Department of Computer Science and Engineering

Punjabi University

Live Updates App

(Live weather + Corona updates + News Feed)

SUBMITTED BY

Kanishka sahu	**11701173**
Vivek sharma	**11701190**
Simarpreet singh	**11701191**
Manvesh Liddar	**11761005**

SUBMITTED TO

Dr. Gaurav Gupta

Project Source code - https://github.com/sahukanishka/Live-Update-College-Project-.git

CONTENTS

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to several individuals and organizations for supporting us throughout our Graduate study. First, we wish to express our sincere gratitude to our Professor Dr. Gaurav Gupta , for his enthusiasm, patience, insightful comments, helpful information, practical advice and unceasing ideas that have helped us tremendously at all times in our projects. His immense knowledge, profound experience and professional expertise in Android development and classroom lectures help us to complete this Project . Without his support and guidance, this project would not have been possible.

Although due to lockdown we are not able to attend the classes but we used our time in a great manner and learned a lot. We worked remotely on this project.

Finally, we would like to say that we are indebted to our parents for everything that they have done for me. All of this would have been impossible without their constant support. And we also thank God for being kind to us and driving us through this journey.

Thank you

Kanishka sahu	11701173
Vivek sharma	11701190
Simarpreet singh	11701191
Manvesh Liddar	11761005

Abstract

The main motive to take this topic as a project is to create one app that has all the daily updates that we have to know. Like due to the coronavirus pandemic everyone wants to know the exact real time data about the pandemic. So for this we are taking live data from the open API and parsing the JSON data in the app so you can get a live update with a single click. The API updates the server in every 10 minutes you will get the real time data about the coronavirus. Weather is very common and day to day uses of features in today’s era. We planned out upcoming events according to the weather forecast and that's why it's very useful information that we all need. So here we are parsing the live data from the open weather API through our App. You can also search your city from the app and it will show you the current weather of your city. We also included news feeds directly from google with Web View so you can scroll the news all around the world or you can login for personal recommending news feeds.
Currently our have 3 fragments and you can navigate all of them from the Bottom Navigation Bar.

Key features --

- Live weather update
- Search your city’s current weather all around the world
- Live corona pandemic tracker
- Real time Data about the corona virus from all around the world
- Single touch news feed to scroll the information

Attractive points ---

- Android Lollipop 5.0 (It gives the power to run approximately 100% of the devices)
- Fragment view for fast Navigation and experience
- Material UI that gives great user experience
- WebView
- Volley library that makes networking very fast (HTTP)
- REST json API for Real time Data parsing

Technologies Used

1. JAVA
2. XML
3. Android
4. REST API

1.JAVA

1.1 What is Java?

Java is a popular programming language, created in 1995.

It is owned by Oracle, and more than 3 billion devices run Java.

It is used for:

- Mobile applications (specially Android apps)
- Desktop applications
- Web applications
- Web servers and application servers
- Games
- Database connection
- And much, much more!

1.2 Why Use Java?

- Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
- It is one of the most popular programming language in the world
- It is easy to learn and simple to use

- It is open-source and free
- It is secure, fast and powerful
- It has a huge community support (tens of millions of developers)
- Java is an object oriented language which gives a clear structure to programs and allows code to be reused, lowering development costs
- As Java is close to C++ and C#, it makes it easy for programmers to switch to Java or vice versa

1.3 Applications of Java Programming

The latest release of the Java Standard Edition is Java SE 8. With the advancement of Java and its widespread popularity, multiple configurations were built to suit various types of platforms. For example: J2EE for Enterprise Applications, J2ME for Mobile Applications.

The new J2 versions were renamed as Java SE, Java EE, and Java ME respectively. Java is guaranteed to be Write Once, Run Anywhere.

- Multithreaded − With Java's multithreaded feature it is possible to write programs that can perform many tasks simultaneously. This design feature allows the developers to construct interactive applications that can run smoothly.
- Interpreted − Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light-weight process.
- High Performance − With the use of Just-In-Time compilers, Java enables high performance.
- Distributed − Java is designed for the distributed environment of the internet.

- Dynamic − Java is considered to be more dynamic than C or C++ since it is designed to adapt to an evolving environment. Java programs can carry an extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.

1.4 Hello World using Java Programming

```
public class MyFirstJavaProgram {
   /* This is my first java program.
    * This will print 'Hello World' as the output
    */
   public static void main(String []args) {
      System.out.println("Hello World"); // prints Hello World
   }
}
```

2. XML

2.1 What is XML?

XML stands for Extensible Markup Language. It is a text-based markup language derived from Standard Generalized Markup Language (SGML).

XML tags identify the data and are used to store and organize the data, rather than specifying how to display it like HTML tags, which are used to display the data. XML is not going to replace HTML in the near future, but it introduces new possibilities by adopting many successful features of HTML.

2.2 Why XML?

There are three important characteristics of XML that make it useful in a variety of systems and solutions −

- XML is extensible − XML allows you to create your own self-descriptive tags, or language, that suits your application.
- XML carries the data, does not present it − XML allows you to store the data irrespective of how it will be presented.
- XML is a public standard − XML was developed by an organization called the World Wide Web Consortium (W3C) and is available as an open standard.

2.3 What is Markup?

XML is a markup language that defines set of rules for encoding documents in a format that is both human-readable and machine-readable. So *what exactly is a markup language?* Markup is information added to a document that enhances its meaning in certain ways, in that it identifies the parts and how they relate to each other. More specifically, a markup language is a set of symbols that can be placed in the text of a document to demarcate and label the parts of that document.

Following example shows how XML markup looks, when embedded in a piece of text −

```
<message>

  <text>Hello, world!</text>

</message>
```

This snippet includes the markup symbols, or the tags such as <message>...</message> and <text>... </text>. The tags <message> and </message> mark the start and the end of the XML code fragment. The tags <text> and </text> surround the text Hello, world!.

3. Android

3.1 What is Android?

Android is an operating system and programming platform developed by Google for smartphones and other mobile devices (such as tablets). It can work on many different devices from many different manufacturers. Android installs software development code for writing original code and integrating software modules to create apps for Android users. Also it provides a marketplace for distribution apps. All together, Android represents the ecosystem of mobile apps.

3.2 Why to make android apps?

Apps are created for a variety of reasons: addressing business needs, building new services, creating new ones businesses, and providing games and other content for users. Developers prefer to upgrade Android respectively to reach the majority of mobile device users.

3.2.1 The most popular platform for mobile applications

As the world's most popular mobile platform, Android enables over 190 million mobile devices. countries around the world. It has the largest installed base of any mobile platform and is growing fast. Every day another million users install their Android devices for the first time and start searching for apps, games, and other digital content.

3.2.2 A great experience for app users

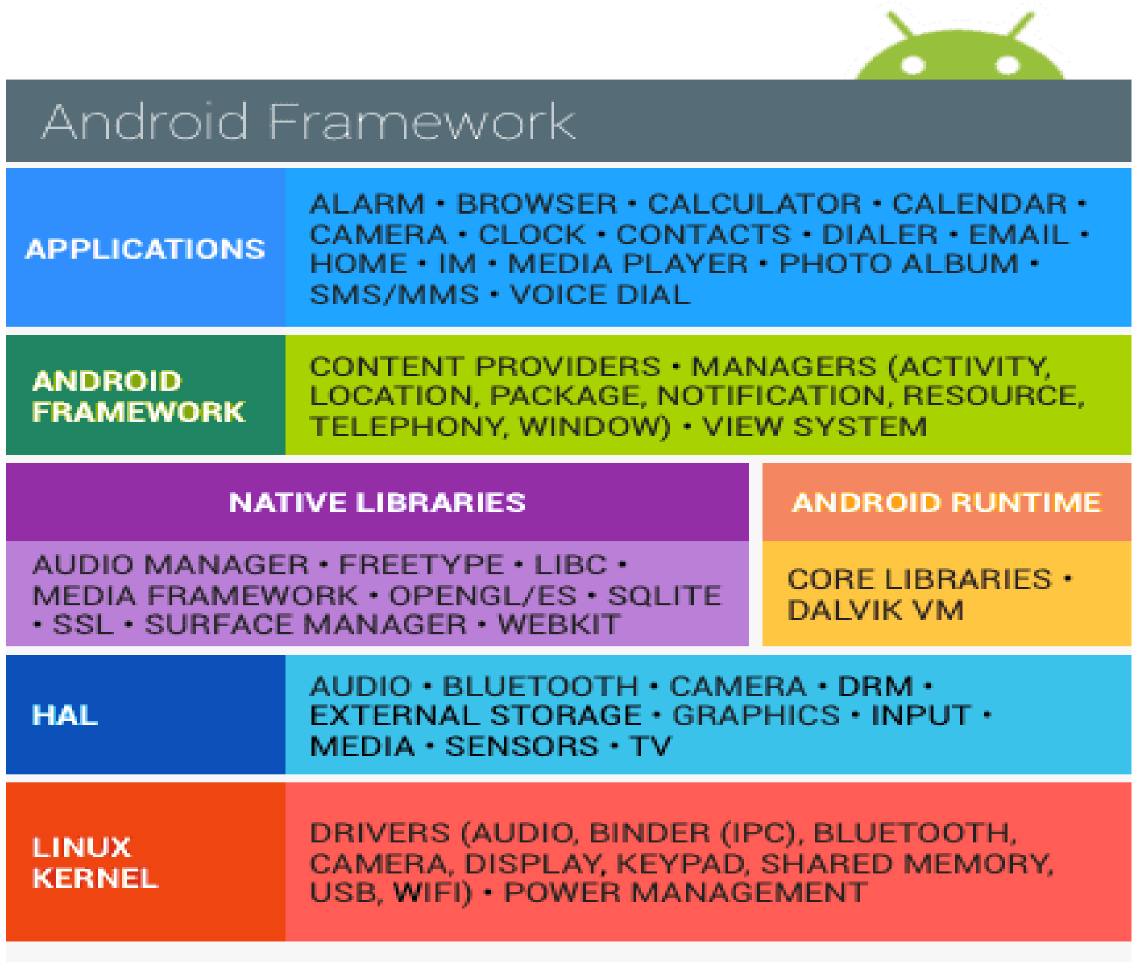
Android provides a user interface for touch-screen interface (UI) in conjunction with applications. Android user experience is largely based to deceive directly in addition to the keyboard, there is a virtual keyboard for custom text input. Android can also support game controllers and full-size keyboards connected via Bluetooth or USB.

3.2.3 It's easy to develop apps

Use the Android software development kit (SDK) to develop apps that take advantage of the Android app as well UI. The SDK includes a complete set of development tools including debugger, written software libraries code, device emulator, documentation, sample code, and tutorial. Use these tools to create apps that look great too use the hardware power available on each device.

To develop applications using the SDK, use Java programming language for application development and Boost capacity
Language files (XML) for describing data resources. By writing code in Java and building a single binary application, you will be able to
have an app that can work on both phone and tablet features. You can declare your UI with lightweight XML sets
resources, one set of UI components common to all aspects of methods, and other sets of specific features in the literature or
pills. At launch, Android uses a set of relevant sources based on its screen size, quantity, location, and so on.

3.3 Android Architecture


Android Framework
APPLICATIONS
ALARM • BROWSER • CALCULATOR • CALENDAR • CAMERA • CLOCK • CONTACTS • DIALER • EMAIL • HOME • IM • MEDIA PLAYER • PHOTO ALBUM • SMS/MMS • VOICE DIAL
ANDROID FRAMEWORK
CONTENT PROVIDERS • MANAGERS (ACTIVITY, LOCATION, PACKAGE, NOTIFICATION, RESOURCE, TELEPHONY, WINDOW) • VIEW SYSTEM
NATIVE LIBRARIES
AUDIO MANAGER • FREETYPE • LIBC • MEDIA FRAMEWORK • OPENGL/ES • SQLITE • SSL • SURFACE MANAGER • WEBKIT
ANDROID RUNTIME
CORE LIBRARIES • DALVIK VM
HAL
AUDIO • BLUETOOTH • CAMERA • DRM • EXTERNAL STORAGE • GRAPHICS • INPUT • MEDIA • SENSORS • TV
LINUX KERNEL
DRIVERS (AUDIO, BINDER (IPC), BLUETOOTH, CAMERA, DISPLAY, KEYPAD, SHARED MEMORY, USB, WIFI) • POWER MANAGEMENT

3.3.1 The Linux Kernel

The foundation of the Android platform is the Linux kernel. For example, the Android Runtime (ART) relies on the Linux kernel for underlying functionalities such as threading and low-level memory management.

Using a Linux kernel allows Android to take advantage of key security features and allows device manufacturers to develop hardware drivers for a well-known kernel.

3.3.2 Hardware Abstraction Layer (HAL)

The hardware abstraction layer (HAL) provides standard interfaces that expose device hardware capabilities to the higher-level Java API framework. The HAL consists of multiple library modules, each of which implements an interface for a specific type of hardware component, such as the camera or bluetooth module. When a framework API makes a call to access device hardware, the Android system loads the library module for that hardware component.

3.3.3 Android Runtime

For devices running Android version 5.0 (API level 21) or higher, each app runs in its own process and with its own instance of the Android Runtime (ART). ART is written to run multiple virtual machines on low-memory devices by executing DEX files, a bytecode format designed specially for Android that's optimized for minimal memory footprint. Build toolchains, such as Jack, compile Java sources into DEX bytecode, which can run on the Android platform.

Some of the major features of ART include the following:

- Ahead-of-time (AOT) and just-in-time (JIT) compilation

- Optimized garbage collection (GC)
- On Android 9 (API level 28) and higher, conversion of an app package's Dalvik Executable format (DEX) files to more compact machine code.
- Better debugging support, including a dedicated sampling profiler, detailed diagnostic exceptions and crash reporting, and the ability to set watchpoints to monitor specific fields

Prior to Android version 5.0 (API level 21), Dalvik was the Android runtime. If your app runs well on ART, then it should work on Dalvik as well, but the reverse may not be true.

Android also includes a set of core runtime libraries that provide most of the functionality of the Java programming language, including some Java 8 language features, that the Java API framework uses.

3.3.4. Native C/C++ Libraries

Many core Android system components and services, such as ART and HAL, are built from native code that require native libraries written in C and C++. The Android platform provides Java framework APIs to expose the functionality of some of these native libraries to apps. For example, you can access OpenGL ES through the Android framework's Java OpenGL API to add support for drawing and manipulating 2D and 3D graphics in your app.

If you are developing an app that requires C or C++ code, you can use the Android NDK to access some of these native platform libraries directly from your native code.

3.3.5. Java API Framework

The entire feature-set of the Android OS is available to you through APIs written in the Java language. These APIs form the building blocks you need to create Android apps by simplifying the reuse of core, modular system components and services, which include the following:

A rich and extensible View System you can use to build an app’s UI, including lists, grids, text boxes, buttons, and even an embeddable web browser
A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files
A Notification Manager that enables all apps to display custom alerts in the status bar
An Activity Manager that manages the lifecycle of apps and provides a common navigation back stack
Content Providers that enable apps to access data from other apps, such as the Contacts app, or to share their own data
Developers have full access to the same framework APIs that Android system apps use.

3.3.6. System Apps

Android comes with a set of core apps for email, SMS messaging, calendars, internet browsing, contacts, and more. Apps included with the platform have no special status among the apps the user chooses to install. So a third-party app can become the user's default web browser, SMS messenger, or even the default keyboard (some exceptions apply, such as the system's Settings app).

The system apps function both as apps for users and to provide key capabilities that developers can access from their own app. For example, if your app would like to deliver an SMS message, you don't need to build that functionality yourself—you can instead invoke whichever SMS app is already installed to deliver a message to the recipient you specify.

3.3.7 Android Versions :

Cupcake
1.5
Donut
1.6
Eclair
2.0/2.1
Froyo
2.2
Gingerbread
2.3
Honeycomb
3.0/3.1
Ice Cream Sandwich
4.0
Android Versions List
#hikkart
Jelly Bean
4.1/4.2/4.3
KitKat
4.4
Lollipop
5.0
Marshmallow
6.0
Nougat
7.0
Oreo
8.0
Pie
9.0

Code name	Version numbers	API level	Release date
No codename	1.0	1	September 23, 2008
No codename	1.1	2	February 9, 2009
Cupcake	1.5	3	April 27, 2009
Donut	1.6	4	September 15, 2009
Eclair	2.0 - 2.1	5 - 7	October 26, 2009
Froyo	2.2 - 2.2.3	8	May 20, 2010
Gingerbread	2.3 - 2.3.7	9 - 10	December 6, 2010
Honeycomb	3.0 - 3.2.6	11 - 13	February 22, 2011
Ice Cream Sandwich	4.0 - 4.0.4	14 - 15	October 18, 2011
Jelly Bean	4.1 - 4.3.1	16 - 18	July 9, 2012

KitKat	4.4 - 4.4.4	19 - 20	October 31, 2013
Lollipop	5.0 - 5.1.1	21- 22	November 12, 2014
Marshmallow	6.0 - 6.0.1	23	October 5, 2015
Nougat	7.0	24	August 22, 2016
Nougat	7.1.0 - 7.1.2	25	October 4, 2016
Oreo	8.0	26	August 21, 2017
Oreo	8.1	27	December 5, 2017
Pie	9.0	28	August 6, 2018
Android 10	10.0	29	September 3, 2019

3.4 The challenges of Android app development

While the Android platform provide rich functionality for app development, there are still a number of challenges you need to address, such as:

- Building for a multi-screen world
- Getting performance right
- Keeping your code and your users secure
- Remaining compatible with older platform versions
- Understanding the market and the user.

The development process
An Android app project begins with an idea and a definition of the requirements necessary to realize that idea. As the
project progresses, it goes through design, development, and testing.
The above diagram is a high-level picture of the development process, with the following steps:

1. Defining the idea and its requirements: Most apps start with an idea of what it should do, bolstered by market and user research. During this stage the app's requirements are defined.
2. Prototyping the user interface: Use drawings, mock ups and prototypes to show what the user interface would look like, and how it would work.
3. Developing and testing the app: An app consists of one or more activities. For each activity you can use Android
   Studio to do the following, in no particular order:
    - Create the layout: Place UI elements on the screen in a layout, and assign string resources and menu items, using
    - the Extensible Markup Language (XML).

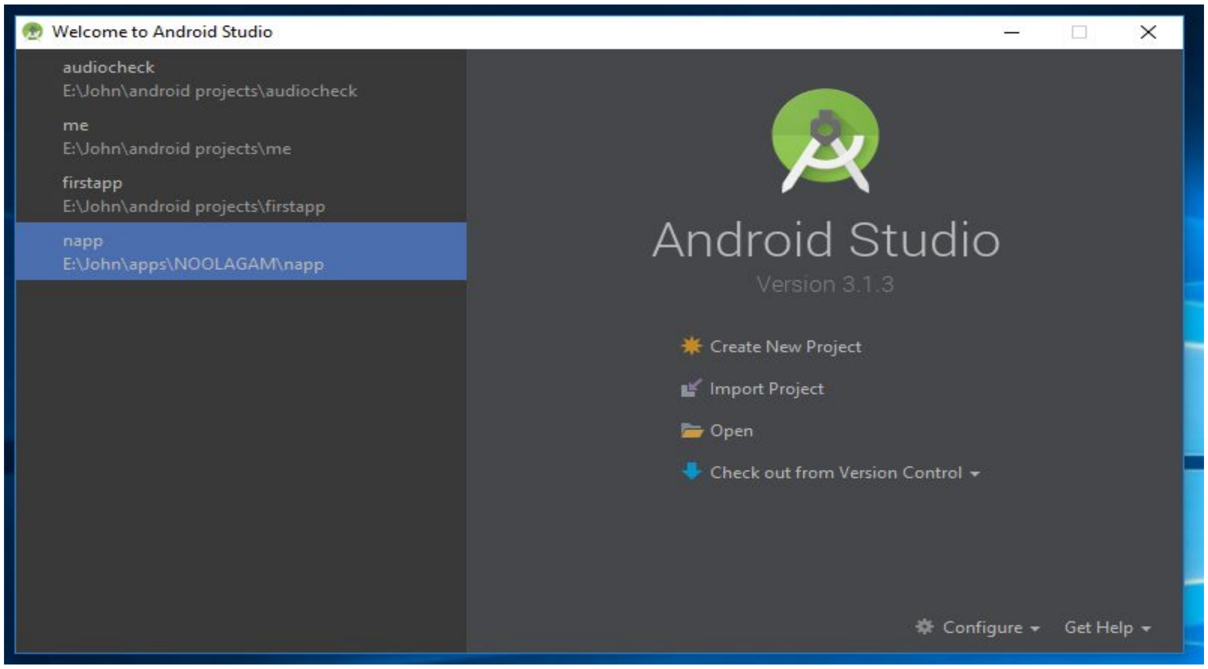
- Write the Java code: Create source code for components and tests, and use testing and debugging tools.
- Register the activity: Declare the activity in the manifest file.
- Define the build: Use the default build configuration or create custom builds for different versions of your app.

4. Publishing the app: Assemble the final APK (package file) and distribute it through channels such as the Google Play

3.5 Using Android Studio

Android Studio provides tools for the testing, and publishing phases of the development process, and a unified development environment for creating apps for all Android devices. The development environment includes code templates with sample code for common app features, extensive testing tools and frameworks, and a flexible build system.

3.5.1 Starting an Android Studio project

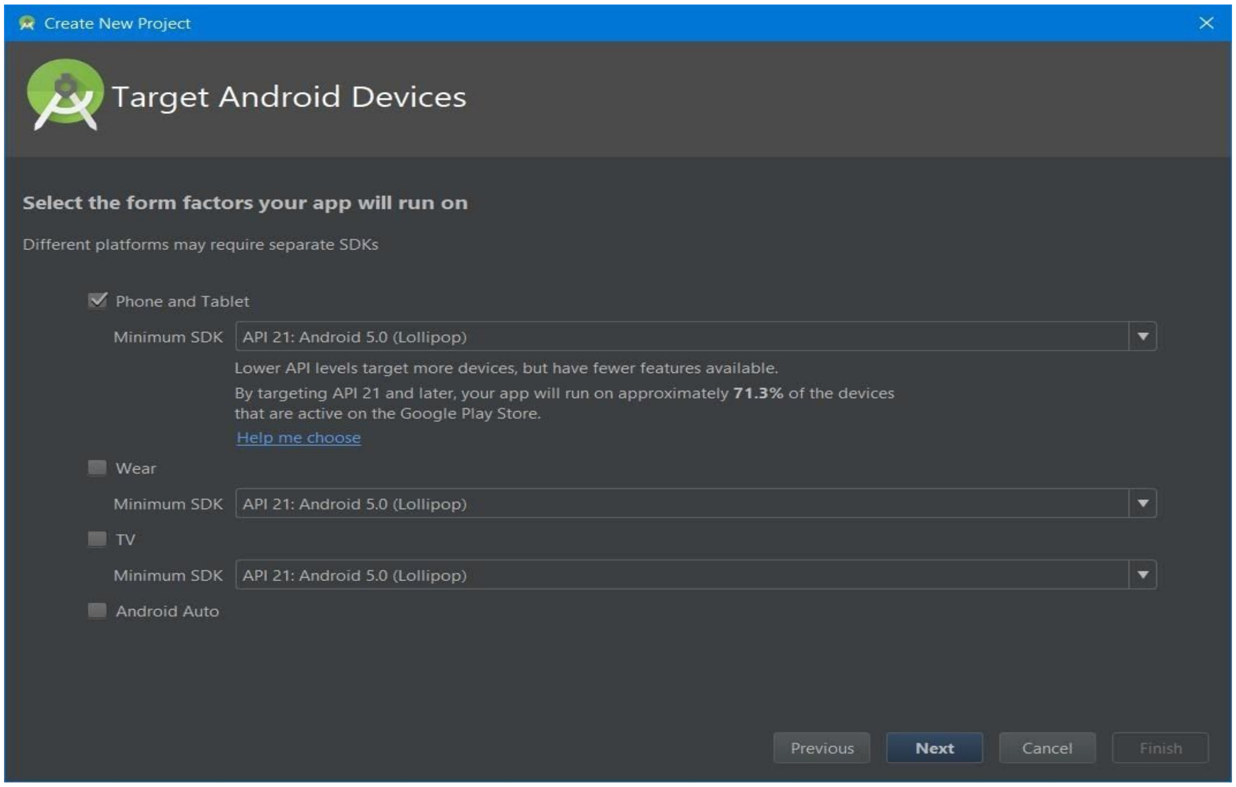

Welcome to Android Studio
audiocheck
E:\John\android projects\audiocheck
me
E:\John\android projects\me
firstapp
E:\John\android projects\firstapp
napp
E:\John\apps\NOOLAGAM\napp
Android Studio
Version 3.1.3
Create New Project
Import Project
Open
Check out from Version Control
Configure
Get Help

After you have successfully installed the Android Studio IDE, double-click the Android Studio application icon to start it.

Choose Start a new Android Studio project in the Welcome window, and name the project the same name that you want to use for the app.

When choosing a unique Company Domain, keep in mind that apps published to the Google Play must have a unique package name. Since domains are unique, prepending the app's name with your name, or your company's domain name, should provide an adequately unique package name. If you are not planning to publish the app, you can accept the default
example domain. Be aware that changing the package name later is extra work.

3.5.2 Choosing target devices and the minimum SDK



Select the form factors your app will run on

Different platforms may require separate SDKs

☑ Phone and Tablet

Minimum SDK: API 21: Android 5.0 (Lollipop)

Lower API levels target more devices, but have fewer features available.
By targeting API 21 and later, your app will run on approximately **71.3%** of the devices that are active on the Google Play Store.

Help me choose

☐ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ Android Auto

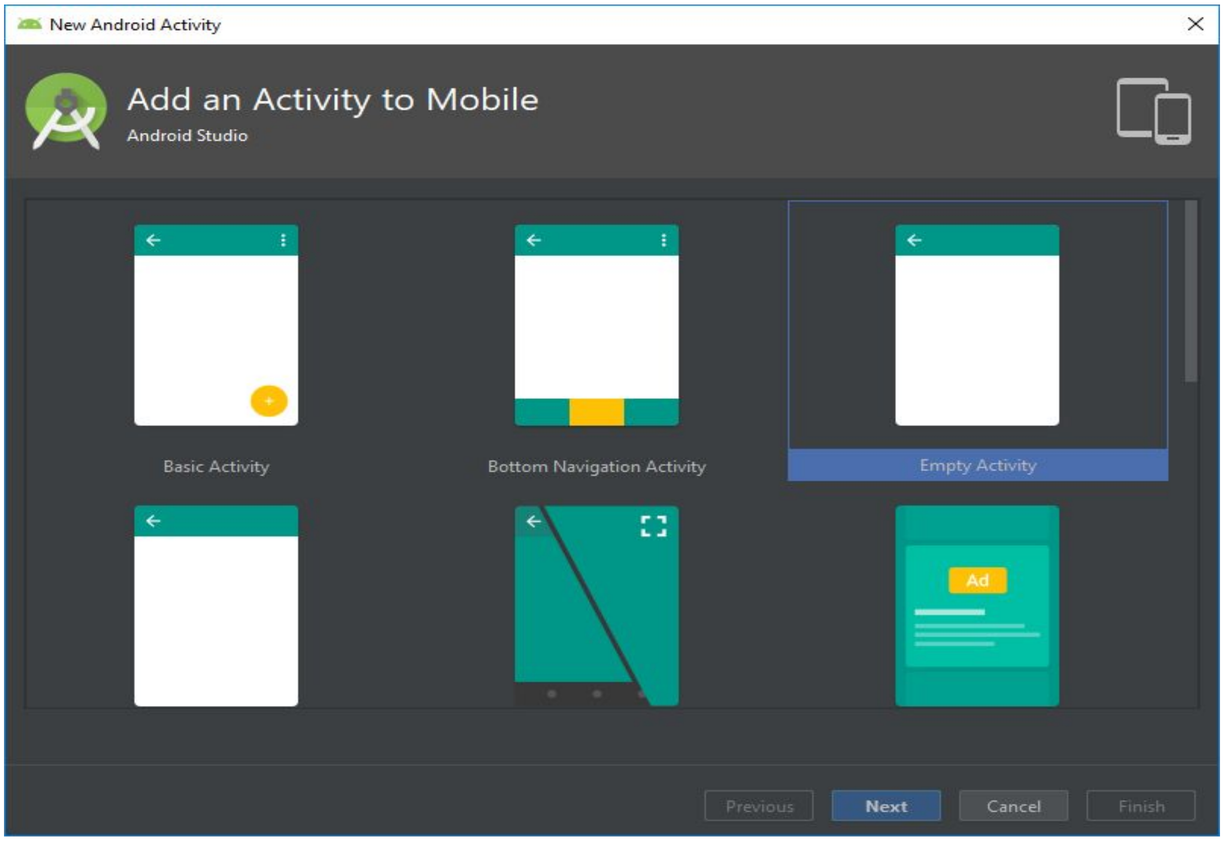
When choosing Target Android Devices, Phone and Tablet are selected by default, as shown in the figure below. The choice shown in the figure for the Minimum SDK — API 15: Android 4.0.3 (IceCreamSandwich) — makes your app compatible with 97% of Android devices active on the Google Play Store.

Different devices run different versions of the Android system, such as Android 4.0.3 or Android

version often adds new APIs not available in the previous version. To indicate which set of APIs are available, each version specifies an API level. For instance, Android 1.0 is API level 1 and Android 4.0.3 is API level 15.

The Minimum SDK declares the minimum Android version for your app. Each successive version of Android provides compatibility for apps that were built using the APIs from previous versions, so your app should always be compatible with future versions of Android while using the documented Android APIs.

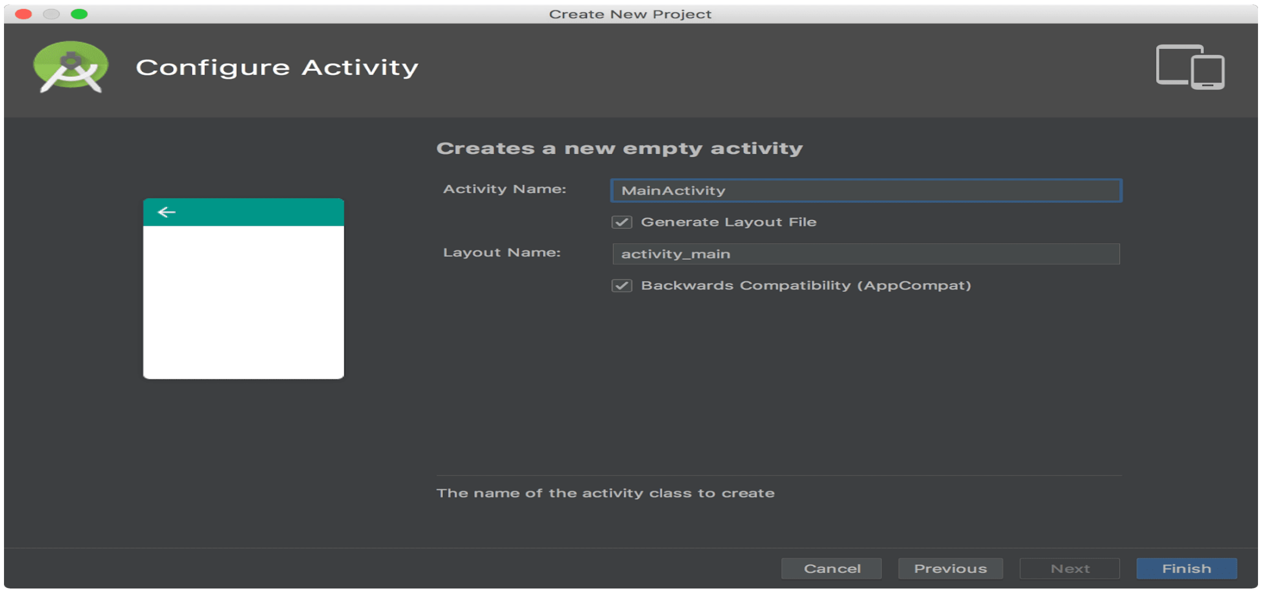
3.5.3 Choosing a template



Android Studio pre-populates your project with minimal code for an activity and a screen layout based on a template. A variety of templates are available, ranging from a virtually blank template (Add No Activity) to various types of activities.
You can customize the activity after choosing your template. For example, the Empty Activity template provides a single activity accompanied by a single layout resource for the screen. You can choose to accept the commonly used name for the activity (such as MainActivity) or change the name on the Customize the Activity screen. Also, if you use the Empty Activity template, be sure to check the following if they are not already checked:

- Generate Layout file: Leave this checked to create the layout resource connected to this activity, which is usually named activity_main.xml. The layout defines the user interface for the activity.

- Backwards Compatibility (AppCompat): Leave this checked to include the AppCompat library so that the app is compatible with previous versions of Android even if it uses features found only in newer versions.



Create New Project

Configure Activity

Creates a new empty activity

Activity Name: MainActivity

☑ Generate Layout File

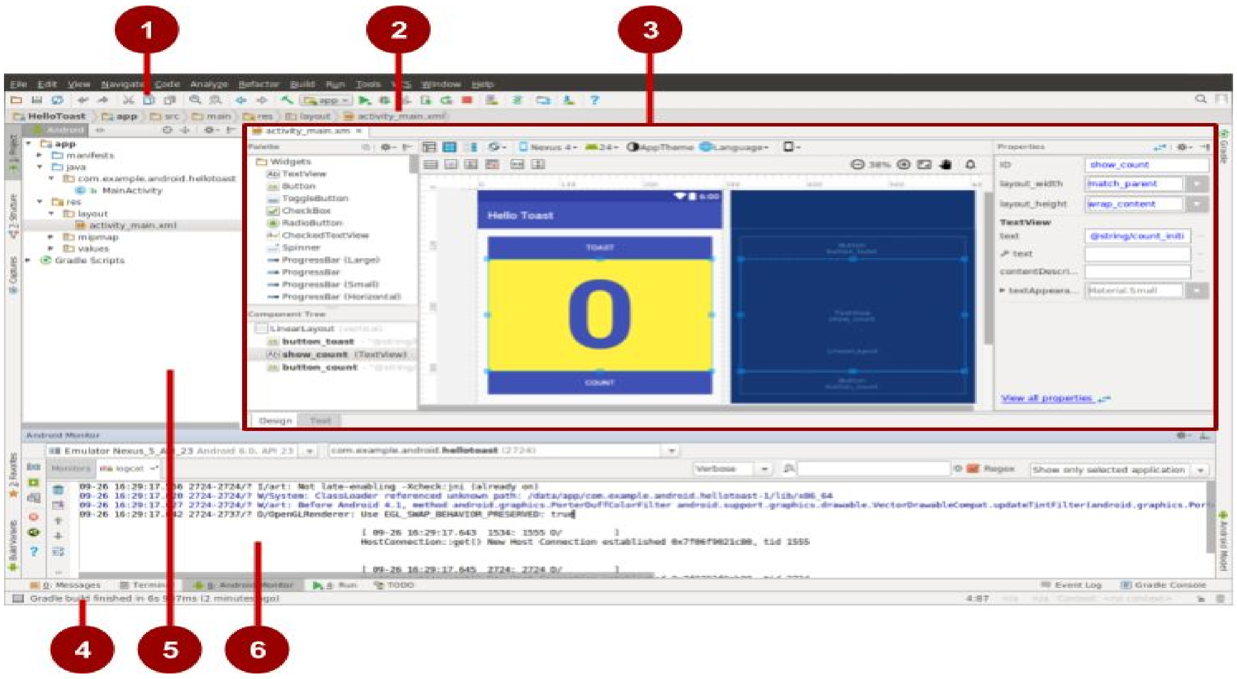
Layout Name: activity_main

☑ Backwards Compatibility (AppCompat)

The name of the activity class to create

Android Studio creates a folder for the newly created project in the AndroidStudioProjects folder on your computer.

3.5.4 Android Studio window panes



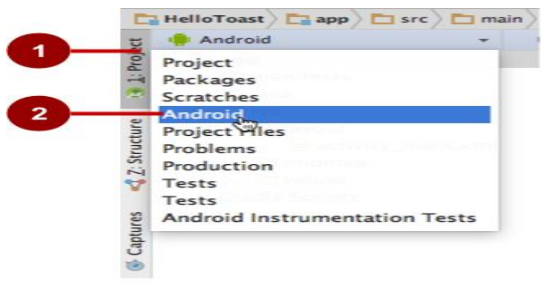
```
09-26 16:29:17.?06 2724-2724/? I/art: Not late-enabling -Xcheck:jni (already on)
09-26 16:29:17.?20 2724-2724/? W/System: ClassLoader referenced unknown path: /data/app/com.example.android.hellotoast-1/lib/x86_64
09-26 16:29:17.?27 2724-2724/? W/art: Before Android 4.1, method android.graphics.PorterDuffColorFilter android.support.graphics.drawable.VectorDrawableCompat.updateTintFilter(android.graphics.Por
09-26 16:29:17.?42 2724-2737/? D/OpenGLRenderer: Use EGL_SWAP_BEHAVIOR_PRESERVED: true

                                          [ 09-26 16:29:17.643  1534: 1555 D/         ]
                                          HostConnection::get() New Host Connection established 0x7f86f9021c80, tid 1555
```

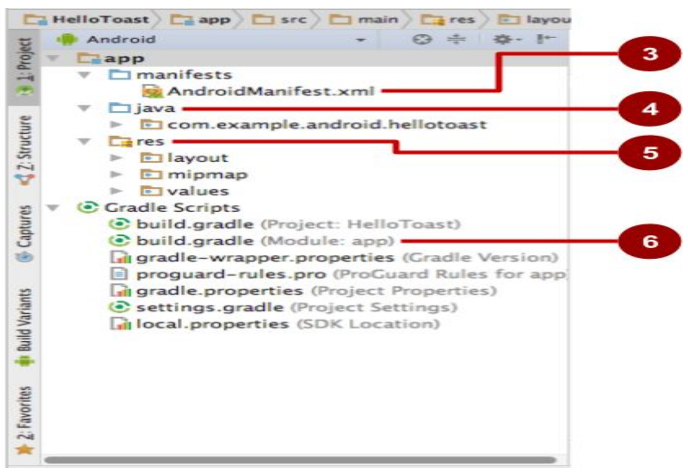
Gradle build finished in 6s 5?7ms (2 minutes ago)

The Android Studio main window is made up of several logical areas, or panes, as shown in the figure below. In the above figure:

1. The Toolbar.The toolbar carries out a wide range of actions, including running the Android app and launching Android tools.
2. The Navigation Bar. The navigation bar allows navigation through the project and open files for editing. It provides a more compact view of the project structure.
3. The Editor Pane. This pane shows the contents of a selected file in the project. For example, after selecting a layout (as shown in the figure), this pane shows the layout editor with tools to edit the layout. After selecting a Java code file, this pane shows the code with tools for editing the code.
4. The Status Bar. The status bar displays the status of the project and Android Studio itself, as well as any warnings or messages. You can watch the build progress in the status bar.
5. The Project Pane. The project pane shows the project files and project hierarchy.
6. The Monitor Pane. The monitor pane offers access to the TODO list for managing tasks, the Android Monitor for monitoring app execution (shown in the figure), the logcat for viewing log messages, and the Terminal application for performing Terminal activities.

3.5.5 Exploring a project




HelloToast
app
src
main
res
layou
Android
app
manifests
AndroidManifest.xml
java
com.example.android.hellotoast
res
layout
mipmap
values
Gradle Scripts
build.gradle (Project: HelloToast)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules for app)
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)
1: Project
Z: Structure
Captures
Build Variants
2: Favorites
3
4
5
6

Each project in Android Studio contains the AndroidManifest.xml file, component source-code files, and associated resource files. By default, Android Studio organizes your project files based on the file type, and displays them within the Project: Android view in the left tool pane, as shown below. The view provides quick access to your project's key files.

To switch back to this view from another view, click the vertical Project tab in the far left column of the Project pane, and choose Android from the pop-up menu at the top of the Project pane, as shown in the figure below. In the figure above:

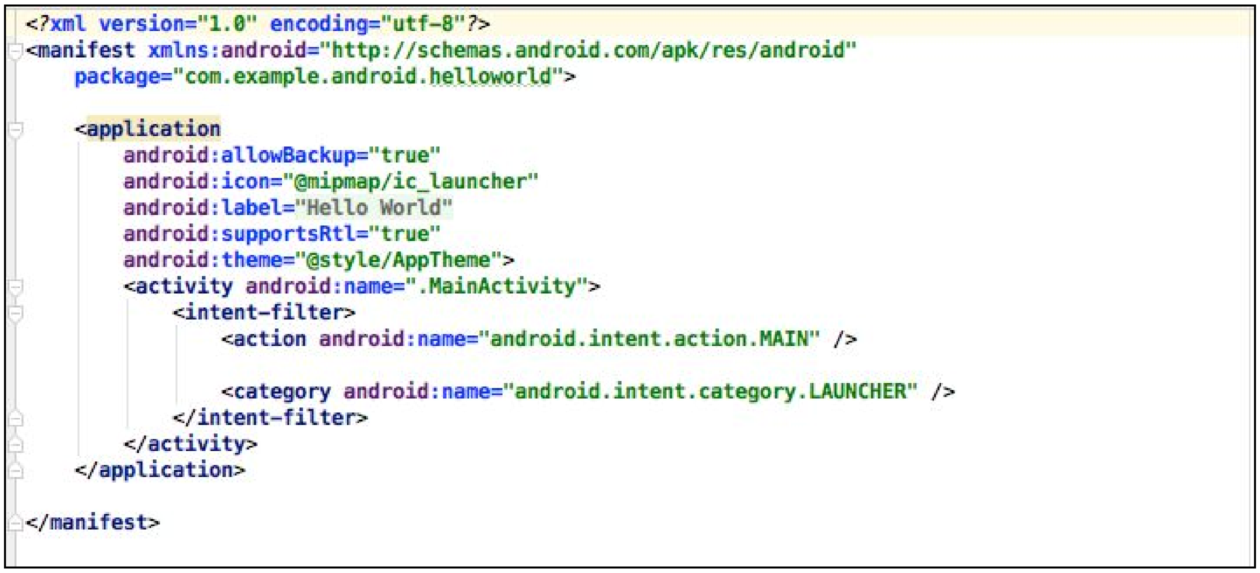
1. The Project tab. Click to show the project view.
2. The Android selection in the project drop-down menu.
3. The AndroidManifest.xml file. Used for specifying information about the app for the Android runtime environment. The template you choose creates this file.
4. The java folder. This folder includes activities, tests, and other components in Java source code. Every activity, service, and other component is defined as a Java class, usually in its own file. The name of the first activity (screen) the user sees, which also initializes app-wide resources, is customarily MainActivity.
5. The res folder. This folder holds resources, such as XML layouts, UI strings, and images. An activity usually is associated with an XML resource file that specifies the layout of its views. This file is usually named after its activity or function.
6. The build.gradle (Module: App) file. This file specifies the module's build configuration. The template you choose creates this file, which defines the build configuration, including the minSdkVersion attribute that declares the minimum version for the app, and the targetSdkVersion attribute that declares the highest (newest) version for which the app has been optimized. This file also includes a list of dependencies, which are libraries required by the code — such as the AppCompat library for supporting a wide range of Android versions.

3.5.6 Viewing the Android Manifest

Before the Android system can start an app component, the system must know that the component exists by reading the app's AndroidManifest.xml file. The app must declare all its components in this file, which must be at the root of the app project directory.

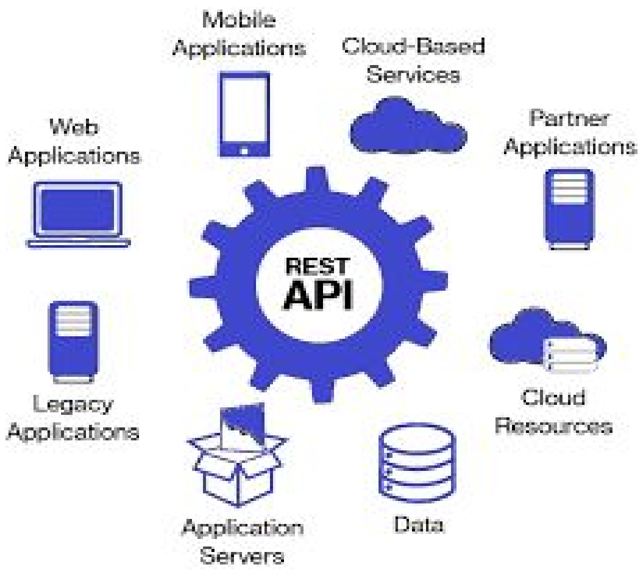
To view this file, expand the manifests folder in the Project: Android view, and double-click the file (AndroidManifest.xml).

Its contents appear in the editing pane as shown in the figure below.



```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.android.helloworld">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Hello World"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

4. REST API



4.1 What is REST

REST is acronym for REpresentational State Transfer. It is architectural style for distributed hypermedia systems and was first presented by Roy Fielding in 2000 in his famous dissertation.

Like any other architectural style, REST also does have it’s own 6 guiding constraints which must be satisfied if an interface needs to be referred as RESTful. These principles are listed below.

4.2 Guiding Principles of REST

1. Client–server – By separating the user interface concerns from the data storage concerns, we improve the portability of the user interface across multiple platforms and improve scalability by simplifying the server components.
2. Stateless – Each request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server. Session state is therefore kept entirely on the client.
3. Cacheable – Cache constraints require that the data within a response to a request be implicitly or explicitly labeled as cacheable or non-cacheable. If a response is cacheable, then a client cache is given the right to reuse that response data for later, equivalent requests.
4. Uniform interface – By applying the software engineering principle of generality to the component interface, the overall system architecture is simplified and the visibility of interactions is improved. In order to obtain a uniform interface, multiple architectural constraints are needed to guide the behavior of components. REST is defined by four interface constraints: identification of resources; manipulation of resources through

representations; self-descriptive messages; and, hypermedia as the engine of application state.

5. Layered system – The layered system style allows an architecture to be composed of hierarchical layers by constraining component behavior such that each component cannot “see” beyond the immediate layer with which they are interacting.
6. Code on demand (optional) – REST allows client functionality to be extended by downloading and executing code in the form of applets or scripts. This simplifies clients by reducing the number of features required to be pre-implemented.

4.3 REST Methods

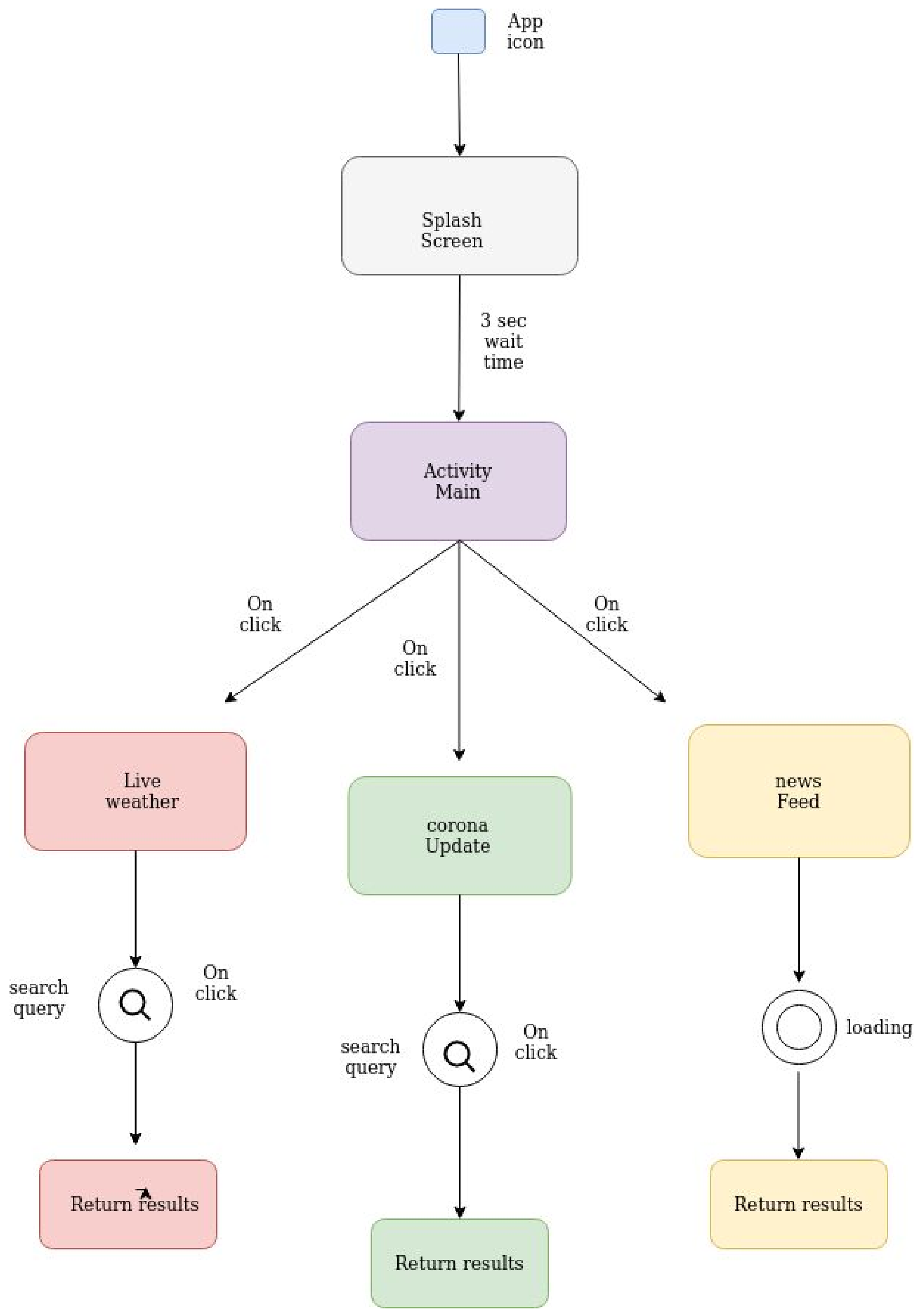
Method Name	Request Meaning
\`GET\`	This request is used to get a resource from a server. If you perform a \`GET\` request, the server looks for the data you requested and sends it back to you. In other words, a \`GET\` request performs a \`READ\` operation. This is the default request method.
\`POST\`	This request is used to create a new resource on a server. If you perform a \`POST\` request, the server creates a new entry in the database and tells you whether the creation is successful. In other words, a

	`POST` request performs an `CREATE` operation.
`PUT` and `PATCH`	These two requests are used to update a resource on a server. If you perform a `PUT` or `PATCH` request, the server updates an entry in the database and tells you whether the update is successful. In other words, a `PUT` or `PATCH` request performs an `UPDATE` operation.
`DELETE`	This request is used to delete a resource from a server. If you perform a `DELETE` request, the server deletes an entry in the database and tells you whether the deletion is successful. In other words, a `DELETE` request performs a `DELETE` operation.

PROJECT-DESIGN

Designing is the first step to creating any project or application.

So we started designing our app with simple control flow diagrams as below.



Project Modules

Project is divided into 5 basic modules to decrease the complexity. There are basically one MainActivity.java file and 3 other fragments files and one splash screen file.Every Activity and fragment have its own layout file that is written in XML.
To enhance the UI we are using Material Design guidelines by google.
All the icons and images used in the app are located in the drawable folders. And finally all the used colors and style are located in the value folder.

Activities and Fragments file name as --

1. MainActivity.java
2. Splash.java
3. Corona_update.java
4. Live_weather.java
5. News_feed.java

Layout files name as --

1. activity_main.xml
2. corona_update.xml
3. splash.xml
4. news_feed_layout.xml
5. Weather_layout.xml

Colors , Style and string files --

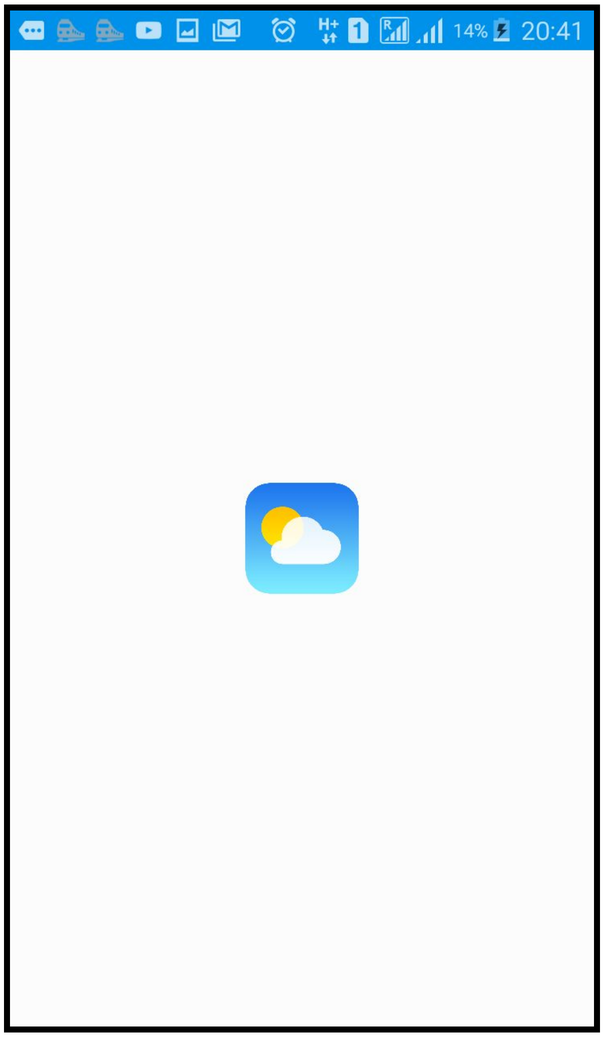
1. Colors.xml
2. Strings.xml
3. Style.xml

Permission Required from the devices --

1. Internet and network

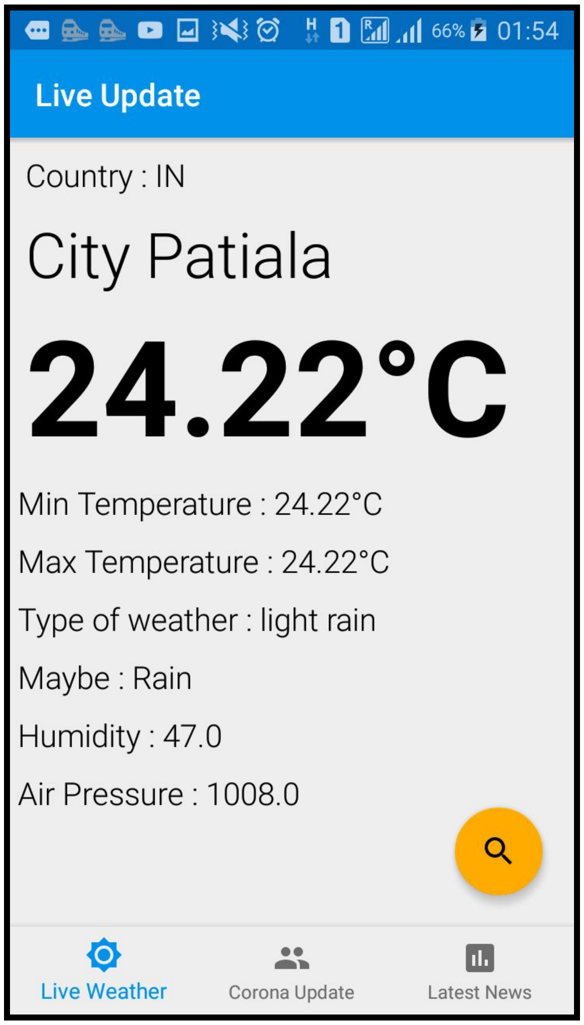
First Screen (Splash Screen)

Splash screen is the first screen when we open the app. It helps us to load the and fetch the data from the internet in background and when the app launches it will show you the current updated information. We created this screen with one file in the name as splashscreen.java It is also a kind of activity and it call from the mainActivity.java file.



Home Screen (3 Fragments)

After the splash screen ends, the first and default fragment is called from the mainActivity.java file . In the main activity file we created some case statements to check which fragment is selected but the default fragment is the first fragment that is live weather. In the bottom of the navigation bar there is yellow round button that is allows us to search the desired city all around the world.



Country : IN

City Patiala

24.22°C

Min Temperature : 24.22°C

Max Temperature : 24.22°C

Type of weather : light rain

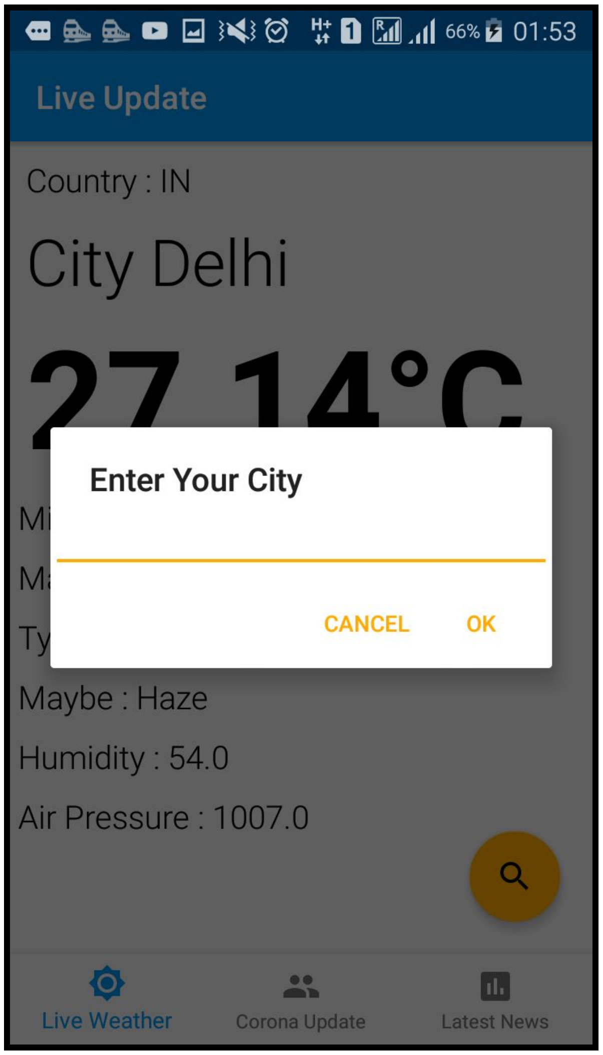
Maybe : Rain

Humidity : 47.0

Air Pressure : 1008.0

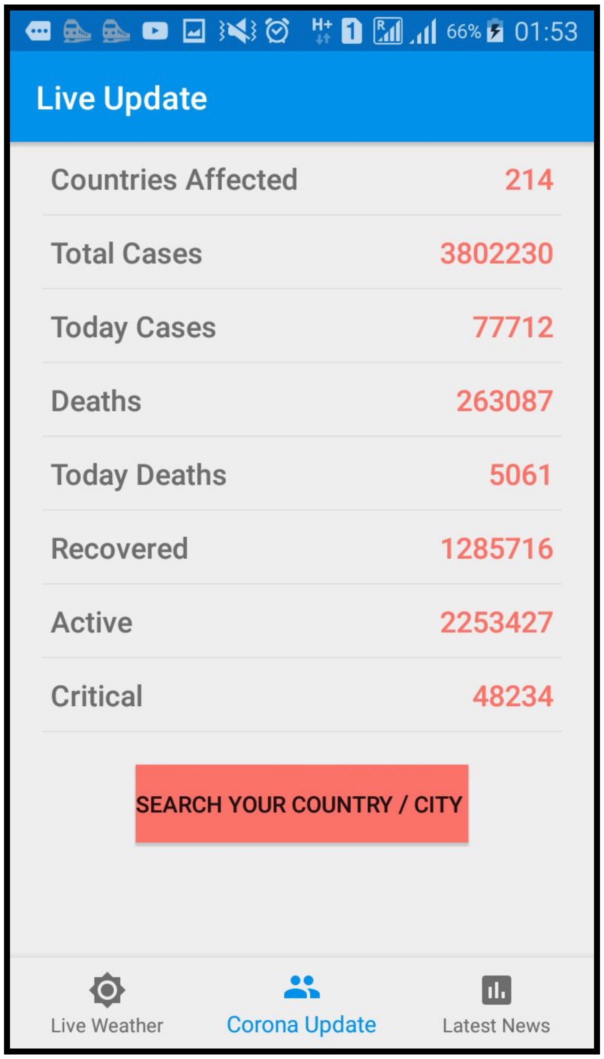
Search Pop-up Screen(Live weather fragment)

For the search button we called the onClickListener method from the button that will call a function. That function contains the alert box with one input field and tow button one is OK and the other one is Cancel. When we type something with this pop up alert box it will take text and pass with another function that will call the REST API and fetch the data in JSON format. Then we are using the JSON object method to extract the desired data.



Corona Updates Screen(Second Fragments)

Corona updates is the second fragment in the app. It works the same as the first fragment. The key difference is here we are using another open api to fetch the data from the server. In this we can also search our country and city to get the latest updates about corona.



66% 01:53

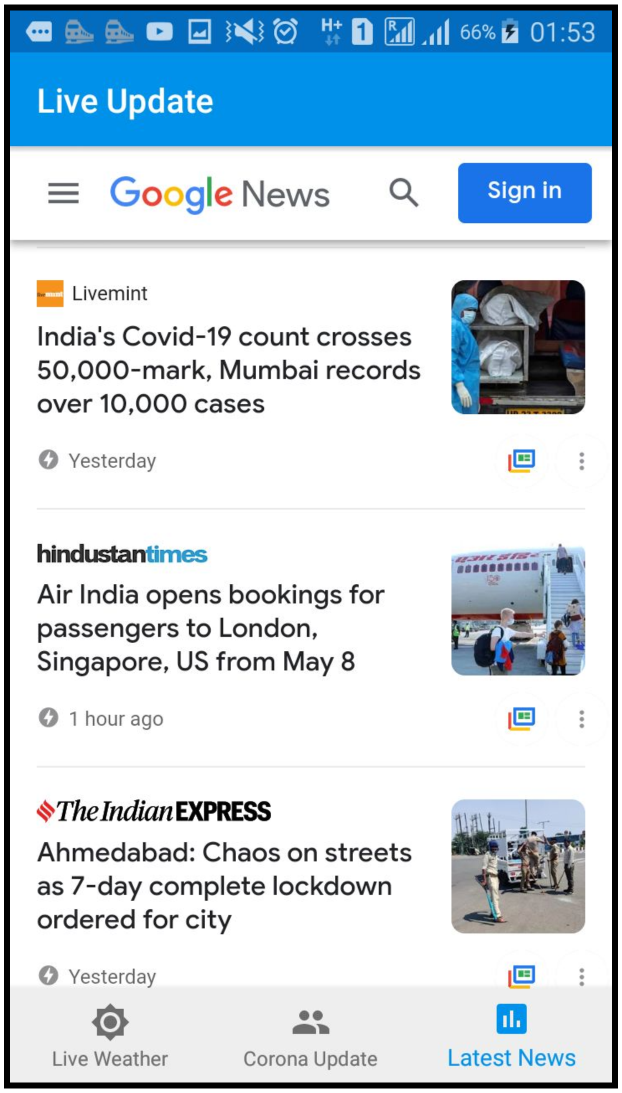
Live Update

Countries Affected	214
Total Cases	3802230
Today Cases	77712
Deaths	263087
Today Deaths	5061
Recovered	1285716
Active	2253427
Critical	48234

News Feed Screen(Third Fragment)

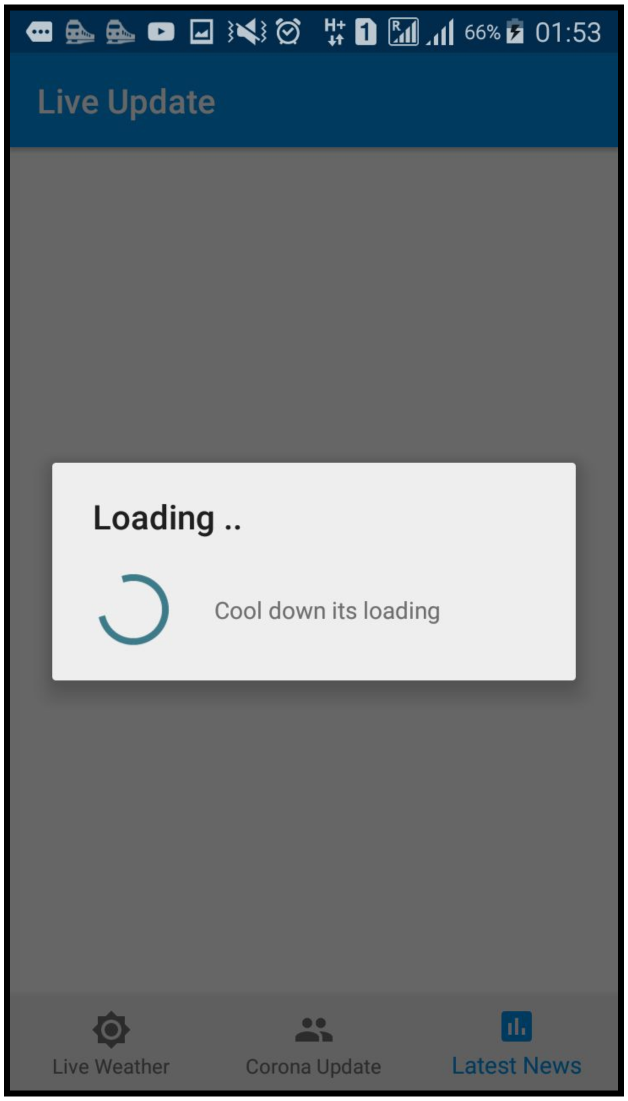
News feed is a very interesting feature of our app. It is using the webview to get the latest news from the google news site. We have added the internet network permission in the Androidmanifest.xml file. It allows us to search the news from google.

```
<uses-permission android:name="android.permission.INTERNET" />
```


66% 01:53
Live Update
Google News
Sign in
Livemint
India's Covid-19 count crosses 50,000-mark, Mumbai records over 10,000 cases
Yesterday
hindustantimes
Air India opens bookings for passengers to London, Singapore, US from May 8
1 hour ago
The Indian EXPRESS
Ahmedabad: Chaos on streets as 7-day complete lockdown ordered for city
Yesterday
Live Weather
Corona Update
Latest News

Web View Loading pop-up (Third Fragment)

It is very boring to wait for the loading but it is necessary in the webview so to avoid some static situation of the app we added loading pop up in the webView so when its load it will pop up the loading message.



Conclusion

The main objective is to develop an application like this to save the time of the user to jump into one to another app for similar information. This is the first version of our app but we are also working to enhance its performance and user experience. Via this app we learned how the fragments are worked together and how the REST api works.

Working on a project is one of the easiest ways to learn new things. We can easily say we learned alot from this project.

And last i would like to thanks our Mobile app development professor Dr. Gaurav Gupta sir for this project.

References

- https://www.geeksforgeeks.org/android-app-development-fundamentals-for-beginners/
- https://developer.android.com/
- https://www.tutorialspoint.com/android/index.htm
- https://www.javatpoint.com/android-tutorial