

AS 916917 and 91635 Level 3 Digital Technologies

Mobile App Journal by Lyle Branzuela

Introduction

In today's society, phones and mobile devices are getting more and more prominent, there's a demand for easier and accessible ways to do something, this something can be as small as making a checklist or something big like managing important files. One of the few things that's happening in today's society is that people are watching less and less television and is transitioning to watching the same television channels and streams to mobile phones and computers, understanding this demand is key in creating my app. The project would be a mobile app where it streams tv channels (TV3, BRAVO TV, and THE EDGE TV) on a phone and has its own timetables of the show that is currently playing or what's next.

The client would be my father, has worked in the IT field for decades, giving him a lot of experience in regards to programming, this could help critique and give more realistic and non-vague ideas to my project. The goal is to create a mobile application that can be helpful wherever, and whenever the user wants it.

Table of Contents	
<u>Introduction</u> - Page 1	<u>Efficiency and Time Management</u> - Page 4-5
<u>Digital Tools Used</u> - Page 1-2	<u>Legal, Moral, and Ethical Considerations</u> - Pages 5
<u>Feedback and Inspirations</u> - Page 3	<u>Final Product</u> - Pages 6-14
<u>Testing and Issues in Development</u> - Page 4	<u>Conclusion</u> - Page 14

Digital Tools Used

Research and Resources
<p>Android Tutorials - I used Android Tutorials to create an idea of what I can realistically do to make the app, and by designing through this philosophy on only designing it if I have the time and ability to do it affected the final outcome very much.</p> <p>Material Design - Material Design Color Tool and Information about how to</p>

design good layouts and color schemes were useful in creating the design part of the project, helping me avoid common mistakes made by other people in the development of the app.

Youtube - Youtube was mainly used on looking at how other people design their Mobile User Interface, and what they considered to be good. By gathering knowledge across multiple videos, I managed to create a more simple layout than before.

Stack Overflow - One of the most helpful resource i've used, this website provides solutions to some of the problems or issues i've encountered while programming.

Udemy - This website is good for people who wants to learn a new programming language, the problem is that not all is for free, and I spent 20\$ on a sale to buy a program that teaches me the basics of Android Studio.

Programming Tools



Main Programming Languages - The main tools I've used in the mobile development is the IDE (Integrated Development Environment) **Android Studio** and its main language **Java** for the main programming front end of the app, **JSON** (JavaScript Object Notation) for the **Web API** (Application Programming Interface), and **XML** (Extensible Markup Language) for the layouting of the app. These programming languages is popular in the Android Development community, by using this, I can search a broader range of information and guides than less popular programming languages.

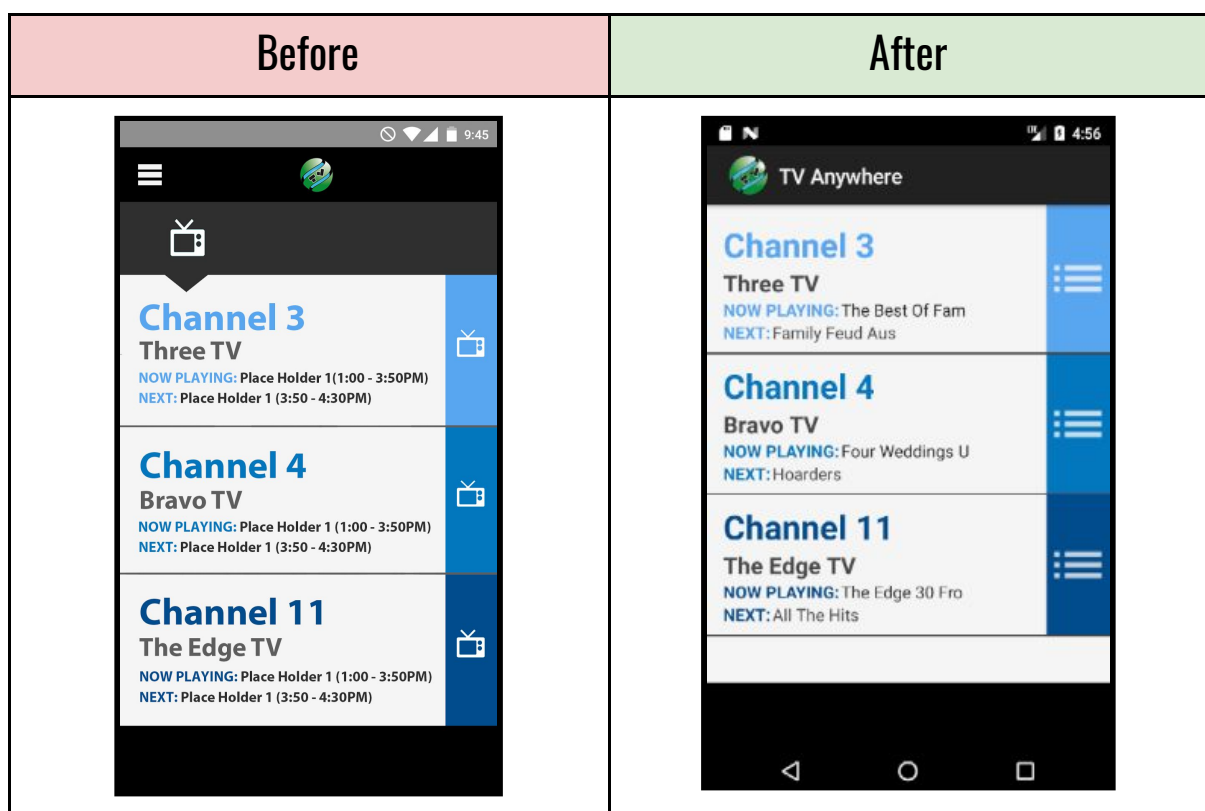
Other Alternatives - Another tool that I could've used is Javascript's React Native/Javascript, which is easier, faster, and can create better looking designs and animations. I've discovered this programming tool long after I've learned the basics of Java, because of time restrictions, I do not have enough time to learn or use this language, and would just try it out on other projects.

Storage and Backups - To store and backup my documents i use Git and Github, so that if my computer ever gets broken, the data would still have it saved on the internet/cloud data storage, lowering the risk of losing all the data in one go, if my computer ever gets a virus or broken.

Feedback and Inspirations

Most of my inspirations came from my hobby of watching films and tv series, computer, and mobile phones. What I realized is that most streaming apps on the mobile phone does not provide a good list of time, as well as having the channel streaming, this means that it's difficult to tell when the movie/film you want to watch when you don't have a good list of these movies/films.

Most of the feedback I got was from my client, some from my peers in class. Almost all of the feedback was about the design and layout of the whole app, such as the change from the initial design to the final design of the main activity, as my client and I decided that the final design is better than the initial one. The final design is more consistent and realistic, if I added the time (Place holder 1 (1:00 - 2:50PM)), then the word would be too long and wouldn't fit the app, the final design is implemented to adjust to this. After talking with the client, we found out that the hamburger button on the top left is useless as the app only has 3 activities running and a user doesn't need to move around too much, so we got rid of it and just stayed with just one action bar.



Some feedback like what features should be added was talked about with my client, such as the use of JSON in creating the list of channel names and their time. This serves the purpose of the goal of making an easy accessible mobile app that can stream tv channels and give information on what's next and currently playing automatically and always updated.

Testing and Issues in Development

Testing is one of the most important thing in programming, without testing my Mobile Application would be full of bugs and would instantly crash and break. I tested my Mobile Application through 2 different ways, one way is the use of my personal phone to test the apps, this makes it more physical and I can see what's wrong in both the UI as well as if the lines of codes I recently added is working. A feature in Android Studio is that it can show the errors and a log on why it crashed, this is one of the most useful things that Android Studio IDE can offer since it helps me find the problem much more easier, making fixing it faster.

A Lot of the issues i've faced is due to my inexperience and lack of knowledge, after encountering these errors i've learned a lot and started to adapt and predict where the error might be or what causes the error, this skill I developed is very useful in future programming projects. Although I improved on some stuff, one of the major issues that I got early on is that my initial start got erased/disappeared from the computer, this major issue sets back my project for about 1 week of work, after this I decided to use Git and Github to save my project, so that I will never face this issue ever again.

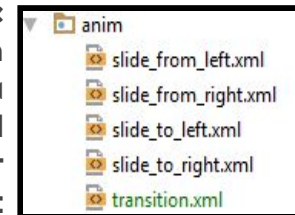
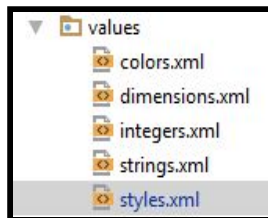
One of the main uses of github is that I can revert back the project to a certain committed time, I only used this once after I messed up a major part of the program making everything in errors, and it really helped and saved me a lot of hours and weeks of development. Another major issue is how I should make the whole mobile app responsive, this is the hardest issue since there's a lot of devices out there, and that making it responsive for all of them is a hard task, so what I did was to focus on certain groups of devices, mainly mobile Galaxy Nexus phones.

Efficiency and Time Management

As a person with no prior experience to the programming tools i've used (Java, JSON, and XML), my biggest issue would be from managing my time in learning how to use the tools, and then actually completing the application on time. Because of this time restraint, my lack of experience, and that these tools aren't available in the school's computers, I did all of my learning how to use the programming tools and creating the mobile application exclusively outside of class, and on my personal laptop.

I spent most of my first few weeks learning how to use Java, JSON, and XML, while my peers were already starting on their websites. I mainly use the time at school to read about things on the tools i was gonna use, and trying to know if i can implement the techniques i research at school, because I couldn't do the actual mobile app. Without proper time management, efficiency, and passion in learning how and creating the application, then I would've been behind my peers and wouldn't have finished it at all.

Another one of the things that I did so that I can be more time efficient is the use of file management and values such as styles/colors/integers to manage my all of my styles in a very efficient manner, this means that if I want to change a certain style or animation that is in the app, I would just go to one of these files and change them, this changes the whole thing that uses these styles, saving me time rather than editing it individually.



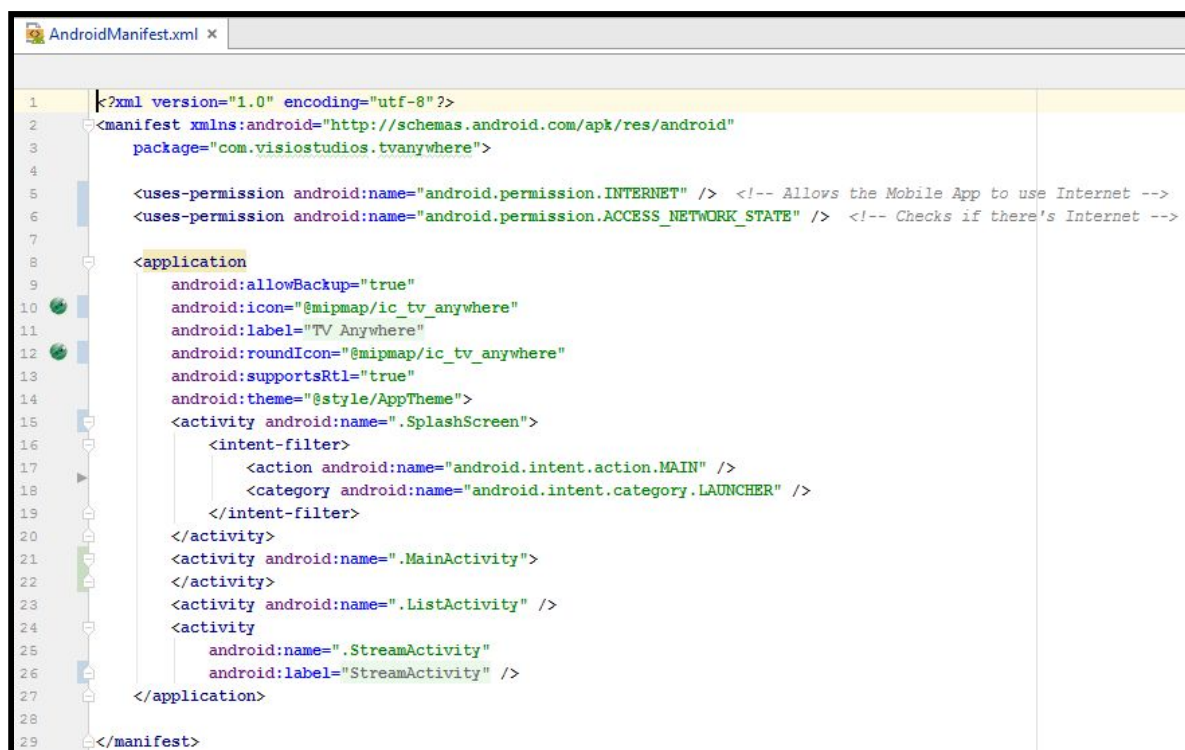
JSON is one of the best ways in implementing automated stuff from the web, by using JSON to grab the array of data or the list of channel names and times, then I don't need to manually include the list myself, and this saves a lot of time and helps increase the life of the application, as it's self sustaining.

Legal, Ethical, and Moral Responsibilities

The product itself can be accessible by anyone at any age, and since I've used tv channels that can be considered as everyone and has their own age guidelines such as G (Guardian), and PG (Parental Guidance), which makes the tv channels itself safe for everyone. As for the mobile application itself, the app itself is in a neutral position in where it's safe for everyone, and that the data itself is public, and can be accessible anywhere if you have internet, as it's taken from the official <https://www.threenow.co.nz/> website. All rights and conditions belong to Mediaworks.

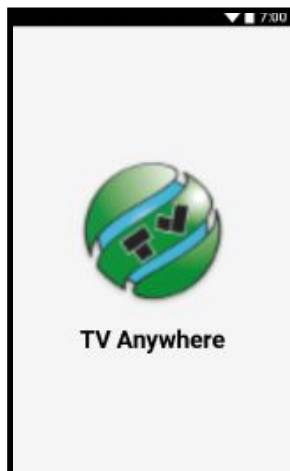
Final Product

Android Manifest



Android Manifest is where the activities are located, and the things that the application needs outside of the app. This can mean from what activity should run first or if the user has internet, this is important because my Mobile Application exclusively relies on using the internet, as it gets all the data from the Internet, without the internet my application's features won't work at all. This is where I put my logos and removing the default action bar, by doing this, I can customize further from the android studio's defaults, and make themes that is consistent throughout the app.

Splash Screen



decided to add a splash screen because of how I felt that something was lacking at the start, and then I realized it was the opening splash screen every app has.

The first thing that people will see is the splash screen, the splash screen provides a short introduction of what the name of the app is as well as the logo of it. It's useful in setting up the mood of the app, as well as introduce the new users to what the UI or what they will be expecting in the app on the short 3 seconds it is presented.

```
// Identifies the Views in the XML
splashText = (TextView) findViewById(R.id.splashText);
splashImage = (ImageView) findViewById(R.id.splashImage);
// Identifies the Animation in the XML anim Folder
Animation mySplash = AnimationUtils.loadAnimation(this,R.anim.transition);
// Starts The Animation
splashText.startAnimation(mySplash);
splashImage.startAnimation(mySplash);

// Transitions from the Splash Screen to the Main Activity
final Intent mainIntent = new Intent(SplashScreen.this,MainActivity.class);
Thread timer = new Thread() {
    public void run () {
        try {
            sleep(5000); // Sleeps the thread
        } catch (InterruptedException e) {
            e.printStackTrace();
        } finally {
            startActivity(mainIntent); // Fade into The Main Activity
            finish();
        }
    }
};

timer.start(); // Starts the Thread
```

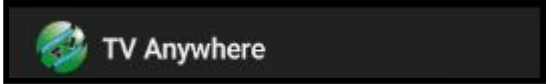
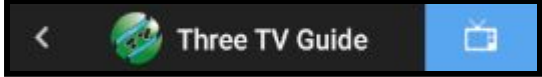
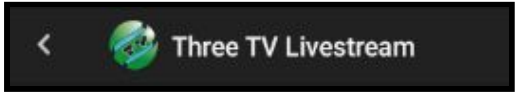
Toolbar

One of the main recurring things inside the app is the use of my own custom Toolbar, this custom toolbar is present in all activities except for the initial splash activity. The main purpose of a toolbar is so that it provides the feeling of consistency in between activities as well navigate through activity to activity. This is so that users won't be confused in navigating the mobile app.

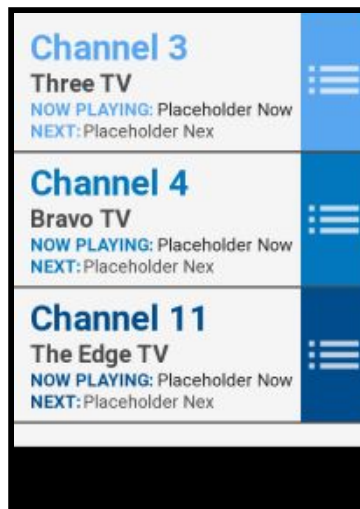

```
// Setting up the Custom Toolbar for the Main Activity
Toolbar myToolbar = (Toolbar) findViewById(R.id.grayBar); // Makes the grayBar into a Toolbar Variable
setSupportActionBar(myToolbar); // Sets Toolbar Variable into the SupportActionBar of the App
getSupportActionBar().setLogo(R.mipmap.ic_tv_anywhere); // Sets what Image the Logo will be
getSupportActionBar().setDisplayUseLogoEnabled(true); // Displays The Logo
getSupportActionBar().setTitle(" TV Anywhere"); // Sets The Title Name on top
```

The main difference between the Action Bar of the Main Activity (Photo on Top), and the Action Bar of other activities is the presence of a HomeAsUpindicator or a back button (Photo on the Bottom), and the lack of set titles (The Titles are programmed (More on that later on) to change depending on what channel the user clicks on) since the Main Activity is the first page, it doesn't need one.

```
getSupportActionBar().setDisplayHomeAsUpEnabled(true); // Displays The Back Button
getSupportActionBar().setHomeAsUpIndicator(R.drawable.ic_back_button); // Uses my custom Back button instead of the Default
```

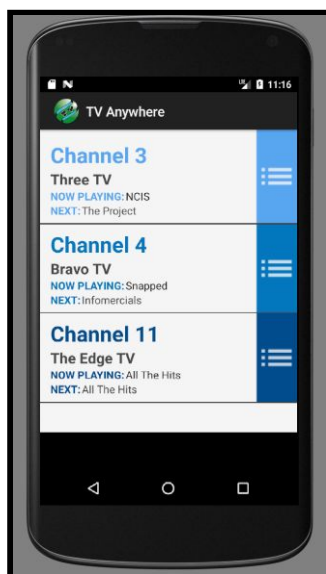
Main Activity	Other Activity
	 
Toolbar Style	
<pre><!-- General Use/Colored Bars --> <style name="Graybar"> <item name="android:layout_width">match_parent</item> <item name="android:layout_height">?attr/actionBarSize</item> <item name="android:background">@color/colorPrimary</item> <item name="android:layout_marginTop">0dp</item> <item name="android:layout_marginRight">0dp</item> <item name="android:layout_marginLeft">0dp</item> </style> <android.support.v7.widget.Toolbar android:id="@+id/grayBar" style="@style/Graybar" android:layout_height="?attr/actionBarSize" android:elevation="4dp" android:theme="@style/ThemeOverlay.AppCompat.ActionBar" app:layout_constraintLeft_toLeftOf="parent" app:layout_constraintRight_toRightOf="parent" app:layout_constraintTop_toTopOf="parent" android:layout_gravity="center" app:popupTheme="@style/ThemeOverlay.AppCompat.Light" app:titleTextColor="@color/backgroundcolor"> </android.support.v7.widget.Toolbar></pre>	<p>The style of each Toolbar is simple and changes height depending on the user's phone, this is because of the line: (<code>android:layout_height="?attr/actionBarSize"</code>), which makes it so that it matches the default height of the phone's default action bar, this helps makes it more responsive. After that, it's just a bunch of constraints that ensures that the toolbar stays on top and doesn't move too much and designs such as color and themes to customize the color.</p>

Bottom Blackbar



The black bar shows up throughout activities except the splash and the stream activity, the bottom black bar provides the feeling of subconsciously locking in and making the app feel more symmetrical. It has the same style as the Graybar, and uses the same style and functions as the gray bar style, without it being a toolbar. The black bar itself doesn't do anything, but it helps in creating the consistency between the Main Activity as well as the List Activity, as both of them have this blackbar, it would seem like they are just one activity.

Main Activity



The main activity is the activity where the user can navigate in between the three given tv channels. It's presented the what's currently playing, and what's next, above is its channel number and name. This part provides the essential information quickly to the user, without lengthy explanations on what they are, this again fulfills the purpose of the product to provide fast and easy to understand information. The Main Activity itself is pretty complex, it uses **JSON** and **Java** to grab the data of what is playing right now to from the JSONArray online to the main activity's, making it automated, and everything updates itself (Image below):

```
private JSONObject getLiveTV () {
    JSONObject response = null;
    try {
        URL url = new URL("http://now-api.mediaworks.nz/now-api/v3/live-epg"); // Specifying the URL taken
        HttpURLConnection conn = (HttpURLConnection) url.openConnection(); // Connects to the internet
        conn.setRequestMethod("GET"); // Specifies the Request Method
        InputStream in = new BufferedInputStream(conn.getInputStream()); // Reads the response
        // Converts the stream data into string, which can be used to compare to other objects
        String jsonStr = convertStreamToString(in);
        if (jsonStr != null) {
            response = new JSONObject(jsonStr);
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return response;
}
```


This was used in taking the channel name as well:

```
// Function to get the Now Playing from the JSON List
private String getNowPlaying (JSONObject jsonList , String channel , int index) {
    String playing = "";
    // A try catch and an inside for loop to get the JSON List's Title
    try {
        JSONArray jsonArray = jsonList.getJSONArray("channels"); // Specifies the Array used, in this case channels
        for (int x = 0; x < jsonArray.length(); x++) {
            if (jsonArray.getJSONObject(x).getString("title").equals(channel)) {
                JSONObject channelObject = jsonArray.getJSONObject(x);
                JSONArray broadcasts = channelObject.getJSONArray("broadcasts");
                playing = broadcasts.getJSONObject(index).getString("title"); // Take the name of the Channel
            }
        }
    } catch (Exception e) {
        // Return Nothing if there's an error
        e.printStackTrace();
    }
    return playing; // Send the name of the Channel
}
```

And then changed the text in the Main Activity:

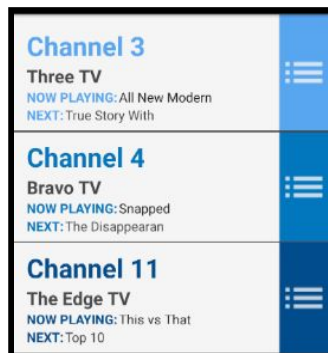
```
// Changes the Text inside the Main Activity
// Putting the first item in the JSON Array of Now Playing to the Main Activity
nowPlayingListC3.setText(getNowPlaying(liveTV , "tv3" , 0)); // (0 means the first/latest title/name of the series playing)
nowPlayingListC4.setText(getNowPlaying(liveTV , "bravo" , 0));
nowPlayingListC11.setText(getNowPlaying(liveTV , "theedgetv" , 0));
// Putting the second item in the JSON Array of what's next of the current series playing to the Main Activity (this means 1)
nextListC3.setText(getNowPlaying(liveTV , "tv3" , 1));
nextListC4.setText(getNowPlaying(liveTV , "bravo" , 1));
nextListC11.setText(getNowPlaying(liveTV , "theedgetv" , 1));
```

The dates and channel names are also taken and then sent to the list activity:

```
// Function to get the Format and the Date to be used in the List Activity
private ArrayList<String> getChannelList (JSONObject jsonList , String channel) {
    ArrayList<String> response = new ArrayList<>();
    SimpleDateFormat dF = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ssZ"); // Date Format of the JSON List
    SimpleDateFormat dF2 = new SimpleDateFormat("hh:mm:ss"); // Date Format of the Start Date
    dF.setTimeZone(TimeZone.getTimeZone("GMT")); // Declaring the Timezone
    dF2.setTimeZone(TimeZone.getTimeZone("Pacific/Auckland")); // Declaring The Specific Timezone
    try {
        JSONArray jsonArray = jsonList.getJSONArray("channels");
        for (int x = 0; x < jsonArray.length(); x++) {
            if (jsonArray.getJSONObject(x).getString("title").equals(channel)) {
                JSONObject channelObject = jsonArray.getJSONObject(x);
                JSONArray broadcasts = channelObject.getJSONArray("broadcasts");
                for (int y = 0; y < broadcasts.length(); y++) {
                    Date start = dF.parse(broadcasts.getJSONObject(y).getString("startDate"));
                    response.add(dF2.format(start) + " - " + broadcasts.getJSONObject(y).getString("title"));
                }
            }
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return response;
}
```

This line of code uses the functions created above to get the information needed to take the information needed for the next two activities.

```
// Using the function to get URL of the LiveTV, and then declare them as a JSON Object
JSONObject liveTV = getLiveTV();
// Getting the Channel List, and setting a variable for each list
channel3 = getChannelList(liveTV , "tv3");
channel4 = getChannelList(liveTV , "bravo");
channel11 = getChannelList(liveTV , "theedgetv");
```



It is then sent through the buttons (shown on the left). These buttons have different functions depending on what the user clicks (Shown Below), if the user clicks on Channel 3, it takes them on the Activity list with the JSON Array of Channel 3, which then shows the list of information about what comes next or what's currently playing in the next activity. It also sends the media player for the user to stream in the stream activity.

```
// Function that identifies if the ArrayList is either Channel 3, Channel 4, or Channel 11
private void launchActivity(ArrayList<String> schedule , String playUrl , String channel) {
    Intent intent = new Intent(MainActivity.this, ListActivity.class); // Declaring what Activities the Intent should move to
    intent.putStringArrayListExtra("schedule", schedule); // Sends the Schedule to the next activity
    intent.putExtra("playUrl", playUrl); // Sends the Stream URL to List Activity
    intent.putExtra("channelName", channel); // Sends the Channel name to List Activity
    startActivity(intent); // Switches to the Next Activity
    overridePendingTransition(R.anim.slide_from_right, R.anim.slide_to_left); // Sliding Animation
}
```

It then uses the function above to move between activities:

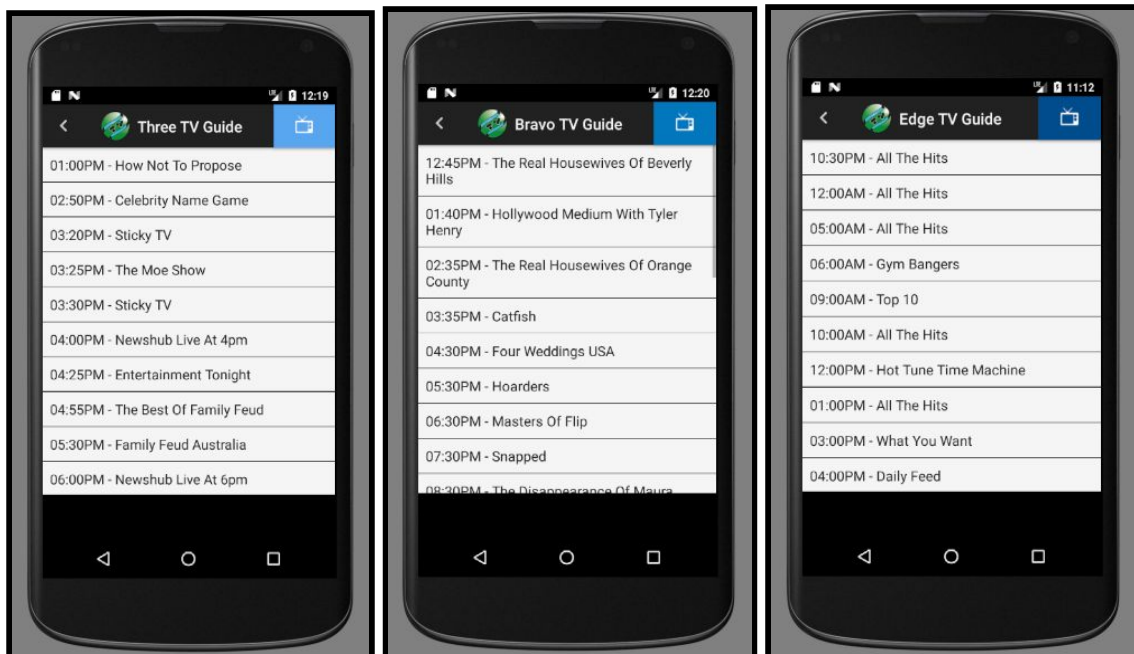
```
// Sending JSON Array List, Video Stream Link, and The Channel Title to the other Activity
ImageButton buttonC3 = (ImageButton)findViewById(R.id.buttonC3);
buttonC3.setOnClickListener(new View.OnClickListener() { // Listens when the user clicks on the image button
    @Override
    public void onClick(View view) {
        launchActivity(channel3 , "http://players.brightcove.net/3812193411001/ryftAvCY_default/index.html?videoId=5455129193001", "channel3"); // Sends all the data taken to the next activity
    }
});

ImageButton buttonC4 = (ImageButton)findViewById(R.id.buttonC4);
buttonC4.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        launchActivity(channel4 , "http://players.brightcove.net/3812193411001/ryftAvCY_default/index.html?videoId=54551543064001", "channel4");
    }
});

ImageButton buttonC11 = (ImageButton)findViewById(R.id.buttonC11);
buttonC11.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        launchActivity(channel11 , "http://players.brightcove.net/3812193411001/ryftAvCY_default/index.html?videoId=545512812200", "channel11");
    }
});
```

The main activity is important in creating a concise and quick information that the mobile app's purpose provides to the user's needs and gives it to them in a short and not overwhelming way, which may discourage them from using it if they get too overwhelmed from the information.

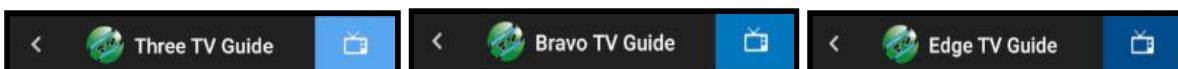
List Activity



The first thing that the List Activity does is that it checks what channel the Main Activity sent the List Activity this is done through this block of code:

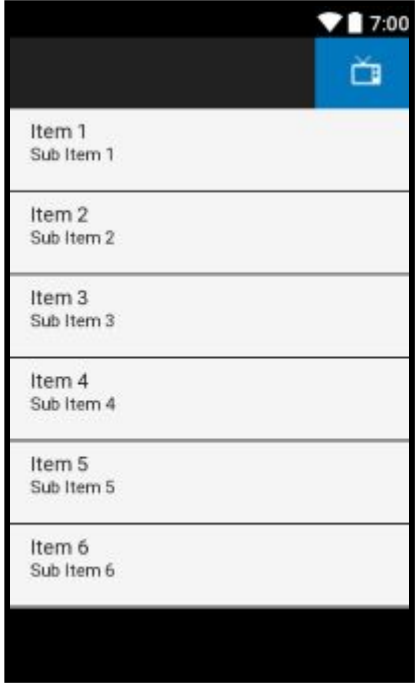
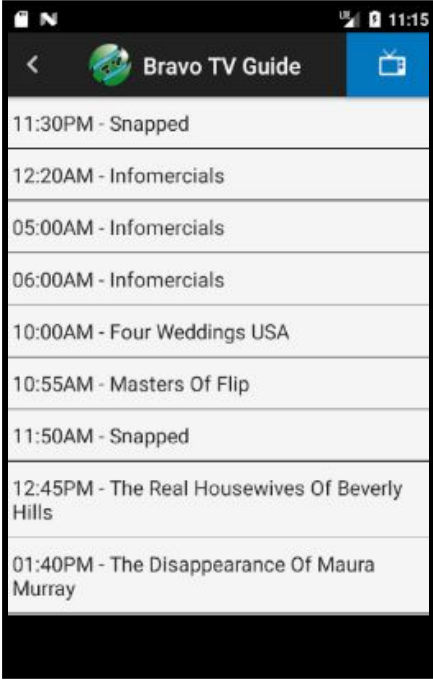
```
// Calling the Intent from MainActivity
Intent intent = getIntent();
final String channelName = intent.getStringExtra("channelName");
```

```
// Checks what Channel No. the Main Activity sent the List Activity
// After checking, it then changes the title and the color of the image depending on that Channel No.
if (channelName.equals("channel13")) {
    getSupportActionBar().setTitle(" Three TV Guide");
    liveStreamButton.setImageResource(R.drawable.television_lb);
    liveStreamButton.setBackgroundResource(R.color.secondaryColorPrimaryLight); // Changes the Color of the Button
} else if (channelName.equals("channel14")) {
    getSupportActionBar().setTitle(" Bravo TV Guide");
    liveStreamButton.setImageResource(R.drawable.television_b);
    liveStreamButton.setBackgroundResource(R.color.secondaryColorPrimary);
} else {
    getSupportActionBar().setTitle(" Edge TV Guide");
    liveStreamButton.setImageResource(R.drawable.television_db);
    liveStreamButton.setBackgroundResource(R.color.secondaryColorPrimaryDark);
}
```



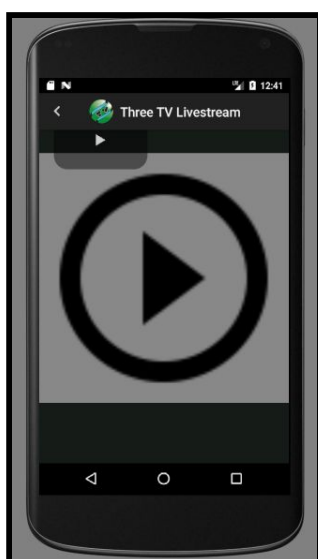
It then converts the whole custom list into the JSON Array that was extracted:

```
// Gets the Schedule from the Main Activity's Intent
ArrayList<String> schedule = intent.getStringArrayListExtra("schedule");
// This changes my custom List on the XML into the JSON Array list
ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(this, R.layout.custom_list_view_1, schedule);
scheduleListView.setAdapter(arrayAdapter);
```

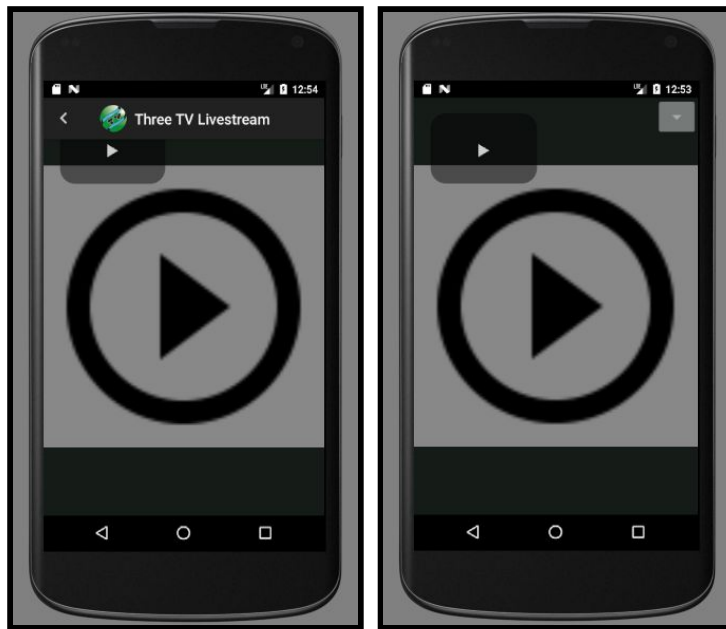
Before	After
	

The list activity is one of the main purpose of the mobile app, a list of what's playing and what's next, the user can plan on when they can watch their favorite television series before even watching, this means that if they don't like what is currently playing, they can just skip it, and not watch the stream. This makes it so that the user doesn't waste his time, which is one of the purposes of the mobile application.

Stream Activity



The final activity and again one of the key purposes of the application is the stream activity, or where clients can watch the tv channel they chose. The livestream activity had simple codes in, such as the action bar changing similar from the List Activity, simple buttons that does simple actions, and a whole view of the live stream itself. One of the main problem in programming this activity is that I can't edit the media player inside, as it's the same media player in the website, this means that I do not have control over this. The link of the media player was passed down from the JSON Array extraction in Main Activity to the Stream Activity.



However, a key difference between this application, is the ability to hide the actionbar by pressing it, and to reappear it through pressing the transparent drop down button on top right. This provides a better view for the user and prevents the action bar from obstructing the media player/livestream.

Other Parts of the App

Things such as the colors, strings, transition animations, and the slide animations were a key part in making the app smooth and as well as time efficient. I can change a single code from this and it would change the whole thing that uses these styles.

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <resources>
3  <color name="colorPrimary">#212121</color>
4  <color name="colorPrimaryDark">#000000</color>
5  <color name="colorAccent">#484848</color>
6  <color name="backgroundColor">#f5f5f6</color>
7  <color name="secondaryColorPrimary">#0277bd</color>
8  <color name="secondaryColorPrimaryDark">#004c8c</color>
9  <color name="secondaryColorPrimaryLight">#58a5f0</color>
10 </resources>
  
```

```

1  <resources>
2  <string name="app_name">TV Anywhere</string>
3  <!-- Channel Strings -->
4  <string name="channelThree">Channel 3</string>
5  <string name="channelFour">Channel 4</string>
6  <string name="channelEleven">Channel 11</string>
7  <!-- Channel Name Strings -->
8  <string name="threeTV">Three TV</string>
9  <string name="bravoTV">Bravo TV</string>
10 <string name="theEdgeTV">The Edge TV</string>
11 <!-- End of Channel Strings -->
12
13
14 <!-- Start of Now Playing Strings -->
15 <string name="nowPlaying">NOW PLAYING</string>
16 <string name="nowPlayingListC3">Placeholder Now Playing</string>
17 <string name="nowPlayingListC4">Placeholder Now Playing</string>
18 <string name="nowPlayingListC11">Placeholder Now Playing</string>
19 <!-- End of Now Playing Strings -->
20
21 <!-- Start of ...
22 <string name="...
23 <string name="...
24 <string name="...
25 <string name="...
26 <string name="...
27 <string name="...
28 <!-- End of ...
29 </resources>
  
```

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <alpha
3  xmlns:android="http://schemas.android.com/apk/res/android"
4  android:fromAlpha="0.0"
5  android:toAlpha="1.0"
6  android:duration="2000">
7  </alpha>
  
```

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <!-- XML Code from https://kylewbanks.com/ tutorial on Activity Animations -->
3  <set xmlns:android="http://schemas.android.com/apk/res/android">
4  <translate android:fromXDelta="-100%p" android:toXDelta="0"
5  android:interpolator="@android:anim/accelerate_decelerate_interpolator"
6  android:duration="@integer/slide_animation_duration"/>
7  </set>
  
```


Conclusion

In conclusion, the product itself is decent enough to fulfill the purpose of the app, which is to create a mobile app that's accessible and easy to use for the users. The app looks different from its initial design, the initial design was before I even knew what the limitations of both my current experience and skill, this means that some functions that I probably didn't even consider in the initial design is in the final outcome such as the Action Bar. Overall, the app itself is something that achieves its goals and purpose, it still has a lot of room for improvements that can be filled if I had enough experience and skill.