# CHAPTER 1

# INTRODUCTION

# INTRODUCTION

**Converton** is a simple, smart and elegant utility tool with 15 categories of units that are used in daily life. Our unit converter app has the likes of various users ranging from students to a high level Professional who use conversion tools for their daily activities, which serves their purpose. All this was possible because of the Simple and engaging user experience along with most number of important and highly useful conversion tools. There are many unit conversion apps on the market. However, most are inconvenient and difficult to use because of poor and complicated UI.

This app has intuitive and simple UI, which is designed for casual user like you.

I have sorted essential unit sets for your daily life into 4 categories:-

- Basic: Length (distance), Area, Weight (mass), Data Storage (MB, GB).
- Living: Currency Rates, Temperature, Time, Velocity (speed).
- Science: Work (energy), Power, Volume.
- Misc.: Angle, Data Transfer Rate (Mbps), Fuel Efficiency, Cooking.

# CHAPTER 2
# PREVIOUS WORK

# PREVIOUS WORK

From the beginning of time converting different units was an issue that bothered the most brilliant minds of science. Many tried and failed. Different parts of the world adapted different units and measures, as globalization kicked in it become a prominent issue from people to trade measure and quantize things. More than thousand types of units are present. Steps were taken to normalize all this. The SI units of measurement were adapted, MKG and SKG were implemented but still the US and some other countries still use the metric system. Many different mathematicians provided theorems and formulas for converting these units into different desired forms. A thousand unit conversion charts were formed, then computers were used to convert these units. Many programs have been developed recently to tackle the same issue. For mobile several different apps are there that people currently use. Apps like Ultimate unit converter, every unit converter, and final unit converter are the apps that people use widely. The apps also provide functionality for converting currencies which is atone of the most important issue. We used the formulas developed by those previous workers into our algorithm.

# CHAPTER 3

# PROBLEM DEFINITION

# PROBLEM DEFINITION

- Have you ever felt the Need to Convert Miles to Kilometers?
- Have you ever felt the Need to Convert Dollars to Rupees?
- Have you ever felt the Need to Check Fuel Consumption?
- Have you ever felt the Need to Convert Pounds to Kilograms?

**YES!!!**

Then what you did, assume some Numbers and converted it yourself, but was it Accurate because Humans are bound to Make Mistakes.

You Doubt yourself, you try to convert it Again or To Cross Check your Answer.

Now you don't have to do all the Hard Work as Converton (Unit Converter) comes in role. Converton is an App on which you can rely on and all you have to do is just Enter the Numbers select Types and click CONVERT.

**THAT'S IT!!!**

You get the Accurate Answer which your Brain may or may not achieve plus it also reduces the Hard Work you do by Stressing your Brain to Try and Solve Complex Conversion.

# CHAPTER 4

# DEVELOPED SYSTEM

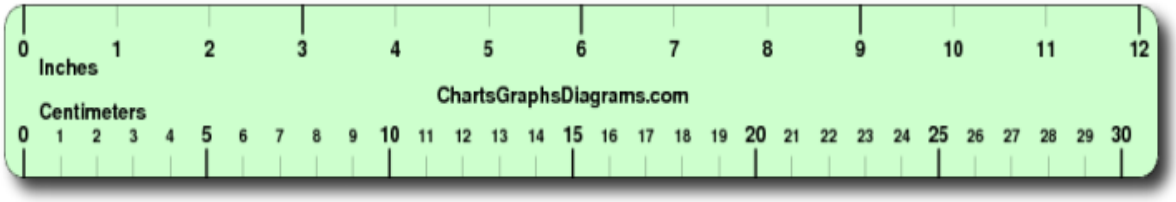
# DEVELOPED SYSTEM

## 4.1 Analysis and Design:

### This is how Unit is Converted Mathematically: -

Converting unit's means using different terms of measurements, but not changing the thing that is actually measured - length, mass, etc.

For example, on the ruler shown below, there are markings for both inches and centimetres. They both measure length, but in different units.




But if your ruler only has inch markings, and you need to know the length of something in centimetres, then you are going to have to convert units.

The trick for doing this is to have a special way of multiplying by 1, so that the number changes, and the units change, but the thing being measured does not change.

The 'special way of multiplying by one' is to use the correct conversion factor.

Here's an example: we want to convert feet to meters - say we have a piece of wood 2 feet long; we want to know how many meters that is.

Equation 1 **2 feet X (conversion factor) = ? meters**

The **conversion factor** will be a ratio or fraction. It will have the unit we want - meters, in this case - in the numerator (top) and the unit we have - feet - in the denominator (bottom).

Equation 2 **2 feet X ? meters / ? feet = ? meters**

This is so that the old units will 'cancel out', leaving us with the new units.

Equation 3 **2 feet X ? meters / ? feet = ? meters**

Now we need to assign numbers to the question marks in the conversion factor. This is easy in the denominator; we will just set it equal to 1. This way the numerator becomes the 'real' conversion factor.

Equation 4

$$\text{2 feet} \times \frac{\text{? meters}}{\text{? feet}} = \text{? meters}$$

You can't 'figure out' the conversion factor in the numerator - you have to look it up, or at least determine it with information outside the problem. It's determined by the ratio between the two different measurement systems.

I looked it up for you. The feet-to-meters conversion factor is 0.3048.

Equation 5

$$\text{2 feet} \times \frac{\text{0.3048 meters}}{\text{1 foot}} = \text{? meters}$$

Notice - this is a key point - that 0.3048 meters' **equals** 1 foot. Because of this, the conversion factor is the 'special way of multiplying by 1' that we mentioned earlier.

$$\text{0.3048 meters} = \text{1 foot} \qquad \text{So} \qquad \frac{\text{0.3048 meters}}{\text{1 foot}} = 1$$

Now do the math.

Equation 6

$$\text{2 feet} \times \frac{\text{0.3048 meters}}{\text{1 foot}} = \text{0.6096 meters}$$

**SOLUTION FOR ALL THIS IS: -**

### 4.1.1 Data Flow Diagram (DFD):

LEVEL-0

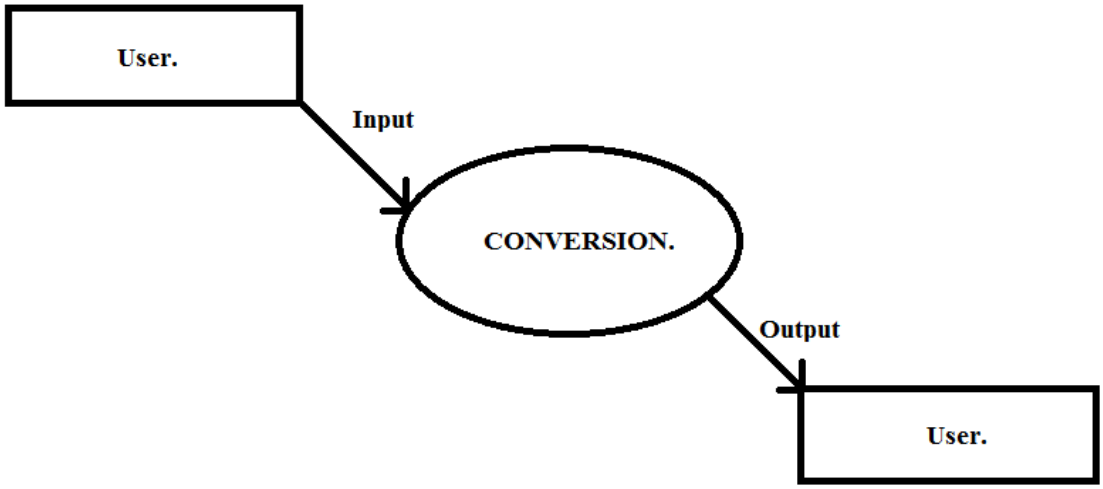



**Fig no.4.1.1**

LEVEL-1

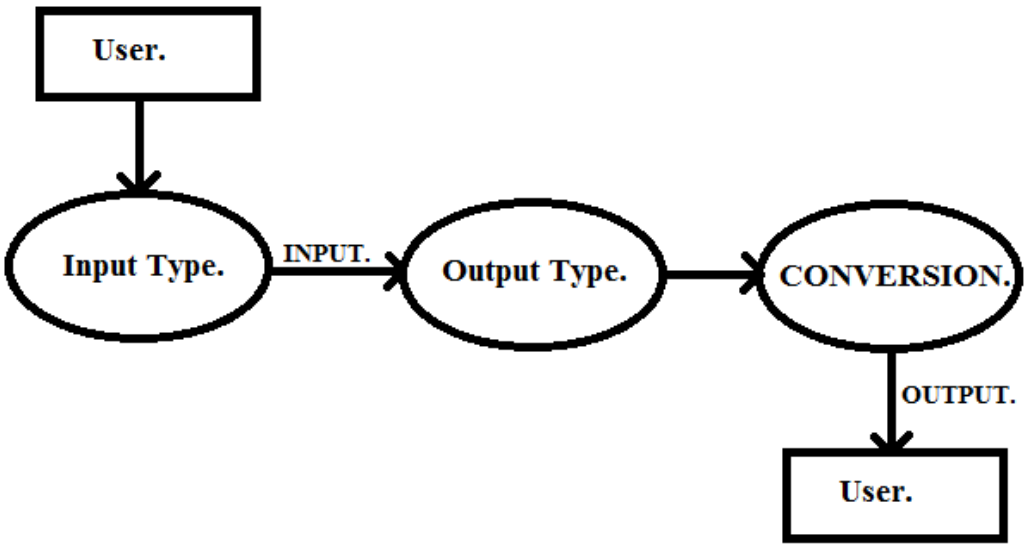



**Fig no.4.1.2**

## 4.2 Requirements:

HARWARE CONFIGURATION:

| | | |
|---|---|---|
| PROCESSOR | : | 1.1 GHZ Dual Core Processor |
| ROM CAPACITY | : | 1 GB |
| EXTERNAL MEMORY | : | NOT REQUIRED |

SOFTWARE CONFIGURATION:

| | | |
|---|---|---|
| OPERATING SYSTEM | : | LOLLIPOP (API 21) |
| FRONT END | : | XML |
| BACK END | : | JAVA |

### 4.2.1 ANDROID STUDIO

Android Studio is the official Integrated Development Environment (IDE) for Android platform Development.

It was announced on May 16, 2013 at the Google I/O conference. Android Studio is freely available under the Apache License 2.0.

Android Studio was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0.

Based on JetBrains' IntelliJ IDEA software, Android Studio is designed specifically for Android development. It is available for download on Windows, Mac OS X and Linux, and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

Android Studio mostly uses XML and JAVA to build an App. It also provide a built-in app editor plus user friendly environment for writing scripts similar to Notepad++. It also provides an Emulator on which we can test written codes simultaneously while writing them.

### 4.2.2 XML

In computing, **Extensible Markup Language** (**XML**) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The W3C's XML 1.0 Specification and several other related specifications all of them free open standards define XML.

The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services.

Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data.

The characters making up an XML document are divided into *markup* and *content*, which may be distinguished by the application of simple syntactic rules. Generally, strings that constitute markup either begin with the character `<` and end with a `>`, or they begin with the character `&` and end with a `;`. Strings of characters that are not markup are content. However, in a CDATA section, the delimiters `<![CDATA[` and `]]>` are classified as markup, while the text between them is classified as content. In addition, whitespace before and after the outermost element is classified as markup.

XML documents may begin with an *XML declaration* that describes some information about themselves. An example is `<?xml version="1.0" encoding="UTF-8"?>`.

### 4.2.3 JAVA

**Java** is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java Virtual Machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in

1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The Java language is a key pillar in Android, an open source mobile operating system. Although Android, built on the Linux kernel, is written largely in C, the Android SDK uses the Java language as the basis for Android applications. The bytecode language supported by the Android SDK is incompatible with Java bytecode and runs on its own virtual machine, optimized for low-memory devices such as smartphones and tablet computers. Depending on the Android version, the bytecode is either interpreted by the Dalvik virtual machine, or compiled into native code by the Android Runtime.

Android does not provide the full Java SE standard library, although the Android SDK does include an independent implementation of a large subset of it. It supports Java 6 and some Java 7 features, offering an implementation compatible with the standard library (Apache Harmony).

## Example #1 an introductory example

```
Class Hell{
Public static void main(String args[]){
System.out.println(“HELLO THERE!!!”);
}}
```

### 4.2.4 ADOBE ILLUSTRATOR

**Adobe Illustrator** is a vector graphics editor developed and marketed by Adobe Systems. The latest version, Illustrator CC 2017, is the twenty-first generation in the product line. The Apple Macintosh first developed adobe Illustrator in December 1986 (shipping in January 1987) as a commercialization of Adobe's in-house font development software and PostScript file format. Adobe Illustrator is the companion product of Adobe Photoshop. Photoshop is primarily gears toward digital photo manipulation and photorealistic styles of computer illustration, while Illustrator provides results in the typesetting and logographic areas of design. Early magazine advertisements (featured in graphic design trade magazines such as Communication Arts) referred to the product as "The Adobe Illustrator". Illustrator 88, the product name for version 1.7, was released in 1988 and introduced many new tools and features.

## 4.3 Implementation

### 4.3.1 - HOME SCREEN CODING: -

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.fahad.converton.MainActivity1">

    <TableLayout
        style="@style/Base.Widget.AppCompat.Button.Borderless"
        android:layout_width="400dp"
        android:layout_height="500dp"
        android:layout_alignParentStart="true"
        android:isScrollContainer="false"
        android:layoutMode="opticalBounds">

        <TableRow
            android:id="@+id/row5"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_margin="0dp"
            android:layout_marginLeft="0dp"
            android:layout_marginTop="0dp"
            android:showDividers="beginning|middle|end">

            <Button
                android:id="@+id/Currency"
                android:layout_width="100dp"
                android:layout_height="80dp"
                android:layout_marginLeft="12dp"
```

```
        android:background="@drawable/ic_currency2"
        android:elevation="?android:attr/windowTitleSize"
        android:freezesText="false" />

    <Button
        android:id="@+id/Area"
        android:layout_width="90dp"
        android:layout_height="80dp"
        android:layout_marginLeft="20dp"
        android:background="@drawable/ic_area" />

    <Button
        android:id="@+id/Cooking"
        android:layout_width="90dp"
        android:layout_height="80dp"
        android:layout_marginLeft="20dp"
        android:background="@drawable/ic_cook" />
</TableRow>

<TableRow
    android:id="@+id/row4"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="20dp">

    <Button
        android:id="@+id/Length"
        android:layout_width="100dp"
        android:layout_height="80dp"
        android:layout_marginLeft="12dp"
        android:background="@drawable/ic_scale" />

    <Button
        android:id="@+id/Data"
        android:layout_width="100dp"
```

```
        android:layout_height="80dp"
        android:layout_marginLeft="20dp"
        android:background="@drawable/ic_data" />

    <Button
        android:id="@+id/Volume"
        android:layout_width="100dp"
        android:layout_height="80dp"
        android:layout_marginLeft="20dp"
        android:background="@drawable/ic_volume"
        android:elevation="0dp" />

</TableRow>

<TableRow
    android:id="@+id/row1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="20dp">

    <Button
        android:id="@+id/Fuel"
        android:layout_width="100dp"
        android:layout_height="80dp"
        android:layout_marginLeft="12dp"
        android:background="@drawable/ic_fuel" />

    <Button
        android:id="@+id/Weight"
        android:layout_width="100dp"
        android:layout_height="80dp"
        android:layout_marginLeft="20dp"
        android:background="@drawable/ic_weight" />

    <Button
```

```
        android:id="@+id/Temperature"
        android:layout_width="100dp"
        android:layout_height="80dp"
        android:layout_marginLeft="20dp"
        android:background="@drawable/ic_temperature" />
    </TableRow>

    <TableRow
        android:id="@+id/row2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="20dp">

        <Button
            android:id="@+id/Velocity"
            android:layout_width="100dp"
            android:layout_height="80dp"
            android:layout_marginLeft="12dp"
            android:background="@drawable/ic_velocity" />

        <Button
            android:id="@+id/Angle"
            android:layout_width="100dp"
            android:layout_height="80dp"
            android:layout_marginLeft="20dp"
            android:background="@drawable/ic_angle" />

        <Button
            android:id="@+id/Energy"
            android:layout_width="100dp"
            android:layout_height="80dp"
            android:layout_marginLeft="20dp"
            android:background="@drawable/ic_energy" />

    </TableRow>
```

```
        <TableRow
            android:id="@+id/row3"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_marginTop="20dp">

            <Button
                android:id="@+id/Power"
                android:layout_width="100dp"
                android:layout_height="80dp"
                android:layout_marginLeft="12dp"
                android:background="@drawable/ic_power"
                android:elevation="24dp" />

            <Button
                android:id="@+id/Time"
                android:layout_width="100dp"
                android:layout_height="80dp"
                android:layout_marginLeft="20dp"
                android:background="@drawable/ic_time"
                android:elevation="24dp" />

            <Button
                android:id="@+id/DTS"
                android:layout_width="100dp"
                android:layout_height="80dp"
                android:layout_marginLeft="20dp"
                android:background="@drawable/ic_dts" />

        </TableRow>

    </TableLayout>

</RelativeLayout>
```

### 4.3.2 – CONVERSION PHASE CODING: -

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_length"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.fahad.converton.length">

    <EditText
        android:id="@+id/len_input"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/len_spinner2"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="35dp"
        android:ems="10"
        android:hint="          Enter Number"
        android:inputType="number|numberSigned|numberDecimal"
        android:textAppearance="@style/TextAppearance.AppCompat.Body2"
        android:textColor="@android:color/black"
        android:textSize="20sp" />

    <Spinner
        android:layout_width="match_parent"
        android:layout_marginTop="65dp"
        android:id="@+id/len_spinner1"
        android:layout_height="50dp"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true" />
```

```
<Spinner
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:layout_centerVertical="true"
    android:layout_centerHorizontal="true"
    android:id="@+id/len_spinner2" />

<Button
    android:id="@+id/con_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="68dp"
    android:background="@color/colorPrimary"
    android:clickable="true"
    android:elevation="24dp"
    android:text="CONVERT"
    android:visibility="visible"
    tools:clickable="true" />

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:ems="10"
    android:layout_marginBottom="35dp"
    android:id="@+id/len_output"
    android:textAppearance="@style/TextAppearance.AppCompat.Body2"
    android:textSize="20sp"
    android:textColor="@android:color/black"
    android:clickable="false"
    android:hint="               Result"
    android:layout_above="@+id/con_button"
    android:layout_alignStart="@+id/len_input"
```

```
        tools:contextClickable="false" />
    </RelativeLayout>
```

### 4.3.3 – LENGTH CONVERRSION JAVA CODE: -

```
    package com.example.fahad.converton;

    import android.content.Context;
    import android.os.Bundle;
    import android.support.v7.app.AppCompatActivity;
    import android.view.Gravity;
    import android.view.View;
    import android.widget.AdapterView;
    import android.widget.ArrayAdapter;
    import android.widget.Button;
    import android.widget.EditText;
    import android.widget.Spinner;
    import android.widget.Toast;

    public class length extends AppCompatActivity implements View.OnClickListener,
    AdapterView.OnItemSelectedListener {

       private EditText in_length;
       private Spinner C2M;
       private Button Converto;
       private Spinner M2C;
       private EditText output;
       ArrayAdapter<String> unitarrayadapter;
       private Strategy currentStrategy;
       private String unitfrom;
       private String unitto;
       private static length instance;



       /**
```

```
     * ATTENTION: This was auto-generated to implement the App Indexing API.
     * See https://g.co/AppIndexing/AndroidStudio for more information.
     */

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_length);

        C2M = (Spinner) findViewById(R.id.len_spinner1);
        C2M.setOnItemSelectedListener(this);

        M2C = (Spinner) findViewById(R.id.len_spinner2);
        M2C.setOnItemSelectedListener(this);

        unitarrayadapter = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item);

unitarrayadapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown
_item);
        fillSpinnerWithLengthUnit();
        C2M.setAdapter(unitarrayadapter);
        M2C.setAdapter(unitarrayadapter);

        unitarrayadapter.setNotifyOnChange(true);

        output = (EditText) findViewById(R.id.len_output);
        output.setClickable(false);

        Converto = (Button) findViewById(R.id.con_button);
        Converto.setOnClickListener(this);

        in_length = (EditText) findViewById(R.id.len_input);

        currentStrategy = new LengthStrategy();
```

```
    instance = this;

}

private void fillSpinnerWithLengthUnit() {
    unitarrayadapter.clear();
    unitarrayadapter.add(getResources().getString(R.string.lengthunitmile));
    unitarrayadapter.add(getResources().getString(R.string.lengthunitkm));
    unitarrayadapter.add(getResources().getString(R.string.lengthunitm));
    unitarrayadapter.add(getResources().getString(R.string.lengthunitcm));
    unitarrayadapter.add(getResources().getString(R.string.lengthunitmm));
    unitarrayadapter.add(getResources().getString(R.string.lengthunitinch));
    unitarrayadapter.add(getResources().getString(R.string.lengthunitfeet));
    unitarrayadapter.notifyDataSetChanged();
}


public static length getInstance() {
    return instance;
}

public void onItemSelected(AdapterView<?> parent) {

}

public void onNothingSelected(AdapterView<?> parent) {
}

public void onItemSelected(AdapterView<?> parent, View v, int position, long id) {
    if (v.getParent() == C2M) {
        unitfrom = (String) (C2M.getSelectedItem().toString());
    }
    if (v.getParent() == M2C) {
        unitto = (String) (M2C.getSelectedItem().toString());
    }
```

```
    }

    public void onClick(View v) {
        if (v == Converto) {
            if (!in_length.getText().toString().equals("")) {
                double in = Double.parseDouble(in_length.getText().toString());
                double result = currentStrategy.Convert(unitfrom, unitto, in);
                if(result>=9999999)
                {
                    String resultmain = String.format(" %f", result);
                    output.setText(resultmain);
                }
                else
                    output.setText(Double.toString(result));
            } else {
                Context context = getApplicationContext();
                CharSequence text = "Enter Data";
                int duration = Toast.LENGTH_SHORT;
                Toast toast = Toast.makeText(context, text, duration);
                toast.setGravity(Gravity.BOTTOM| Gravity.CENTER_HORIZONTAL, 0, 20);
                toast.show();
            }
        }
    }
}
```

**4.3.4 – LENGTH TYPE COMPARISON JAVA CODE: -**

```
package com.example.fahad.converton;

import android.content.Context;
import android.view.Gravity;
import android.widget.Toast;

/**
 * Created by Fahad on 2/13/2017.
```

```
 */
public class LengthStrategy implements Strategy {
   public double Convert(String from, String to, double input) {
      // TODO Auto-generated method stub

      //Application app = UnitConverter.StrategyClass

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitkm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmile)))){
         //if((from.equals("km")) && (to.equals("mile"))){
         double ret = 0.62137*input;
         return ret;
      }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmile))                                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitkm)))){
         //if((from.equals("mile")) && (to.equals("km"))){
         double ret = 1.60934*input;
         return ret;
      }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmile))                                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitm)))){
         //if((from.equals("mile")) && (to.equals("m"))){
         double ret = 1609.34*input;
         return ret;
      }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
```

```
g.lengthunitm))                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmile))))){
        //if((from.equals("m")) && (to.equals("mile"))){
        double ret = input/1609.34;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmile))                                                 &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitcm))))){
        //if((from.equals("mile")) && (to.equals("cm"))){
        double ret = 160934*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitcm))                                                   &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmile))))){
        //if((from.equals("cm")) && (to.equals("mile"))){
        double ret = input/160934;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmile))                                                 &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmm))))){
        //if((from.equals("mile")) && (to.equals("mm"))){
        double ret = input*1609340;
        return ret;
    }
```

```
if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitmm)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitmile)))){
        //if((from.equals("mm")) && (to.equals("mile"))){
        double ret = input/1609340;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitmile)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitinch)))){
        //if ((from.equals("mile")) && (to.equals("inch"))){
        double ret = 63360*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitinch)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitmile)))){
        //if ((from.equals("inch")) && (to.equals("mile"))){
        double ret = input/63360;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitmile)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitfeet)))){
        //if((from.equals("mile")) && (to.equals("ft"))){
        double ret = 5280*input;
        return ret;
```

```
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitfeet))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmile))))){
        //if((from.equals("ft")) && (to.equals("mile"))){
        double ret = input/5280;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitkm))                                                                      &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitm))))){
        //if ((from.equals("km")) && (to.equals("m"))){
        double ret = input*1000;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitm))                                                                       &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitkm))))){
        //if((from.equals("m")) && (to.equals("km"))){
        double ret = 0.001*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitkm))                                                                      &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitcm))))){
        //if((from.equals("km")) && (to.equals("cm"))){
        double ret = 100000*input;
```

```
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitcm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitkm)))){
            //if((from.equals("cm")) && (to.equals("km"))){
            double ret = input/100000;
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitkm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmm)))){
            //if((from.equals("km")) && (to.equals("mm"))){
            double ret = 1000000*input;
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitkm)))){
            //if((from.equals("mm")) && (to.equals("km"))){
            double ret = input/1000000;
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitkm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitfeet)))){
            //if((from.equals("km")) && (to.equals("ft"))){
```

```
            double ret = input*3280.84;
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitfeet))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitkm)))){
            //if((from.equals("ft")) && (to.equals("km"))){
            double ret = input/3280.84;
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitkm))                                                                      &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitinch)))){
            //if((from.equals("km")) && (to.equals("inch"))){
            double ret = input*39370.1;
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitinch))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitkm)))){
            //if((from.equals("inch")) && (to.equals("km"))){
            double ret = input/39370.1;
            return ret;
        }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitm))                                                                       &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitcm)))){
```

```
        //if((from.equals("m")) && (to.equals("cm"))){
        double ret = 100*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitcm)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitm)))){
        //if((from.equals("cm")) && (to.equals("m"))){
        double ret = input/100;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitm)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitmm)))){
        //if((from.equals("m")) && (to.equals("mm"))){
        double ret = 1000*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitmm)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitm)))){
        //if((from.equals("mm")) && (to.equals("m"))){
        double ret = input/1000;
        return ret;
    }


if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitm)) &&
```

```
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitinch))){
        //if((from.equals("m")) && (to.equals("inch"))){
        double ret = 100*input/2.54;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitinch)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitm))){
        //if((from.equals("inch")) && (to.equals("m"))){
        double ret = 2.54*input/100;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitm)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitfeet))){
        //if((from.equals("m")) && (to.equals("ft"))){
        double ret = input*3.28084;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitfeet)) &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.lengthunitm))){
        //if((from.equals("ft")) && (to.equals("m"))){
        double ret = input/3.28084;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
```

```
g.lengthunitcm))                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmm))))){
        //if((from.equals("cm")) && (to.equals("mm"))){
        double ret = 10*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmm))                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitcm))))){
        //if((from.equals("mm")) && (to.equals("cm"))){
        double ret = input/10;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitinch))                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitcm))))){
        //if((from.equals("inch")) && (to.equals("cm"))){
        double ret = 2.54*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitcm))                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitinch))))){
        //if((from.equals("cm")) && (to.equals("inch"))){
        double ret = input/2.54;
        return ret;
    }
```

```
if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitcm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitfeet))))){
        //if((from.equals("cm")) && (to.equals("ft"))){
        double ret = input*0.03281;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitfeet))                                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitcm))))){
        //if((from.equals("ft")) && (to.equals("cm"))){
        double ret = input*30.48;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitfeet))))){
        //if((from.equals("mm")) && (to.equals("ft"))){
        double ret = 0.00328*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitfeet))                                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmm))))){
        //if((from.equals("ft")) && (to.equals("mm"))){
        double ret = input*304.8;
        return ret;
```

```
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitmm))                                                                    &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitinch)))){
        //if((from.equals("mm")) && (to.equals("inch"))){
        double ret = input/25.4;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitinch))                                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitmm))){
        //if((from.equals("inch")) && (to.equals("mm"))){
        double ret = input*25.4;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitfeet))                                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitinch)))){
        //if((from.equals("ft")) && (to.equals("inch"))){
        double ret = 12*input;
        return ret;
    }

if((from.equals(length.getInstance().getApplicationContext().getResources().getString(R.strin
g.lengthunitinch))                                                                  &&
to.equals(length.getInstance().getApplicationContext().getResources().getString(R.string.leng
thunitfeet))){
        //if((from.equals("inch")) && (to.equals("ft"))){
        double ret = input/12;
```

```
            return ret;
        }
            if(from.equals(to)){
            Context context = length.getInstance().getApplicationContext();
            CharSequence text = "Same Types Selected";
            int duration = Toast.LENGTH_SHORT;
            Toast toast = Toast.makeText(context, text, duration);
            toast.setGravity(Gravity.BOTTOM| Gravity.CENTER_HORIZONTAL, 0, 20);
            toast.show();
            return input;
        }
        return 0.0;
    }
}
```

## 4.4 Results and Reports

**Home Screen:**

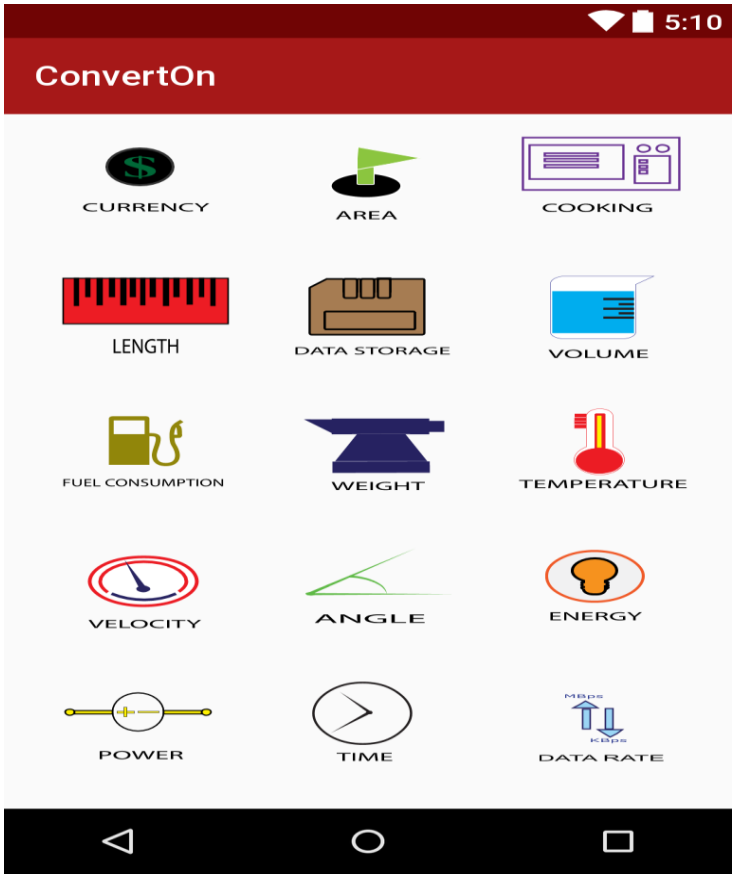



**Fig no: 4.4.1**

**Conversion Phase:**

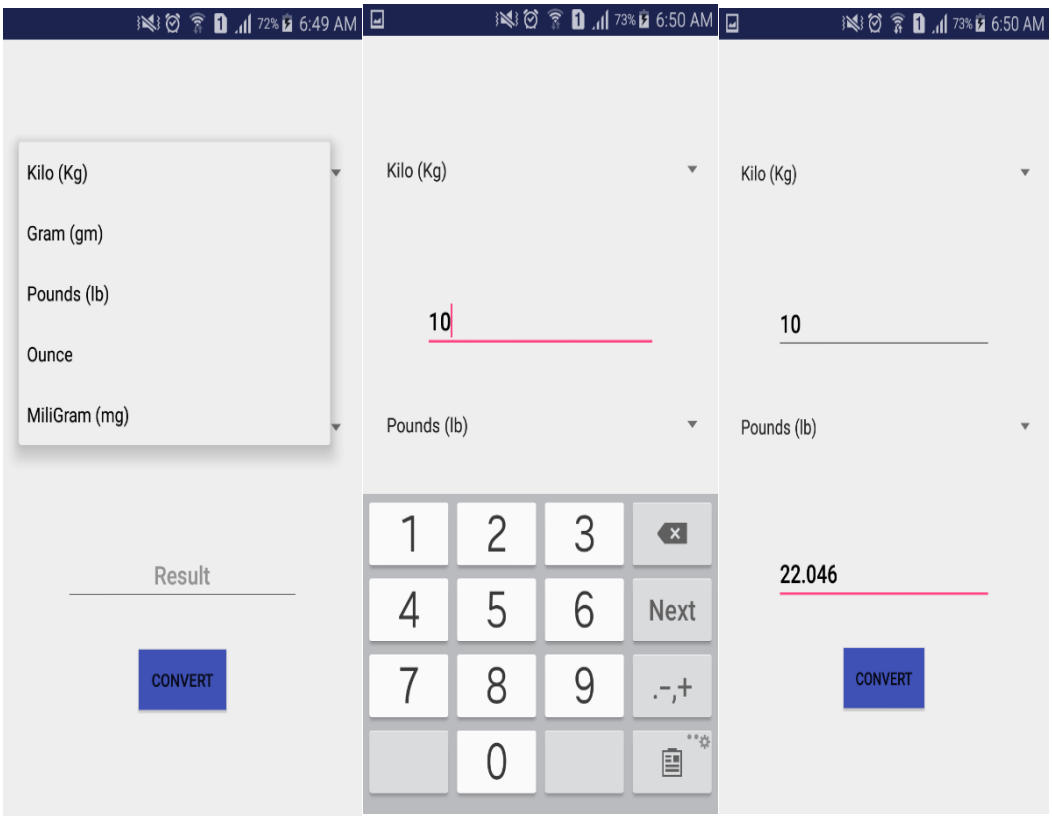



**Fig no: 4.4.2**

### 4.5 Cost Estimation

The constructive cost model was developed by Barry W.Boehm in the late 1970s and published in Boehm's 1981 book Software Engineering Economics as a model for estimating effort, cost, and schedule for software projects. It drew on a study of 63 projects at TRW Aerospace where Boehm was Director of Software Research and Technology. The study examined projects ranging in size from 2,000 to 100,000 lines of code, and programming languages ranging from assembly to PL/I. These projects were based on the waterfall model of software development which was the prevalent software development process in 1981.

COCOMO consists of a hierarchy of three increasingly detailed and accure forms. The first level, Basic COCOMO is good for quick, early, rough order of magnitude estimates of software costs

COCOMO applies to three modes:

**Organic projects** - "small" teams with "good" experience working with "less than rigid" requirements

**Semi-detached projects** - "medium" teams with mixed experience working with a mix of rigid and less than rigid requirements

**Embedded projects** - developed within a set of "tight" constraints (hardware, software, operational)

- ✓ **Project Duration :**

**TDEV = $c(E)^d$**

Where,

TDEV is time for development

C and d are constants to be determined

E is the effort

## Organic Mode :

The project is developed in a familiar, stable environment, and the product is similar to previously developed products. The product is relatively small, and requires little innovation.It is relatively very small, simple software projects in which a small team with good application experience work to a set of less than rigid requirement. The equation is :

- ✓ E = 626 staff months
- ✓ TDEV = $2.5(626)^{0.38}$ = 29months

## Effort Applied (E) = $a_b(KLOC)^{b}{}_b$ [ man-months ] :

Where,

E is the effort in the staff months

A and b are coefficients to be determine.

KLOC is thousands of lines of code

## Average Staff size:

$$SS = \frac{E}{TDEV} = \frac{[\text{Staff - } \sout{\text{months}}]}{[\sout{\text{months}}]} = [\text{Staff}]$$

## Productivity:

P = Size [KLOC]

$\frac{\text{KLOC}}{\text{E}} = \frac{\text{KLOC}}{\text{[Staff – months]}} = \text{KLOC/ Staff – month}$

**Table no: 4.5.1**

| NO | Item | Organic mode |
|---|---|---|
| 1 | Effort (Staff - month) | 44 |
| 2 | Development time | 11 |
| 3 | Average Staff | 4 |
| 4 | Productivity | 375 |

# CHAPTER 5
# FUTURE WORK

# FUTURE WORK

- ✓ Data redundancy will be remove.
- ✓ More Categories will be added.
- ✓ Search Tab will be added.
- ✓ Currency will be Dynamic.
- ✓ More Units will be added.
- ✓ GUI will Improve.
- ✓ Favourites Tab will be added.
- ✓ File Conversion will be provided.

# CHAPTER 6
# CONCLUSION

# CONCLUSION

Unit converter was designed with the goal of supporting wide range of devices from small screen phone devices to large screen tablets plus it will feature worldwide languages and their conversion system. We aim to add and support more range of units to the application in the near future. This project has helped us a lot in learning different languages, how to tackle problems, work in group and many other things that will be beneficial to us in the near future and in our careers.

We plan to add more features mentioned above in our existing project and make it more useful for the people.

# CHAPTER 8
# BIBLOGRAPHY & REFERENCES

# REFERENCES AND BIBLIOGRAPHY

## Web sites:


- ✓ https://www.google.co.in
- ✓ http://www.wikipedia.com
- ✓ http://www.youtube.com
- ✓ http://stackoverflow.com
- ✓ https://developer.android.com