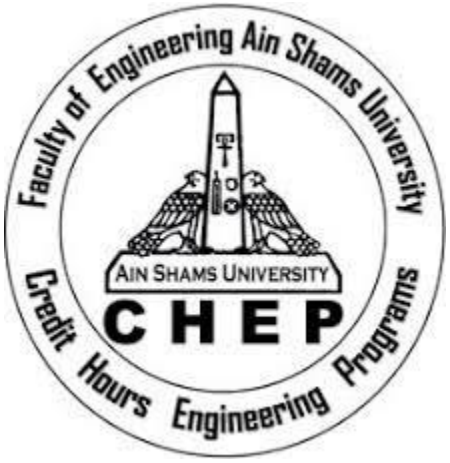


CSE 430

Advanced Mobile Computing

Project

Submitted by: Ibrahim Ahmed Yehia Salama (16P6036)

App Description

This app is smart remote controller app where user can control devices in his/her house such as TVs and ACs from their mobile regardless of the location they are at. It allows user to add as many devices as they want.

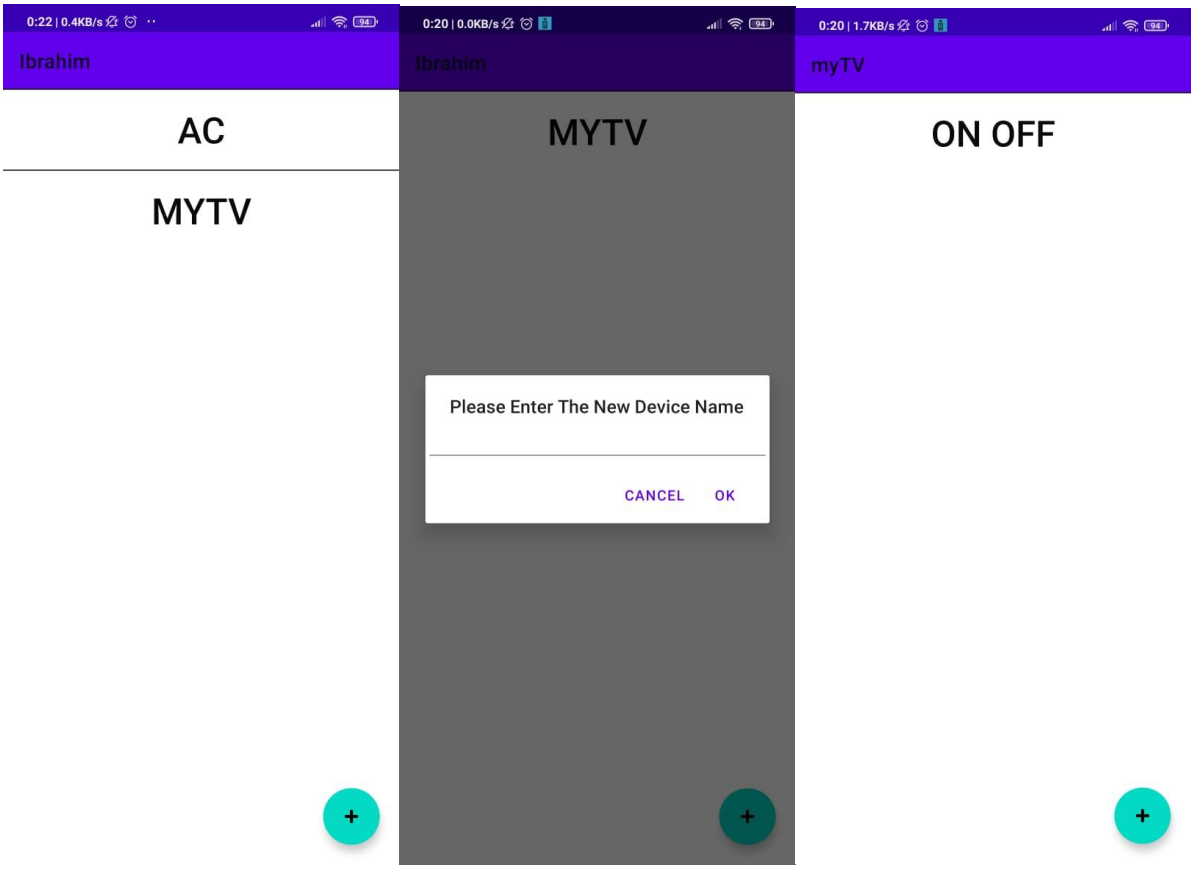
How the app works

The app works by connecting to a database to retrieve saved devices and display list on the phone user can add, remove and edit device or button name, when a user click on a device from the displayed devices it displays the list of buttons of this device. When a button is clicked the app writes the IR function code of this particular device in the database where the nodeMCU can read and send this code via IR to perform the function. When adding a new button user puts nodeMCU in receiving mode so it can receive IR code from the device remote controller then user enters buttons name press ok then press the button he wants to add from device remote controller so that the app can read code from database and save it.

List of features

- Add Device
- Remove Device
- Edit Device Name
- Add Button
- Remove Button
- Edit Device Button
- Multi-selection
- Control Devices that works with IR

Screenshots



Test Cases

- Open app click a device a list of buttons of this device should appear.
- Open app click on add button a dialog will appear where user can type name of new device and add it, new device should appear in the list.
- Open app long press on a device the app should enter selection mode where appBar changes color and view number of selected items as well as the edit and remove buttons in appBar.
- In the previous test case after entering selection mode when selecting multiple items the edit button will be disabled and when clicking remove all selected items will be removed. While clicking edit will make a dialog appear where user can change device name.
- When click on a certain device a list of buttons of this device will appear where user can also add, remove or edit button name.

- When clicking on a button the app writes the value of the button function code in DB so the hardware can read and perform it.
- Add and edit features won't work when there is no internet connection instead it displays a toast so user can know that they are not connected to internet.

MainActivity.java

```
package com.example.smartremotecontroller;


import android.app.AlertDialog;

import android.content.DialogInterface;

import android.graphics.Color;

import android.graphics.drawable.ColorDrawable;

import android.os.Build;

import android.os.Bundle;

import android.text.InputType;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.FrameLayout;

import android.widget.Toast;


import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

import androidx.appcompat.app.AppCompatActivity;

import androidx.appcompat.widget.Toolbar;
```

```
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProviders;
import androidx.recyclerview.selection.SelectionPredicates;
import androidx.recyclerview.selection.SelectionTracker;
import androidx.recyclerview.selection.StableIdKeyProvider;
import androidx.recyclerview.selection.StorageStrategy;
import androidx.recyclerview.widget.RecyclerView;

import com.google.android.material.floatingactionbutton.FloatingActionButton;

import java.io.IOException;
import java.util.Iterator;

public class MainActivity extends AppCompatActivity {
    private String userName;
    private FloatingActionButton addDeviceButton;
    private String []deviceList = {""};
    private CustomAdapter adapter;
    private RecyclerView r;
    private MyViewModel model;
    private Toolbar toolbar = null;
    private SelectionTracker tracker;
    private Menu menu;
    private boolean selected = false;
    private String selectedItems[];
    private MenuItem edit;
    private MenuItem remove;
```

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

        if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.LOLLIPOP) {
            toolbar = (Toolbar) findViewById(R.id.toolbar);
            showOverflowMenu(false);
        }
        setSupportActionBar(toolbar);
        r = findViewById(R.id.DeviceList);
        adapter = new CustomAdapter(deviceList, true);
        showOverflowMenu(false);
        AlertDialog.Builder alertDialogueBuilder = new AlertDialog.Builder(this);
        model = ViewModelProviders.of(this).get(MyViewModel.class);
        model.getData().observe(this, new Observer<User>() {
            @Override
            public void onChanged(@Nullable User user) {
                if (user != null) {
                    userName = user.getUserName();
                    toolbar.setTitle(userName);
                    deviceList = user.getDevicesList().keySet().toArray(new
String[user.getDevicesList().size()]);
                    adapter.setLocalDataSet(deviceList);
                    if (deviceList.length > 0) {
                        r.setAdapter(adapter);
                        if(tracker==null)
```

```
                tracker = new SelectionTracker.Builder<Long>(
                        "my-selection-id",
                        r,
                        new StableIdKeyProvider(r),
                        new MyDetailsLookup(r),
                        StorageStrategy.createLongStorage())
                        .withSelectionPredicate(SelectionPredicates.createSelectAnything())
                        .build();
            adapter.setTracker(tracker);
            tracker.addObserver(new SelectionTracker.SelectionObserver<Long>(){
                @Override
                public void onSelectionChanged(){
                    int numItems = tracker.getSelection().size();
                    if(numItems == 1){
                        edit.setEnabled(true);
                        selected = true;
                        showOverflowMenu(true);
                        getSupportActionBar().setTitle(numItems+" items selected");
                        getSupportActionBar().setBackgroundDrawable(new
ColorDrawable((Color.parseColor("#ef6c00"))));
                    }
                    else if(numItems > 0) {
                        edit.setEnabled(false);
                        selected = true;
                        showOverflowMenu(true);
                        getSupportActionBar().setTitle(numItems+" items selected");
                        getSupportActionBar().setBackgroundDrawable(new
ColorDrawable((Color.parseColor("#ef6c00"))));
```

```
                    } else {
                        selected = false;
                        showOverflowMenu(false);
                        // Reset color and title to default values
                        getSupportActionBar().setTitle(userName);
                        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                            toolbar = findViewById(R.id.toolbar);
                            setSupportActionBar(toolbar);
                            getSupportActionBar().setBackgroundDrawable(new
ColorDrawable(getColor(R.color.design_default_color_primary)));
                        }
                    }
                }
            });
        }
        adapter.notifyItemChanged(0,deviceList.length);
        adapter.notifyDataSetChanged();
      }
    }
  });

  addDeviceButton =  findViewById(R.id.floatingActionButton);
  addDeviceButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      alertDialogueBuilder.setTitle("Please Enter The New Device Name");
      // Set up the input
      final EditText input = new EditText(getBaseContext());
```

```
        // Specify the type of input expected; this, for example, sets the input as a password,
and will mask the text
        input.setInputType(InputType.TYPE_CLASS_TEXT);
        alertDialogueBuilder.setView(input);

        // Set up the buttons
        alertDialogueBuilder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
           @Override
           public void onClick(DialogInterface dialog, int which) {
              String deviceName = input.getText().toString();
              if (((deviceName.contains("/")|| deviceName.contains(".")||
deviceName.contains("#")
                    || deviceName.contains("$")||deviceName.contains("[") ||
deviceName.contains("]")
                    || deviceName.equals("")))) {
                 Toast.makeText(getApplicationContext(),
                       "Name must not contain special characters or an empty name",
Toast.LENGTH_LONG).show();
                 return;
              }
              if(isInternetAvailable())
                 model.addDevice(deviceName);
              else
                 Toast.makeText(getApplicationContext(),
                       "No Internet Connection, Please make sure you are connected to internet",
Toast.LENGTH_LONG).show();
           }
        });
```

```
            alertDialogueBuilder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    dialog.cancel();
                }
            });
            alertDialogueBuilder.show();
        }
    });

}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    this.menu = menu;
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    remove = menu.findItem(R.id.Remove).setEnabled(true);
    edit = menu.findItem(R.id.Edit);//.setEnabled(true);
    return true;
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if(item.getItemId() == R.id.Remove ) {
        int numItems = tracker.getSelection().size();
        selectedItems = new String[numItems];
```

```
        Iterator itr = tracker.getSelection().iterator();
        for (int i = 0; i < numItems; i++) {
            Long l = (Long) itr.next();
            FrameLayout f = (FrameLayout) r.getChildAt(l.intValue());
            Button b = (Button) f.getChildAt(0);
            selectedItems[i] = b.getText().toString();
        }
        tracker.clearSelection();
        for (int i = 0; i < selectedItems.length; i++)
        model.removeDevices(selectedItems);
        selectedItems = null;
        tracker.clearSelection();
    }
    else if(item.getItemId() == R.id.Edit){
        Iterator itr = tracker.getSelection().iterator();
        Long l = (Long) itr.next();
        FrameLayout f = (FrameLayout) r.getChildAt(l.intValue());
        Button b = (Button) f.getChildAt(0);
        String selectedDeviceName = b.getText().toString();;
        AlertDialog.Builder alertDialogueBuilder = new AlertDialog.Builder(this);
        alertDialogueBuilder.setTitle("Please Enter The New Device Name");
        // Set up the input
        final EditText input = new EditText(getBaseContext());
        // Specify the type of input expected; this, for example, sets the input as a password, and
will mask the text
        input.setInputType(InputType.TYPE_CLASS_TEXT);
        alertDialogueBuilder.setView(input);
```

```
        // Set up the buttons
        alertDialogueBuilder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                String deviceName = input.getText().toString();
                model.changeDeviceName(deviceName, selectedDeviceName);
                tracker.clearSelection();
            }
        });
        alertDialogueBuilder.setNegativeButton("Cancel", new DialogInterface.OnClickListener()
{
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.cancel();
            }
        });
        alertDialogueBuilder.show();
      }
      return true;
   }

   public void showOverflowMenu(boolean showMenu){
      if(menu == null)
         return;
      menu.setGroupVisible(R.id.main_menu_group, showMenu);
   }

   @Override
```

```
    public void onBackPressed() {
        if (selected)
            tracker.clearSelection();
        else
            onDestroy();
    }

    public static boolean isInternetAvailable() {
        String command = "ping -c 1 google.com";
        try {
            return Runtime.getRuntime().exec(command).waitFor() == 0;
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return false;
    }

}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/linearLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context=".MainActivity">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="?colorPrimary"
        android:minHeight="?attr/actionBarSize"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:popupTheme="@style/ThemeOverlay.AppCompat.Light" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/DeviceList"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
```

```
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/toolbar"
    app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"/>

  <com.google.android.material.floatingactionbutton.FloatingActionButton
    android:id="@+id/floatingActionButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="20dp"
    android:layout_marginRight="20dp"
    android:layout_marginBottom="20dp"
    android:clickable="true"
    android:contentDescription="Add Device"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:srcCompat="@android:drawable/ic_input_add" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Video Link

https://youtu.be/fILV3nfqY7k