Coursework

Petru-Marian Burlacu
40285539@live.napier.ac.uk
Edinburgh Napier University - Mobile Applications Development — Set08114

1 Introduction

This coursework is a great opportunity to learn the process of creating a mobile application for Android Platform using Android SDK. My application idea remained the same from the beginning. As we discussed from the first day, I decided to create a voice recording application that would work on a locked screen and give users the option to record a voice memo on the go after an event (shake or pressing a button). The application would have a simple voice recording interface for the main app, with the options to record, play, and see the recordings. After doing some research I stumbled upon Secret Voice Recorder Application from Google Play. It had everything that I wanted to implement.

2 Software Design

For design process I wanted to stick, as much as possible, to the initial plan. I knew that I wanted to create my own images for drawable resource folder. On the Main Activity Recording screen I wanted to have buttons that would allow users to record, stop the recording, basic timer to show for audio recordings and a button to send users to another activity where they could see all of the audio recordings. For the locked screen recording feature I knew that I had to create a connection between a physical button/ gesture and recording process.

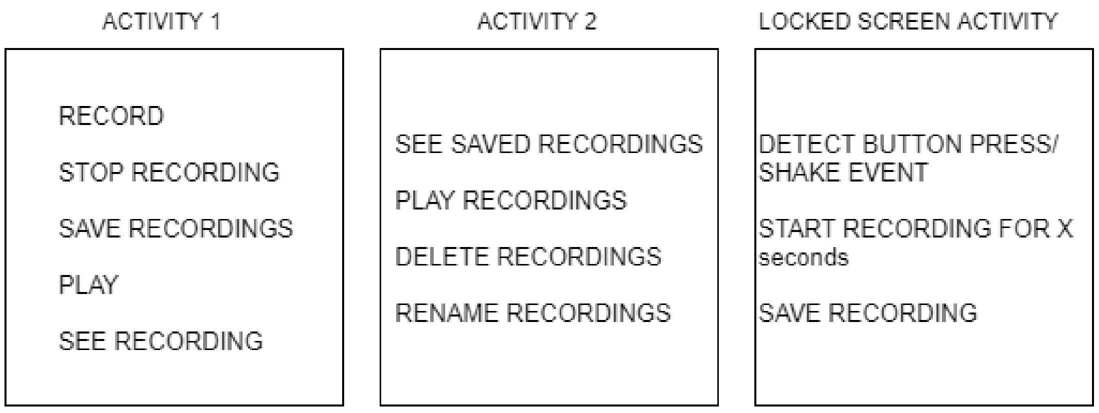
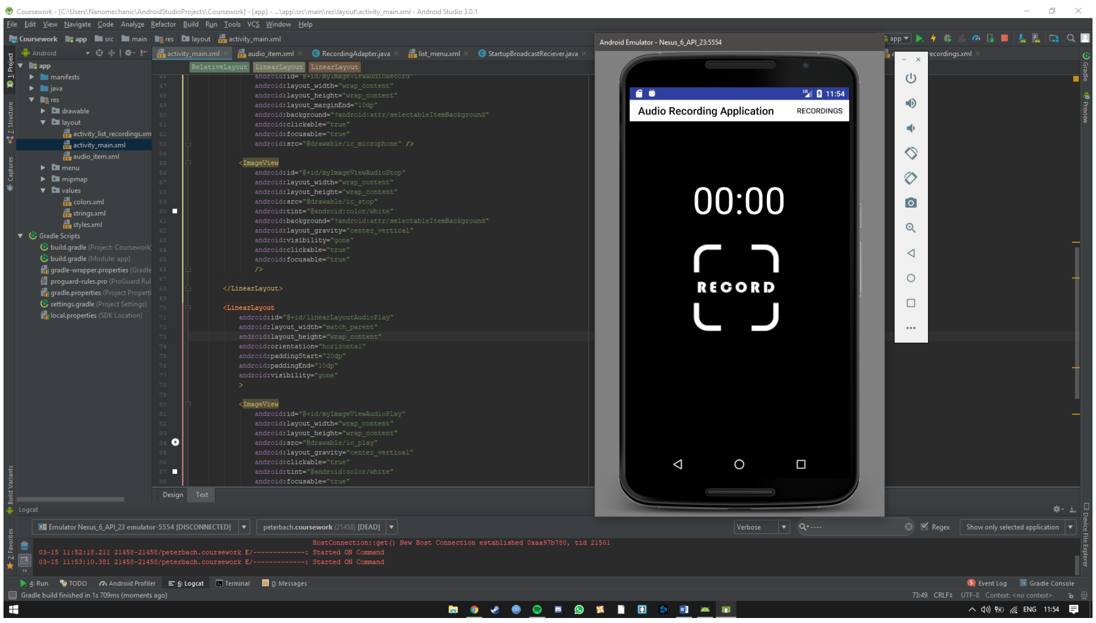


Figure 1: **Inititial Plan**

3 Implementation

The xml layouts were created using Relative Layout, instead of constrains. I found it annoying to design my views using the default constrains. Relative Layout feels similar to web design in some cases and intuitive to design with in xml.The main screen (main activity) contains a chronometer, a Seek Bar, 4 buttons and manage the recording process.


Coursework - [C:\Users\Nanomechanic\AndroidStudioProjects\Coursework] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 3.0.1
activity_main.xml
audio_item.xml
RecordingAdapter.java
list_menu.xml
StartupBroadcastReciever.java
Android Emulator - Nexus_6_API_23:5554
Audio Recording Application
RECORDINGS
00:00
RECORD
Design
Text

```
            android:id="@+id/myImageViewAudioRecord"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginEnd="10dp"
            android:background="?android:attr/selectableItemBackground"
            android:clickable="true"
            android:focusable="true"
            android:src="@drawable/ic_microphone" />

        <ImageView
            android:id="@+id/myImageViewAudioStop"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/ic_stop"
            android:tint="@android:color/white"
            android:background="?android:attr/selectableItemBackground"
            android:layout_gravity="center_vertical"
            android:visibility="gone"
            android:clickable="true"
            android:focusable="true"
            />

    </LinearLayout>

    <LinearLayout
        android:id="@+id/linearLayoutAudioPlay"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:paddingStart="20dp"
        android:paddingEnd="10dp"
        android:visibility="gone"
        >

        <ImageView
            android:id="@+id/myImageViewAudioPlay"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/ic_play"
            android:layout_gravity="center_vertical"
            android:clickable="true"
            android:tint="@android:color/white"
            android:focusable="true"
```

```
Emulator Nexus_6_API_23 emulator-5554 [DISCONNECTED]    peterbach.coursework (21458) [DEAD]
                                                                   HostConnection::get() New Host Connection established 0xaa97b780, tid 21501
03-15 11:52:18.211 21458-21458/peterbach.coursework E/-------------: Started ON Command
03-15 11:53:10.381 21458-21458/peterbach.coursework E/-------------: Started ON Command
Gradle build finished in 1s 709ms (moments ago)
```

Figure 2:

In this project, I used RecyclerView library to display the audio recordings. (build.gradle Module: app): compile 'com.android.support:recyclerview-v7:26.0.0-alpha1'

The main view starts with a timer and a record button. The stop button will appear after the record button is pressed. The Seek Bar with the play button would remain hidden until a recording is completed and the stop button is hit. This is done using the onClick method and changing the visibility of objects.

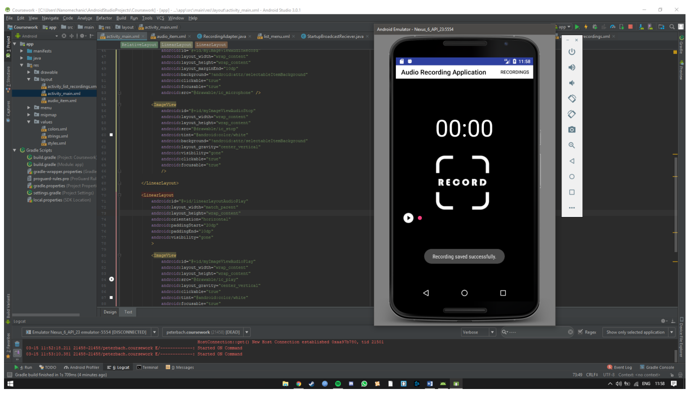
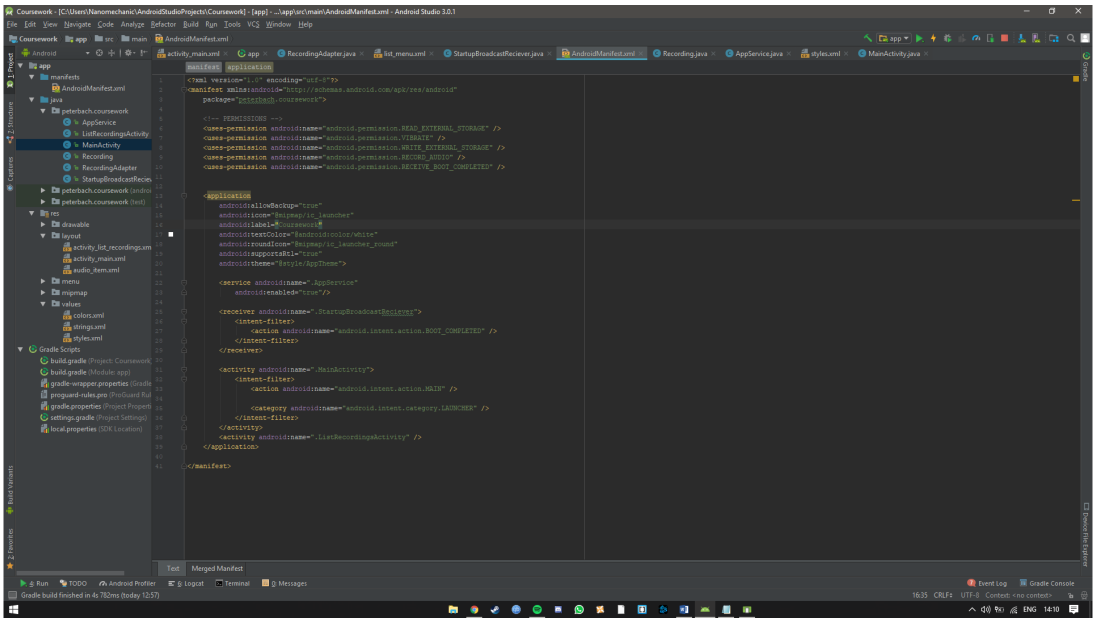


Figure 3:

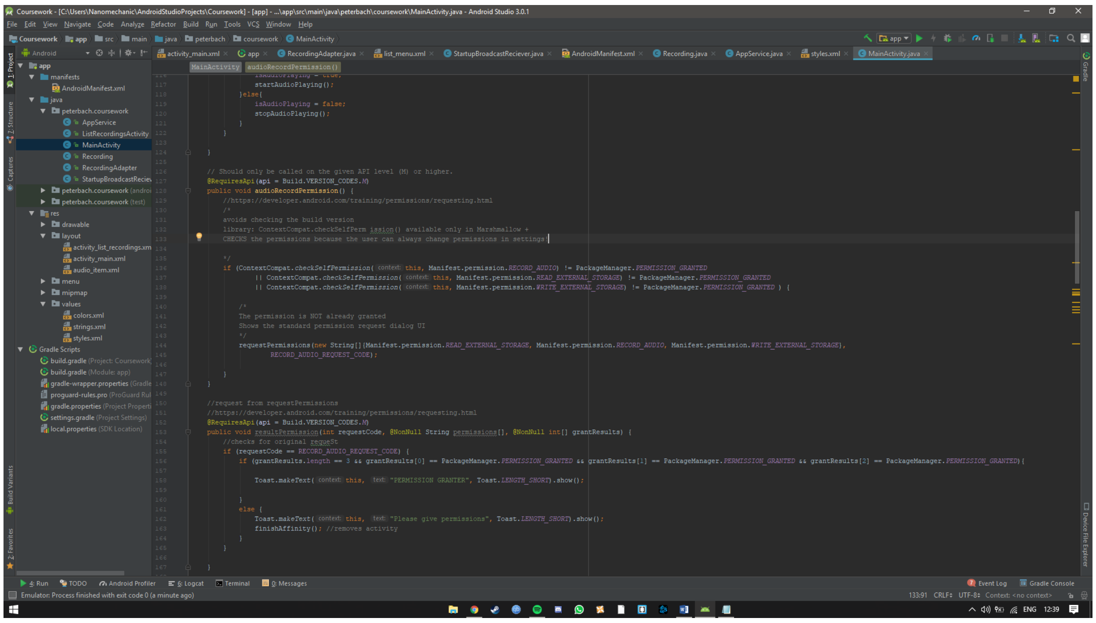
Some permissions had to be mentioned in the AndroidManifest.xml in order to set up the recording process. We know that to record we need permission to record and audio, save somewhere this audio file and read it whenever its needed. The following permissions allow us to do this:



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

    <!-- PERMISSIONS -->
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.VIBRATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />


    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Coursework"
        android:textColor="@android:color/white"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <service android:name=".AppService"
            android:enabled="true"/>

        <receiver android:name=".StartupBroadcastReciever">
            <intent-filter>
                <action android:name="android.intent.action.BOOT_COMPLETED" />
            </intent-filter>
        </receiver>

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

Figure 4:

For Marshmallow android version user has to accept permissions in the runtime. In order to do so, I discovered that I must include a request for permissions and user input. (https://developer.android.com/training/permissions/requesting.html). At this point I understood that programming logic is different. You have to check for permissions for every feature. Not just call a function and expect for the application to work. Ignored permissions would mean an application crash.



```java
                    isAudioPlaying = true;
                    startAudioPlaying();
                }else{
                    isAudioPlaying = false;
                    stopAudioPlaying();
                }
            }

    }

    // Should only be called on the given API level (M) or higher.
    @RequiresApi(api = Build.VERSION_CODES.M)
    public void audioRecordPermission() {
        //https://developer.android.com/training/permissions/requesting.html
        /*
        avoids checking the build version
        library: ContextCompat.checkSelfPerm ission() available only in Marshmallow +
        CHECKS the permissions because the user can always change permissions in settings!

        */
        if (ContextCompat.checkSelfPermission(this, Manifest.permission.RECORD_AUDIO) != PackageManager.PERMISSION_GRANTED
                || ContextCompat.checkSelfPermission(this, Manifest.permission.READ_EXTERNAL_STORAGE) != PackageManager.PERMISSION_GRANTED
                || ContextCompat.checkSelfPermission(this, Manifest.permission.WRITE_EXTERNAL_STORAGE) != PackageManager.PERMISSION_GRANTED ) {

            /*
            The permission is NOT already granted
            Shows the standard permission request dialog UI
            */
            requestPermissions(new String[]{Manifest.permission.READ_EXTERNAL_STORAGE, Manifest.permission.RECORD_AUDIO, Manifest.permission.WRITE_EXTERNAL_STORAGE},
                    RECORD_AUDIO_REQUEST_CODE);

        }
    }

    //request from requestPermissions
    //https://developer.android.com/training/permissions/requesting.html
    @RequiresApi(api = Build.VERSION_CODES.M)
    public void resultPermission(int requestCode, @NonNull String permissions[], @NonNull int[] grantResults) {
        //checks for original requeSt
        if (requestCode == RECORD_AUDIO_REQUEST_CODE) {
            if (grantResults.length == 3 && grantResults[0] == PackageManager.PERMISSION_GRANTED && grantResults[1] == PackageManager.PERMISSION_GRANTED && grantResults[2] == PackageManager.PERMISSION_GRANTED){

                Toast.makeText(this, "PERMISSION GRANTER", Toast.LENGTH_SHORT).show();

            }
            else {
                Toast.makeText(this, "Please give permissions", Toast.LENGTH_SHORT).show();
                finishAffinity(); //removes activity
            }
        }

    }
```

Figure 5:

More than that, sometimes checking and adding permissions would not be enough. For example I encountered an Error for the PERMISSION RECORD AUDIO REQUEST CODE. To fix the error I had to initialize a default code for the request code, equal to 123. (https://inthecheesefactory.com/blog/things-you-need-to-know-about-android-m-permission-developer-edition/en).

Seek Bar progress is updated by calling the runnable every 100ms. Audio recording is played by declaring a Media Player object. It is initialized in the startAudioPlaying() Method. The audio file address is called in the method.

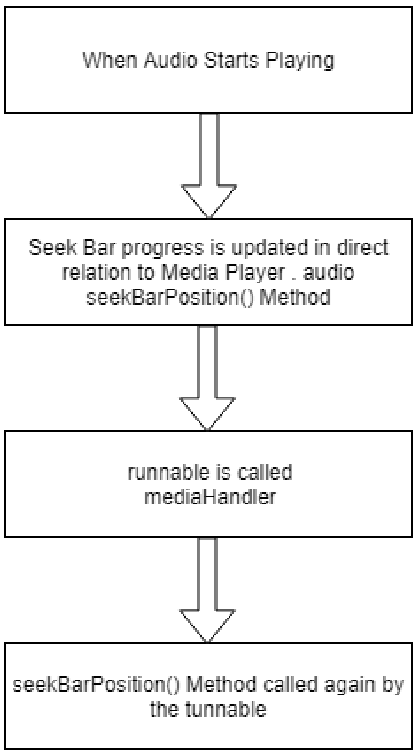
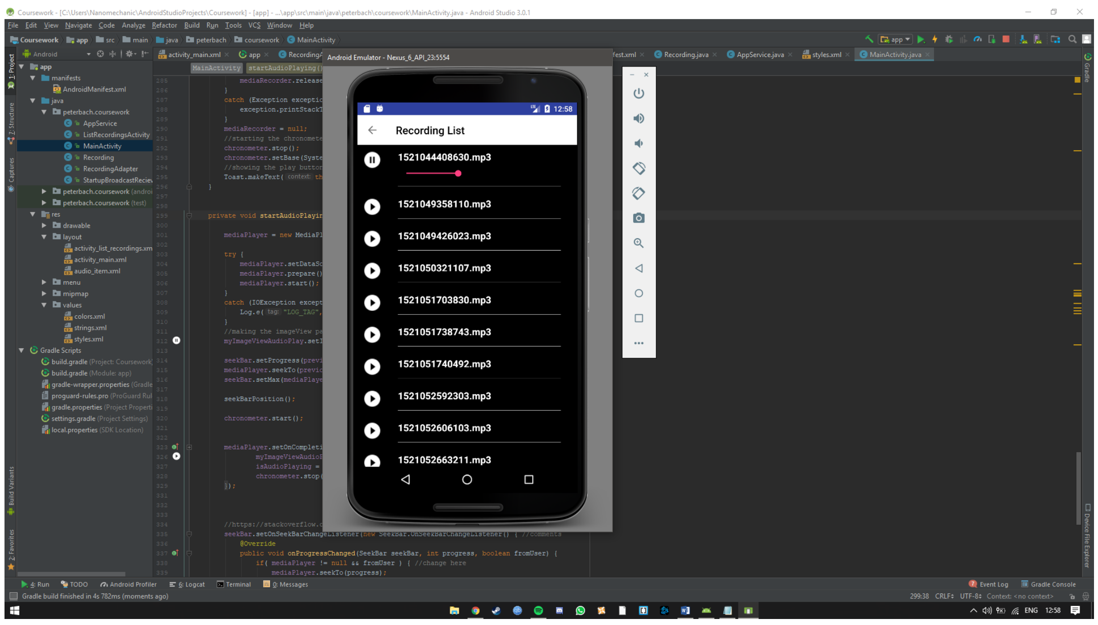

When Audio Starts Playing
Seek Bar progress is updated in direct relation to Media Player . audio seekBarPosition() Method
runnable is called mediaHandler
seekBarPosition() Method called again by the tunnable

Figure 6: **Audio Play / Seek Bar Combo** - flow

In the list of recordings, there is a seek bar and play/stop button for each recording. Following the same principle.



```
            mediaRecorder.releas
        }
        catch (Exception exceptic
            exception.printStackT
        }
        mediaRecorder = null;
        //starting the chronomete
        chronometer.stop();
        chronometer.setBase(Syste
        //showing the play button
        Toast.makeText( context: th
    }

    private void startAudioPlayin

        mediaPlayer = new MediaPl

        try {
            mediaPlayer.setDataSc
            mediaPlayer.prepare()
            mediaPlayer.start();
        }
        catch (IOException except
            Log.e( tag: "LOG_TAG",
        }
        //making the imageView pa
        myImageViewAudioPlay.setI

        seekBar.setProgress(previ
        mediaPlayer.seekTo(previc
        seekBar.setMax(mediaPlaye

        seekBarPosition();

        chronometer.start();


        mediaPlayer.setOnCompleti
                myImageViewAudioE
                isAudioPlaying =
                chronometer.stop(
        });



        //https://stackoverflow.c
        seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() { //comments
            @Override
            public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
                if( mediaPlayer != null && fromUser ) { //change here
                    mediaPlayer.seekTo(progress);
```

Figure 7:

To take the audio files from storage, put them in the ArrayList and update the recycker view of audio recordings shown in the previous picture, we use RecordingAdapter Java class.

To access the list of audio recordings I created a directory by the name menu and used on click options menu. (Top Right Corner of App main screen)

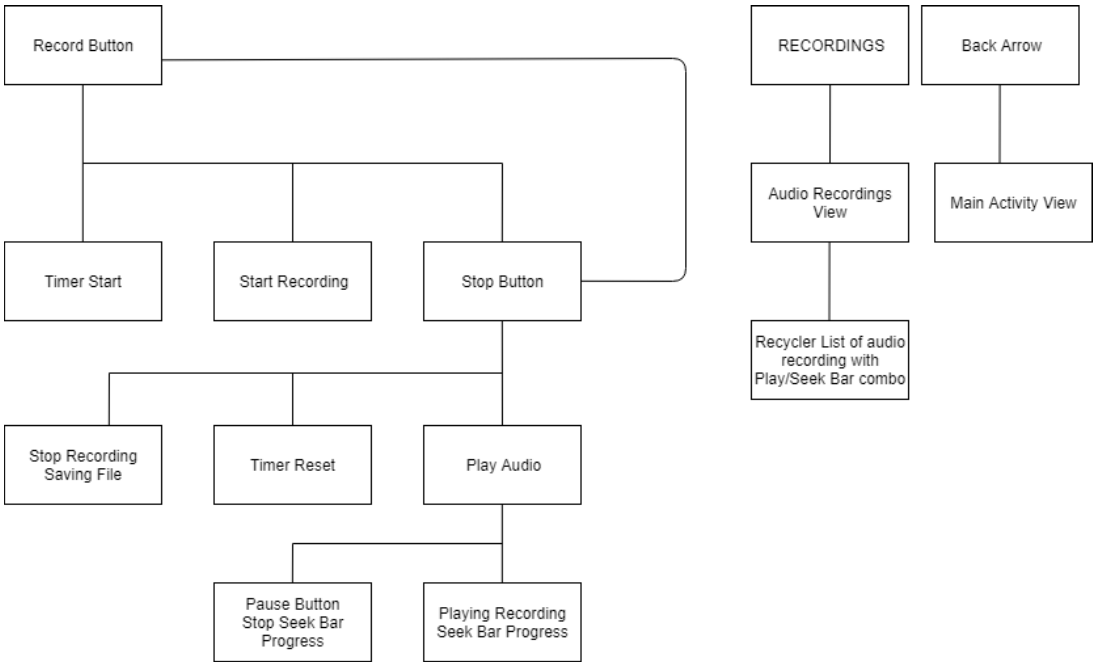
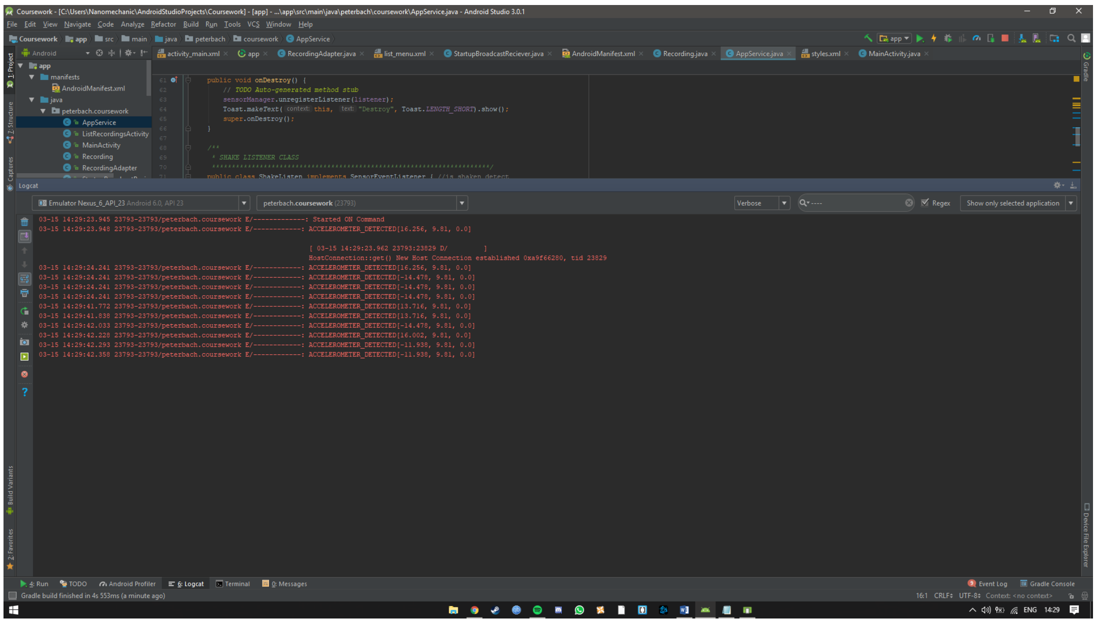

Record Button
Timer Start
Start Recording
Stop Button
Stop Recording Saving File
Timer Reset
Play Audio
Pause Button Stop Seek Bar Progress
Playing Recording Seek Bar Progress
RECORDINGS
Audio Recordings View
Recycler List of audio recording with Play/Seek Bar combo
Back Arrow
Main Activity View

Figure 8:

The last part is the background voice recording feature. In order to be able to run on a locked screen, I created a Startup Broadcast Receiver to listen for a shake event. I managed to detect the shake event using Sensor Event Listener and changes in Accelerometer values. My problem would be implementing the voice recording methods during the shake. I couldnâĂŹt figure out a way to restrinct the methods to initialize only once after the shake event begins. Bacause it is receiving multiple constant changes in accelerometer values it proved to be difficult not to crash the app while trying to record once an audio. I learned how to record an audio for a specific amount of time without overlapping with my other methods. I know where and how to start a method when the app is detecting sudden motion. This can be proven by logs.



```java
    public void onDestroy() {
        // TODO Auto-generated method stub
        sensorManager.unregisterListener(listener);
        Toast.makeText( context: this,  text: "Destroy", Toast.LENGTH_SHORT).show();
        super.onDestroy();
    }

    /**
     * SHAKE LISTENER CLASS
     ************************************************************************/
    public class ShakeListen implements SensorEventListener { //is shaken detect
```

```
03-15 14:29:23.945 23793-23793/peterbach.coursework E/-------------: Started ON Command
03-15 14:29:23.948 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[16.256, 9.81, 0.0]

                                                                    [ 03-15 14:29:23.962 23793:23829 D/         ]
                                                                    HostConnection::get() New Host Connection established 0xa9f66280, tid 23829
03-15 14:29:24.241 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[16.256, 9.81, 0.0]
03-15 14:29:24.241 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[-14.478, 9.81, 0.0]
03-15 14:29:24.241 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[-14.478, 9.81, 0.0]
03-15 14:29:24.241 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[-14.478, 9.81, 0.0]
03-15 14:29:41.772 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[13.716, 9.81, 0.0]
03-15 14:29:41.838 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[13.716, 9.81, 0.0]
03-15 14:29:42.033 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[-14.478, 9.81, 0.0]
03-15 14:29:42.228 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[16.002, 9.81, 0.0]
03-15 14:29:42.293 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[-11.938, 9.81, 0.0]
03-15 14:29:42.358 23793-23793/peterbach.coursework E/------------: ACCELEROMETER_DETECTED[-11.938, 9.81, 0.0]
```

Figure 9:

4 Critical Evaluation

The application is a bit heavy on battery performance due to the Broadcast Receiver that is running in the background on os start up. I had in mind, from the beginning to add a feature to record audio after an event, even on a locked screen. My issue was that whenever the app would get a shake event I could not stop it or limit the interaction. In other words, it would feel like a constant loop once in progress. This would mean that my audio recorder should start every time a shake would be detected, even if the recording was not finished / stopped. This was the main cause behind the crashes. I tried several approaches using conditions for media recorder, limit the shake sensibility and even trying to do only one recording. If I had more time, I would have probably discovered a way to do it. The current state of this feature would be halfway through. It detects the shake event and adds statements in the log.

The rest of the app contains two activities. The Main activity contains the classic features of a audio recorder. The activity that shows a list of audio recordings has recycler view in the main body. With the options to play and stop the recordings. To complete the application I would need to add a function to delete individual recordings and fix the audio recording after a shake event. I learned to understand how a mobile application works and I can see the dangers and potential in mobile development.

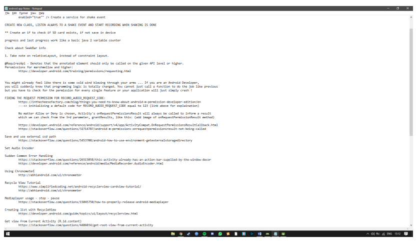
5 Personal Evaluation

From the last coursework I learned that it is a safe practice to always save the references in a file while implementing them and take notes while developing. This helped me to keep a structure and a flow in mind.

One of the challenging parts was to learn how different the mobile application development is from traditional programming. I had to research new methods and techniques available for android studio, understand how important permissions are and interdependencies between elements. Even now, after completing the coursework I believe that I have so much more to learn.

I started by following my initial idea and tried to get as close as possible to the end result that I had in mind. During the process I understood my limits and always tried to research new ways to overcome them. This ended up in being one of the more interesting parts of the coursework for me. In my opinion, this coursework was a good experience for me to understand the challenges behind mobile applications development.

Overall, planning my steps in advance game me a clear path to follow. The challenges encountered required me to ask for help and do serious research on different areas. I enjoyed working on this project but I always have the feeling that I could have done better.



```
        enabled="true"" /> Create a service for shake event

CREATE NEW CLASS, LISTEN ALWAYS TO A SHAKE EVENT AND START RECORDING WHEN SHAKING IS DONE

** Create an if to check if SD card exists, if not save in device

progress and last progress work like a basic java 2 variable counter

Check about SeekBar info

1. Take note on relativeLayout, instead of constraint layout.

@RequiresApi - Denotes that the annotated element should only be called on the given API level or higher.
Permissions for marshmellow and higher:
        https://developer.android.com/training/permissions/requesting.html


You might already feel like there is some cold wind blowing through your arms ... If you are an Android Developer,
you will suddenly know that programming logic is totally changed. You cannot just call a function to do the job like previous
but you have to check for the permission for every single feature or your application will just simply crash !

FIXING THE REQUEST PERMISION FOR RECORD_AUDIO_REQUEST_CODE:
        https://inthecheesefactory.com/blog/things-you-need-to-know-about-android-m-permission-developer-edition/en
        --->> initializing a default code for RECORD_AUDIO_REQUEST_CODE equal to 123 (link above for explaination)

        No matter Allow or Deny is chosen, Activity's onRequestPermissionsResult will always be called to inform a result
        which we can check from the 3rd parameter, grantResults, like this: (add image of onRequestPermissionResult method)

        https://developer.android.com/reference/android/support/v4/app/ActivityCompat.OnRequestPermissionsResultCallback.html
        https://stackoverflow.com/questions/32714787/android-m-permissions-onrequestpermissionsresult-not-being-called

Save and use external ssd path
        https://stackoverflow.com/questions/5453708/android-how-to-use-environment-getexternalstoragedirectory

Set Audio Encoder

Sudden Common Error Handling
        https://stackoverflow.com/questions/26515058/this-activity-already-has-an-action-bar-supplied-by-the-window-decor
        https://developer.android.com/reference/android/media/MediaRecorder.AudioEncoder.html

Using Chronometer
        http://abhiandroid.com/ui/chronometer

Recycle View Tutorial
        https://www.simplifiedcoding.net/android-recyclerview-cardview-tutorial/
        http://abhiandroid.com/ui/chronometer

Mediaplayer usage - stop - pause
        https://stackoverflow.com/questions/15045750/how-to-properly-release-android-mediaplayer

Creating list with RecycleView
        https://developer.android.com/guide/topics/ui/layout/recyclerview.html

Get view from Current Activity (R.id.content)
        https://stackoverflow.com/questions/4486034/get-root-view-from-current-activity
```

Figure 10:

References

PERMISSIONS FOR MARSHMALLOW AND HIGHER
https://developer.android.com/training/permissions/requesting.html

FIXING THE REQUEST PERMISION ERRORS
https://inthecheesefactory.com/blog/things-you-need-to-know-about-android-m-permission-dev
https://developer.android.com/reference/android/support/v4/app/ActivityCompat.OnRequestPer
https://www.simplifiedcoding.net/audio-recording-android-example/
https://stackoverflow.com/questions/32714787/android-m-permissions-onrequestpermissionsres

SAVE AND USE EXTERNAL SD FILE
https://stackoverflow.com/questions/5453708/android-how-to-use-environment-getexternalstor

SUDDEN COMMON ERRORS HANDLING
ttps://stackoverflow.com/questions/26515058/this-activity-already-has-an-action-bar-suppli
https://developer.android.com/reference/android/media/MediaRecorder.AudioEncoder.html

USING CHRONOMETER
http://abhiandroid.com/ui/chronometer

RECYCLER VIEW TUTORIAL
https://www.simplifiedcoding.net/android-recyclerview-cardview-tutorial/

MEDIAPLAYER USAGE - SOP - PLAY - PAUSE
https://stackoverflow.com/questions/15045750/how-to-properly-release-android-mediaplayer

CREATING LIST WITH RECYCLER VIEW
https://developer.android.com/guide/topics/ui/layout/recyclerview.html

GETTING ROOT VIEW
https://stackoverflow.com/questions/4486034/get-root-view-from-current-activity

RECYCLER VIEW ERROR SOLVING
https://stackoverflow.com/questions/24440852/how-to-import-recyclerview-for-android-l-prev

NONEXISTENT FILE AVOID CRASH
https://docs.oracle.com/javase/6/docs/api/java/io/File.htmlmkdirs

MAIN TECHNIQUES AND TUTORIALS USED
Android Programming for Beginners - by John Horton

ON BOOT BROADCAST RECEIVER
https://stackoverflow.com/questions/20595337/how-to-start-service-at-device-boot-in-androi

SERVICE
https://developer.android.com/reference/android/app/Service.html
https://androidammy.blogspot.co.uk/2015/05/accelerometer-shake-events-example-with.html

SENSOR MANAGER
https://developer.android.com/reference/android/hardware/SensorManager.html

AUDIO FORMAT
https://developer.android.com/guide/topics/media/media-formats.html

SEEK BAR
http://abhiandroid.com/ui/seekbar
http://mrbool.com/how-to-play-audio-files-in-android-with-a-seekbar-feature-and-mediaplaye
https://stackoverflow.com/questions/8956218/android-seekbar-setonseekbarchangelistener

ACCELEROMETER DETECT

`https://stackoverflow.com/questions/2317428/android-i-want-to-shake-it`

MAIN TECHNIQUES AND TUTORIALS USED

`Android Programming for Beginners - by John Horton`

ON BOOT BROADCAST RECEIVER

`https://stackoverflow.com/questions/20595337/how-to-start-service-at-device-boot-in-androi`

SERVICE

`https://developer.android.com/reference/android/app/Service.html`

`https://androidammy.blogspot.co.uk/2015/05/accelerometer-shake-events-example-with.html`

SENSOR MANAGER

`https://developer.android.com/reference/android/hardware/SensorManager.html`