**C. SOURCE CODE**

**Arduino Code:**

**int Motor\_A\_Enable = 9;**

**int Motor\_A\_Reverse = 3;**

**int Motor\_A\_Forward = 4;**

**int Motor\_B\_Enable = 10;**

**int Motor\_B\_Reverse = 12;**

**int Motor\_B\_Forward = 7;**

**void setup() {**

**// put your setup code here, to run once:**

**Serial.begin(9600);**

**pinMode(Motor\_A\_Enable, OUTPUT);**

**pinMode(Motor\_A\_Forward, OUTPUT);**

**pinMode(Motor\_A\_Reverse, OUTPUT);**

**pinMode(Motor\_B\_Enable, OUTPUT);**

**pinMode(Motor\_B\_Forward, OUTPUT);**

**pinMode(Motor\_B\_Reverse, OUTPUT);**

**}**

**void loop() {**

**if(Serial.available() > 0)**

**{**

**char data;**

**data = Serial.read();**

**Serial.write(Serial.read());**

**switch (data)**

**{**

**case '1': //FORWARD**

**analogWrite(Motor\_B\_Enable, 100);**

**analogWrite(Motor\_A\_Enable, 100);**

**digitalWrite(Motor\_A\_Reverse, LOW);**

**digitalWrite(Motor\_B\_Reverse, LOW);**

**digitalWrite(Motor\_A\_Forward, HIGH);**

**digitalWrite(Motor\_B\_Forward, HIGH);**

**break;**

**case '2': //REVERSE**

**analogWrite(Motor\_B\_Enable, 100);**

**analogWrite(Motor\_A\_Enable, 100);**

**digitalWrite(Motor\_A\_Forward, LOW);**

**digitalWrite(Motor\_B\_Forward, LOW);**

**digitalWrite(Motor\_A\_Reverse, HIGH);**

**digitalWrite(Motor\_B\_Reverse, HIGH);**

**break;**

**case '3': //FORWARD LEFT**

**analogWrite(Motor\_A\_Enable, 100);**

**analogWrite(Motor\_B\_Enable, 0);**

**digitalWrite(Motor\_A\_Reverse, LOW);**

**digitalWrite(Motor\_A\_Forward, HIGH);**

**break;**

**case '4': //FORWARD RIGHT**

**analogWrite(Motor\_B\_Enable, 100);**

**analogWrite(Motor\_A\_Enable, 0);**

**digitalWrite(Motor\_B\_Reverse, LOW);**

**digitalWrite(Motor\_B\_Forward, HIGH);**

**break;**

**case '5': //REVERSE LEFT**

**analogWrite(Motor\_A\_Enable, 100);**

**analogWrite(Motor\_B\_Enable, 0);**

**digitalWrite(Motor\_A\_Reverse, HIGH);**

**digitalWrite(Motor\_A\_Forward, LOW);**

**break;**

**case '6': //REVERSE RIGHT**

**analogWrite(Motor\_B\_Enable, 100);**

**analogWrite(Motor\_A\_Enable, 0);**

**digitalWrite(Motor\_B\_Reverse, HIGH);**

**digitalWrite(Motor\_B\_Forward, LOW);**

**break;**

**default: //If bluetooth module receives any value not listed above, both motors turn off**

**analogWrite(Motor\_A\_Enable, 0);**

**analogWrite(Motor\_B\_Enable, 0);**

**}**

**}**

**}**

**Java Code:**

**package** com.example.philipgo.arduinobluetoothrccar;  
  
**import** android.bluetooth.BluetoothAdapter;  
**import** android.bluetooth.BluetoothDevice;  
**import** android.bluetooth.BluetoothSocket;  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.MotionEvent;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.Toast;  
**import** java.io.IOException;  
**import** java.io.OutputStream;  
**import** java.util.Set;  
**import** java.util.UUID;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 **private final** String **DEVICE\_ADDRESS** = **"98:D3:61:F5:EC:33"**; *//MAC Address of Bluetooth Module* **private final** UUID **PORT\_UUID** = UUID.*fromString*(**"00001101-0000-1000-8000-00805f9b34fb"**);  
  
 **private** BluetoothDevice **device**;  
 **private** BluetoothSocket **socket**;  
 **private** OutputStream **outputStream**;  
  
 Button **forward\_btn**, **forward\_left\_btn**, **forward\_right\_btn**, **reverse\_btn**, **reverse\_left\_btn**, **reverse\_right\_btn**, **bluetooth\_connect\_btn**;  
  
 String **command**; *//string variable that will store value to be transmitted to the bluetooth module* @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *//declaration of button variables* **forward\_btn** = (Button) findViewById(R.id.***forward\_btn***);  
 **forward\_left\_btn** = (Button) findViewById(R.id.***forward\_left\_btn***);  
 **forward\_right\_btn** = (Button) findViewById(R.id.***forward\_right\_btn***);  
 **reverse\_btn** = (Button) findViewById(R.id.***reverse\_btn***);  
 **reverse\_right\_btn** = (Button) findViewById(R.id.***reverse\_right\_btn***);  
 **reverse\_left\_btn** = (Button) findViewById(R.id.***reverse\_left\_btn***);  
 **bluetooth\_connect\_btn** = (Button) findViewById(R.id.***bluetooth\_connect\_btn***);  
  
 *//OnTouchListener code for the forward button (button long press)* **forward\_btn**.setOnTouchListener(**new** View.OnTouchListener() {  
 @Override  
 **public boolean** onTouch(View v, MotionEvent event) {  
  
 **if** (event.getAction() == MotionEvent.***ACTION\_DOWN***) *//MotionEvent.ACTION\_DOWN is when you hold a button down* {  
 **command** = **"1"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes()); *//transmits the value of command to the bluetooth module* }  
 **catch** (IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **else if**(event.getAction() == MotionEvent.***ACTION\_UP***) *//MotionEvent.ACTION\_UP is when you release a button* {  
 **command** = **"10"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch**(IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **return false**;  
 }  
 });  
  
 *//OnTouchListener code for the reverse button (button long press)* **reverse\_btn**.setOnTouchListener(**new** View.OnTouchListener(){  
 @Override  
 **public boolean** onTouch(View v, MotionEvent event)  
 {  
 **if**(event.getAction() == MotionEvent.***ACTION\_DOWN***)  
 {  
 **command** = **"2"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch** (IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **else if**(event.getAction() == MotionEvent.***ACTION\_UP***)  
 {  
 **command** = **"10"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch**(IOException e)  
 {  
 }  
 }  
 **return false**;  
 }  
 });  
 *//OnTouchListener code for the forward left button (button long press)* **forward\_left\_btn**.setOnTouchListener(**new** View.OnTouchListener(){  
 @Override  
 **public boolean** onTouch(View v, MotionEvent event)  
 {  
 **if**(event.getAction() == MotionEvent.***ACTION\_DOWN***)  
 {  
 **command** = **"3"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch** (IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **else if**(event.getAction() == MotionEvent.***ACTION\_UP***)  
 {  
 **command** = **"10"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch**(IOException e)  
 {  
 }  
 }  
 **return false**;  
 }  
 });  
 *//OnTouchListener code for the forward right button (button long press)* **forward\_right\_btn**.setOnTouchListener(**new** View.OnTouchListener(){  
 @Override  
 **public boolean** onTouch(View v, MotionEvent event)  
 {  
 **if**(event.getAction() == MotionEvent.***ACTION\_DOWN***)  
 {  
 **command** = **"4"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch** (IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **else if**(event.getAction() == MotionEvent.***ACTION\_UP***)  
 {  
 **command** = **"10"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch**(IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **return false**;  
 }  
 });  
 *//OnTouchListener code for the reverse left button (button long press)* **reverse\_left\_btn**.setOnTouchListener(**new** View.OnTouchListener(){  
 @Override  
 **public boolean** onTouch(View v, MotionEvent event)  
 {  
 **if**(event.getAction() == MotionEvent.***ACTION\_DOWN***)  
 {  
 **command** = **"5"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch** (IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **else if**(event.getAction() == MotionEvent.***ACTION\_UP***)  
 {  
 **command** = **"10"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch**(IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **return false**;  
 }  
 });  
 *//OnTouchListener code for the reverse right button (button long press)* **reverse\_right\_btn**.setOnTouchListener(**new** View.OnTouchListener(){  
 @Override  
 **public boolean** onTouch(View v, MotionEvent event)  
 {  
 **if**(event.getAction() == MotionEvent.***ACTION\_DOWN***)  
 {  
 **command** = **"6"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch** (IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **else if**(event.getAction() == MotionEvent.***ACTION\_UP***)  
 {  
 **command** = **"10"**;  
 **try** {  
 **outputStream**.write(**command**.getBytes());  
 }  
 **catch**(IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **return false**;  
 }  
 });  
 *//Button that connects the device to the bluetooth module when pressed* **bluetooth\_connect\_btn**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {

**if**(BTinit())  
 {  
 BTconnect();  
 }  
 }  
 });  
 }  
 *//Initializes bluetooth module* **public boolean** BTinit()  
 {  
 **boolean** found = **false**;

BluetoothAdapter bluetoothAdapter = BluetoothAdapter.*getDefaultAdapter*();  
  
 **if**(bluetoothAdapter == **null**) *//Checks if the device supports bluetooth* {  
 Toast.*makeText*(getApplicationContext(), **"Device doesn't support bluetooth"**, Toast.***LENGTH\_SHORT***).show();  
 }  
  
 **if**(!bluetoothAdapter.isEnabled()) *//Checks if bluetooth is enabled. If not, the program will ask permission from the user to enable it* {  
 Intent enableAdapter = **new** Intent(BluetoothAdapter.***ACTION\_REQUEST\_ENABLE***);  
 startActivityForResult(enableAdapter,0);  
 **try** {  
 Thread.*sleep*(1000);  
 }  
 **catch**(InterruptedException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 Set<BluetoothDevice> bondedDevices = bluetoothAdapter.getBondedDevices();  
 **if**(bondedDevices.isEmpty()) *//Checks for paired bluetooth devices* {  
 Toast.*makeText*(getApplicationContext(), **"Please pair the device first"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 **else** {  
 **for**(BluetoothDevice iterator : bondedDevices)  
 {  
 **if**(iterator.getAddress().equals(**DEVICE\_ADDRESS**))  
 {  
 **device** = iterator;  
 found = **true**;  
 **break**;  
 }  
 }  
 }  
 **return** found;  
 }  
 **public boolean** BTconnect()  
 {  
 **boolean** connected = **true**;  
 **try** {  
 **socket** = **device**.createRfcommSocketToServiceRecord(**PORT\_UUID**); *//Creates a socket to handle the outgoing connection* **socket**.connect();  
 Toast.*makeText*(getApplicationContext(),  
 **"Connection to bluetooth device successful"**, Toast.***LENGTH\_LONG***).show();  
 }  
 **catch**(IOException e)  
 {  
 e.printStackTrace();  
 connected = **false**;  
 }  
 **if**(connected)  
 {  
 **try** {  
 **outputStream** = **socket**.getOutputStream(); *//gets the output stream of the socket* }  
 **catch**(IOException e)  
 {  
 e.printStackTrace();  
 }  
 }  
 **return** connected;  
 }  
 @Override  
 **protected void** onStart()  
 {  
 **super**.onStart();  
 }  
}

**XML Code:**

**AndroidMainfest.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.philipgo.arduinobluetoothrccar"**>  
  
 <**uses-permission android:name = "android.permission.BLUETOOTH"** />  
  
 <**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme"**>  
 <**activity android:name=".MainActivity" android:screenOrientation="portrait"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
 </**application**>  
  
</**manifest**>

**XML Layout:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 tools:context="com.example.philipgo.arduinobluetoothrccar.MainActivity"**>  
  
 <**Button  
 android:id="@+id/forward\_btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginTop="74dp"  
 android:layout\_toStartOf="@+id/forward\_right\_btn"  
 android:layout\_toLeftOf="@+id/forward\_right\_btn"  
 android:text="Forward"** />  
  
 <**Button  
 android:id="@+id/forward\_left\_btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_below="@+id/forward\_btn"  
 android:layout\_toStartOf="@+id/forward\_btn"  
 android:layout\_toLeftOf="@+id/forward\_btn"  
 android:text="forward&#10;LEFT"** />  
  
 <**Button  
 android:id="@+id/forward\_right\_btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_above="@+id/reverse\_btn"  
 android:layout\_toEndOf="@+id/reverse\_btn"  
 android:layout\_toRightOf="@+id/reverse\_btn"  
 android:text="forward&#10;RIGHT"** />  
  
 <**Button  
 android:id="@+id/reverse\_btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_below="@+id/forward\_left\_btn"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="48dp"  
 android:text="REVERSE"** />  
  
 <**Button  
 android:id="@+id/reverse\_left\_btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_below="@+id/forward\_left\_btn"  
 android:layout\_alignStart="@+id/forward\_left\_btn"  
 android:layout\_alignLeft="@+id/forward\_left\_btn"  
 android:layout\_centerHorizontal="true"  
 android:text="Reverse&#10;Left"** />  
  
 <**Button  
 android:id="@+id/reverse\_right\_btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_below="@+id/forward\_right\_btn"  
 android:layout\_alignEnd="@+id/forward\_right\_btn"  
 android:layout\_alignRight="@+id/forward\_right\_btn"  
 android:layout\_toRightOf="@+id/reverse\_btn"  
 android:text="Reverse&#10;Right"** />  
  
 <**Button  
 android:id="@+id/bluetooth\_connect\_btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="50dp"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:text="Connect to Bluetooth Device"** />  
</**RelativeLayout**>

**Colour.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**resources**>  
 <**color name="colorPrimary"**>#3F51B5</**color**>  
 <**color name="colorPrimaryDark"**>#303F9F</**color**>  
 <**color name="colorAccent"**>#FF4081</**color**>  
</**resources**>

**Dimension.xml**

<**resources**>  
 *<!-- Default screen margins, per the Android Design guidelines. -->* <**dimen name="activity\_horizontal\_margin"**>16dp</**dimen**>  
 <**dimen name="activity\_vertical\_margin"**>16dp</**dimen**>  
</**resources**>

**String.xml**

<**resources**>  
 <**string name="app\_name"**>Arduino Bluetooth RC Car</**string**>  
</**resources**>

**Style.xml**

<**resources**>  
 *<!-- Base application theme. -->* <**style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar"**>  
 *<!-- Customize your theme here. -->* <**item name="colorPrimary"**>@color/colorPrimary</**item**>  
 <**item name="colorPrimaryDark"**>@color/colorPrimaryDark</**item**>  
 <**item name="colorAccent"**>@color/colorAccent</**item**>  
 </**style**>  
</**resources**>

**Dimension.xml**

<**resources**>  
 *<!-- Example customization of dimensions originally defined in res/values/dimens.xml(such as screen margins) for screens with more than 820dp of available width. This  
 would include 7" and 10" devices in landscape (~960dp and ~1280dp respectively). -->* <**dimen name="activity\_horizontal\_margin"**>64dp</**dimen**>  
</**resources**>