Making the database accessible to smartwatch required the collaboration of tools provided by android studio. The Firebase database web development application allowed us to wirelessly send and receive data in real-time to a database free of charge and have access to tools to analyze the flow of data and subsequently store it for later use. It also allowed us to maintain a record of accounts that have signed up with our application. The objective of the prototype was to have all three sensors send their data to the mobile application, which would write to the database. The user would then be able to see all the recorded values in the firebase in the history activity.

The process of creating the connection to firebase involved creating a structure called the DataItem activity, this activity structured the data in the following fields:

- itemId
- Temperature
- Heartbeat
- Steps Taken
- Timestamp

Which is structured as the following code below:

```
// Define a data class that would have the following items:
// This is a holder of the data that would be used for your app.
public class DataItem {
    private int itemId;
    private String Heartbeat;
    private String Temperature;
    private String StepsTaken;
    private String timestamp;
```

Each intended entry has a get and sets method all of which are as simple as returning the data.

```
public String getHeartbeat() {
    return Heartbeat;
}

public void setHeartbeat(String Heartbeat) {
    this.Heartbeat = Heartbeat;
}

public String getTemperature() {
    return Temperature;
}

public void setTemperature(String Temperature) {
    this.Temperature = Temperature;
}
```

Each of the sensor's data would be placed in the field that matches the intention of the sensor as well as has an item identifier (itemId) and a timestamp of when this data was written to the database this is the same date displayed when pulled as the history tab uses a listview to display the data. The history activity configuration involves solely using the listview to update using child event listeners to update data live these methods are set to display changes on data when added, changed, removed or reorganized.