How To Technical Documentation

Sixth Sense Project
Multisensory Input Puck

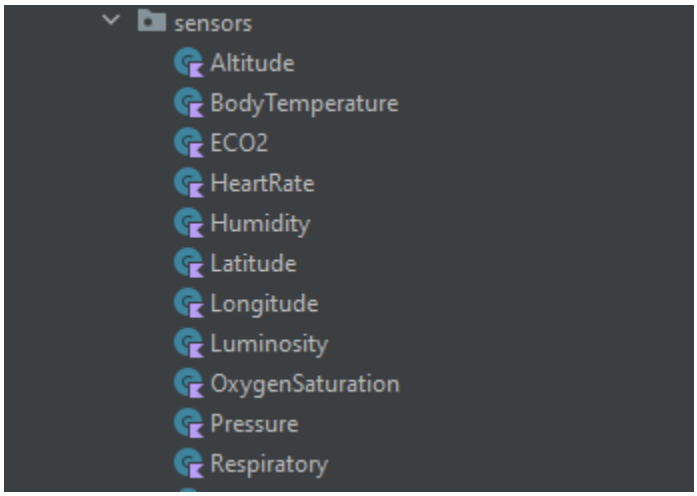
Client
Engineering Students

Created By
Albert Koyikalorthe Siby
Zeel Manvendrakumar Patel
Andrew Bourgeois
Kuang Xu
Parag Salgotra
Abdelrahman Amer
Shady Hussein

Table Of Contents

1.0 Adding new sensor: Backend

In order to add a new sensor to the application, you need to create a new kotlin class called with the name of the sensor inside of the package called “sensor”, like the ones found below



Sensors take 5 parameters, id, which is an int; name, which is a string; data; which is an int, warning, which is an int; and time, which is a LocalDateTime? (nullable). It needs to extend the class Sensor, which will give you the methods required for your new sensor object. An example of the declaration of the sensor “Altitude” can be found below

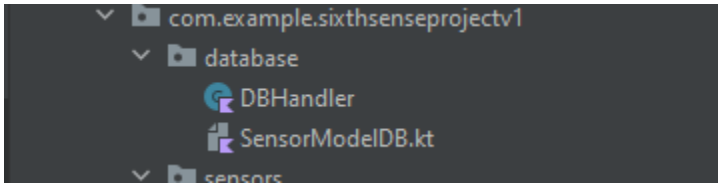
```
class Altitude(id: Int, name: String, data: Int, warning: Int, time: LocalDateTime?) : Sensors(id, name, data, warning, time) {

    //method to get the data of a sensor from the database
    @RequiresApi(Build.VERSION_CODES.O)
```

You will then need to adjust all 8 methods to match your new sensor. This mainly consists of changing the name of the used sensor to the name of the new sensor you are creating.
For example in the getFromDatabase method, in order to adjust it to a new sensor “xyz”,

```
override fun getFromDatabase(context: Context, personID: Int): String {
    var returnData = ""

    try{
        val dbHandler: DBHandler = DBHandler(context)
        var latestData: ArrayList<SensorModelDB> = ArrayList<SensorModelDB>()
        latestData = dbHandler.readLatestPerSensor( sensorName: "Altitude", personID)
        //If condition to return empty string if the table has size 0
        if (dbHandler.readLatestPerSensor( sensorName: "Altitude", personID).size == 0)
        {
            return returnData;
        }

        returnData = latestData.get(0).data + " km"
    }
    catch (e: Exception){
        //handle exception
    }

    return returnData
```

You would need to change the “Altitude” part into “xyz”.
Repeat this for all 8 methods, adjusting as needed.

Then, you would have to go to the dbHandler class, found within the database package



Inside the DBHandler class, you would need to add the following piece of code to the generateLatestData method.

```
else if(sensorName == "xyz"){
    tempQ = "SELECT * FROM $TABLE_NAME WHERE $NAME_COL = 'xyz' AND $USERID_COL = $tempUserID"
}
```

You would also need to add this line to the generateTenMinutesData method. This is so the SQLite queries are automatically generated for you, for a sensor called xyz

```
else if(sensorName == "xyz"){
    tempQ = "SELECT * FROM $TABLE_NAME WHERE $NAME_COL = 'xyz' AND $USERID_COL = $tempUserID" +
            " AND $TIME_COL > datetime('now', '-10 minutes')"
}
```

2.0 Adding new sensor: Frontend

Next, you would move onto the frontend of the application. You would need to move to the globalVariables.kt file, and find the part where the respiratory objects are created within the class, found below.

```
15    val allSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
16    val environmentSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
17    val healthSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
18    val vitalSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
19
20    val myTemperature = Temperature( id: 1,  name: "Temperature",  data: 0,  warning: 0, time: null)
21    val myHumidity = Humidity( id: 1,  name: "Humidity",  data: 0,  warning: 0, time: null)
22    val myPressure = Pressure( id: 1,  name: "Pressure",  data: 0,  warning: 0, time: null)
23    val myAltitude = Altitude ( id: 1,  name: "Altitude",  data: 0,  warning: 0, time: null)
24    val myTVOC = TVOC ( id: 1,  name: "TVOC",  data: 0,  warning: 0, time: null)
25    val myECO2 = ECO2( id: 1,  name: "ECO2",  data: 0,  warning: 0, time: null)
26    val myLuminosity = Luminosity( id: 1,  name: "Luminosity",  data: 0,  warning: 0, time: null)
27    val myLongitude = Longitude( id: 1,  name: "Longitude",  data: 0,  warning: 0, time: null)
28    val myLatitude = Latitude( id: 1,  name: "Latitude",  data: 0,  warning: 0, time: null)
29    val myRespiratory = Respiratory( id: 1,  name: "Respiratory",  data: 0,  warning: 0, time: null)
30
```

then, add this line of code to it

```
val myRespiratory = Respiratory( id: 1,  name: "Respiratory",  data: 0,  warning: 0, time: null)
val myXYZ = XYZ(1, "Respiratory", 0, 0,null)
```

All the sensors are saved into an arrayList called allSensorsArray. So you would need to modify the code and add a line allSensorsArray.add(myXYZ). Now your sensor has been added to the frontend as well

```
52    class GlobalVariable {
53        @RequiresApi(Build.VERSION_CODES.O)
54        fun addSensorsToArray() {
55            if (allSensorsArray.isEmpty()) {
56                allSensorsArray.add(myTemperature)
57                allSensorsArray.add(myHumidity)
58                allSensorsArray.add(myPressure)
59                allSensorsArray.add(myAltitude)
60                allSensorsArray.add(myTVOC)
61                allSensorsArray.add(myECO2)
62                allSensorsArray.add(myLuminosity)
63                allSensorsArray.add(myLongitude)
64                allSensorsArray.add(myLatitude)
65                allSensorsArray.add(myTime)
66                allSensorsArray.add(myHeartRate)
67                allSensorsArray.add(myOxygenSaturation)
68                allSensorsArray.add(myBodyTemperature)
69                allSensorsArray.add(myRespiratory)
70            }
```

3.0 Adding sensor to a category (For example vitals):

If you want to add the sensor to a specific category, for example vitals. You would need to scroll down a little further in the globalVariables.kt file, until you see this part, which should be right underneath the previous code snippet

```
if (vitalSensorsArray.isEmpty()) {
    vitalSensorsArray.add(myHeartRate)
    vitalSensorsArray.add(myOxygenSaturation)
    vitalSensorsArray.add(myBodyTemperature)
    vitalSensorsArray.add(myTemperature)
    vitalSensorsArray.add(myRespiratory)
}
```

In order to add sensor xyz to the vitals category, all you have to do is add the following line inside the if statement: vitalSensorsArray.add(myXYZ).
Now your new sensor has been added to the vitals array, good job!

4.0 Removing a sensor from a category (For example vitals):

In order to remove a sensor from a category, let's say vitals, all you would have to do is remove (or comment) out the sensor's addition to the array. For example, in the following code snippet we commented out the respiratory sensor, removing it from the vitals category

```
if (vitalSensorsArray.isEmpty()) {
    vitalSensorsArray.add(myHeartRate)
    vitalSensorsArray.add(myOxygenSaturation)
    vitalSensorsArray.add(myBodyTemperature)
    vitalSensorsArray.add(myTemperature)
//      vitalSensorsArray.add(myRespiratory)
}
```

5.0 Adding a new sensor category:

To add a new sensor category, you would need to first initialize an arrayList with the name of your sensor in the globalVariables.kt class. In the below code snippet we have added a new sensor category called Essentials:

```
15  val allSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
16  val environmentSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
17  val healthSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
18  val vitalSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
19  val essentialSensorsArray: ArrayList<Sensors> = arrayListOf<Sensors>()
20
```

Then, you would need to create an if statement to see if it's empty, if it is, then you need to add all the sensors you would like to add to the new category.
Scroll down to the bottom of the globalVariables.kt class, and add the following piece of code:

```
92          if (healthSensorsArray.isEmpty()) {
93              healthSensorsArray.add(myHeartRate)
94              healthSensorsArray.add(myOxygenSaturation)
95              healthSensorsArray.add(myBodyTemperature)
96          }
97          if (essentialSensorsArray.isEmpty()) {
98              essentialSensorsArray.add(myHeartRate)
99              essentialSensorsArray.add(myOxygenSaturation)
100             essentialSensorsArray.add(myBodyTemperature)
101         }
102     }
103 }
```

Ofcourse, you can add any sensor you would like to the new category. I used the Heart Rate, Oxygen Saturation and Body Temperature sensors only as examples.

6.0 Changing sensor from fake to real:

Remove sensor from fake sensor array in GlobalVariable.kt

```
if(fakeSensorsArray.isEmpty()){
    fakeSensorsArray.add(myRespiratory)
    fakeSensorsArray.add(myAccelerometer)
    fakeSensorsArray.add(mySystolic)
    fakeSensorsArray.add(myDiastolic)
}
```

Add a new indicator based on the indicator sent from the puck. Note: there cannot be two of the same indicators. Do the following in HomeActivity.kt replace the indicator with the correct one.

```
when (data[0]) {
    'G' -> {
        val trimString = data.substring( startIndex: 1).trim()
        val doubleString = trimString.toDouble()
        val intString = doubleString.toInt()
        val warning = myLongitude.checkWarnings(intString)
        dbHandler.addNewSensorData(id.toString(), myLongitude.getName(),data.substring( startIndex: 1),
            warning.toString(), LocalDateTime.now().toString())
    }
```

7.0 Changing sensor from real to fake:

Do the opposite of the above process. Add sensor to fake sensor array in GlobalVariable.kt
Remove indicator from the HomeActivity.kt

8.0 Adding a fake sensor:

Complete 1.0, 2.0 3.0 first. Add sensor to fake sensor array in GlobalVariable.kt

9.0 Change the Arudino:

Make sure the baud rate is correct in the HomeActivity.kt

```
if (mSerial != null) {
    mSerial!!.open()
    if (mSerial!!.open()) {
        mSerial!!.setBaudRate(115200)
        mSerial!!.setDataBits(UsbSerialInterface.DATA_BITS_8)
        mSerial!!.setStopBits(UsbSerialInterface.STOP_BITS_1)
        mSerial!!.setParity(UsbSerialInterface.PARITY_NONE)
        mSerial!!.setFlowControl(UsbSerialInterface.FLOW_CONTROL_OFF)
```

Change the Vendor ID in HomeActivity.kt

```
    //vendor id for testing, use product id 67
    val VENDORID: Int = 9025
    //vendor id they have, use product id 32779
    //val VENDORID: Int = 9114
    val BAUDRATE: Int = 115200
```

Change the Vendor ID and Device ID in device_filter.xml in res/xml.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <usb-device vendor-id="9025" product-id="67"/>
</resources>
```

10.0 Adding or Removing Teammates:

Add or Remove teammate name and id from the array in GlobalVariables.kt

```
val teammateName = arrayOf("Me", "Baker", "Carter", "Dean", "John", "Tim")
val teammateID = arrayOf(1, 2, 3, 4, 5, 6)
```

If adding a teammate, Add the name of the teammate along with a new pin assignment in returnPins() and returnPinsColor() methods in Teammates. Follow the existing procedure. Change only the name.