COLLECTING ALL SENSOR DATA ON MOBILE DEVICES AND DEVELOP A FRAMEWORK

Burak BOZ, Bahattin KOÇ
Bilgisayar Mühendisliği Bölümü
Yıldız Teknik Üniversitesi, 34220 Istanbul, Türkiye
{l1118706, l1118704}@std.yildiz.edu.tr

***Özetçe* —Sensörler, analog dünyanın dijitalleştirilmesi ve akıllı sistemlerin tasarlanmasında oldukça önemli bir role sahiptir. Günümüzde hemen herkeste bulunan mobil cihazlarda birçok sensör bulunmaktadır. Bu verilerin toplanması ve işlenilebilir hale getirilmesi, bu projenin amacıdır. Android cihazlarda bulunan sensör verileri toplanmış, sıkıştırılarak bir sunucuya iletilmiş ve bu sunucuda kayıt altına alınmıştır. Geliştirilen java frameworkü ile kullanıcılara bu verilere ulaşılabilme ve istenilen şekilde işleme fırsatı sunulmuştur. Kullanıcılar bu frameworku kullanarak, sayısal veri işlemede önde gelen dillerden biri olan matlab ve yapay zeka projelerinde sıkça kullanılan python dilinde bu verilere ulaşabilmektedir.**

Anahtar Kelimeler—Android sensör verileri, Restful API, Huffman, Veri sıkışstırma, JAVA, MATLAB, PYTHON, MYSQL.

***Abstract*—Sensors play a very important role in digitizing the analog world and designing smart systems. Today, there are many sensors in mobile devices, which are found in almost everyone. Collecting this data and making it workable is the aim of this project. Sensor data on Android devices were collected, compressed, transmitted to a server and recorded on this server. With the developed java framework, users are provided with the opportunity to access this data and process it as desired. By using this framework, users can access these data in matlab, which is one of the leading languages in numerical data processing, and in python, which is frequently used in artificial intelligence projects.**

Keywords—Android sensor data, Restful API, Huffman, Data Compression, JAVA, MATLAB, PYTHON, MYSQL.

I. INTRODUCTION

Qualities and quantities existing in nature have been digitized thanks to sensors and have played an important role in making surprising technological discoveries and inventions. Integrating qualities such as seeing, hearing, smelling, and feeling with the skin into technological systems has led to the invention of many devices that make our lives easier. This proves the potential of applications that can be made with sensor data. In our project, there are an average of 10 sensors on mobile devices with android operating system, from which we collect sensor data[1].

With the data we collect in our project, it is possible to obtain various information about the location of the device. Information such as the speed of the device, the air temperature of the environment, and the precipitation can be accessed within the desired date range. This data allowed by the user is collected at frequencies per second and compressed with the frequencies determined by the user and transmitted to the server. In the sent data, there is also the device id produced by the android operating system of the mobile device transmitting the data. In this way, the information of which device produced which data on which date can be stored in the database.

Many systems working with Restful APIs have been examined and it has been observed that generally the data is transmitted as it is. In our project, it was foreseen that data could come from many users and therefore it was decided to send the data by compressing it. While compressing the data, a library based on the huffman algorithm[2] was used. In this way, it was observed that more than 50% gain was achieved in data transmission. Importance of data compression[3].

II. ANDROID APPLICATION

A. Determening the Sensors

The "Sensor" library [1] prepared by Google for the detection of sensors in the Android operating system has been examined. When this document is examined, it has been determined that not all sensors are available in all devices. There is no single hardware for every sensor. Some sensor data are derived by applying mathematical operations to the data received from certain hardware. For example, gravity sensor data is presented to the user by calculating the data received from the gyroscope sensor by applying a mathematical operation.

$$alpha = 0.8$$

gravity = alpha * gravity + (1-alpha) * gyroscope.event.values[2] (1)

The gravity sensor data is calculated as in equation (1). The data of the sensors that are equipped and return a single value are directly received.

The sensors suitable for use in the device are shown to the user with their data and the user is expected to select the sensors to which the data will be transmitted.

B. Preparing the Data to be Transmitted to the Server

The user selects the data to transmit and the transmission frequency. With the start of data transmission, the data of the selected sensors are collected once a second. The date of collection is added to each collected data. The unique ID of

the device is added to the sensor data and date information collected at the time of data transmission..

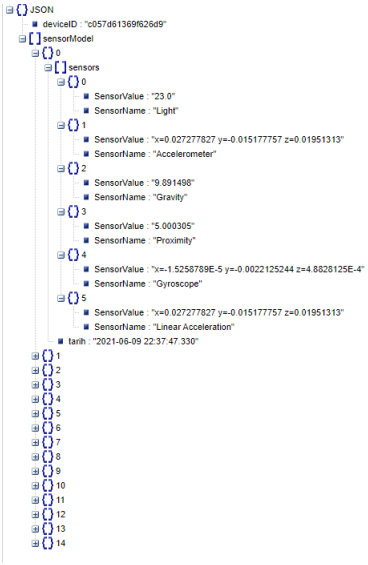

Figure 1 Converted Data to JSON

The data is translated into JSON [4] text as in figure 1. It is now ready to be compressed, as it is fully textual. When the data is carefully examined, the frequency of repetitive textual expressions in it stands out. This situation paves the way for the data to be compressed by an algorithm based on the huffman algorithm. For example, "SensorName" and "SensorValue" keys that represent data are repeated for each data. The Gzip library[5] is used to compress the data. This library converts the textual expression into a compressed byte array. After the data is compressed, it is sent to the server.

III. SERVER

A restful API has been prepared in order to store the sensor data on a continuously serving server. Java language was used while coding the server. This server is responsible for analyzing the incoming data, saving it to the database and ensuring that this data can be accessed at any time. While preparing the server, Jersey framework[6] was used. Tomcat v9.0[7] is used for the server to serve actively. The Mysql database that comes with the Xampp application[8] is used as the database.

IV. FRAMEWORK PREPARED TO ACCESS DATA

This framework was prepared in java language and made executable as a jar package. It offers three different methods to access data. For ease of use, when a new object is derived, a user guide has been added to the constructor method of the relevant object.

The user must first prepare the data group. When deriving the object, the device id, desired date range and server address should be entered. When the constructor method is run with this data, the server transmits the data. After the data group is ready, the user can access the data by choosing from three different methods. The first method is to reach the data of a certain sensor on a certain date. The second method retrieves data from all date points of a sensor in the data group. The last selection returns all sensor data at a particular date point. The user can access the data with these three selections.

V. MATLAB AND PYTHON

The prepared framework was tested on MATLAB and python and the results were successful.

```
>> isikSensorleri = nesne.getSensorsWithoutDate(dataModel,"Işık Sensörü")

isikSensorleri =

[[2021-06-08 19:49:14.0, 204.0], [2021-06-08 19:49:14.0, 204.0], [2021-06-08 19:49:14.0, 204.0]
```

Figure 2 All data of MATLAB Light Sensor

```
Main:  <com.sensors.jar.Main at 0x1bd92c7fdb0 jclass=com
DataModel:  <com.sensors.jar.DataModel at 0x1bd951825e0
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
Tarih:  2021-06-07 14:50:47.054  -> Veri:  9.913188
```

Figure 3 All Data of PYTHON Gravity Sensor

VI. RESEARCH RESULTS

In this study, the minimum operating system version for android devices that will collect sensor data is determined as android 8.0. As a result of our research, the minimum version that is still active today with hardware support and usability is android 8.0. Java and kotlin are the most commonly used languages when preparing an Android application. While determining the language to be used, the Java language, which has been supported for a long time due to its dominant position in the market, and has many libraries and resources, was chosen.

Server system planning is again based on the java language. While the system was being designed, the versions that could work in harmony were determined as follows:

- Jersey 2.31
- Tomcat 9.0
- Javax servlet 2.5
- Xampp 3.3.0
- Mysql Connector 5.1.39

VII. RESULT

In this study, a frontend application that collects sensor data from android 8.0 and above devices and compresses this data and sends it to the server, a RESTful API that saves this data to the database, and a framework that allows

access to this data from different software languages has been developed.

As a result of our work, we will offer some suggestions to those who want to develop a similar project or to contribute to our work further:

- There is only android application in our project. An application can also be developed for devices with IOS operating system. Having a server in the system provides support for every device connected to the internet. Different applications can be developed for different platforms.
- All data is stored as textual expression. Users who will use the framework should parse this data according to themselves. This situation can be prevented by adding extra functions.

REFERENCES

[1] Google. (2019) Developers.android. [Online]. Available: https://developer.android.com/guide/topics/sensors/sensors$_overview$

[2] geeksforgeeks.org. (2021) Huffman coding. [Online]. Available: https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/

[3] Google. (2009) Google search central. [Online]. Available: https://www.youtube.com/watch?v=Mjab$_aZsdxw$

[4] Json.org. (2009) Json. [Online]. Available: https://www.json.org/json-en.html

[5] Google. (2021) Developers.android. [Online]. Available: https://developer.android.com/reference/java/util/zip/GZIPOutputStream

[6] J. Brains. (2015) Developing restful apis with jax-rs. [Online]. Available: https://www.youtube.com/playlist?list=PLqq-6Pq4lTTZh5U8RbdXq0WaYvZBz2rbn

[7] tomcat.apache.org. (2018) Tomcat 9 software. [Online]. Available: https://tomcat.apache.org/download-90.cgi

[8] apachefriends.org. (2021) Xampp apache + mariadb + php + perl. [Online]. Available: https://tomcat.apache.org/download-90.cgi