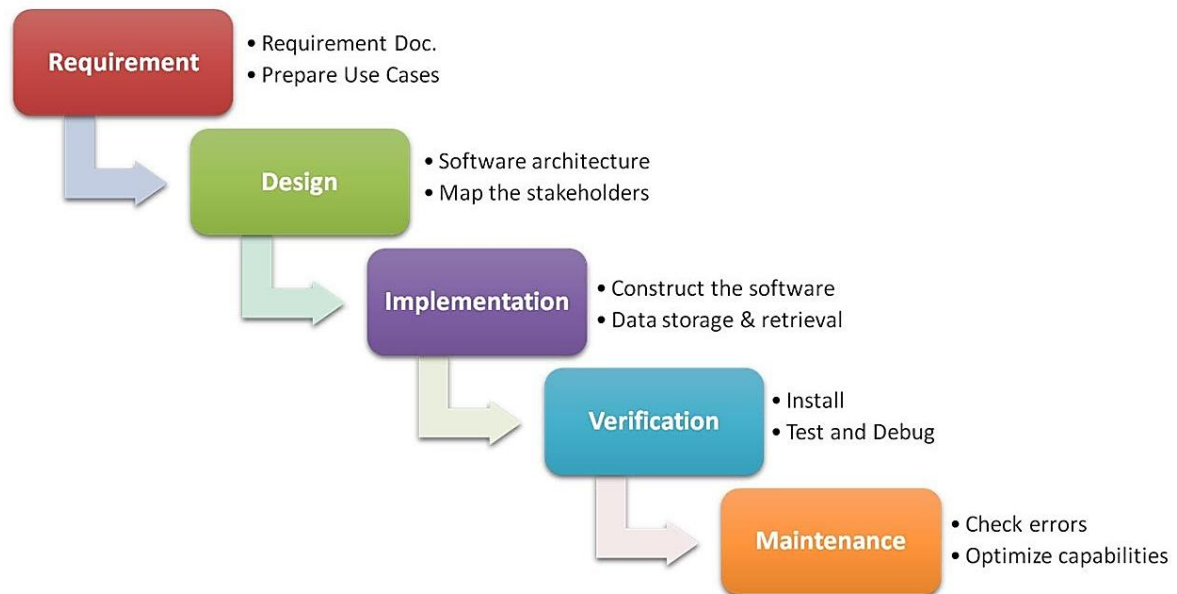
- Considering the anomalies in the existing system computerization of the whole activity is being suggested after initial analysis.
- The android application is developed using Android Studio with JAVA as a programming language.
- Proposed system is accessed by one entity namely, User.
- User can perform task such as blind people can speak and give command to open camera and camera will click the picture of note and tell the user by speech medium how much rupee note it is.
- This system uses speech to text to convert command given by blind people, Speech recognition is the interdisciplinary subfield of computational linguistics that develops methodologies and technologies that enables the recognition and translation of spoken language into text.
- For the result purpose this system has text to speech concept which helps to read the value of note and then it converts the text value into speech. Android allows to convert your text into voice. Not only you can convert it but it also allows you to speak text in variety of different languages.

Gantt Chart



Project Lifecycle Details

Waterfall Model

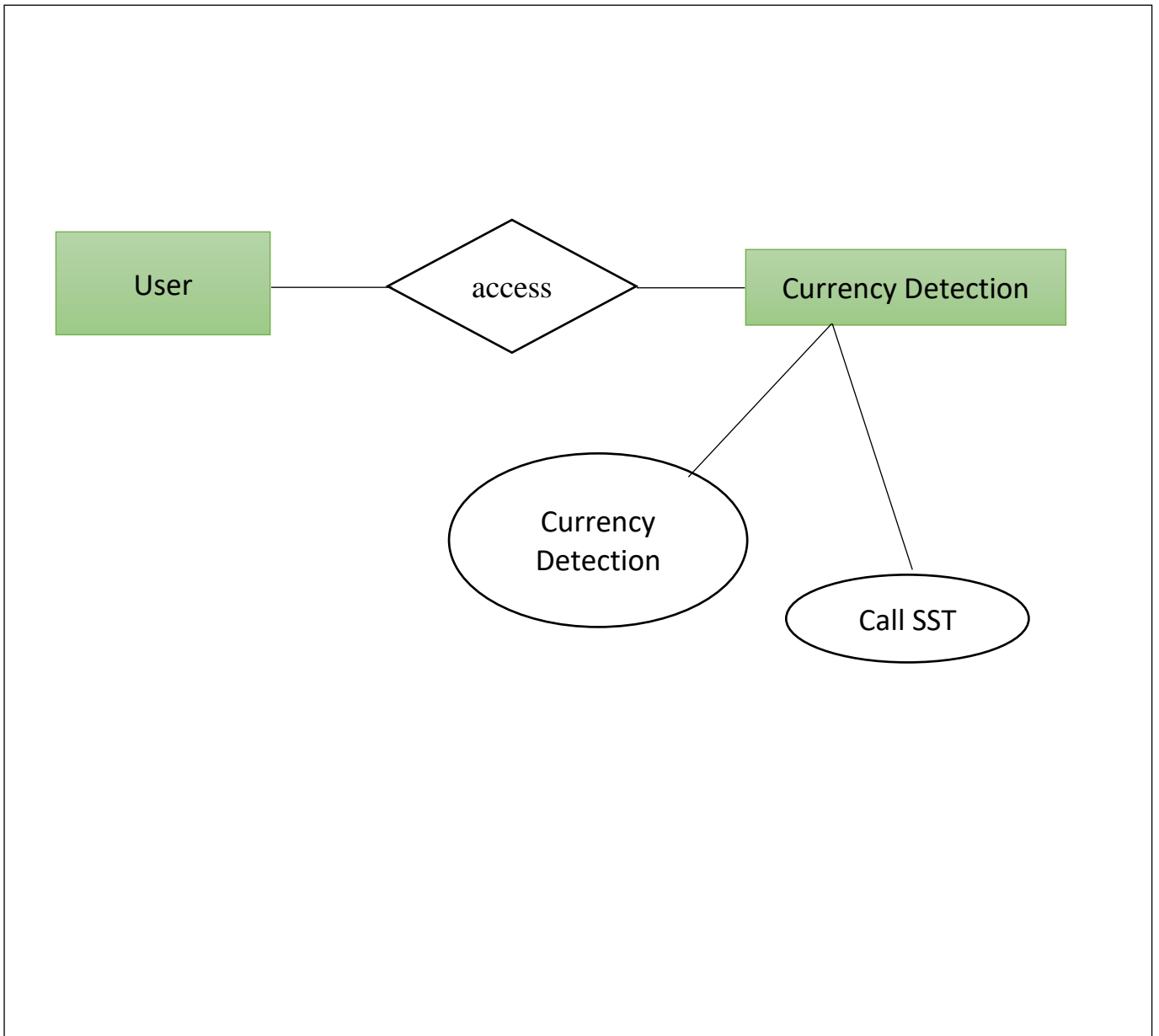


Description

The waterfall Model is a linear sequential flow. In which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation. This means that any phase in the development process begins only if the previous phase is complete. The waterfall approach does not define the process to go back to the previous phase to handle changes in requirement. The waterfall approach is the earliest approach that was used for software development.

PROJECT DESIGN

E-R Diagram



Use Case Diagram

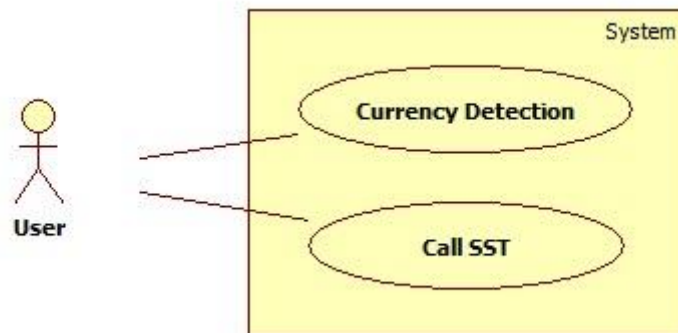


Fig. Use Case Diagram of User

Sequence Diagram

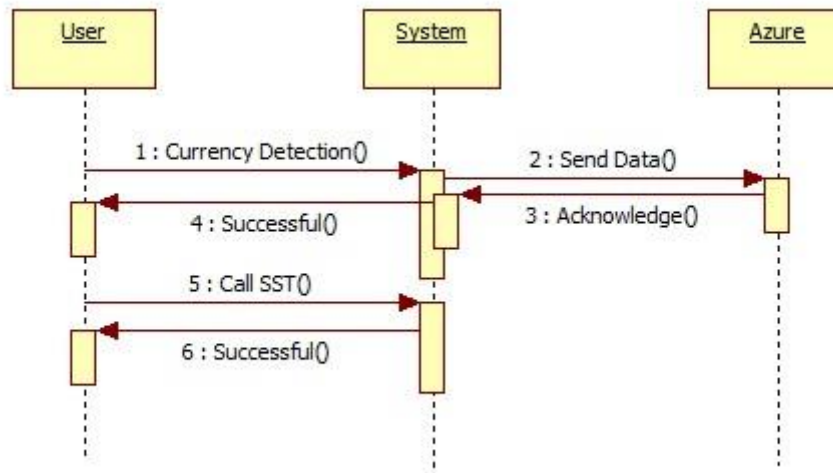


Fig. Sequence Diagram of User

Activity Diagram

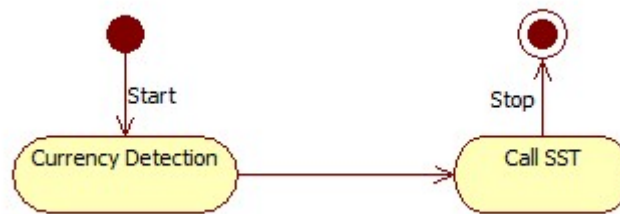


Fig. Activity Diagram of User

Data Flow Diagram (DFD's)

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD's is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The top-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

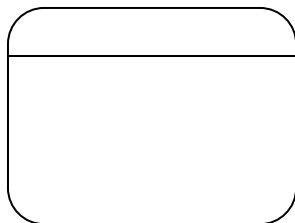
Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical form, this lead to the modular design.

A DFD is also known as a “bubble Chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

DFD SYMBOLS:

In the DFD, there are four symbols

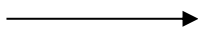
1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data



Process that transforms data flow.



Source or Destination of data



Data flow



Data Store

CONSTRUCTING A DFD:

Several rules of thumb are used in drawing DFD's:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numbered.
4. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each word capitalized

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

Questionnaires should contain all the data elements that flow in and out.

Missing interfaces redundancies and like is then accounted for often through interviews.

SAILENT FEATURES OF DFD's

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the data flows take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

TYPES OF DATA FLOW DIAGRAMS

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

CURRENT PHYSICAL:

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly, data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

CURRENT LOGICAL:

The physical aspects at the system are removed as much as possible so that the current system is reduced to its essence to the data and the processors that transform them regardless of actual physical form.

NEW LOGICAL:

This is exactly like a current logical model if the user were completely happy with the user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

NEW PHYSICAL:

The new physical represents only the physical implementation of the new system.

RULES GOVERNING THE DFD'S

PROCESS

- 1) No process can have only outputs.
- 2) No process can have only inputs. If an object has only inputs than it must be a sink.
- 3) A process has a verb phrase label.

DATA STORE

- 1) Data cannot move directly from one data store to another data store, a process must move data.
- 2) Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
- 3) A data store has a noun phrase label.

SOURCE OR SINK

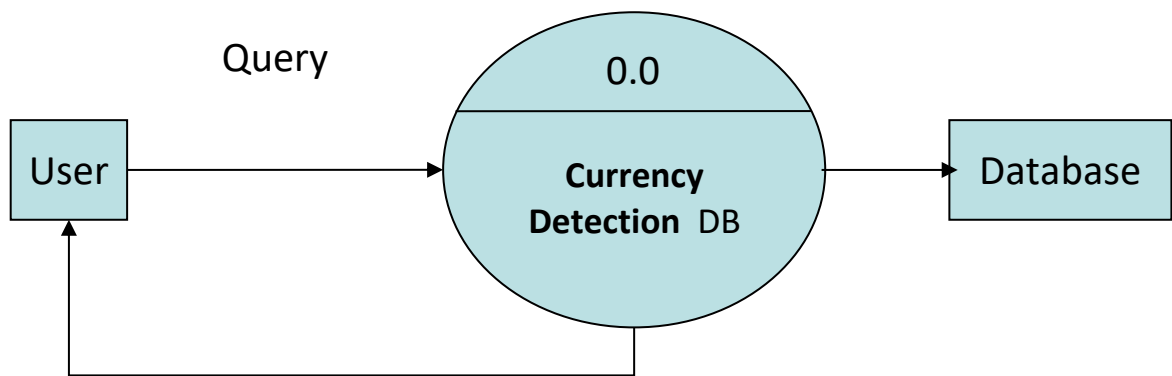
The origin and /or destination of data.

- 1) Data cannot move direly from a source to sink it must be moved by a process
- 2) A source and /or sink has a noun phrase land

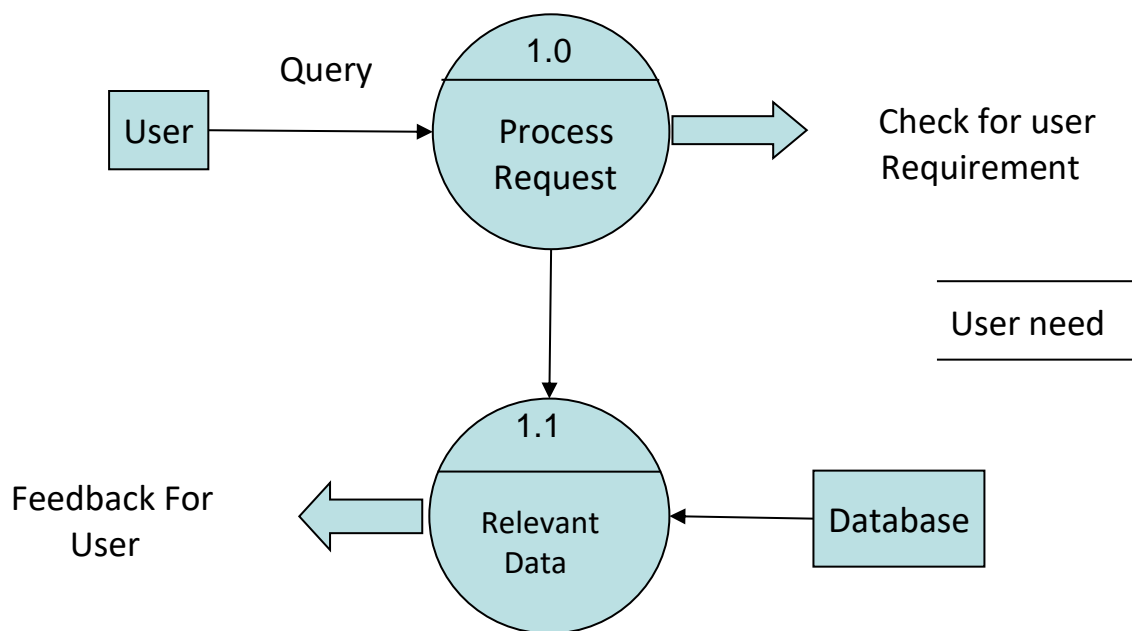
DATA FLOW

- 1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later it usually indicated however by two separate arrows since these happen at different type.
- 2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
- 3) A data flow cannot go directly back to the same process it leads. There must be at least one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
- 4) A Data flow to a data store means update (delete or change).
- 5) A data Flow from a data store means retrieve or use.

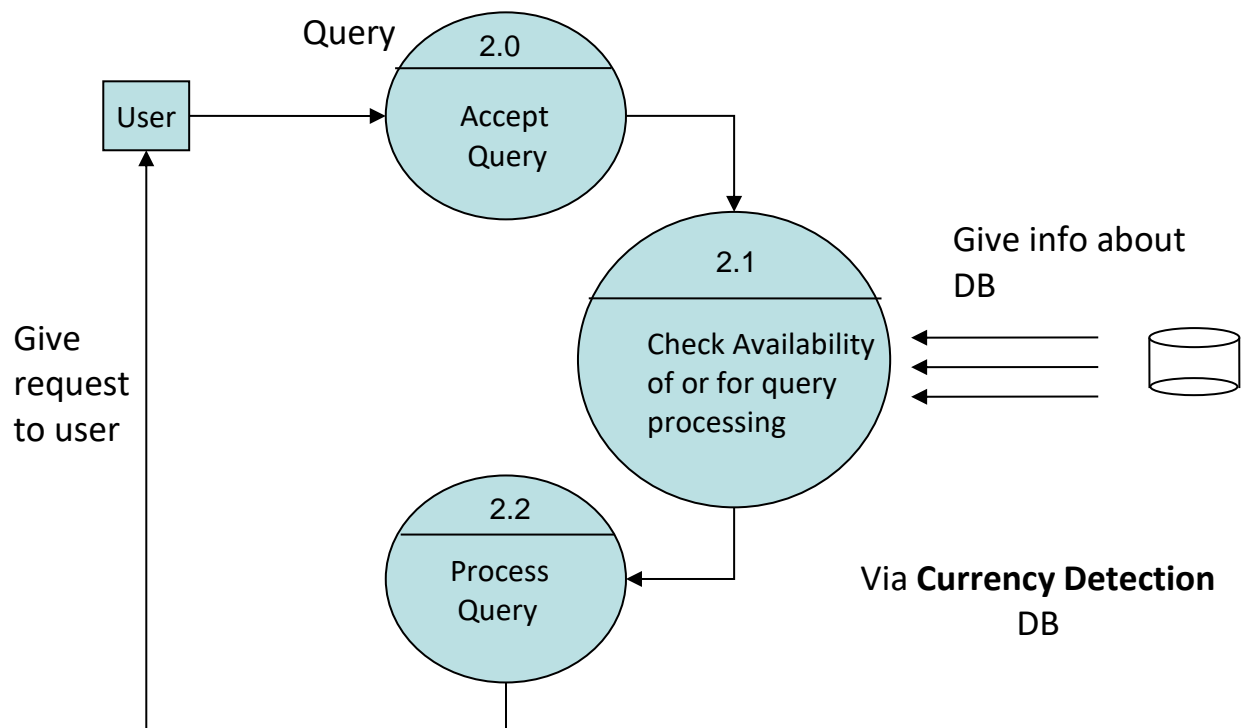
Data Flow Diagrams (DFD's)



DATABASE DETAIL

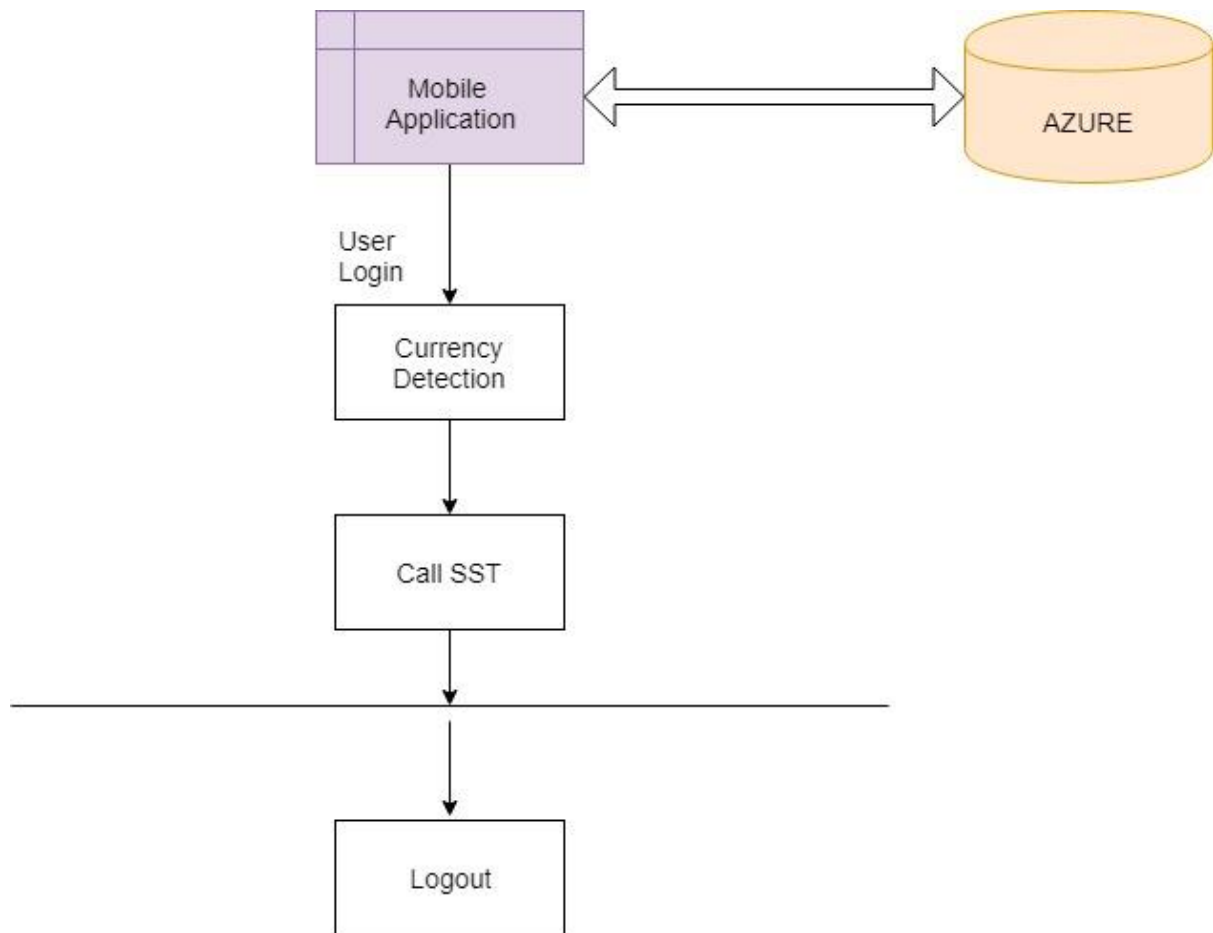


LEVEL 1 DFD



LEVEL 2 DFD: PREDICTION

System Architecture



Snapshots

7:14



7:18



Blind assistant

RESTART



INDIAN CURRENCY DETECTION

CLICK TO SELECT IMAGE

SCAN CURRENCY



PROJECT IMPLEMENTATION

Project Implementation Technology

The Project application is loaded in Android Studio. We used Android Studio for Design and coding of project. Created and maintained all databases into SQL Server, in that we create tables, write query for store data or record of project.

❖ **Hardware Requirement:**

1. Laptop or PC

- i3 Processor Based Computer or higher
- 1GB RAM
- 5 GB Hard Disk

2. Android Phone or Tablet

- 1.2 Quad core Processor or higher
- 1 GB RAM

❖ **Software Requirement:**

1. Laptop or PC

- Windows 7 or higher.
- Java
- Android Studio

2. Android Phone or Tablet

- Android v5.0 or Higher

Introduction to Android

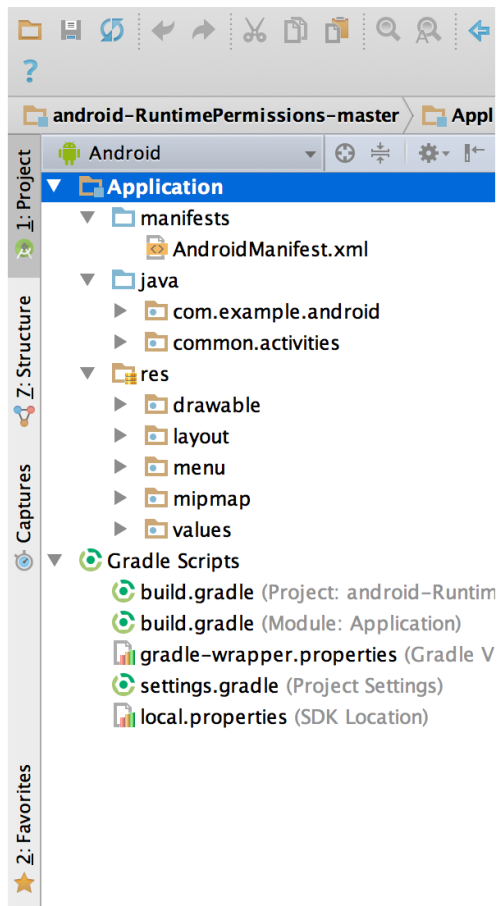
Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Instant Run to push changes to your running app without building a new APK
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine.

Project Structure

Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

- Android app modules
- Library modules
- Google App Engine modules



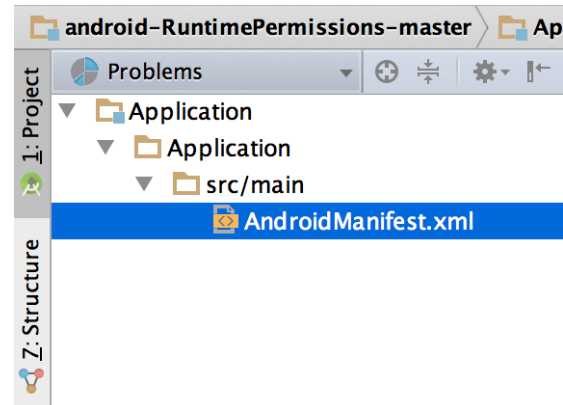
By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.

All the build files are visible at the top level under **Gradle Scripts** and each app module contains the following folders:

- **manifests:** Contains the AndroidManifest.xml file.
- **java:** Contains the Java source code files, including JUnit test code.
- **res:** Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select **Project** from the **Project** dropdown (in figure 1, it's showing as **Android**).

You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the **Problems** view of your project displays links to the source files containing any recognized coding and

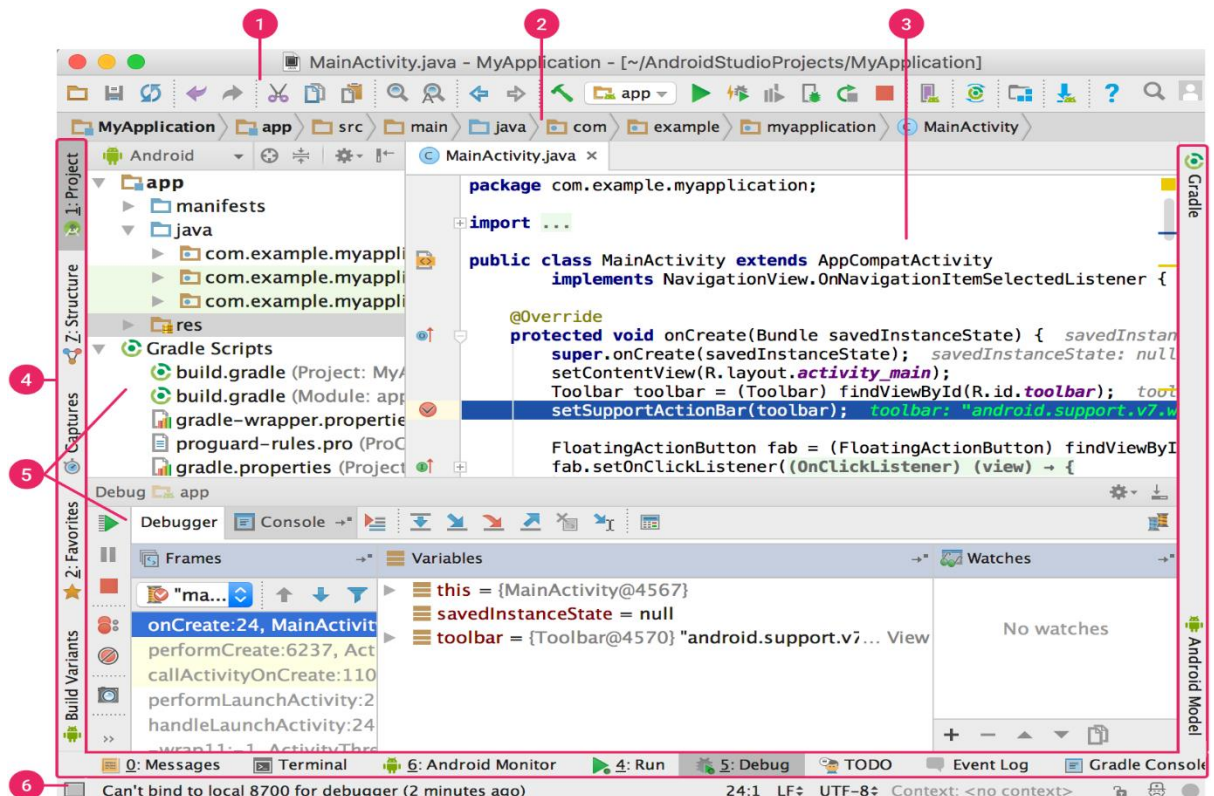


syntax errors, such as a missing XML element closing tag in a layout file.

The User Interface

1. The **toolbar** lets you carry out a wide range of actions, including running your app and launching Android tools.
2. The **navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the **Project** window.
3. The **editor window** is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.
4. The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.

5. The **tool windows** give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.
6. The **status bar** displays the status of your project and the IDE itself, as well as any warnings or messages.




You can organize the main window to give yourself more screen space by hiding or moving toolbars and tool windows. You can also use keyboard shortcuts to access most IDE features.

At any time, you can search across your source code, databases, actions, elements of the user interface, and so on, by double-pressing the Shift key, or clicking the magnifying glass in the upper right-hand corner of the Android Studio window. This can be very useful if, for example, you are trying to locate a particular IDE action that you have forgotten how to trigger.

Tool Windows

Instead of using preset perspectives, Android Studio follows your context and automatically brings up relevant tool windows as you work. By default, the most commonly used tool windows are pinned to the tool window bar at the edges of the application window.

- To expand or collapse a tool window, click the tool's name in the tool window bar. You can also drag, pin, unpin, attach, and detach tool windows.
- To return to the current default tool window layout, click **Window > Restore Default Layout** or customize your default layout by clicking **Window > Store Current Layout as Default**.
- To show or hide the entire tool window bar, click the window icon  in the bottom left-hand corner of the Android Studio window.
- To locate a specific tool window, hover over the window icon and select the tool window from the menu.

Navigation

Here are some tips to help you move around Android Studio.

- Switch between your recently accessed files using the *Recent Files* action. Press **Control+E** (**Command+E** on a Mac) to bring up the Recent Files action. By default, the last accessed file is selected. You can also access any tool window through the left column in this action.
- View the structure of the current file using the *File Structure* action. Bring up the File Structure action by pressing **Control+F12** (**Command+F12** on a Mac). Using this action, you can quickly navigate to any part of your current file.

- Search for and navigate to a specific class in your project using the *Navigate to Class* action. Bring up the action by pressing **Control+N**(**Command+O** on a Mac). Navigate to Class supports sophisticated expressions, including camel humps, paths, line navigate to, middle name matching, and many more. If you call it twice in a row, it shows you the results out of the project classes.
- Navigate to a file or folder using the *Navigate to File* action. Bring up the Navigate to File action by pressing **Control+Shift+N** (**Command+Shift+O** on a Mac). To search for folders rather than files, add a / at the end of your expression.
- Navigate to a method or field by name using the *Navigate to Symbol* action. Bring up the Navigate to Symbol action by pressing **Control+Shift+Alt+N**(**Command+Shift+Alt+O** on a Mac).
- Find all the pieces of code referencing the class, method, field, parameter, or statement at the current cursor position by pressing **Alt+F7**

Gradle Build System

Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android plugin for Gradle. This build system runs as an integrated tool from the Android Studio menu, and independently from the command line. You can use the features of the build system to do the following:

- Customize, configure, and extend the build process.
- Create multiple APKs for your app, with different features using the same project and modules.
- Reuse code and resources across source sets.

By employing the flexibility of Gradle, you can achieve all of this without modifying your app's core source files. Android Studio build files are named `build.gradle`. They are plain text files that use Groovy syntax to configure the build with elements provided by the Android plugin for Gradle. Each project has one top-level build file for the entire project and separate module-level build files for each module. When you import an existing project, Android Studio automatically generates the necessary build files.

Multiple APK Support

Multiple APK support allows you to efficiently create multiple APKs based on screen density or ABI. For example, you can create separate APKs of an app for the `hdpi` and `mdpi` screen densities, while still considering them a single variant and allowing them to share test APK, `javac`, `dx`, and ProGuard settings.

Debug and Profile Tools

Android Studio assists you in debugging and improving the performance of your code, including inline debugging and performance analysis tools.

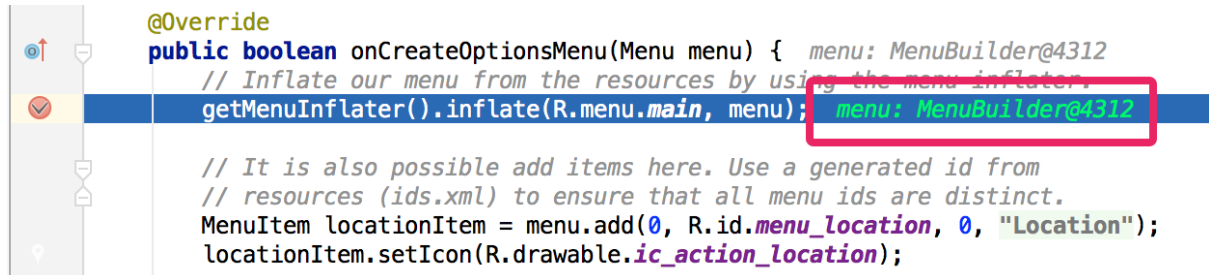
Inline debugging

Use inline debugging to enhance your code walk-throughs in the debugger view with inline verification of references, expressions, and variable values.

Inline debug information includes:

- Inline variable values
- Referring objects that reference a selected object
- Method return values

- Lambda and operator expressions
- Tooltip values



The screenshot shows a code editor with a Java method `onCreateOptionsMenu`. A tooltip is visible over the `menu` parameter, displaying `menu: MenuBuilder@4312`. The code includes comments about inflating the menu and adding items. The tooltip is highlighted with a red box.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate our menu from the resources by using the menu inflater.
    getMenuInflater().inflate(R.menu.main, menu);

    // It is also possible add items here. Use a generated id from
    // resources (ids.xml) to ensure that all menu ids are distinct.
    MenuItem locationItem = menu.add(0, R.id.menu_location, 0, "Location");
    locationItem.setIcon(R.drawable.ic_action_location);
}
```

Performance monitors

Android Studio provides performance monitors so you can more easily track your app's memory and CPU usage, find deallocated objects, locate memory leaks, optimize graphics performance, and analyze network requests. With your app running on a device or emulator, open the **Android Monitor** tool window, and then click the **Monitors** tab.

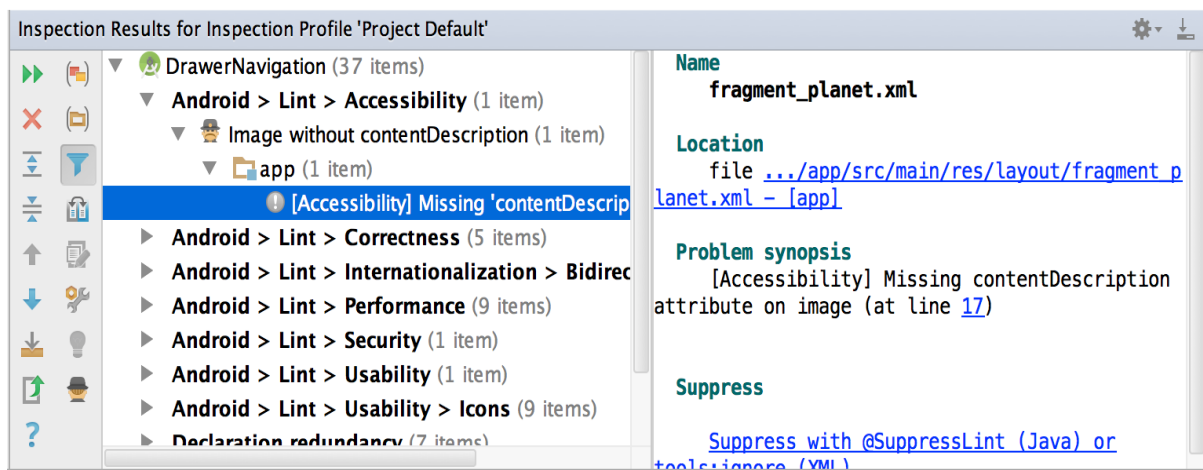
Allocation tracker

Android Studio allows you to track memory allocation as it monitors memory use. Tracking memory allocation allows you to monitor where objects are being allocated when you perform certain actions. Knowing these allocations enables you to optimize your app's performance and memory use by adjusting the method calls related to those actions.

Code inspections

Whenever you compile your program, Android Studio automatically runs configured Lint and other IDE inspections to help you easily identify and correct problems with the structural quality of your code.

The Lint tool checks your Android project source files for potential bugs and optimization improvements for correctness, security, performance, usability, accessibility, and internationalization.



Coding

Autofittextview.java

```
package com.website.currencydetectandvoicecommandindia;

import android.content.Context;
import android.util.AttributeSet;
import android.view.TextureView;

public class AutoFitTextureView extends TextureView {

    private int mRatioWidth = 0;
    private int mRatioHeight = 0;

    public AutoFitTextureView(Context context) {
        this(context, null);
    }

    public AutoFitTextureView(Context context, AttributeSet attrs) {
        this(context, attrs, 0);
    }

    public AutoFitTextureView(Context context, AttributeSet attrs, int defStyle) {
        super(context, attrs, defStyle);
    }

    /**
     * Sets the aspect ratio for this view. The size of the view will be measured based on the
     * ratio
     * calculated from the parameters. Note that the actual sizes of parameters don't matter, that
     * is, calling setAspectRatio(2, 3) and setAspectRatio(4, 6) make the same result.
     *
     * @param width Relative horizontal size
     * @param height Relative vertical size
     */
    public void setAspectRatio(int width, int height) {
        if (width < 0 || height < 0) {
            throw new IllegalArgumentException("Size cannot be negative.");
        }
        mRatioWidth = width;
        mRatioHeight = height;
        requestLayout();
    }

    @Override
    protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
        super.onMeasure(widthMeasureSpec, heightMeasureSpec);
        int width = MeasureSpec.getSize(widthMeasureSpec);
        int height = MeasureSpec.getSize(heightMeasureSpec);
        if (0 == mRatioWidth || 0 == mRatioHeight) {
```

```

        setMeasuredDimension(width, height);
    } else {
        if (width < height * mRatioWidth / mRatioHeight) {
            setMeasuredDimension(width, width * mRatioHeight / mRatioWidth);
        } else {
            setMeasuredDimension(height * mRatioWidth / mRatioHeight, height);
        }
    }
}
}
}

```

Callactivity.java

```
package com.website.currencydetectandvoicecommandindia;
```

```

import android.Manifest;
import android.app.Dialog;
import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.net.ConnectivityManager;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.os.Environment;
import android.provider.ContactsContract;
import android.provider.MediaStore;
import android.speech.RecognitionListener;
import android.speech.RecognizerIntent;
import android.speech.SpeechRecognizer;
import android.speech.tts.TextToSpeech;
import android.speech.tts.UtteranceProgressListener;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

```

```
import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import com.airbnb.lottie.LottieAnimationView;

import org.apache.commons.net.ftp.FTP;
import org.apache.commons.net.ftp.FTPClient;
import org.apache.commons.net.io.CopyStreamAdapter;
import org.json.JSONArray;
import org.json.JSONObject;
```

```
import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.SocketException;
import java.net.URL;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.Locale;
import java.util.Objects;
```

```
public class CallActivity extends AppCompatActivity implements RecognitionListener,
TextToSpeech.OnInitListener {
```

```
    private static final String SPEAK_NOW = "Speak now";
    private static final CharSequence CALL_CMD = "call";
    public static final String IMAGE_CAPTURE = "ImageCaptured";
    private static final int RC_CALL = 4;
    private static final String INDIA = "India";
    private static String utteranceId;
    private String contactNameSpeech, contactPhone;
    private SpeechRecognizer speechRecog; //stt
    private Intent intentRecog;
    private TextToSpeech tts;
    private ImageView micTapImageView;
    public static final int CAMERA_PERMISSIONS_REQUEST = 2;
```

```

public static final int CAMERA_IMAGE_REQUEST = 3;
public static final String FILE_NAME = "temp.jpg";
private static final String TAG = "TAG";
private Bitmap resultBitmap;
private ImageView imageSelected;
private Dialog dialogLoader;
private RelativeLayout llayout;
private TextView bestGuessLabelTV;
private boolean noInputNeeded;
private int tts_id = 0, firstCmd = 0;
private boolean isTTSready;
private Runnable runnable;
private String url;
private boolean flowOnceStarted;
private boolean backFromCallActivity;
private static final int MAX_DIMENSION = 1200;
private Bitmap origBitmap;
private String imageURL;

private ProgressDialog pd;
private FTPClient mFTPClient;
private static final int RC_PERM = 107;
private static final int RC_FILE_GET = 10;
private CopyStreamAdapter streamListener;
private String fPath;
private String extension;
private String destUploadedPath;
private String getImageUrl;
private static final String MY_KEY = "0a1326bbe8e640359c8b7cdc664815e2";
private static final String UPLOADPATH = "AOrganDon/currencyd/";
final String API_URL =

```

```

"https://northcentralus.api.cognitive.microsoft.com/customvision/v3.0/Prediction/1032cf44-1e62-4ebe-8c42-4aad60be5b88/classify/iterations/Iteration14/url";
final String API_URL_IMG =

```

```

"https://northcentralus.api.cognitive.microsoft.com/customvision/v3.0/Prediction/1032cf44-1e62-4ebe-8c42-4aad60be5b88/classify/iterations/Iteration14/image";

```

```

@Override

```

```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_call);
    Toolbar toolbar = findViewById(R.id.toolbar);

```

```

//toolbar.setNavigationIcon(android.support.v7.appcompat.R.drawable.abc_ic_ab_back_material);
    setSupportActionBar(toolbar);
    getSupportActionBar().setBackgroundDrawable(null);
    getSupportActionBar().setTitle("Blind assistant");

```

```

        init();
    }

    //checks for the internet if available else will show the dialog saying "looks like you're
offline..
    private void internetCheck() {
        if (!isConnectedToInternet()) {
            speakOut("Looks like you're offline, Connect to the internet!", true);
            AlertDialog.Builder builder = new AlertDialog.Builder(CallActivity.this);
            builder.setTitle("Looks like you're offline");
            builder.setCancelable(false);
            builder.setMessage("No internet connection");
            builder.setPositiveButton("ok", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    finish();
                }
            });
            builder.show();
        } else {
            isTTSready = true;
            Log.i(TAG, "internetCheck: isTTSready " + isTTSready);
            if (PermissionUtils.requestPermission(CallActivity.this, 0,
Manifest.permission.RECORD_AUDIO,
Manifest.permission.CALL_PHONE,
Manifest.permission.READ_CONTACTS,
Manifest.permission.WRITE_EXTERNAL_STORAGE,
Manifest.permission.CAMERA)
&& tts_id == 0) {
                speakOut("Do you want to Detect currency or make a call? Say 1 to make a call,
say 2 to detect currency.", false);
                flowOnceStarted = true;
                firstCmd = 0;
            }
        }
    }

    // returns true if connected to internet
    private boolean isConnectedToInternet() {
        ConnectivityManager mgr = (ConnectivityManager)
getSystemService(CONNECTIVITY_SERVICE);
        return mgr.getActiveNetworkInfo() != null &&
mgr.getActiveNetworkInfo().isConnected();
    }

    //AsyncTask matching spoken contactname from your address book in device
    private void ContactMatchTask(String s) {
        ContactsTask task = new ContactsTask();
        task.execute(s);
    }

```

```

//initializing the android UI widgets and texttospeech
private void init() {
    micTapImageView = findViewById(R.id.iv_mic);
    tts = new TextToSpeech(this, this);
    tts.setOnUtteranceProgressListener(new UtteranceProgressListener() {

        @Override
        public void onStart(String utteranceId) {
            Log.i(TAG, "onStart utteranceId: " + utteranceId);
        }

        @Override
        public void onDone(String utteranceId) {
            //pause or some indication to inputnext
            if (!noInputNeeded) {
                CallActivity.utteranceId = utteranceId;
                runOnUiThread(runnable);

                Log.i(TAG, "onDone: utteranceId " + utteranceId);
            }
        }

        @Override
        public void onError(String utteranceId) {
            Log.i(TAG, "onError: utteranceId" + utteranceId);
        }
    });
    runnable = () -> onMicTapped(utteranceId);
    imageSelected = findViewById(R.id.iv_currency_display);
    llayout = findViewById(R.id.ll);
    bestGuessLabelTV = findViewById(R.id.tv_bestguesslabel);
}

//speaks the String s
private void speakOut(String s, boolean noInputNeeded) {
    tts_id++;
    this.noInputNeeded = noInputNeeded;
    HashMap<String, String> mapTTSid = new HashMap<>();
    mapTTSid.put(TextToSpeech.Engine.KEY_PARAM_UTTERANCE_ID,
String.valueOf(tts_id));
    Log.i(TAG, "speakOut: isSpeaking " + tts.isSpeaking());
    tts.speak(s, TextToSpeech.QUEUE_FLUSH, mapTTSid);
}

/*@Override
protected void onStop() {
    if (tts != null) {
        tts.stop();
        tts.shutdown();
    }
}

```

```

    }
    if (speechRecog != null) {
        speechRecog.destroy();
    }
    super.onStop();
}
*/
//when mic image is tapped start speech recognizing
public void onMicTapped(View view) {
    //recognise speech
    //if (SpeechRecognizer.isRecognitionAvailable(this)) {
    Log.i(TAG, "onMicTapped: ");
    if (speechRecog == null) {
        if (SpeechRecognizer.isRecognitionAvailable(CallActivity.this)) {
            Toast.makeText(this, "Speech recognition not available on this device",
Toast.LENGTH_SHORT).show();
            return;
        } //TEST
        speechRecog = SpeechRecognizer.createSpeechRecognizer(this);
        speechRecog.setRecognitionListener(this);
        intentRecog = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        intentRecog.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
        intentRecog.putExtra(RecognizerIntent.EXTRA_LANGUAGE, Locale.ENGLISH);
        intentRecog.putExtra(RecognizerIntent.EXTRA_MAX_RESULTS, 1);
    }
    if (!PermissionUtils.requestPermission(CallActivity.this, 0,
Manifest.permission.RECORD_AUDIO))
        return;
    if (!tts.isSpeaking()) {
        speechRecog.stopListening();
        speechRecog.startListening(intentRecog);
    }
}

@Override
protected void onDestroy() {
    if (tts != null) {
        tts.stop();
        tts.shutdown();
    }
    if (speechRecog != null) {
        speechRecog.destroy();
    }
    super.onDestroy();
}

//method of ContactMatch Asyctask
private String matchContact(String contactNameSpeech) {

```



```

        Cursor cursor =
getContentResolver().query(ContactsContract.Contacts.CONTENT_URI, null, null,
        null, null, null);
        if (cursor != null) {
            cursor.moveToFirst();
            String hasPhone, contactName, contactId;
            while (!cursor.isAfterLast()) {
                contactName =
cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME));
                contactId =
cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts._ID));
                //matches?
                Log.i(TAG, "matchContact: " + contactName + contactNameSpeech);
                if (contactNameSpeech.equalsIgnoreCase(contactName)) {
                    hasPhone =
cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.HAS_PHONE_NUMBE
R));
                    Log.i(TAG, "matchContact: " + hasPhone);
                    // if (Integer.parseInt(hasPhone) > 0) {
                    Log.d(TAG, "matchContact: hasPhone" + hasPhone);
                    Cursor phones =
getContentResolver().query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
null, ContactsContract.CommonDataKinds.Phone.CONTACT_ID + " = " + contactId, null,
null);
                    while (phones.moveToNext()) {
                        String phoneNumber =
phones.getString(phones.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NU
MBER));
                        phones.close();
                        Log.i(TAG, "matchContact matched: " + phoneNumber);
                        return phoneNumber;
                    }
                } else if (Integer.parseInt(hasPhone) == 0) {
                    return null;*/
                }
            }
            cursor.moveToNext();
        }
        cursor.close();
    }
    return null;
}

```

@Override

```

public void onInit(int status) {
    //takes 3s
    if (status == TextToSpeech.SUCCESS) {
        int resultLang = tts.setLanguage(Locale.getDefault());
        if (resultLang == TextToSpeech.LANG_MISSING_DATA ||
            resultLang == TextToSpeech.LANG_NOT_SUPPORTED) {

```

```

        // missing data, install it
        Intent install = new Intent();
        install.setAction(
            TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA);
        startActivity(install);
    } else {
        internetCheck();
    }
}
Log.d(TAG, "onInit() called with: status = [" + status + "]);
}

```

//when upload image is clicked the ImageAnalysisTask is executed for Microsoftcognitive custom vision api service

```

/* public void uploadImage() {
    //ftp upload and image renamed as temp.jpg or jpg or what format, keep the same format
    for upload but name
    uploadFTP();
}*/

```

```

public void uploadImage(View view) {
    /* if (resultBitmap != null)
        uploadImage();*/
    if (origBitmap != null && origBitmap.getBytesCount() > 4_000000) {
        //resize and send image
        Log.i(TAG, "onActivityResult: >4mb");
        new ImageAnalysisTaskImage().execute();
    } else {
        //upload from url
        uploadFTP();
        imageSelected.setImageBitmap(resultBitmap);
    }
}
}

```

//AsyncTask for matching contact spoken

```

private class ContactsTask extends AsyncTask<String, Void, String> {

```

```

    @Override
    protected String doInBackground(String... strings) {
        return matchContact(strings[0]);
    }

```

```

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
    }

```

```

    @Override
    protected void onPostExecute(String phoneNum) {
        super.onPostExecute(phoneNum);
    }

```

```

//make a call
Log.i(TAG, "onPostExecute: " + contactNameSpeech + phoneNum);
if (phoneNum != null) {
    speakOut("calling " + contactNameSpeech, true);
    Intent intentCall = new Intent(Intent.ACTION_CALL);
    intentCall.setData(Uri.parse("tel:" + phoneNum));
    if (ActivityCompat.checkSelfPermission(CallActivity.this,
Manifest.permission.CALL_PHONE) != PackageManager.PERMISSION_GRANTED) {
        // TODO: Consider calling
        //   ActivityCompat#requestPermissions
        // here to request the missing permissions, and then overriding
        //   public void onRequestPermissionsResult(int requestCode, String[]
permissions,
        //                                     int[] grantResults)
        // to handle the case where the user grants the permission. See the documentation
        // for ActivityCompat#requestPermissions for more details.
        return;
    }
    startActivityForResult(intentCall, RC_CALL);
} else {
    speakOut("CONTACT " + contactNameSpeech + " not found ,\n " +
askAgainToCallString(), false);
    Log.i(TAG, "onPostExecute: " + contactNameSpeech);
}
}
}

@Override
public boolean onSupportNavigateUp() {
    onBackPressed();
    return super.onSupportNavigateUp();
}

@Override
protected void onResume() {
    super.onResume();
    permission();
    if (backFromCallActivity) {
        Log.d(TAG, "onResume() called" + isTTSSready);
        speakOut("\n " + askAgainToCallString(), false);
        backFromCallActivity = false;
    }
}

//dynamic permissions are asked checked and asked from user to grant them
private void permission() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        if (ContextCompat.checkSelfPermission(this, Manifest.permission.CAMERA) !=
PackageManager.PERMISSION_GRANTED ||

```

```

        ContextCompat.checkSelfPermission(this,
Manifest.permission.RECORD_AUDIO) != PackageManager.PERMISSION_GRANTED ||
        ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_CONTACTS) != PackageManager.PERMISSION_GRANTED
||
        ContextCompat.checkSelfPermission(this,
Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED ||
        ContextCompat.checkSelfPermission(this, Manifest.permission.CALL_PHONE)
!= PackageManager.PERMISSION_GRANTED) {
    Log.d(TAG, "permission() called inside");
    PermissionUtils.requestPermission(CallActivity.this, 0,
Manifest.permission.RECORD_AUDIO,
    Manifest.permission.CALL_PHONE,
    Manifest.permission.READ_CONTACTS,
    Manifest.permission.WRITE_EXTERNAL_STORAGE,
    Manifest.permission.CAMERA);
} else {
    Log.d(TAG, "permission() called in else");
}
} else {
    Log.d(TAG, "permission() called ");
}
}
}

```

```

@Override
public void onReadyForSpeech(Bundle params) {
    //play a sound
    Log.d(TAG, "onReadyForSpeech() called with: params = [" + params + "]");
    Animation animation = AnimationUtils.loadAnimation(getApplicationContext(),
R.anim.zoom_in_out);
    //animation.setRepeatCount(3);
    micTapImageView.setAnimation(animation);
}

```

```

@Override
public void onBeginningOfSpeech() {
    Log.d(TAG, "onBeginningOfSpeech() called");
}

```

```

@Override
public void onRmsChanged(float rmsdB) {
    //Log.d(TAG, "onRmsChanged() called with: rmsdB = [" + rmsdB + "]");
}

```

```

@Override
public void onBufferReceived(byte[] buffer) {
    Log.d(TAG, "onBufferReceived() called with: buffer = [" + buffer + "]");
}

```

```

//restart
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu, menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    if (item.getItemId() == R.id.restart) {
        restart();
    }
    return super.onOptionsItemSelected(item);
}

@Override
public void onEndOfSpeech() {
    Log.d(TAG, "onEndOfSpeech() called");
}

@Override
protected void onPause() {
    Log.i(TAG, "onPause: before" + firstCmd + flowOnceStarted);
    super.onPause();
    /*if (!flowOnceStarted) {
        speakOut("Do you want to Detect currency or make a call? Say 1 to make a call, say 2
to detect currency.", false);
        firstCmd = 0;
    }*/
    Log.i(TAG, "onPause: after" + firstCmd);
}

@Override
public void onError(int error) {
    Log.d(TAG, "onError() called with: error = [" + error + "]");
}

@Override
public void onResults(Bundle results) {
    //play a sound
    //isSpeechRecognized = 0;
    String command =
Objects.requireNonNull(results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGN
ITION)).get(0);
    ArrayList<String> cmdList = new ArrayList<>();
    cmdList =
Objects.requireNonNull(results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGN
ITION));
    Log.d(TAG, "onResults: firstcmd = " + firstCmd + "command : " + command + ", " +
Arrays.asList(cmdList));

```

```

        if (firstCmd == 0) {
            for (int i = 0; i < cmdList.size(); i++) {
                String cmd = cmdList.get(i);
                if (cmd.equalsIgnoreCase("1") || cmd.equalsIgnoreCase("one") ||
cmd.equalsIgnoreCase("CALL")) {
                    firstCmd = 1;
                    Log.d(TAG, "onResults: To make a call to your contact, say ");
                    speakOut("To make a call to your contact, say \"CALL CONTACT NAME\"",
false);
                    break;
                } else if (cmd.equalsIgnoreCase("2") || cmd.equalsIgnoreCase("two") ||
cmd.equalsIgnoreCase("DETECT")) {
                    firstCmd = 1;
                    speakOut("Starting camera.. Picture will be taken in next 3 seconds", true);
                    Log.d(TAG, "onResults: Please Choose an Image to Analyse");
                    startCamera();
                    break;
                } else if (cmd.equalsIgnoreCase("EXIT")) {
                    firstCmd = 1;
                    finish();
                    break;
                }
            }
            if (firstCmd != 1) {
                speakOut("Didn't catch that, please try again", false);
            }
        } else if (command.contains(CALL_CMD)) {
            Log.i(TAG, "onResults: before replace " + command);
            contactNameSpeech = command.replace("call ", "");
            ContactMatchTask(contactNameSpeech.trim());
            Log.i(TAG, "onResults: replaced " + command);
        } else {
            //no match
            speakOut("Didn't catch that, please try again", false);
        }
        Log.d(TAG, "onResults() called with: results = [" + results + "]");
        //speechRecog.stopListening();
    }

```

@Override

```

public void onPartialResults(Bundle partialResults) {
    Log.d(TAG, "onPartialResults() called with: partialResults = [" + partialResults + "]");
}

```

@Override

```

public void onEvent(int eventType, Bundle params) {
    Log.d(TAG, "onEvent() called with: eventType = [" + eventType + "], params = [" +
params + "]");
}

```

```

public void onMicTapped(String utteranceId) {
    //recognise speech
    if (CallActivity.utteranceId.equals(utteranceId)) {
        if (SpeechRecognizer.isRecognitionAvailable(this)) {
            Log.i(TAG, "onMicTapped: ");
            if (speechRecog == null) {
                speechRecog = SpeechRecognizer.createSpeechRecognizer(this);
                speechRecog.setRecognitionListener(this);
                intentRecog = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
                intentRecog.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
                intentRecog.putExtra(RecognizerIntent.EXTRA_LANGUAGE,
Locale.ENGLISH);
                intentRecog.putExtra(RecognizerIntent.EXTRA_MAX_RESULTS, 1);
            }
            if (!PermissionUtils.requestPermission(CallActivity.this, 0,
Manifest.permission.RECORD_AUDIO))
                return;
            if (! tts.isSpeaking()) {
                speechRecog.stopListening();
                speechRecog.startListening(intentRecog);
            }
        } else {
            Log.e(TAG, "Speech Recognition not available on this device.");
        }
    }
}

```

//DETECT CURRENCY

```

public void onCurrencyClicked(View view) {
    speakOut("Starting camera.. Picture will be taken in next 3 seconds", true);
    startCamera();
}

public void startCamera() {
    if (PermissionUtils.requestPermission(
        this,
        CAMERA_PERMISSIONS_REQUEST,
        Manifest.permission.CAMERA)) {
        Intent intentCam = new Intent(CallActivity.this, MainActivity.class);
        /* Uri photoUri = FileProvider.getUriForFile(this,
getApplicationContext().getPackageName() + ".provider",
getCameraFile());
        intentCam.putExtra(MediaStore.EXTRA_OUTPUT, photoUri);
        intentCam.addFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);*/
        startActivityForResult(intentCam, CAMERA_IMAGE_REQUEST);
    }
}

```

```

public File getCameraFile() {
    File dir = getExternalFilesDir(Environment.DIRECTORY_PICTURES);
    return new File(dir, FILE_NAME);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == CAMERA_IMAGE_REQUEST && resultCode == RESULT_OK)
    {
        Log.i(TAG, "onActivityResult: " + data);
        Uri uri = Uri.fromFile(new
File(data.getStringExtra(Utils.CAM_RESULT_OK_KEY)));
        Log.i(TAG, "onActivityResult: uri" + uri);
        Bitmap bt = ImageHelper.loadSizeLimitedBitmapFromUri(uri,
getContentResolver());
        Log.i(TAG, "onActivityResult: bt" + bt);
        /* Uri photoUri = FileProvider.getUriForFile(this,
getApplicationContext().getPackageName()
+ ".provider", getCameraFile());*/
        //Log.i(TAG, "onActivityResult: " + photoUri.getPath());
        if (data != null)
            fPath = data.getStringExtra(Utils.CAM_RESULT_OK_KEY);
        //Log.i(TAG, "onActivityResult: getData" + data.getData().getLastPathSegment() +
data.getType());
        getImageUrl = ImagePath_MarshMallow.getPath(CallActivity.this, uri);
        // if (getImageUrl != null) {
        Log.i(TAG, "onActivityResult: getImageUrl "+getImageUrl);
        speakOut("processing the image".toUpperCase(), true);
        // getImageUrl = uri.getPath();
        //setImg(Uri.parse(fPath));
        setImg(uri);
        extension = Uri.parse(fPath).getLastPathSegment();
        /* String ext[] = extension.split(".");*/
        imageURL = "temp.jpg";
        Log.i(TAG, "onActivityResult: " + extension);
        if (origBitmap != null && origBitmap.getByteCount() > 4_000000) {
            //resize and send image
            Log.i(TAG, "onActivityResult: >4mb");
            new ImageAnalysisTaskImage().execute();
        } else {
            //upload from url
            uploadFTP();
            imageSelected.setImageBitmap(resultBitmap);
        }
        // uploadImage();
        //}
    } else if (requestCode == RC_CALL) {
        Log.i(TAG, "onActivityResult: RC_CALL" + RC_CALL);
    }
}

```



```

        backFromCallActivity = true;
    }
}

//start the text to speech commands again
private void restart() {
    firstCmd = 0;
    Log.d(TAG, "onrestart() called with firstCmd=" + firstCmd);
    if (!tts.isSpeaking()) {
        if (isTTSSready) {
            Log.d(TAG, "onrestart() called isttsready=" + isTTSSready);
            speakOut("Do you want to Detect currency or make a call? Say 1 to make a call,
say 2 to detect currency.", false);
        }
    } else {
        tts.stop();
        if (isTTSSready) {
            Log.d(TAG, "onrestart() called isttsready=" + isTTSSready);
            speakOut("Do you want to Detect currency or make a call? Say 1 to make a call,
say 2 to detect currency.", false);
        }
    }
}

@Override
public void onRequestPermissionsResult(
    int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    switch (requestCode) {
        case CAMERA_PERMISSIONS_REQUEST:
            if (PermissionUtils.permissionGranted(requestCode,
CAMERA_PERMISSIONS_REQUEST, grantResults)) {
                startCamera();
            }
            break;
    }
    if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
        speakOut("Do you want to Detect currency or make a call? Say 1 to make a call, say 2
to detect currency.", false);
        flowOnceStarted = true;
        firstCmd = 0;
    }
}

public void setImg(Uri uri) {
    if (uri != null) {
        Log.i(TAG, "setImg: " + uri.getPath());
        try {

```

```

        // scale the image to save on bandwidth
        origBitmap = MediaStore.Images.Media.getBitmap(getContentResolver(), uri);
        resultBitmap = scaleBitmapDown(
            origBitmap,
            MAX_DIMENSION);
        imageSelected.setImageBitmap(resultBitmap);
    } catch (IOException e) {
        Log.d(TAG, "Image picking failed because " + e.getMessage());
        Toast.makeText(this, R.string.image_picker_error,
            Toast.LENGTH_LONG).show();
    }
} else {
    Log.d(TAG, "Image picker gave us a null image.");
    Toast.makeText(this, R.string.image_picker_error, Toast.LENGTH_LONG).show();
}
}

//image size if too large is reduced
private Bitmap scaleBitmapDown(Bitmap bitmap, int maxDimension) {

    int originalWidth = bitmap.getWidth();
    int originalHeight = bitmap.getHeight();
    int resizedWidth = maxDimension;
    int resizedHeight = maxDimension;

    if (originalHeight > originalWidth) {
        resizedHeight = maxDimension;
        resizedWidth = (int) (resizedHeight * (float) originalWidth / (float) originalHeight);
    } else if (originalWidth > originalHeight) {
        resizedWidth = maxDimension;
        resizedHeight = (int) (resizedWidth * (float) originalHeight / (float) originalWidth);
    } else if (originalHeight == originalWidth) {
        resizedHeight = maxDimension;
        resizedWidth = maxDimension;
    }
    return Bitmap.createScaledBitmap(bitmap, resizedWidth, resizedHeight, false);
}

private void stopAnimation() {
    if (dialogLoader.isShowing())
        dialogLoader.cancel();
}

private void startAnimation() {
    dialogLoader = new Dialog(CallActivity.this, R.style.AppTheme_NoActionBar);
    dialogLoader.getWindow().setBackgroundDrawable(new
        ColorDrawable(Color.parseColor("#8D000000")));
    final View view =
        CallActivity.this.getLayoutInflater().inflate(R.layout.custom_dialog_loader, null);
    LottieAnimationView animationView = view.findViewById(R.id.loader);

```

```

        animationView.playAnimation();
        dialogLoader.setContentView(view);
        dialogLoader.setCancelable(false);
        dialogLoader.show();
    }

    public byte[] getImage() {
        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        Log.i(TAG, "byte[] getImage: " + resultBitmap);
        if (origBitmap.getBytesCount() < 4_000_000) {
            origBitmap.compress(Bitmap.CompressFormat.PNG, 100, stream);
            Log.i(TAG, "getImage: origBitmap Bytes:<4Mb" + origBitmap.getBytesCount());
        } else
            resultBitmap.compress(Bitmap.CompressFormat.PNG, 100, stream);
        final byte[] byteArray = stream.toByteArray();
        return byteArray;
    }
    // return Base64.encodeToString(byteArray,Base64.DEFAULT);
}

class ImageAnalysisTaskImage extends AsyncTask<String, Void, BufferedReader> {
    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        startAnimation();
    }

    @Override
    protected BufferedReader doInBackground(String... params) {
        HttpURLConnection urlConnection = null;
        BufferedReader br = null;

        try {
            URL url = new URL(API_URL_IMG);
            urlConnection = (HttpURLConnection) url.openConnection();

            urlConnection.setRequestProperty("Prediction-Key", MY_KEY);
            urlConnection.setRequestProperty("Content-Type", "application/octet-stream");

            urlConnection.setRequestMethod("POST");
            urlConnection.setDoOutput(true);

            OutputStream os = urlConnection.getOutputStream();
            os.write(getImage());
            os.close();

            urlConnection.connect();

            if (200 <= urlConnection.getResponseCode() &&
                urlConnection.getResponseCode() <= 299) {

```

```

        br = new BufferedReader(new
InputStreamReader(urlConnection.getInputStream()));
    } else {
        br = new BufferedReader(new
InputStreamReader(urlConnection.getErrorStream()));
    }

    } catch (Exception e) {
        Log.d("ERROR", "ERROR " + e.toString()); //IO Exception Prints in log cat not
recognizing URL
        e.printStackTrace();
    } finally {
        urlConnection.disconnect();
    }

    return br;
}

@Override
protected void onPostExecute(BufferedReader s) {
    super.onPostExecute(s);
    if (s != null) {
        // Toast.makeText(CallActivity.this, "not null", Toast.LENGTH_SHORT).show();

        try {
            StringBuilder sb = new StringBuilder();
            String output;
            while ((output = s.readLine()) != null) {
                sb.append(output);
            }

            String result = sb.toString();
            Log.d("REPLY", result);

            JSONObject jsonObject = new JSONObject(result);
            if (result != null) {
                parseResultJson(jsonObject);
            }
        } catch (Exception e) {
            Toast.makeText(CallActivity.this, "EXP:" + e.getMessage(),
Toast.LENGTH_SHORT).show();
        }
    } else {
        Toast.makeText(CallActivity.this, "null", Toast.LENGTH_SHORT).show();
    }
}

private void parseResultJson(JSONObject response) {
    try {

```

```

JSONObject jo = response;
JSONArray jarr = jo.getJSONArray("predictions");
float prediction = 0;
String Tag = "";
DecimalFormat dFormat = new DecimalFormat("#.##");
boolean foundINDIAtag = false;
for (int i = 0; i < jarr.length(); i++) {
    prediction = 10.0f; //should be greaterthan 10% probability
    JSONObject jobj = jarr.getJSONObject(i);
    float f = Float.parseFloat(jobj.getString("probability"));
    f = f * 100;
    Log.i(TAG, "onResponse: " + jobj.getString("tagName"));
    Log.i(TAG, "onResponse: " + f + ">," + prediction);
    if (f > prediction) {
        String[] tagName = jobj.getString("tagName").split("_");
        if (tagName[0].equalsIgnoreCase(INDIA)) {
            foundINDIAtag = true;
            if (f >= 30) {
                Tag = getTagFiltered(jobj.getString("tagName"));
                bestGuessLabelTV.setText(Tag + " (probability " +
String.valueOf(dFormat.format(f)) + ")");
            } else {
                Tag = "Best Guess is " + getTagFiltered(jobj.getString("tagName")) + "
with a Low Probability";
                bestGuessLabelTV.setText("Best Guess - " +
getTagFiltered(jobj.getString("tagName")) + " (probability " +
String.valueOf(dFormat.format(f)) + ")");
            }
            break;
        }
    } else {
        Tag = "not a currency";
        bestGuessLabelTV.setText(Tag);
        Log.i(TAG, "onResponse: " + Tag + " - " + response.toString());
        break;
    }
}
if (!foundINDIAtag) {
    Tag = "not a currency";
    bestGuessLabelTV.setText(Tag);
}
stopAnimation();
//Float.valueOf(dFormat.format(prediction));
speakOut(Tag + "\n " + askAgainForCurrencyString(), false);
} catch (Exception e) {
    Toast.makeText(CallActivity.this, "catch-" + e.getMessage(),
Toast.LENGTH_SHORT).show();
    Log.e(TAG, "onResponse: " + e.getMessage());
}
}
}

```

```

public String getTagFiltered(String s) {
    String temp[] = s.split(" ");
    String res = temp[0].replaceAll("_", " ").toUpperCase();
    return res;
}

private String askAgainForCurrencyString() {
    firstCmd = 0;
    return "Do you want to Detect currency or make a call? Say CALL to make a call or say
DETECT to detect currency or say EXIT to exit.";
}

private String askAgainToCallString() {
    firstCmd = 0;
    return "Do you want to Detect currency or make a call? Say DETECT to detect currency
or say CALL to make a call or say EXIT to exit.";
}

//FTP upload for link
private void uploadFTP() {
    if (extension != null) {
        destUploadedPath = UPLOADPATH + imageUrl;
        UploadFTPTask uploadFTP = new UploadFTPTask();
        uploadFTP.execute(getImageUrl, destUploadedPath);
        Log.i(TAG, "UploadFTPCall: " + getImageUrl + ", " + destUploadedPath);
    }
}

class ImageAnalysisTask extends AsyncTask<String, Void, BufferedReader> {
    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        startAnimation();
    }

    @Override
    protected BufferedReader doInBackground(String... params) {
        HttpURLConnection urlConnection = null;
        BufferedReader br = null;

        try {
            URL url = new URL(API_URL);
            urlConnection = (HttpURLConnection) url.openConnection();

            urlConnection.setRequestProperty("Prediction-Key",
"0a1326bbe8e640359c8b7cdc664815e2");
            urlConnection.setRequestProperty("Content-Type", "application/json");

            urlConnection.setRequestMethod("POST");

```

```

        urlConnection.setDoOutput(true);
        JSONObject jsonObject = new JSONObject();
        jsonObject.put("Url", "http://aorgandon.hostoise.com/currencyd/" + imageURL);
        //{"Url": "http://aorgandon.hostoise.com/currencyd/naira10.jpeg"}
        //ftp://Hostoise@182.50.132.7/AOrganDon/currencyd/naira10.jpeg
        OutputStreamWriter os = new
OutputStreamWriter(urlConnection.getOutputStream());
        os.write(jsonObject.toString());
        os.close();

        urlConnection.connect();

        if (200 <= urlConnection.getResponseCode() &&
urlConnection.getResponseCode() <= 299) {
            br = new BufferedReader(new
InputStreamReader(urlConnection.getInputStream()));
        } else {
            br = new BufferedReader(new
InputStreamReader(urlConnection.getErrorStream()));
        }

        } catch (Exception e) {
            Log.d("ERROR", "ERROR " + e.toString());//IO Exception Prints in log cat not
recognizing URL
            e.printStackTrace();
        } finally {
            urlConnection.disconnect();
        }

        return br;
    }

    @Override
    protected void onPostExecute(BufferedReader s) {
        super.onPostExecute(s);
        if (s != null) {
            // Toast.makeText(CallActivity.this, "not null", Toast.LENGTH_SHORT).show();

            try {
                StringBuilder sb = new StringBuilder();
                String output;
                while ((output = s.readLine()) != null) {
                    sb.append(output);
                }

                String result = sb.toString();
                Log.d("REPLY", result);

                JSONObject jsonObject = new JSONObject(result);

```

```

        /*JSONParser parser = new JSONParser();*/
        if (result != null) {
            parseResultJson(jsonObject);
        }
    } catch (Exception e) {
        Toast.makeText(CallActivity.this, "EXP:" + e.getMessage(),
Toast.LENGTH_SHORT).show();
    }
    } else {
        Toast.makeText(CallActivity.this, "null", Toast.LENGTH_SHORT).show();
    }
}
}
}

```

//srcpath/destpath

```

public class UploadFTPTask extends AsyncTask<String, Void, String> {
    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        pd = new ProgressDialog(CallActivity.this);
        pd.requestWindowFeature(Window.FEATURE_NO_TITLE);
        pd.setMessage("Uploading...");
        pd.setIndeterminate(false);
        pd.setCancelable(false);
        pd.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
        pd.setProgress(0);
        pd.show();
    }
}

```

```

@Override
protected String doInBackground(String... params) {
    String ans = "";
    try {
        mFTPClient = new FTPClient();
        mFTPClient.connect("182.50.132.7");

        if (mFTPClient.login("Hostoise", "Tyfo61*1")) {
            mFTPClient.setFileType(FTP.BINARY_FILE_TYPE);

            BufferedInputStream buffIn = null;
            final File file = new File(params[0]);
            buffIn = new BufferedInputStream(new FileInputStream(file));
            mFTPClient.enterLocalPassiveMode();
            streamListener = new CopyStreamAdapter() {

```

```

                @Override
                public void bytesTransferred(long totalBytesTransferred,
                    int bytesTransferred, long streamSize) {

```

```

                    int percent = (int) (totalBytesTransferred * 100 / file.length());

```



```

        pd.setProgress(percent);
        publishProgress();

        if (totalBytesTransferred == file.length()) {
            System.out.println("100% transfered");

            removeCopyStreamListener(streamListener);
        }
    }
};
mFTPClient.setCopyStreamListener(streamListener);
Boolean status = mFTPClient.storeFile(params[1], buffIn);
if (status) {
    ans = "true";
} else {
    ans = "false";
}

buffIn.close();
mFTPClient.logout();
mFTPClient.disconnect();

}
} catch (FileNotFoundException e) {
    ans = "U-file not found-" + "\n" + e.getMessage();
} catch (SocketException e) {
    ans = "U-socket-" + "\n" + e.getMessage();
} catch (IOException e) {
    ans = "U-IO-" + "\n" + e.getMessage();
}
return ans;
}

protected void onProgressUpdate(String... values) {
    pd.setProgress(Integer.parseInt(values[0]));
}

@Override
protected void onPostExecute(String s) {
    super.onPostExecute(s);
    pd.cancel();
    if (s.compareTo("true") == 0) {
        //if upload completes then give that link to analysis
        analyseImage();
        Log.i("TAG", "onPostExecute: " + s);
    } else {
        Toast.makeText(CallActivity.this, s, Toast.LENGTH_SHORT).show();
        Log.e("TAG", "onPostExecute: " + s);
    }
}
}

```

```

    }

    private void analyseImage() {
        new ImageAnalysisTask().execute();
    }

}
Mainactivity.java
package com.website.currencydetectandvoicecommandindia;

import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.hardware.Camera;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.widget.FrameLayout;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    private int REQUEST_CODE_PERMISSIONS = 10; //arbitrary number, can be changed
    accordingly
    private final String[] REQUIRED_PERMISSIONS = new
String[]{"android.permission.CAMERA",
"android.permission.WRITE_EXTERNAL_STORAGE"};

    private Camera mCamera;
    private CameraPreview mPreview;
    FrameLayout preview;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

```

```

super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
if (allPermissionsGranted()) {
    startCamera(); //start camera if permission has been granted by user

    } else {
        ActivityCompat.requestPermissions(this, REQUIRED_PERMISSIONS,
REQUEST_CODE_PERMISSIONS);
    }
}

public void startCamera() {
    try {
        mCamera = getCameraInstance("Back", this);
        if (mCamera != null) {
            mPreview = new CameraPreview(MainActivity.this, mCamera);
            preview = (FrameLayout) findViewById(R.id.cam_surface);
            Camera.Parameters parameters = mCamera.getParameters();
            List<String> focusModes = parameters.getSupportedFocusModes();

            if(focusModes.contains(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE)){

                parameters.setFocusMode(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE)
                ;

            } else
            if(focusModes.contains(Camera.Parameters.FOCUS_MODE_AUTO)){
                parameters.setFocusMode(Camera.Parameters.FOCUS_MODE_AUTO);
            }

            mCamera.setParameters(parameters);
            preview.addView(mPreview);
            try {
                new Handler().postDelayed(new Runnable() {
                    @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
                    @Override
                    public void run() {
                        mCamera.takePicture(null, null, new Camera.PictureCallback() {
                            @Override
                            public void onPictureTaken(byte[] bytes, Camera camera) {
                                Bitmap pic = BitmapFactory.decodeByteArray(bytes, 0, bytes.length);

                                camera.stopPreview();
                                camera.startPreview();

                                String Fname = "";
                                try {
                                    //File file = Environment.getExternalStoragePublicDirectory(
                                    // Environment.DIRECTORY_PICTURES);
                                    File file = new File("/sdcard/CurrencyIndia/Photos");
                                    Log.i("TAG", "onPictureTaken: "+file.mkdirs());
                                }
                            }
                        });
                    }
                }, 1000);
            }
        }
    }
}

```

```

        String filename = "IMG" + System.currentTimeMillis() + ".jpg";
        file.mkdirs();
        File finalfile = new File(file.getAbsolutePath(), filename);
        Log.i("TAG", "onPictureTaken: " + finalfile.getAbsolutePath());
        Fname = finalfile.getAbsolutePath();
        OutputStream out = new FileOutputStream(finalfile);
        pic.compress(Bitmap.CompressFormat.JPEG, 100, out);
        out.flush();
        out.close();
        getResultsBackToPrevActivity(Fname);
    } catch (IOException e) {
        e.printStackTrace();
        Toast.makeText(MainActivity.this, e.getMessage(),
Toast.LENGTH_SHORT).show();
    }
    });
}
}, 6000);

    } catch (Exception e) {
        e.printStackTrace();
        Toast.makeText(this, e.getMessage(), Toast.LENGTH_SHORT).show();
    }
    } else {
        Toast.makeText(this, "camera null", Toast.LENGTH_SHORT).show();
    }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

private void getResultsBackToPrevActivity(String filename) {
    Intent intent = new Intent();
    intent.putExtra(Utils.CAM_RESULT_OK_KEY, filename);
    setResult(RESULT_OK, intent);
    finish();
}

public static Camera getCameraInstance(String mode, Context con) {
    Camera c = null;
    try {
        if (mode.compareTo("Front") == 0) {
            c = Camera.open(Camera.CameraInfo.CAMERA_FACING_FRONT);
        } else if (mode.compareTo("Back") == 0) {
            c = Camera.open(Camera.CameraInfo.CAMERA_FACING_BACK);
        }
    } catch (Exception e) {
        Toast.makeText(con, "getinstance:" + e.getMessage(),
Toast.LENGTH_SHORT).show();
    }
}

```

```

    }
    return c;
}

@Override
protected void onResume() {
    super.onResume();
    if (mCamera != null) {
        mPreview = new CameraPreview(MainActivity.this, mCamera);
        preview = (FrameLayout) findViewById(R.id.cam_surface);
        preview.addView(mPreview);
    }
}

@Override
protected void onPause() {
    super.onPause();
    if (mCamera != null) {
//        mCamera.release();
        mCamera.stopPreview();
    }
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
                                       @NonNull int[] grantResults) {
    //start camera when permissions have been granted otherwise exit app
    if (requestCode == REQUEST_CODE_PERMISSIONS) {
        if (allPermissionsGranted()) {
            startCamera();
        } else {
            Toast.makeText(this, "Permissions not granted by the user.",
Toast.LENGTH_SHORT).show();
            finish();
        }
    }
}

private boolean allPermissionsGranted() {
    //check if req permissions have been granted
    for (String permission : REQUIRED_PERMISSIONS) {
        if (ContextCompat.checkSelfPermission(this, permission) !=
PackageManager.PERMISSION_GRANTED) {
            return false;
        }
    }
    return true;
}
}
Permissionutility.java

```

```

/*
 * Copyright 2016 Google Inc. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
 * implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

```

```

package com.website.currencydetectandvoicecommandindia;

```

```

import android.app.Activity;
import android.content.pm.PackageManager;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

```

```

import java.util.ArrayList;

```

```

public class PermissionUtils {
    public static boolean requestPermission(
        Activity activity, int requestCode, String... permissions) {
        boolean granted = true;
        ArrayList<String> permissionsNeeded = new ArrayList<>();

        for (String s : permissions) {
            int permissionCheck = ContextCompat.checkSelfPermission(activity, s);
            boolean hasPermission = (permissionCheck ==
PackageManager.PERMISSION_GRANTED);
            granted &= hasPermission;
            if (!hasPermission) {
                permissionsNeeded.add(s);
            }
        }

        if (granted) {
            return true;
        } else {
            ActivityCompat.requestPermissions(activity,
                permissionsNeeded.toArray(new String[permissionsNeeded.size()]),
                requestCode);
            return false;
        }
    }
}

```

```

    }

    public static boolean permissionGranted(
        int requestCode, int permissionCode, int[] grantResults) {
        return requestCode == permissionCode && grantResults.length > 0 && grantResults[0]
        == PackageManager.PERMISSION_GRANTED;
    }
}

```

Imageview.java

```

/*
 * Copyright 2016 Google Inc. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
 * implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

```

```

package com.website.currencydetectandvoicecommandindia;

```

```

import android.app.Activity;
import android.content.pm.PackageManager;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

```

```

import java.util.ArrayList;

```

```

public class PermissionUtils {
    public static boolean requestPermission(
        Activity activity, int requestCode, String... permissions) {
        boolean granted = true;
        ArrayList<String> permissionsNeeded = new ArrayList<>();

        for (String s : permissions) {
            int permissionCheck = ContextCompat.checkSelfPermission(activity, s);
            boolean hasPermission = (permissionCheck ==
            PackageManager.PERMISSION_GRANTED);
            granted &= hasPermission;
            if (!hasPermission) {
                permissionsNeeded.add(s);
            }
        }
    }
}

```

```

    }
}

if (granted) {
    return true;
} else {
    ActivityCompat.requestPermissions(activity,
        permissionsNeeded.toArray(new String[permissionsNeeded.size()]),
        requestCode);
    return false;
}
}

public static boolean permissionGranted(
    int requestCode, int permissionCode, int[] grantResults) {
    return requestCode == permissionCode && grantResults.length > 0 && grantResults[0]
== PackageManager.PERMISSION_GRANTED;
}
}

```

Imagehelper.java

```

//
// Copyright (c) Microsoft. All rights reserved.
// Licensed under the MIT license.
//
// Microsoft Cognitive Services (formerly Project Oxford):
// https://www.microsoft.com/cognitive-services
//
// Microsoft Cognitive Services (formerly Project Oxford) GitHub:
// https://github.com/Microsoft/Cognitive-Face-Android
//
// Copyright (c) Microsoft Corporation
// All rights reserved.
//
// MIT License:
// Permission is hereby granted, free of charge, to any person obtaining
// a copy of this software and associated documentation files (the
// "Software"), to deal in the Software without restriction, including
// without limitation the rights to use, copy, modify, merge, publish,
// distribute, sublicense, and/or sell copies of the Software, and to
// permit persons to whom the Software is furnished to do so, subject to
// the following conditions:
//
// The above copyright notice and this permission notice shall be
// included in all copies or substantial portions of the Software.
//
// THE SOFTWARE IS PROVIDED ""AS IS"", WITHOUT WARRANTY OF ANY KIND,
// EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES
OF

```



```
// MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
// NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
// HOLDERS BE
// LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN
// ACTION
// OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
// CONNECTION
// WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
//
package com.website.currencydetectandvoicecommandindia;
```

```
import android.content.ContentResolver;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Matrix;
import android.graphics.Rect;
import android.media.ExifInterface;
import android.net.Uri;
import android.provider.MediaStore;
```

```
import java.io.IOException;
import java.io.InputStream;
```

```
/**
 * Defined several functions to load, draw, save, resize, and rotate images.
 */
```

```
public class ImageHelper {
```

```
    // The maximum side length of the image to detect, to keep the size of image less than
    4MB.
```

```
    // Resize the image if its side length is larger than the maximum.
```

```
    private static final int IMAGE_MAX_SIDE_LENGTH = 1280;
```

```
    // Ratio to scale a detected face rectangle, the face rectangle scaled up looks more natural.
```

```
    private static final double FACE_RECT_SCALE_RATIO = 1.3;
```

```
    // Decode image from imageUri, and resize according to the
    expectedMaxImageSideLength
```

```
    // If expectedMaxImageSideLength is
```

```
    // (1) less than or equal to 0,
```

```
    // (2) more than the actual max size length of the bitmap
```

```
    // then return the original bitmap
```

```
    // Else, return the scaled bitmap
```

```
    public static Bitmap[] bt;
```

```
    public static void setBitmap(Bitmap[] b)
```

```
    {
```

```
        bt=b;
```

```

    }

    public static Bitmap[] getBitmap()
    {
        return bt;
    }

    public static Bitmap loadSizeLimitedBitmapFromUri(
        Uri imageUri,
        ContentResolver contentResolver) {
        try {
            // Load the image into InputStream.
            InputStream imageInputStream = contentResolver.openInputStream(imageUri);

            // For saving memory, only decode the image meta and get the side length.
            BitmapFactory.Options options = new BitmapFactory.Options();
            options.inJustDecodeBounds = true;
            Rect outPadding = new Rect();
            BitmapFactory.decodeStream(imageInputStream, outPadding, options);

            // Calculate shrink rate when loading the image into memory.
            int maxSideLength =
                options.outWidth > options.outHeight ? options.outWidth: options.outHeight;
            options.inSampleSize = 1;
            options.inSampleSize = calculateSampleSize(maxSideLength,
IMAGE_MAX_SIDE_LENGTH);
            options.inJustDecodeBounds = false;
            if (imageInputStream != null) {
                imageInputStream.close();
            }

            // Load the bitmap and resize it to the expected size length
            imageInputStream = contentResolver.openInputStream(imageUri);
            Bitmap bitmap = BitmapFactory.decodeStream(imageInputStream, outPadding,
options);
            maxSideLength = bitmap.getWidth() > bitmap.getHeight()
                ? bitmap.getWidth(): bitmap.getHeight();
            double ratio = IMAGE_MAX_SIDE_LENGTH / (double) maxSideLength;
            if (ratio < 1) {
                bitmap = Bitmap.createScaledBitmap(
                    bitmap,
                    (int)(bitmap.getWidth() * ratio),
                    (int)(bitmap.getHeight() * ratio),
                    false);
            }

            return rotateBitmap(bitmap, getImageRotationAngle(imageUri, contentResolver));
        } catch (Exception e) {
            return null;
        }
    }

```

```

    }

    // Draw detected face rectangles in the original image. And return the image drawn.
    // If drawLandmarks is set to be true, draw the five main landmarks of each face.

    // Highlight the selected face thumbnail in face list.

    // Crop the face thumbnail out from the original image.
    // For better view for human, face rectangles are resized to the rate faceRectEnlargeRatio

    // Return the number of times for the image to shrink when loading it into memory.
    // The SampleSize can only be a final value based on powers of 2.
    private static int calculateSampleSize(int maxSideLength, int
expectedMaxImageSideLength) {
        int inSampleSize = 1;

        while (maxSideLength > 2 * expectedMaxImageSideLength) {
            maxSideLength /= 2;
            inSampleSize *= 2;
        }

        return inSampleSize;
    }

    // Get the rotation angle of the image taken.
    public static int getImageRotationAngle(
        Uri imageUri, ContentResolver contentResolver) throws IOException {
        int angle = 0;
        Cursor cursor = contentResolver.query(imageUri,
            new String[] { MediaStore.Images.ImageColumns.ORIENTATION }, null, null,
null);
        if (cursor != null) {
            if (cursor.getCount() == 1) {
                cursor.moveToFirst();
                angle = cursor.getInt(0);
            }
            cursor.close();
        } else {
            ExifInterface exif = new ExifInterface(imageUri.getPath());
            int orientation = exif.getAttributeInt(
                ExifInterface.TAG_ORIENTATION,
ExifInterface.ORIENTATION_NORMAL);

            switch (orientation) {
                case ExifInterface.ORIENTATION_ROTATE_270:
                    angle = 270;
                    break;
                case ExifInterface.ORIENTATION_ROTATE_180:
                    angle = 180;
                    break;
            }
        }
    }
}

```

```

        case ExifInterface.ORIENTATION_ROTATE_90:
            angle = 90;
            break;
        default:
            break;
    }
}
return angle;
}

// Rotate the original bitmap according to the given orientation angle
private static Bitmap rotateBitmap(Bitmap bitmap, int angle) {
    // If the rotate angle is 0, then return the original image, else return the rotated image
    if (angle != 0) {
        Matrix matrix = new Matrix();
        matrix.postRotate(angle);
        return Bitmap.createBitmap(
            bitmap, 0, 0, bitmap.getWidth(), bitmap.getHeight(), matrix, true);
    } else {
        return bitmap;
    }
}

// Resize face rectangle, for better view for human
// To make the rectangle larger, faceRectEnlargeRatio should be larger than 1, recommend
1.3
}

```

FEASIBILITY REPORT

Feasibility Study is a high-level capsule version of the entire process intended to answer a number of questions like: What is the problem? Is there any feasible solution to the given problem? Is the problem even worth solving? Feasibility study is conducted once the problem clearly understood. Feasibility study is necessary to determine that the proposed system is Feasible by considering the technical, Operational, and Economical factors. By having a detailed feasibility study the management will have a clear-cut view of the proposed system.

The following feasibilities are considered for the project in order to ensure that the project is variable and it does not have any major obstructions.

Feasibility study encompasses the following things:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

In this phase, we study the feasibility of all proposed systems, and pick the best feasible solution for the problem. The feasibility is studied based on three main factors as follows.

❖ Technical Feasibility

In this step, we verify whether the proposed systems are technically feasible or not. i.e., all the technologies required to develop the system are available readily or not.

Technical Feasibility determines whether the organization has the technology and skills necessary to carry out the project and how this should be obtained. The system can be feasible because of the following grounds:

- All necessary technology exists to develop the system.
- This system is too flexible and it can be expanded further.
- This system can give guarantees of accuracy, ease of use, reliability and the data security.
- This system can give instant response to inquire.

Our project is technically feasible because, all the technology needed for our project is readily available.

Operating System	: Android v5.0 or Higher (For Android Devices)
Languages	: JAVA
Database System	: MS-SQL Server
Documentation Tool	: MS - Word

❖ Economic Feasibility

Economically, this project is completely feasible because it requires no extra financial investment and with respect to time, it's completely possible to complete this project in 6 months.

In this step, we verify which proposal is more economical. We compare the financial benefits of the new system with the investment. The new system is economically feasible only when the financial benefits are more than the investments and expenditure. Economic Feasibility determines whether the project goal can be within the resource limits allocated to it or not. It must determine whether it is worthwhile to process with the entire project or whether the benefits obtained from the new system are not worth the costs. Financial benefits must be equal or exceed the costs. In this issue, we should consider:

- The cost to conduct a full system investigation.
- The cost of h/w and s/w for the class of application being considered.
- The development tool.
- The cost of maintenance etc...

Our project is economically feasible because the cost of development is very minimal when compared to financial benefits of the application.

❖ Operational Feasibility

In this step, we verify different operational factors of the proposed systems like man-power, time etc., whichever solution uses less operational resources, is the best operationally feasible solution. The solution should also be operationally possible to implement. Operational Feasibility determines if the proposed system satisfied user objectives could be fitted into the current system operation.

- The methods of processing and presentation are completely accepted by the clients since they can meet all user requirements.
- The clients have been involved in the planning and development of the system.
- The proposed system will not cause any problem under any circumstances.

Our project is operationally feasible because the time requirements and personnel requirements are satisfied. We are a team of four members and we worked on this project for three working months.

TESTING

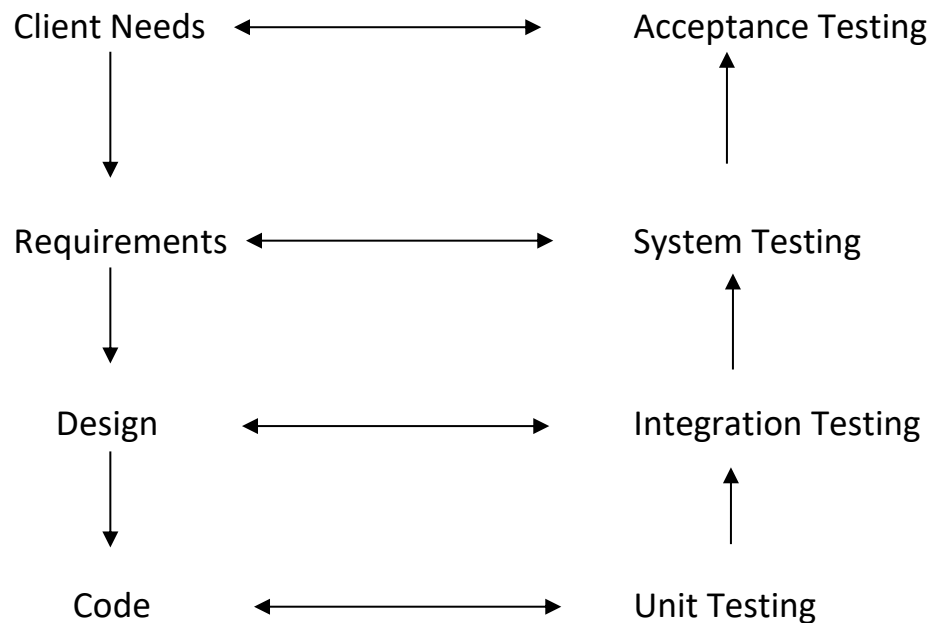
As the project is on bit large scale, we always need testing to make it successful. If each components work properly in all respect and gives desired output for all kind of inputs then project is said to be successful. So the conclusion is-to make the project successful, it needs to be tested.

The testing done here was System Testing checking whether the user requirements were satisfied. The code for the new system has been written completely using JAVA as the coding language and Android Studio as the interface for front-end designing. The new system has been tested well with the help of the users and all the applications have been verified from every nook and corner of the user.

Although some applications were found to be erroneous these applications have been corrected before being implemented. The flow of the forms has been found to be very much in accordance with the actual flow of data.

Levels of Testing

In order to uncover the errors present in different phases we have the concept of levels of testing. The basic levels of testing are:



A series of testing is done for the proposed system before the system is ready for the user acceptance testing.

The steps involved in Testing are:

✓ **Unit Testing**

Unit testing focuses verification efforts on the smallest unit of the software design, the module. This is also known as “Module Testing”. The modules are tested separately. This testing carried out during programming stage itself. In this testing each module is found to be working satisfactorily as regards to the expected output from the module.

✓ **Integration Testing**

Data can be grossed across an interface; one module can have adverse efforts on another. Integration testing is systematic testing for construction the program structure while at the same time conducting tests to uncover errors associated with in the interface. The objective is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole. Here correction is difficult because the isolation of cause is complicate by the vast expense of the entire program. Thus in the integration testing stop, all the errors uncovered are corrected for the text testing steps.

✓ **System testing**

System testing is the stage of implementation that is aimed at ensuring that the system works accurately and efficiently for live operation commences. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, then goal will be successfully achieved.

✓ **Validation Testing**

At the conclusion of integration testing software is completely assembled as a package, interfacing errors have been uncovered and corrected and a final series of software tests begins, validation test begins. Validation test can be defined in many ways. But the simple definition is that validation succeeds when the software function in a manner that can reasonably expected by the customer. After validation test has been conducted one of two possible conditions exists.

One is the function or performance characteristics confirm to specifications and are accepted and the other is deviation from specification is uncovered and a deficiency list is created. Proposed system under consideration has been tested by using validation testing and found to be working satisfactorily.

✓ **Output Testing**

After performing validation testing, the next step is output testing of the proposed system since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated by the system under consideration. Here the output format is considered in two ways, one is on the screen and other is the printed format. The output format on the screen is found to be correct as the format was designed in the system designed phase according to the user needs.

For the hard copy also the output comes as the specified requirements by the users. Hence output testing does not result any corrections in the system.

✓ **User Acceptance Testing**

User acceptance of a system is the key factor of the success of any system. The system under study is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required.

ADVANTAGES OF PROJECT

Advantages:

- This application will help blind people to know the value of the money.
- Blind person can also call the person from contact by giving command.

Limitations

- **This application requires active internet connection.**
- **User need to put correct data or else it behaves abnormally.**

Features

1) Load Balancing:

Since the system will be available only the admin logs in the amount of load on server will be limited to time period of admin access.

2) Easy Accessibility:

Records can be easily accessed and store and other information respectively.

3) User Friendly:

The application will be giving a very user-friendly approach for all user.

4) Efficient and reliable:

Maintaining the all secured and database on the server which will be accessible according the user requirement without any maintenance cost will be a very efficient as compared to storing all the customer data on the spreadsheet or in physically in the record books.

5) Easy maintenance:

Currency Detection is design as easy way. So maintenance is also easy.

CONCLUSION

This was our project of System Design about “**Currency Detection**” developed in Android application based on Java programming language. The Development of this system takes a lot of efforts from us. We think this system gave a lot of satisfaction to all of us. Though every task is never said to be perfect in this development field even more improvement may be possible in this application. We learned so many things and gained a lot of knowledge about development field. We hope this will prove fruitful to us.

BIBLIOGRAPHY

► Websites

- ✓ en.wikipedia.org
- ✓ <https://ieeexplore.ieee.org/document/8284300>
- ✓ <http://ieeexplore.ieee.org/document/7916838/>