11/5/2021

Research Manual

Balance Health

Name : Diarmuid Brennan
Student No : C00133947
Supervisor : Joseph Kehoe

Abstract

This document will discuss the research carried out in selecting the technological tools needed in developing the Balance Health project. The projects aim is to create an application that can be easily used by elderly people or patients recovering from illnesses such as stroke. The patient will carry out balance activities set by a medical personal while wearing a Movesense sensor device. Movesense is a wearable smart electronic device that can be easily worn by the user and used to monitor and transmit data collected through sensors over Bluetooth to a mobile device.

The medical personal can set balance activities for a patient and store them to a cloud database from a web application. The patient will be able download the activities to an application on their mobile device. The patient will then carry out the activities while wearing the Movesense device which will gather sensor data and transmit the data to the mobile application. The results of the activity are then stored to the cloud database from where the medical personal can retrieve the data for monitoring and analysis.

The software technologies used for developing the mobile and web application for this project will be determined from the research carried out and demonstrated within this document.

Table of Contents

Table of Figures

Introduction

The Internet of Things (IoT) is one of the fastest growing sectors in the technology industry. IoT are devices which are embedded with sensors, software and high processing capabilities which can connect with other devices and systems to transmit and exchange data. In recent years, technological advancements in embedded systems, wireless technology and data monitoring have brought about a revolution in smart devices.

Wearable technologies, also known as smart wearables, are one of the areas in which this growth can be seen most. Wearables are electronic devices which can be worn by the user as an accessory, such as smartwatches, rings, or wristbands. These devices contain numerous sensors used to gather large amounts of data from the wearer. Brands such as Fitbit, Apple watch or Google Glass are just a few of the products currently available changing the way in which we are receiving, monitoring, and sharing data through wearables.

Wearables are becoming increasingly used in healthcare to monitor elderly individuals medical and health conditions. They can provide continuous monitoring of the wearers movements and activities to gather data such as heart rate, body temperature, balance, posture, or strength. The data generated from these devices can be used for monitoring purposes or data collection for analysis and research.

The purpose of this project is to develop an application that can be used to monitor a patient's balance using a wearable Movesense sensor device. The patient will carry out activities set by medical personnel while wearing the device. The medical personnel will then be able to view the patient's data and monitor their progress from a web application.

The document will discuss wearable technologies, its growth in recent years and its increasing influence in the healthcare industry. It will describe the Movesense platform, its APIs, its hardware and the software components and its programming environments. The document will also discuss existing applications that have similar functionality to how this project will be developed. Lastly, it will discuss the research carried in selecting the frontend and backend technologies that will be used in developing the mobile and web applications to complete the project.

Wearable Technologies

Wearable technologies, also known as wearables, have gained extreme popularity in recent times and are one of the fastest growing consumer technology devices. Advancements in electronic chips, sensors, GPS, Wi-Fi, and Bluetooth technologies have transformed the way in which we can now access and monitor data in health and physical activities. Wearable technologies are one of the most important fields which have evolved from these continuous technological advancements (Tao, 2005).

Wearables applications can be divided into three main categories, wearable health technologies, wearable consumer electronics and wearable textile technologies. They are electronic devices that "are physically worn by individuals in order to track, analyse and transmit personal data" (BuiltIn, 2021) gathered through sensors embedded on the device (**Figure 1**).

Figure 1 Wearables (Source Trainerize, 2019)

Wright and Keith (2014) describe wearable technology and wearable devices as electronics that have been integrated into accessories that can be worn on the body by the user. The device can be incorporated into wearable items such as clothing, watches, jewellery, and glasses which can then communicate and transmit data to a wearable app, generally on a mobile device.

Raised awareness and interest in personal health issues and physical activity has led to an increased consumer demand for ways in which the consumer can track and manage their physical activities and health conditions. Wearables provide a cheap and efficient component that allows the consumer to achieve this.

Wearable Technologies Growth

Wearables are one of leading contributors to the growth in IoT devices. Apple watch and Fitbit are just two examples of existing wearable products available, but a variety of other gadgets are now being created to collect and record users' data in various industries from health to education to gaming. As shown in **Figure 2**, the wearable technology market has grown year on year and forecasts predict that the market is showing no signs of slowing down.

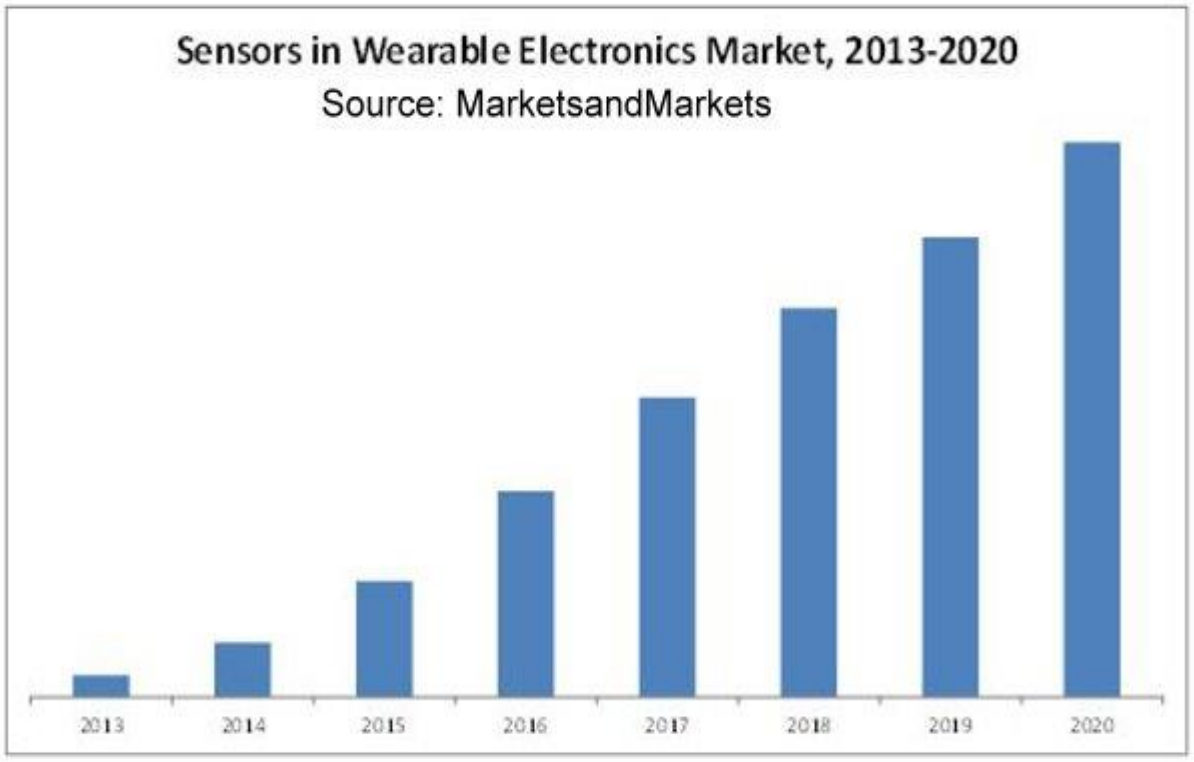

Figure 2 Sensors in Wearable Electronic Market, 2013-2020. Source (MarketsandMarkets, 2020)

As technological advancements continue to improve the way in which we can collect, monitor, and analyse our activities in different environments so too does the demand for wearable technologies. In 2019, the Wearable Technology Market was valued at approx. $28 billion and is expected to continue to grow to $74billion by 2025 (LinkedIn, 2021). **Figure 3** shows the expected growth of wearables over the coming years and the increase in growth in each wearable technology region.

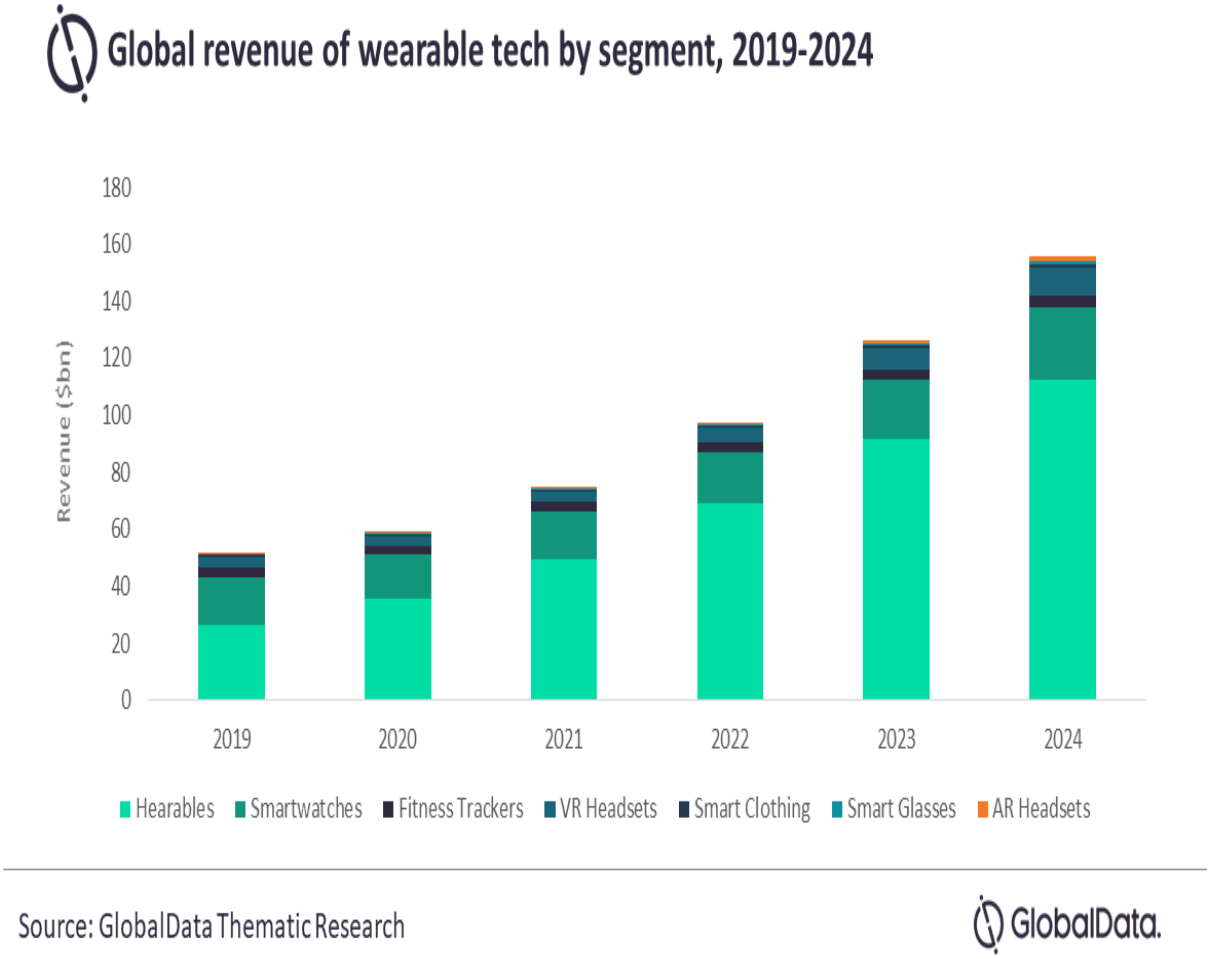

Figure 3 Global revenue of wearable tech by segment, 2019-2024. Source (GlobalData Thematic Research, 2019)

Wearable Technologies in Health

Advances in medical technologies and overall living standards has led to increased life expectancy for much of the global population. American citizens over 65 is projected to double by 2060 (Michigan State University, 2021). Our aging global population has increased demands on our healthcare systems and changed the way in which we must approach healthcare.

Wearables can ease the burden on our health-care personal by providing a means to monitor, record, analyse and transmit the user's data through the wearable device (Rutherford, 2010). Devices such as smartwatches or fitness trackers can be used to provide insights for healthcare providers to develop improved care experiences. Artificial Intelligence (AI) and Machine Learning (ML) algorithms can also be used to monitor and analyse the patient's data to predict chronic conditions such as heart disease or can be used to monitor a patient's gait to predict fall detection.

Patient's data can be gathered through the sensors and be transmitted to their mobile device while the patient is in their home settings (**Figure 4**). The data can be stored to a database or gathered and

sent to a doctor through a connected mobile device, in many cases removing the need for patients to meet with a doctor directly which can reduce some of the burden placed upon healthcare systems. Data analytics and ML carried out on the data to find patterns or trends can also be used as a preventive measure to detect medical issues early and make predictions on possible future medical conditions.

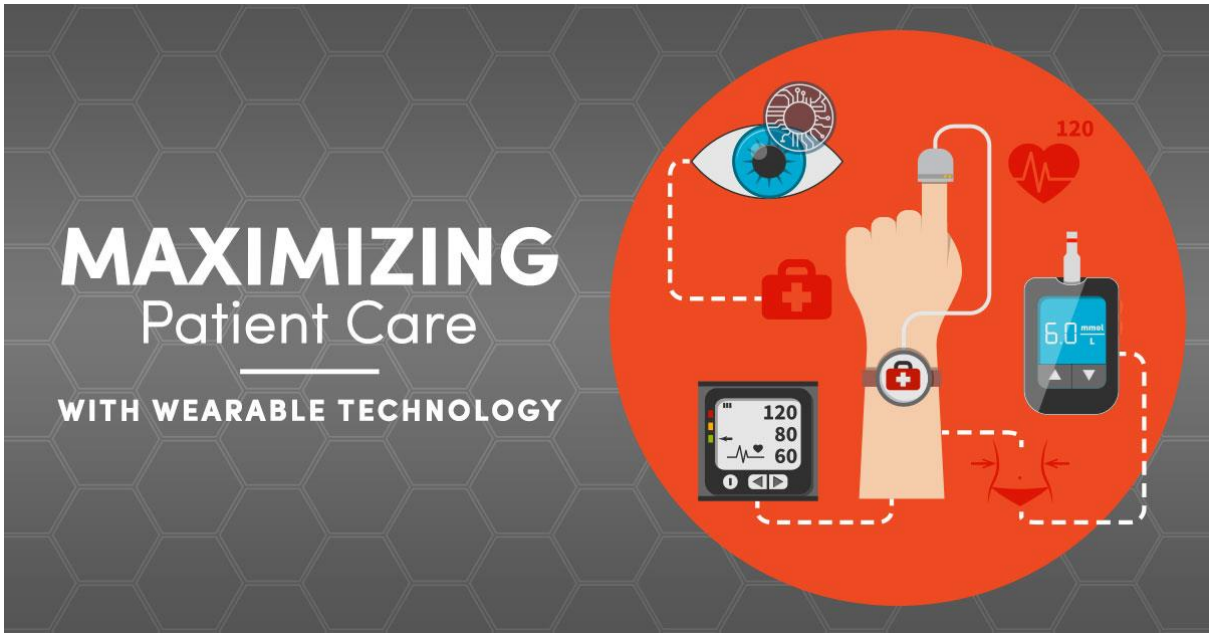

Figure 4 Wearable Health monitoring (Source: gate6, 2020)

Wearable technologies provide an innovative solution that will "enable cost-effective and affordable personalized mobile healthcare monitoring and management (PMHMM) at any place and time" (Vahist, Luong, 2018). They can be used to measure physical activity, blood pressure, blood glucose, body analysis, pulse rate, gait as well as many other characteristics.

Advancements in Bluetooth technologies, cloud computing, telemedicine capabilities and smart phone communication have enabled consumers to monitor and analyse their physical and health condition easier than ever before. Kevin Dang, Medical Data Analyst at GlobalData, states that "Mobile health is in a very exciting stage of development. Its applications span from mainstream daily usage with wearables to specific use cases in healthcare. The possibilities are endless and the potential for growth is astounding" (GlobalData, 2020).

Sensors

Sensors are devices that can detect input from events caused in its environment. This input can be due to changes in temperature, motion, pressure, or from several other environment attributes. The data gathered by the sensor can be stored, processed, displayed at the location of the sensor, or transmitted to a nearby device using Bluetooth or a wireless technology.

Sensors are used in many applications in medical, entertainment and security fields to provide "accurate and reliable information on people's activities and behaviours" (Mukhopadhyay, 2014). Advancements in technology and the development of microelectronics and other related technologies has led to vast improvements in the accuracy and reliability in which sensors can measure data.

Sensors embedded in devices such as a Movesense device are ultra-miniature, low cost, and provide high-precision and high reliability. There are various types of sensors which can be used to measure different parameters to monitor human biometric data. Below is a list of some of the more common sensor types:

- Electrocardiogram (ECG) – records electrical impulses sent from the heart. It measures the rate and regularity of our heartbeat and can be used to assess and diagnose cardiovascular diseases.
- 3-Axis Accelerometer – measures the linear accelerations and track the movements in what direction the wearer is moving.
- Gyroscope – tracks the angular rate and velocity of movements.
- Magnetometer – measures the magnetic field of the earth to track coordinates.
- Inertial measurement unit (IMU) - Combines accelerometer, gyroscope, and magnetometer to detect and measure the intensity of movements. It is used in pedometers, fall detection and many other areas.
- Heart Rate (HR) – HR sensor measure blood flow and calculates the heart rate.
- Temperature – records body temperature. This can be extremely useful in determining the physical condition of the wearer and variations in temperature can be used to detect medical symptoms seen in strokes or heart attacks.

Movesense

"Movesense is an open sensor development platform for sports, health, research, manufacturing, and more. It's easy and innovative tools allow you to build your own wearable device quickly and cost-efficiently. Movesense is maintained by the Finnish sports watch expert Sunnto" (Movesense, 2021). Sunnto have developed a sensor platform and open development environment which allows developers to create solutions to a range of motion sensing problems. It provides a set of developer tools, APIs, and a sensor device that a developer can use to easily create their own custom-made sensor application.

Movesense have "eliminated the need to invest in hardware design and manufacturing when building sports sensing solutions" (Movesense, 2021). The Movesense platform provides the hardware and developer tools needed to develop a wearable to track and monitor every type of activity. It enables all developers, regardless of their technical or programming skill level, to easily begin working on and creating their own solutions to create applications that can be used for monitoring physical activities during exercise or for tracking the health conditions of the wearer.

Figure 5 Movesense Sensor. Source (Movesense, 2021)

The Movesense sensor is a versatile, light, and small device with swim and shock proof construction suitable for all types of activities. The device is a lightweight, low powered wearable that provides an innovative tool capable of measuring any movement and can be used to track, analyse, and gain insights into any type of physical activity (**Figure 5**).

Movesense Hardware

The Movesense sensor is a small device which consists of multiple sensors contained within a black plastic water and shock resistant outer casing. This makes the device suitable for collecting data from a range of various activities. The device is powered by a CR 2025 lithium coin cell battery, which has extremely low energy consumption, that can provide months of operating time. It uses a very high performance 64MHz Nordic Semiconductor Micro Controller Unit (MCU) which is an advanced low powered system on chip. This handles the Movesense software platform and Bluetooth connectivity. All components of the device have been specifically chosen to enable low power usage and to achieve long battery life.

The central processing unit (CPU) memory on the chip includes 64kB of RAM, 512Kb of FLASH as well as 384kB of EEPROM. The device can store measurements on its local memory using the log data capabilities. This can reduce the need for requiring a constant BLE connection with the mobile device when monitoring an activity. Once the activity has been completed the data stored in the logs can be accessed and sent in a single request over the API.

The sensors contained within the Movesense device include a 9-axis motion sensor, consisting of Accelerometer, Gyroscope and Magnetometer, a Maxim ECG Analog Frontend, which can measure ECG, heartrate and RR-intervals and a temperature measurement sensor. All the functionality of the sensor, such as accessing sensor data, memory access, communication protocols and system features, are all carried out over wireless data transmission using the Movesense REST API Whiteboard.

Accessories

Several accessories can also be included along with the sensor which allow the sensor to be strapped to the wearer. This includes chest straps, wristbands and other connectors used to attach the device to textiles which can be worn on the body. Deciding which accessory is most suitable to use will generally depend on the activity being carried out. The Movesense Developer website provides information on the accessories available which can be reviewed by developers on the Movesense website.

Architecture

At a high level, Movesense is an open wearable tech platform which consists of a programmable sensor, a mobile SDK and a custom-made embedded REST communication framework called "Whiteboard" built specifically for Movesense devices. The Movesense device is developed and designed to work using state of the art low power circuitry. In general, data is gathered through the sensors mounted on the device which can communicate with a mobile device over the Bluetooth Low Energy (BLE) technology. The platform provides a connectivity stack for the sensor and for a mobile application. The connectivity stacks provide the framework which allows the devices to interoperate and communicate (**Figure 6**).

The development stacks allow for symmetric communication between the devices. The mobile device can access the services of the sensor using the Whiteboard API communication framework. The mobile device and sensor form a client-server connection and communicate over BLE. Whiteboard allows the devices to quickly and easily interact and communicate with each other and other devices.

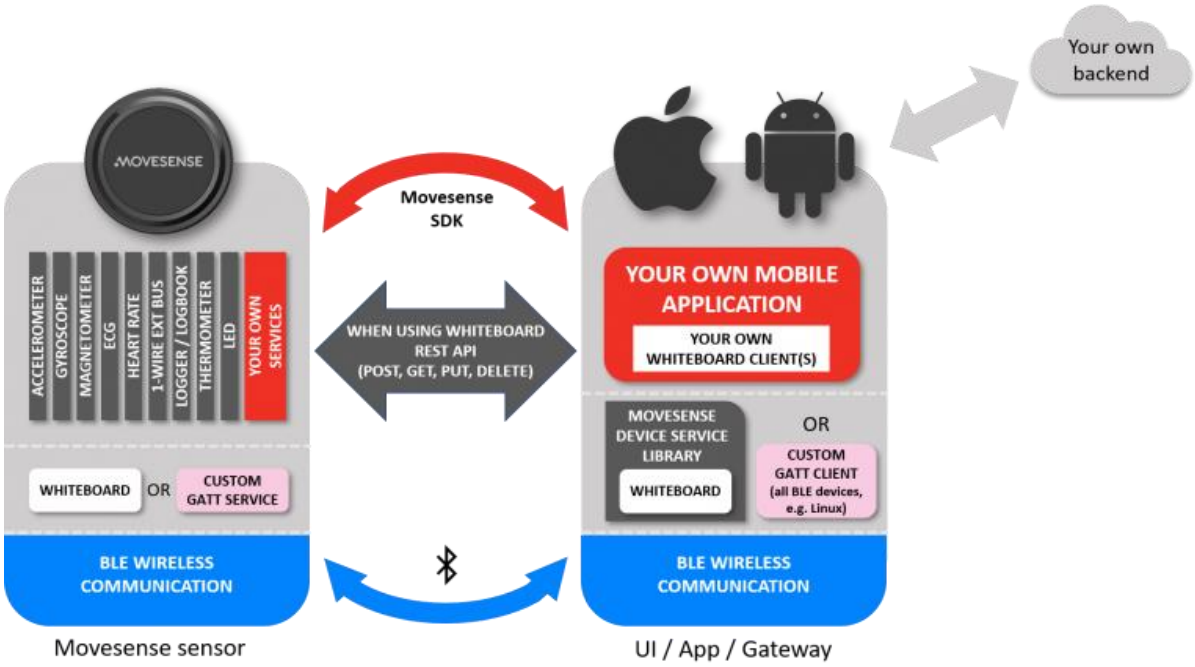

Figure 6 Movesense Architecture. Source (Movesense, 2021)

The architecture allows the developer the freedom and flexibility to easily create and design their own custom sensor solutions. Sample apps and libraries are provided on the Movesense platform in the developer resources. These demonstrate the libraries in use to the developer and allows the developer to begin creating their own applications. Support from the Movesense developer community and the Movesense team is provided to help new users throughout the process.

Movesense Software

The Movesense platform consists of a Sensor development software stack and a Mobile development software stack. The software stacks can be accessed through the Movesense community Bitbucket account from their Movesense-device-lib and Movesense-mobile-lib repositories respectively. The sensors mounted on the device collect the wearers data which can then be utilized by the software. The data captured is accessed through the Whiteboard architecture framework. The services on the device can be accessed over the Bluetooth connection from a mobile device, directly from the sensor itself using a programming jig, or through the creation of a custom-made GATT service. The software stacks provided for the sensor and mobile device handle the communication between the devices.

Sensor Software

Movesense provides the platform that allows developers to program the sensor themselves. To begin working with the sensor software, the developer must first install the sensor programming environment on their machine. This requires Docker Desktop to be installed for Windows or Mac, or for Linux to install docker using the distributions package manager. Docker is a containerized platform which provides a simple build environment, that can be used across differing operating systems, which automatically downloads the sensor programming environment.

The sensor programming environment contains the tools necessary to compile and build the firmware for the device already pre-packaged within the container. The sensor software is written in C/C+ programming language due to its low memory usage but with some limitations. There is no dynamic memory allocation, no STL and there is very limited amount of RAM and Flash memory.

Once the Docker environment has been installed, to begin developing in the sensor programming environment the developer must first pull down the Docker Movesense build environment container.

Note: All code snippets were taken from the Movesense documentation (Home - Movesense)

```
docker pull movesense/sensor-build-env:latest
```

They then clone the Movesense software from the Movesense-device-lab repo.

```
git clone git@bitbucket.org:suunto/movesense-device-lib.git
```

Once moved inside the cloned repository folder, the developer can open the docker image on the terminal in the build environment.

```
docker run -it --rm -v `pwd`:/movesense:delegated movesense/sensor-build-env:lates
t
```

This will open a docker prompt, inside which the developer can create a build directory. After moving into the build directory, the developer can now use the CMake build tool. The CMake command will generate the Movesense core libraries and ninja files from a provided sample application.

```
cmake -G Ninja -DMOVESENSE_CORE_LIBRARY=../MovesenseCoreLib/ -DCMAKE_TOOLCHAIN_FIL
E=../MovesenseCoreLib/toolchain/gcc-nrf52.cmake <sample_directory>
```

<sample_directory> here means the relative path to the CMakeLists.txt file of the application desired to be built. The Movesense build environment container contains various sample apps that the developer can initially use when creating their first build. They can then update the application and program it is using their own specific instructions. The files will be created within the newly created build directory. The final step is to run the ninja command.

```
ninja pkgs
```

The ninja command references the generated ninja files and contains how to build the software. It creates the Movesense_combined.hex files and the device firmware update (DFU) packages.

The DFU zip files can then be used to update the software on the sensor. This can be done using a programming jig, which can be connected from the sensor to a laptop and is used to flash the sensor from the jig. Alternatively, the sensor software can be updated from a mobile device over Bluetooth using the Movesense showcase app.

The Showcase app can be downloaded from the Movesense-mobile-lib repository. The library contains files to create the app for both Android and Apple devices. The Showcase app provides the developer with the functionality to connect to the sensor device over Bluetooth and upload the DFU files to the sensor. The app scans for nearby devices and pairs with any found (**Figure 7**).

The developer can then select the DFU zip files, which must be either downloaded to the mobile device or can be retrieved from a cloud storage location. A constant red light on the sensor will indicate that the device is in DFU mode, and the zip files can be uploaded. Once uploaded the sensor will updated with the build software.

Figure 7 Movesense Showcase app (Source: Movesense - DFU update)

When connecting to the Movesense sensor the initial update must be done using the DFU folder that contains the bootloader. This folder will be shown as Movesense_dfu_w_bootloader.zip. Any further updates to the sensor software can be done then using the smaller DFU packet, Movesense_dfu.zip.

Docker

"Docker is an open-source platform for building, deploying, and managing containerized applications" (IBM Cloud Education, 2021). It is a container virtualization technology which can be "used for building, distributing, and running applications in a portable, lightweight runtime and packaging too, known as Docker Engine" (Rad et al, 2017).

Docker depends on the Linux kernel and uses Linux's in-built capabilities of process isolation and virtualization to enable application components to share the resources of the host operating system (OS). This works in much the same as virtual machines (VM's) using a hypervisor. The Docker Engine is deployed on the host OS which creates the virtual container for hosting applications. Docker container technology has all the benefits of a virtual machine while eliminating the requirements of creating a guest OS for each VM and of creating a separate VM for each application (**Figure 8**). This makes docker containers more light weight, have greater resource efficiency and improved developer productivity (IBM Cloud Education, 2021).

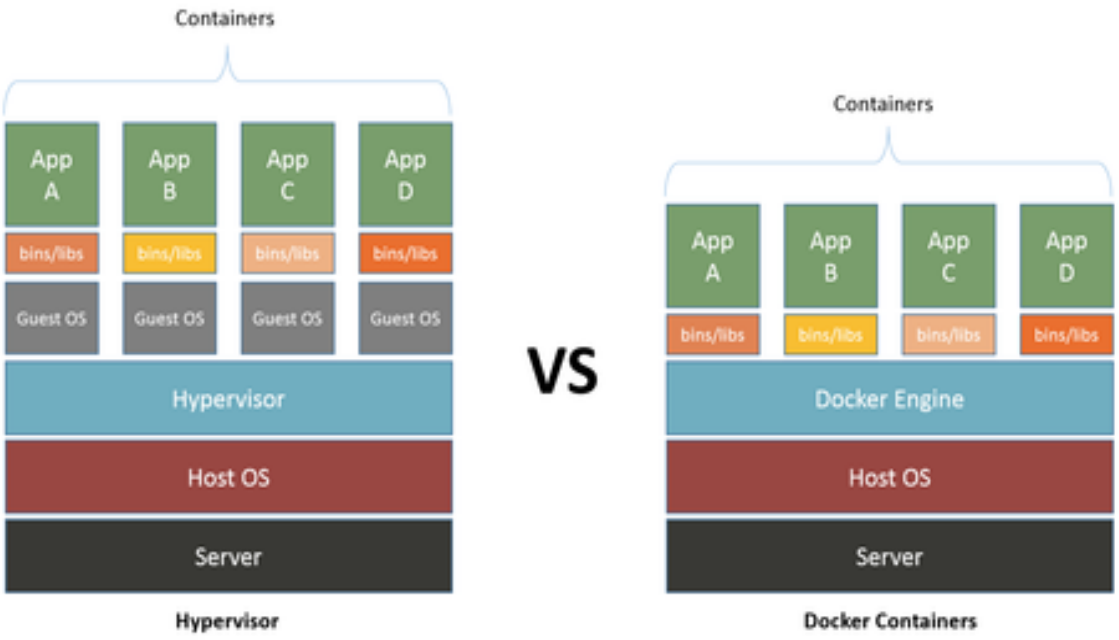

Figure 8 Hypervisor vs Docker container (Source: Network Computing, 2021)

The docker environment allows developers to package their applications into lightweight containers which can run on top of any operating system kernel by providing a layer of abstraction. Docker containers can develop applications to be portable across all types of environments, are more light-weight, scalable and reusable while providing better performance than their virtualization counterparts.

CMake

CMake is a cross platform, free and open-source software for managing application source code builds. "CMake is an open-source tool for managing the process of building programs for a certain number of programming languages, which is independent of the operating system and compiler" (Faust, 2016).

It provides a single definition for building a project which can then be translated into specific build definitions for various supported platforms. It achieves this by pairing with the platform specific build system and generating the build input for the specific platform. For example, on a Linux machine, CMake will generate a MakeFile, while on a Windows machine it will generate a Visual Studio project.

The CMakeLists.txt file is the main project file for CMake. It contains the build behaviour of the project and defines what the build system should do. In the Movesense sensor programming environment, CMakeLists.txt is what CMake uses to generate the Ninja build system files and defines the structure of the project.

Mobile Software

The Movesense platform provides documentation for connecting to the sensor from both Android and iOS devices. It provides sample applications that demonstrate how to connect to the sensor over Bluetooth as well as how to access the senor data and hardware information using the Whiteboard API. The Movesense Bitbucket account also provides support showing how to create apps using third party plugins for Unity3D, Xamarin and React Native.

To create compatible applications for mobile devices to connect to the sensor device, Movesense provides the Movesense Mobile Library (MDS). The MDS library contains the components and interfaces needed to communicate with the sensor over the Whiteboard communication framework. The MDS library allows the developer to create a Mds object which contains methods for communicating with the sensor and for retrieving the data gathered by the sensor using the API request calls provided.

The first step in setting up for mobile development is to clone the Movesense-Mobile-Lib repository from the Movesense Bitbucket account.

```
git clone git@bitbucket.org:suunto/movesense-mobile-lib.git
```

The repository contains subfolders for Android and iOS operating systems (**Figure 9**) which each contain their own specific OS versions of sample code for the Movesense Showcase app, the MDS library, and sample applications demonstrating accessing the sensor data from the mobile device (**Figure 10** (Android)).

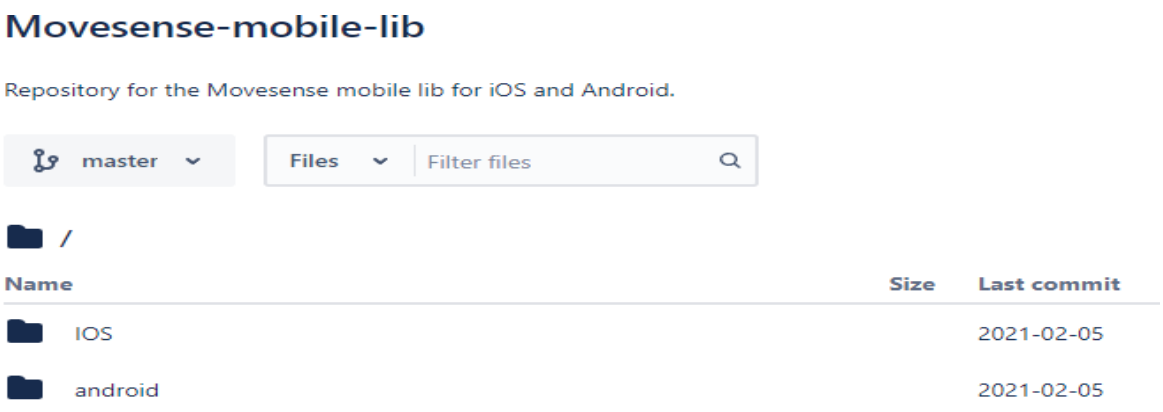

Figure 9 Movesense-mobile-lib repo

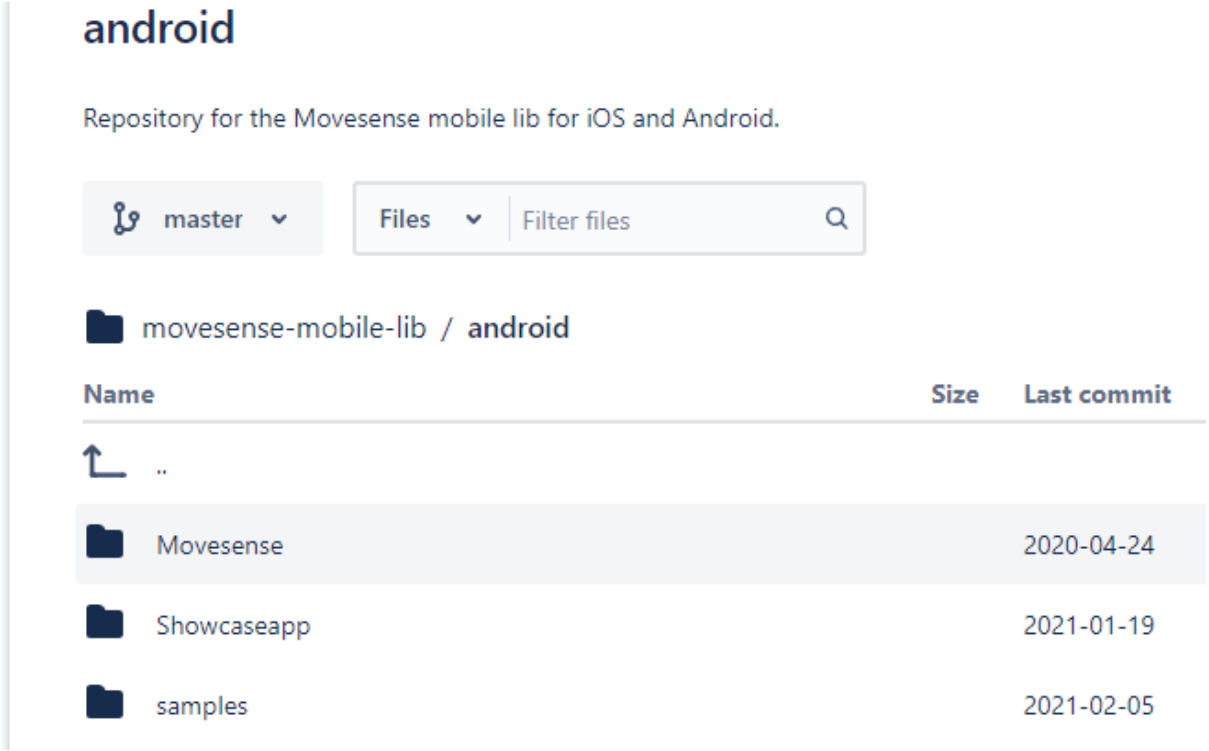

Figure 10 Android folders within Movesense-device-lib repo

To demonstrate connecting to the sensor device, this document will show how to develop for Android using Android studio. The MDS library is contained in the mdslib-1.44.0(1)-release.aar file within the Movesense subfolder. When creating an application, the library must be placed in the root folder of the project and added to the applications build.gradle dependencies (Figure **11**). The MDS library provides a simple interface which will allow the application to communicate with the Movesense device using the Whiteboard communication framework.

```
allprojects {
    repositories {
        jcenter()
        maven {
            url "https://maven.google.com"
        }
        flatDir {
            dirs './','../../../Movesense/' // Folders to look for jar & aar libraries
            dirs './','../Movesense/' // Folders to look for jar & aar libraries
        }
    }
}

dependencies {

    implementation fileTree(dir: 'libs', include: ['*.jar'])

    implementation(name: 'mdslib', version: '1.44.0(1)-release', ext: 'aar')

    implementation "com.polidea.rxandroidble2:rxandroidble:1.13.0"
    implementation 'androidx.appcompat:appcompat:1.3.1'
    implementation 'com.google.android.material:material:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.1'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'

}
```

Figure 11 Project applications build dependencies

Once the MDS library is installed, the library can be imported to the application and a Mds object can be created.

```
import com.movesense.mds.Mds;

Mds mds = Mds.builder().build(context);
```

The Mds object has two main types of methods, MDS Connectivity API and MDS REST API. Both API's use asynchronous requests and results are returned to the mobile device through call-back interfaces. The MDS Connectivity API provides the methods for connecting with and disconnecting from the Movesense device over Bluetooth. The MDS REST API then provides the methods for accessing the sensor data.

```
void get(String uri, String contract, MdsResponseListener callback);

void put(@NonNull String uri, String contract, MdsResponseListener callback);
```

```
void post(@NonNull String uri, String contract, MdsResponseListener callback);

void delete(@NonNull String uri, String contract, MdsResponseListener callback);

void MdsSubscription subscribe(@NonNull String uri, String contract, MdsNotificati
onListener listener);
```

GET, PUT, POST, and DELETE methods are based on the REST architecture and their behaviours are carried out the same as with REST. Requests are sent in JSON format to the sensor and the response is returned through the call-back interface as a JSON string to the MdsResonseListener interface. The SUBSCRIBE () method is then particular to Movesense and can be used to subscribe to receive notifications from specific events that are captured on the sensor. An example of this would be the developer can create a subscribe request that will receive continuous samples of accelerometer data every second from the device until the UNSUBSCRIBE () method is called to end the request.

Once the library dependency has been set up on the Android project and a Mds object can be created, the application must connect with the device over BLE. To connect to the sensor the Mds will need the mac address of the device. This can be retrieved by carrying out a BLE Scan and opening a connection with the sensor and providing the location permission of the client device. Android provides libraries such as RxAndroidBle, which will scan for and connect to devices over BLE. Once these steps have been set up the developer is ready to start creating their application.

Whiteboard

Whiteboard is Movesense's embedded REST API protocol which provides the common REST services. Whiteboard was developed specifically to run on top of low resources. It handles the data communication between the Movesense sensor and the mobile device. It is an asynchronous communication library that uses a client-server design pattern. This allows the client and server components to evolve independently of each other.

Whiteboard is a simplified version of REST which can be ran more efficiently on the embedded devices. The Whiteboard REST framework provides the services, clients, timers, threading support and external communication between devices.

Like REST, the API consists of four main request types; GET, PUT, POST and DELETE. Whiteboard also has a SUBSCRIBE request type. This can used for the continuous transfer of data from a service provided by the sensor to a client. The results of the request sent are returned through call-backs sent in JSON string format to the client. Requests can be carried asynchronously using the Whiteboard library, either internally on the device or externally between devices over BLE.

REST (REpresentational State Transfer)

REST is an architectural style (**Figure 12**) that is used for providing standards for and developing web services. RESTful systems can be characterized as being stateless, for separating the client and server concerns and communicating over HTTP (Hypertext Transfer Protocol) (Khan and Abbasi, 2015).

Separation of client and server allows that implementation of code on the client or server side can be carried out without impacting the operation of the other. This separation allows for each component to evolve independently. Communication between the client and server is carried out over a REST interface using a predetermined data exchange format such as JSON or XML.

REST systems are stateless. This means that the client and server know nothing about what state the other is in. The client is responsible for storing and handling the session information. Each request from the client must contain all the necessary data required and session information within the request for the server to understand and respond.

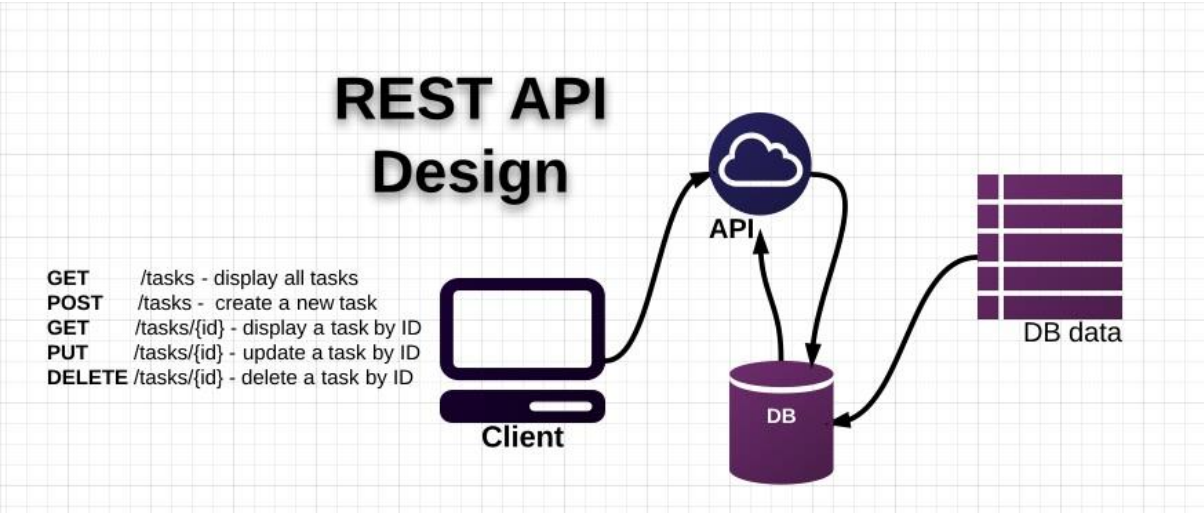

Figure 12 REST Architecture (Source: maxoffsky.com)

The client can send requests to retrieve data from the services provided at the server and the server sends a response back to the client. The four basic HTTP requests used in REST are GET, POST, PUT, and DELETE.

- GET – retrieves a specific resource from a service
- POST – used to create a new resource
- PUT – used to update a resource detail
- DELETE – used to remove a resource

The request also contains the path to the resource that the request is made to. To request data from the sensor device the Movesense API is referenced using the pathways described in the following section.

REST is very popular amongst IoT developers and wearables. The compact formats of RESTful APIs and JSON make them extremely suitable for these kinds of technologies (Zhang et al, 2010).

Movesense REST API

The Movesense REST API can be separated into several different sections which can be used to retrieve sensor device related data. The APIs are split into measurement related, system related and other API categories. The API categories provided are shown in **Figure 13** and describe what data is returned in the call-backs from the request made.

- */Meas* for sensor data (Acc, Gyro, Magn, Temp, HR, ECG)
- */Mem* for data memory access: DataLogger & Logbook
- */Comm* for communication protocols: BLE, 1Wire
- */Component* for low level features: LED, EEPROM, chip specific features
- */System for s*ystem features: Mode, Settings, Energy, Memory, States
- */UI* for user interface (LED blinks)
- */Misc* for the ones that do not fit in the above

and

- */Whiteboard* for Whiteboards own services

Figure 13 Movesense API's (Source: Movesense Developer Workshop, 2021)

Each API service contains several request calls that it provides to the client. For example, the measurement API service, /Meas, can be used to retrieve data from the sensors mounted on the Movesense sensor. The available API measurements can be viewed in **Table 1**.

Table 1. Movesense measurement API's.

Resource	Description
/Meas/Acc	API to gather acceleration data from the accelerometer
/Meas/Gyro	API to gather angular velocity data from the gyroscope
/Meas/Magn	API to gather magnetic field data from the magnetometer
/Meas/Imu	API to gather motion data from the accelerometer, gyroscope, and magnetometer
/Meas/ECG	API to gather electrocardiography data
/Meas/HR	API to gather heart rate data
/Meas/Temp	API to gather temperature data

Other typical API resources are used for system, communication, or memory requests. **Table 2** demonstrates a few of the more commonly used services.

Table 2. Movesense device common API services.

Resource	Description
/System/Debug	API for subscribing messages from device
/System/Energy	API for reading the battery state
/System/Memory	API for reading memory state

/Comm/Ble	API for managing BLE
/Mem/DataLogger	Generic logger capable of logging max. 8 different resources
/Mem/Logbook	Generic Logbook from where the logged data can be read
/Meas/Temp	API to gather temperature data
/Time	API for accessing different time related services

YAML (YAML Ain't Markup Language)

The API requests are defined using Swagger 2.0 notation, which uses the file format YAML. The API definition YAML files can be found in the Movesense-device-lib folder: movesense-device-lib/MovesenseCoreLib/resources/movesense-api in the Movesense-device-lib repository.

"YAML Ain't Markup Language" (abbreviated YAML) is a data serialization language designed to be human-friendly and work well with modern programming languages for common everyday tasks" (Evans et al, 2001). YAML is often written for configuration files that is human readable and easy to understand. **Figure 14** demonstrates an example of the Movesense accelerometer measurement API.

```
paths:

  /Meas/Acc/Info:
    get:
      description: |
        Get supported sample rates and ranges.
      responses:
        200:
          description: Returns accelerometer information.
          schema:
            $ref: '#/definitions/AccInfo'

  /Meas/Acc/Config:
    get:
      description: |
        Get current linear acceleration measurement configuration.
      responses:
        200:
          description: Returns accelerometer configurations.
          schema:
            $ref: '#/definitions/AccConfig'
    put:
      description: |
        Set linear acceleration measurement configuration.
      parameters:
        - name: config
          in: body
          description: New configurations for the accelerometer.
          required: true
          schema:
            $ref: '#/definitions/AccConfig'
      responses:
        200:
          description: Operation completed successfully
        503:
          description: |
            Not allowed to change configuration at the moment. E.g. when
```

Figure 14 Accelerometer measurement API YAMLfile (Source: Movesense, 2021)

Each Movesense service API follows the same structure pattern containing tags for info, paths, parameters, and definitions used to describe the functionality of the API as well the paths to the different types of services which can be accessed through the API. As can be seen in **Figure 14 above,** the /Meas/Acc/Info path contains only a GET request which returns the description of what is returned from the request, "Get supported sample rates and ranges information". It then shows the /Meas/Acc/Config path which contains a GET and a PUT request. The GET request returns the accelerometer measurement configuration data while the PUT request can be used to set the configuration value of the acceleration request.

Bluetooth Low Energy (BLE)

Bluetooth Low Energy (BLE) is a radio standard which was developed as part of the Bluetooth 4.0 Core Specification that has been "specifically optimized for low cost, low bandwidth, low power, and low complexity" (O'Reilly, 2020). BLE is wireless communication interface, which is used by many PC's, mobile devices, smart watches, medical devices, and many others. It provides a wireless protocol (**Figure 15**) solution which can run sensors for an extended time using just a coin battery.

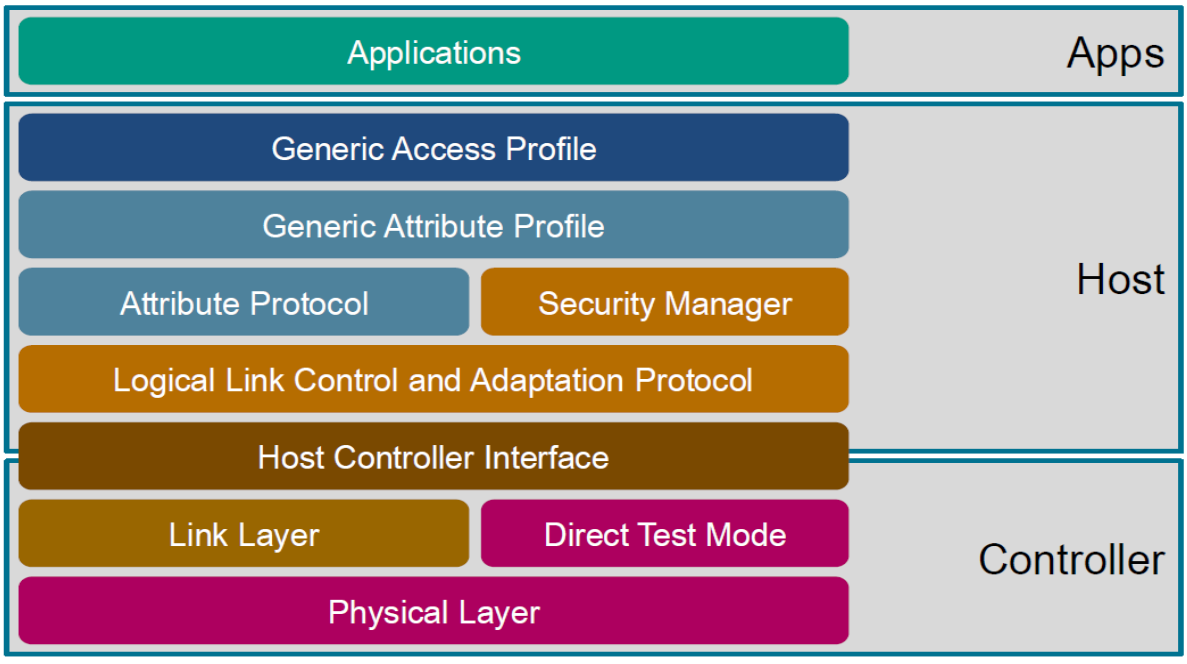

Figure 15 BLE protocol (Source: lpccs-docs.dialog-semiconductor.com)

"Bluetooth Low Energy (BLE) is an emerging low-power wireless technology developed for short-range control and monitoring applications" (Gomez et al, 2012). With continued growth in small electronic devices and on the Internet of Things (IOT), the widespread use of Bluetooth technology BLE is expected to be used in billions of devices worldwide. Due to its low power consumption and performance, it has widely been used for "activity monitors and heart rate sensors to be used to monitor a user's health and fitness levels" (Helge, 2010).

Market Analysis

Ainone Balance

Ainone is a Finnish software development company that creates mobile platform solutions for measuring and analysing human activities using sensor technology and machine learning algorithms. It has developed a unique approach to measuring human performance in health and sporting activities to provide solutions in rehabilitation and coaching environments.

Ainone Balance Software gathers a user's balance measurements. Balance measurements can be used to gain insights into a person's wellbeing during medical, rehabilitation or sporting activities. Balance prevents the body from swaying and involves the central nervous system and sensory feedback to stabilize the body. As balance evaluation is key to our central nervous system, measuring balance can inform us of important neurological features (**Figure 16**).

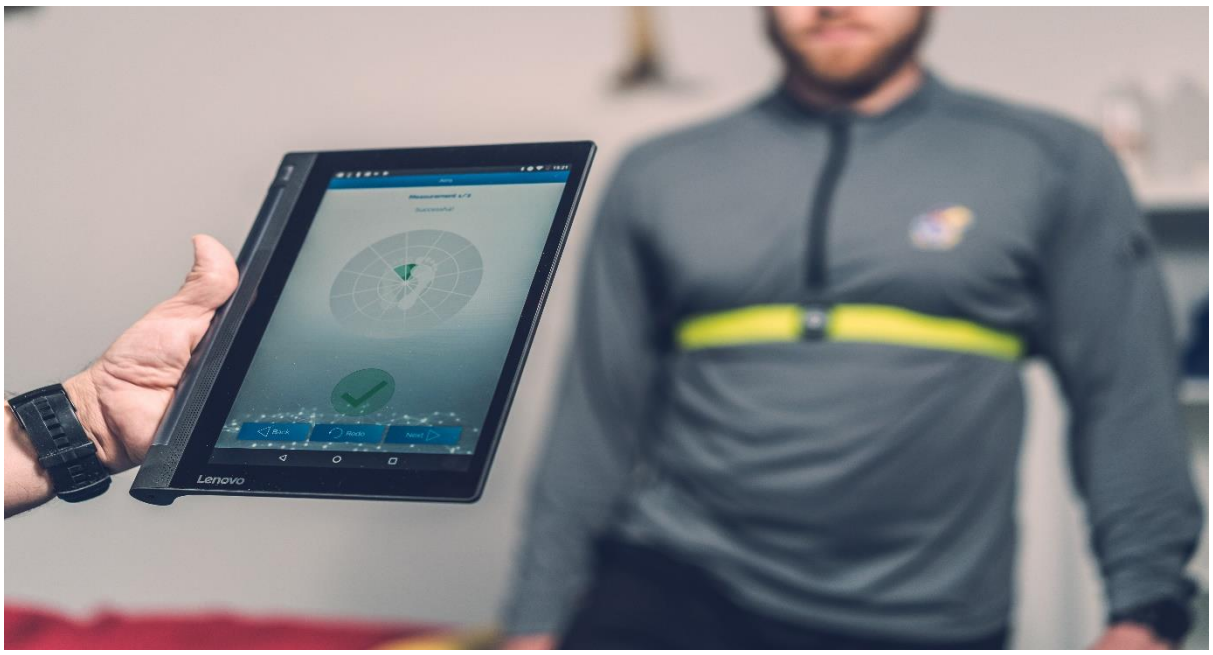

Figure 16 Ainone Balance App (Source: Ainone, 2020)

When analysed, balance measurements can be key indicators in the early diagnosis of neurological diseases such as Parkinson's disease or Alzheimer's. The tests carried out to evaluate the user's balance involves a series of different postures and stance positions (**Figure 17**). The results can then be applied to machine learning algorithms to provide insights to the user's performance and condition. The user's "balance can be determined as a set of movements or exercises while wearing the sensor, which captures precise measurements previously unavailable to healthcare professionals or coaches. With the data, the performance can be monitored more accurately and objectively" (Ainone, 2020).

Figure 17 Balance posture (Source: Ainone, 2020)

ALMA.care

ALMA.care is a Belgian Digital Health tech firm that aims to improve healthcare using sensor technology. ALMA.care collects long term data from the user from foot and wrist worn sensors to monitor data in health assessment and fall risk monitoring (**Figure 18**). The data is stored and passed to proprietary algorithms to be analysed and can provide timely insights to detect falls and tackle any medical issues found early on. “Prevention is the key to healthy ageing. ALMA.care brings you the tools for that” (ALMA care, 2021).

Figure 18 ALMA.care Application (Source: ALMA.care, 2021)

By monitoring how am elderly person walks using the sensors, any changes in their movement can be detected and identified early to help determine the wearers risk of falling. This can lead to prevention of falls and any complications that can stem from falls. Gait analysis can detect if the user has shortened their stride or views any changes in the users regular walking pattern. Alma's data analytics solutions are carried out on the gathered data to find any patterns which can provide insights into the patient that can help make future predictions from the data.

Fitbit

Fitbit products are smart technologies that use advanced sensors designed to track and monitor the wearers health and physical activities. Fitbit smart watches and activity trackers are used to track a range of physical attributes including oxygen saturation, steps taken, skin temperature, breathing rate and heart rate. The data collected on the device can be displayed on the devices interface or the results can be viewed on the Fitbit app (**Figure 19**).

Figure 19 Fitbit Mobile App (Source: Fitbit.com)

The Fitbit app not only gathers data from the Fitbit device, but it also provides functionality to the user that allows them to track their activities, record workouts, log food diaries, monitor heart rate, hydration and sleep, and allows the user to set and manage targets and goals on the app. Fitbit provides an all-round fitness monitoring device that can "help you keep a pulse on not only your fitness levels, but your stress, heart health, sleep, and overall wellbeing" (Fitbit, 2021).

Mobile Application

Operating Systems

To gather and track data from the sensor device, a mobile application will be developed to retrieve the data and display it to the user. In deciding on what mobile platform to develop the application for, research was carried on Android, iOS, and Xamarin developer platforms. The research will discuss each platform and the reason behind choosing which development platform to use for this project.

Android

Android is an open platform operating system owned by Google that allows third party developers to modify and contribute to the source code. It can run on various smart devices such as phones, watches, tablets and more. The Android OS is made of a layered stack of software that provides the frameworks and platforms to mobile applications that will allow the application to run on a multitude of Android enabled devices. **Figure 20** below shows the separate layers contained within an Android OS.

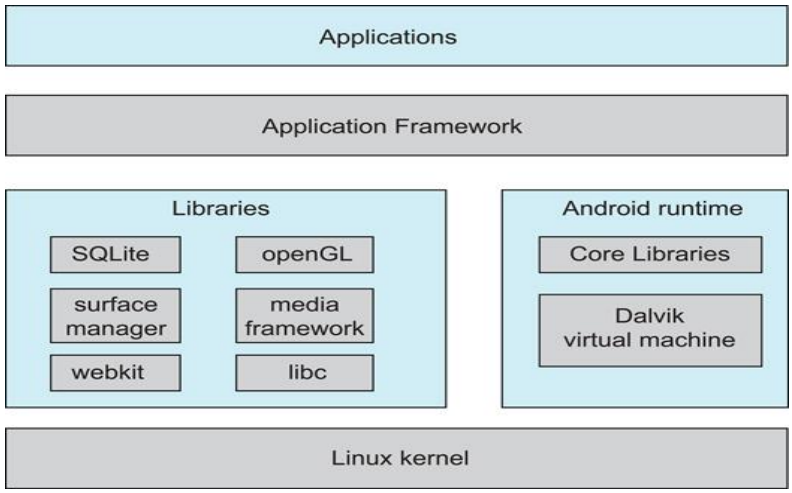

Figure 20 Android OS Software Stack (Source: Operating System Concepts)

Android applications are usually developed using the Java language but do not use the standard Java API, instead using the Android API developed specifically for mobile applications. The applications interact with the application framework which contains the Android API and the Android Runtime (ART), the environment designed for Android to optimize mobile devices which have limited memory and processing capabilities. Java programs are compiled to Java bytecode and then into an executable file by the ART which performs ahead-of-time (AOT) compilation. AOT compilation means that executable files are compiled to native code when first installed on the device, from where they can be executed by the ART. AOT compilation allows more efficient application execution and reduced power consumption, critical features for mobile development. The Android OS also contains a set of

native libraries that can accessed through the API which include frameworks for developing web browsers (webkit), database support (SQLite), network support such as secure sockets (SSLs) and many others. To allow Android to run on multiple different types of hardware, Android includes a hardware abstraction layer (HAL). This provides applications with a consistent view across all devices, independent of the hardware they are currently executing on. This allows applications to be portable across a multitude of different hardware platforms. Android then uses the Bionic standard C library which has a smaller memory footprint than the GNU C library, again developed specifically for Android and mobile devices which have slower CPU's. This is all built on top of the Linux kernel at the bottom of the software stack which communicates with the device's hardware.

Android is one the most widely used Mobile Operating systems with over 80 percent of the market share. Due to its large market share, its accessibility, portability and its easy build, test and evaluate environment, Android is often a popular choice when it comes to mobile development.

iOS

iOS is the proprietary mobile operating system developed by Apple specifically for Apple devices which supports Objective-C or Swift programming languages based on the Macintosh OS X. As shown in **Figure 21**, iOS uses a layered architecture where iOS acts as an intermediary layer between the application and the hardware allowing it to work on various iOS devices that have different hardware capabilities.

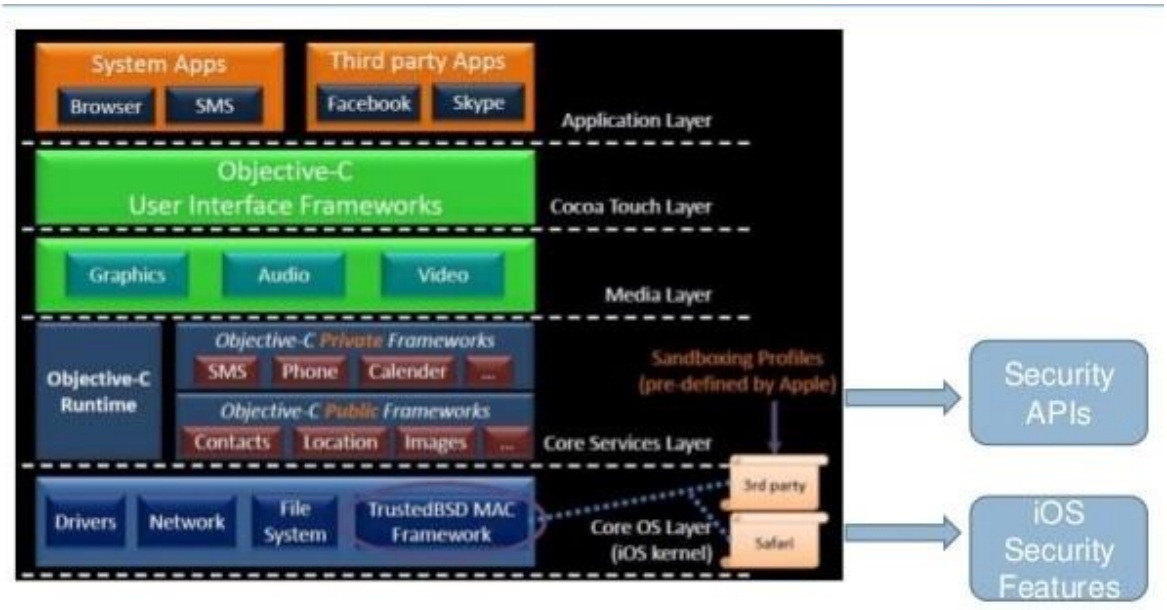

Figure 21 iOS Software Stack (Source: Ahmed-Reza Sadeghi et al)

iOS mobile development is the second most popular mobile development platform. Developing for iOS is faster, cheaper, and easier than for Android which can take up to 40 percent more time than using iOS. This is due Apple's programming language Swift. Swift allows the developer to easily to write, develop, and execute code. Apples Cocoa Touch framework and other interface components

have been developed specifically for iOS for a small selection of standardized devices and OSes. When compared with Android, which is available to a multitude of devices and OSes, standardisation for interfaces and components is a lot more difficult to apply and which follows less strict standards. This means that iOS can be developed in quicker timeframes as development is aimed at a much smaller set of standardized devices.

Xamarin

Xamarin is an open-source development platform created by Microsoft. Xamarin can be used to develop cross platform mobile applications as it can be used to create native applications for Android and iOS using Microsoft's .NET framework and C# programming language. Android and iOS applications created using a single Xamarin codebase have complete access to their own native APIs. Xamarin forms provides an abstraction layer between the native operating systems and the .NET platform that creates a shared UI codebase (**Figure 22**).

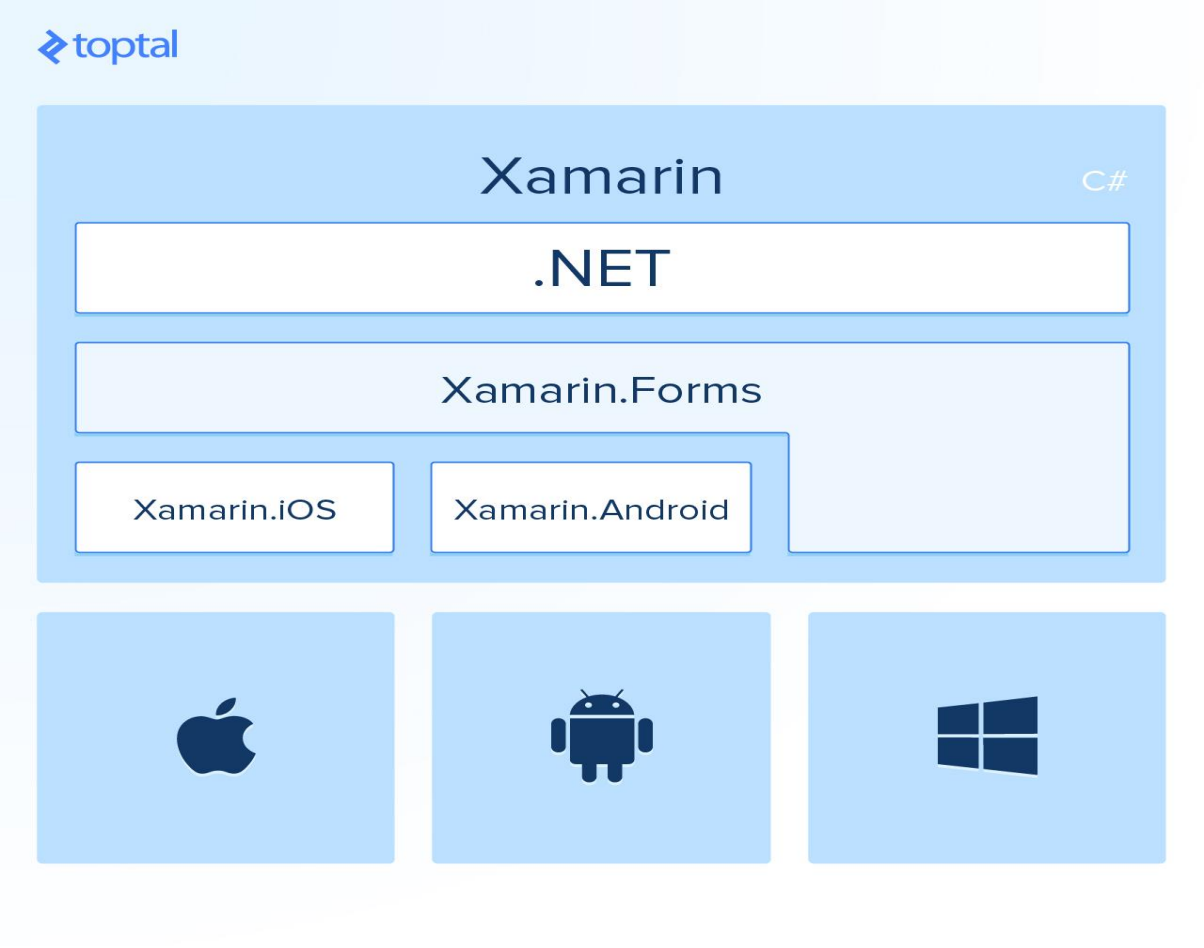

Figure 22 Xamarin Architecture (Source: http://tekhinnovation.blogspot.com/2017/06/building-cross-platform-apps-with.htm)

Xamarin allows the developer to create cross platform, yet native applications. It provides the flexibility to create for both iOS and Android from a single technological stack without compromising any of the aspects provided by the native platforms. Xamarin's shareable codebase can save time when developing for mobile applications and the .NET framework and C# programming language make development, testing and deployment more efficient and effective.

Conclusion

Android and iOS have nearly complete market share for mobile applications. Xamarin provides a cross platform on which an application can be developed for multiple platforms using the one code base making it a prime candidate for developing applications that will target much of the population.

Connecting and communicating with the Movesense device is an integral part of this project and is an unknown area to this developer. While the Movesense platform provides a third-party plugin that demonstrates connecting to the device using Xamarin, the documentation and support are mainly targeted towards Android and iOS applications. For this reason, as seen in projects from previous years connecting to the device can be particularly troublesome, the decision was made to develop the application exclusively for Android or iOS.

Both platforms provide their own advantages and disadvantages. As Android has over 80% market share and having previous experience developing an Android application, the decision was made to develop the mobile application on Android.

Technologies

Android Studio

Android Studio is the official integrated development environment (IDE) developed by Google for applications built on the Android OS. It is the primary IDE for creating native applications for Android. Android studio is available to download for development on Windows, Linux or MacOS based operating systems and supports developing in Java or Kotlin.

Android Studio is built on JetBrains IntelliJ IDEA software which provides built in tools and integration, coding assistance, a plugin ecosystem and support for various programming languages (Wikipedia, 2021). It provides an easy to use and straight forward interface for developing an Android app that handles the organisation of projects and the complexity of file management and maintenance for the developer.

Android Studio uses the Gradle build system which is an advanced build toolkit. Gradle automates and manages the build process. The process compiles the apps source code and resources and packages them into APKs for testing and deployment.

Languages

Android provides technologies that allows the developer to create an application using either the Java or Kotlin programming languages. Java is high level programming language which was formed by Sun Microsystems in 1995 that can run on a variety of platforms. Java is an object orientated language, is platform independent and is simple and secure to use. It also comes with “a vast collection of libraries, frameworks, and tools available for Android development” (medium.com, 2021).

Kotlin is relatively new and was developed by JetBrains in 2010. Backed by Google Kotlin is fast becoming the preferred language for Android app developers as many major companies are moving to Kotlin to develop their applications. It was created to make development more simple, concise, safe, and interoperable across multiple languages and platforms.

Alternatives

Alternatives to using Android Studio include IntelliJ IDEA or Visual Studio. IntelliJ is considered to be one of the best android development tools available. It is a Java programming app which contains a multiplayer framework that allows the developer to “build top-quality applications that are responsive, fast, and most importantly stable” (ErisnTech, 2021).

Visual Studio provides development software that allows developers to develop applications for native or cross platform applications. Visual studio IDE is developed by Microsoft and provides a platform from which developers can edit, debug, and build their software programs. It “is a feature-rich program that supports many aspects of software development” (Microsoft, 2021).

Conclusion

After researching the various options available for developing a mobile application for Android the choice was made to develop the project using the Java programming language on Android Studio. Android Studio is rated as one of the best IDEs for Android development and is used extensively. Developed by Google it provides the tools and features that can make the development process much simpler for the developer. Java is one of the most popular programming languages that contains a vast collection of libraries and a large community of developers that provide support and documentation for much of the features needed for this project. Having previous experience with Java and Android studio also factored in determining the technologies chosen.

Web Application

Front-end Technologies

React

React is an open-source framework developed created by Facebook. React uses a JavaScript library can be used to build fast and interactive front-end development. According to Stack Overflow Developers survey 2021 (Insights, 2021), React is the beat UI framework available today. This is due to its speed of creation, reuse of components and provide a platform on which high levels of traffic can be managed.

Although, due to its significant pace of development there is an absence of much documentation and involves a steep learning curve which requires the developer to have a good knowledge of JavaScript. React provides a platform that allows the developer to create robust and speedy front-end but requires a severe learning curve to developers with no prior experience.

Angular

Angular is an open-source framework based on TypeScript created by Google. Angular differs from other frameworks due to its two-way data binding trait which allows for real time synchronisation between the view and the model. The Angular framework can be used to develop web or mobile applications. Angular provides high performance, countless documentation, as well as a large ecosystem of resources to the developer.

Angular can be quite difficult for inexperienced developers to learn and provides a steep learning for those with no previous experience. As with React the learning curve involved, and time constraints of this project make both frameworks unsuitable.

Back-end Technologies

Node.js

“Node.js is an open source and cross platform JavaScript runtime environment” (NodeJS, 2021). Node.js allows the developer to write server side and client-side code without having to learn a different coding language, though it is most commonly used as a backend development framework. Node.js provides an extensive number of libraries and services that the developer can easily use to create a web application.

While Node.js allows the developer to create the frontend and backend using the same coding language and can be relatively easy to learn it comes with its own issues. It does not provide good scalability, is not suited to CPU-intensive tasks, and does not work well with relational databases.

Flask

Flask is a lightweight Python back-end web application framework "built with a small core and easy-to-extend philosophy" (Full Stack Python, 2021). As one of Python's most popular frameworks for developing web applications, Flask comes with a library of modules and services that allow the developer to create a backend system for a web application. It does this using custom end points that respond to requests made from the frontend without the worry of handling the thread management, protocols, or data transfer.

Flask provides a compact micro-framework that is easy to understand and implement with a large source of documentation and community support. Flask provides libraries and function that allow the developer to generate dynamic webpages in response to requests made to server.

Conclusion

Having previous experience working with Flask and Python, it was decided to develop the web application part of this project using the Flask framework. While each of the options discussed provide their own unique benefits, due to the time constraints of this project, going with something where I have previous experience, and which does not involve a steep learning curve was the correct decision to make. Flask is a popular web development framework with has a large community that can provide support for any issues that may arise and provides a vast library of modules that will help in developing the web application.

Database

SQL vs NoSQL

Traditional database systems were based on relational database models and used SQL database systems. However, with the rise in 'Big Data' NoSQL databases have dramatically risen in popularity and are increasingly being used to store and manage our data (Li, Manoharan, 2013).

SQL databases are called Relational Database Management Systems (RDBMS). Data is stored in tables with can be linked from relationships within the tables. Structured Query Language (SQL) is the standard query language used when interacting with a relational database. SQL databases are structured, organised, vertically scalable and use a predefined schema which must be adhered to when interacting with the database (**Figure** 23). They require specialized database hardware for better performance and whose transactions must conform to ACID principles.

- **A**tomicity – any transaction made must either be completely successful, or no changes are committed.
- **C**onsistency – data entered must meet all the rules set out by the database.
- **I**solation – transaction with the database must be carried in isolation, sequentially one after the other.
- **D**urability – once a transaction has been committed the data will remain in the table even in the event of power failure or crashes (unless explicitly removed).

Members

Field Name	Data Type	Description (Optional)
Member ID	AutoNumber	Identification Number
Name	Short Text	
Address	Short Text	
Gender	Short Text	
Phone	Number	
Email	Short Text	
Registered	Yes/No	
Available	Yes/No	

Figure 23 SQL Table

NoSQL is a distributed database which contains unstructured and unpredictable data. It prioritises high performance, high availability, and scalability over a structured architecture as with SQL. NoSQL databases use JSON format, key value storage with no predefined schema or declarative query language. It is a highly flexible database structure, horizontally scalable, and is bound to no query design accepting of whatever data is sent to it. NoSQL follows CAP (Consistency, Availability, Partition Tolerance) theorem and BASE transactions which can be used for large scale distributed systems.

- **Ba**sic Availability – rather than enforcing consistency, NoSQL ensures availability by spreading its data across multiple nodes of the database.
- **S**oft State – does not enforce immediate consistency
- **E**ventual Consistency – data reads are still possible even though the data stored may not be consistent.

```
{
  "address": {
     "building": "1007",
     "coord": [ -73.856077, 40.848447 ],
     "street": "Morris Park Ave",
     "zipcode": "10462"
  },
  "borough": "Bronx",
  "cuisine": "Bakery",
  "grades": [
     { "date": { "$date": 1393804800000 }, "grade": "A", "score": 2 },
     { "date": { "$date": 1378857600000 }, "grade": "A", "score": 6 },
     { "date": { "$date": 1358985600000 }, "grade": "A", "score": 10 },
     { "date": { "$date": 1322006400000 }, "grade": "A", "score": 9 },
     { "date": { "$date": 1299715200000 }, "grade": "B", "score": 14 }
  ],
  "name": "Morris Park Bake Shop",
  "restaurant_id": "30075445"
}
```


Figure 24 NoSQL (Source: Guide to NoSQL Databases for Developers (crondose.com))

Firebase

Firebase is a Backend-as-a-service (BaaS) platform developed by Google. It provides the platform from which mobile and web application developers can link their applications to a backend cloud storage platform where data can be pulled directly form the cloud with no server involvement. Firebase is NoSQL cloud database from which clients can connect to directly in Realtime.

Realtime Database enables users to upload data to the database and the data can be synced across all available devices instantly. Firebase does not connect through HTTP as with most databases, instead connecting through a WebSocket which are much faster than through HTTP.

Firebase provides many services including real-time database, cloud firestore, authentication, analytics and messaging services which can be accessed by the developer saving on time and complexity for the developer allowing them to focus on other aspects of their application. The firebase dashboard provides a simple interface from which the developer can choose from several services provided on the platform which can be used to carry out common tasks for the user (Firebase, 2021).

Firebase provides a serverless service which can store massive amounts of data for applications which involves minimal setup. Once setup firebase provides inbuilt security, authentication, and quick and easy access to, maintenance and processing of the database.

SQLite

“SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine” (SQLite, 2021). It is built into all mobile devices and most computers and is free for all to use. SQLite contains a complete SQL database contained within a single disk file

which is embedded into the end program. It uses the internal memory of the device to store the data without the need for a separate server process.

SQLite is a cross platform service which requires no configuration and stores data to ordinary disk file. SQLite is a complete SQL database that provides fast interactions with the database where reads and writes are made directly to disk files with no external dependencies. Its small size and efficient use of memory space make it a popular choice amongst many small to medium websites and mobile devices.

SQLite is a fast, reliable, easy to use serverless database that is a great choice for projects that require a small amount of local data to be stored which can be accessed rapidly. However, it has many limitations in terms of accessibility, scalability, and is not suited for large applications.

Conclusion

In choosing what database to use for this project, whether to use a SQL database or NoSQL database, the decision was made to use Firebases NoSQL database system. Firebase allows free data storage up to 1GB and can be easily integrated to mobile and web applications. NoSQL databases will allow a huge amount of flexibility and scalability that cannot be provided using a SQL database and will be of massive benefit as this project expands. Firebase provides authentication, security, plus multiple extra features that will help throughout the project as well as providing a vast amount of documentation and a large community support.

Fitness Wearables and Health

"Wearables and other health technologies are driving a more patient-focused, preventive approach to healthcare" (Clarkston Consulting, 2021). As our aging population continues to increase and demands on the healthcare system grows, new approaches into how healthcare is provided must be established. Wearable technologies provide a way in which real-time monitoring and data collection can be used to provide personalized treatment for individual patients by a medical practitioner remotely from their laptop or PC.

Trends in wearable devices and fitness trackers continues to grow and the impact of this can be seen in the medical device industry as more and more devices are being developed to monitor and analyse a patient's health statistics. Wearables provides an improved approach to providing quality access to healthcare at a much lower cost to the patient and the health system by acting as a proactive and preventive measure to health.

Tech companies are creating a multitude of products for fitness and general wellness tracking purposes. This includes products such as Fitbit, the Amazon Halo band, and the Apple smart watch. These devices can now be used to monitor blood oxygen levels, measure heart rate, measure the wearers gait or provide various other sensory information, allowing users to monitor their health data.

The healthcare industry is following suit and examples can be seen in the industry already. "Phillips developed a wearable biosensor self-adhesive patch for patients to monitor movement, heart and respiratory rate, and temperature, which led to 89% fewer patients deteriorating into preventable issues like cardiac arrest, showing the real impacts of this technology" (Clarkston Consulting, 2021).

Medical devices can now be used to connect and communicate with other system and devices to monitor a patient's data providing various capabilities and benefits. "Digital tools are giving providers a more holistic view of patient health through access to data and giving patients more control over their health. Digital health offers real opportunities to improve medical outcomes and enhance efficiency" (U.S. Food & Drug Administration, 2020).

As Wearables and digital health technologies continue to evolve, they are developing an approach to provide remote healthcare. Personalized data for individual patients can cause a significant impact on patient care by providing real-time monitoring and data analyse for each individual patient.

Why Balance?

Balance measurement can be used by medical professionals to determine a great deal of information about their patients. “Balance control is a result of multimodal coordination in brain and central nervous system” (Ainone, 2021) and balance testing can play an important role in monitoring human performance. Testing can be carried out on a wide category of people from children to elderly people, to elite sports personal to help evaluate physical and neurological performance.

As we grow older the ability to balance can be of particular concern as our balance can become impaired due to aging, disease, physical or neurological issues (Berg, 1989). Balance requires coordination between the brain and nervous system and any deterioration in balance can prove telling in diagnosing the early stages in many neurological issues such as Parkinson’s disease or Alzheimer’s.

By monitoring a patient’s performance while carrying out various balance activities, a medical personal will be able provide a prognosis on the patient. Tests such as Romberg’s test (Khasnis. and Gokula, 2003) can be used to diagnose any problems with a patients balance that help indicate id there is any issues with the patient.

As our populations life expectancy continues to grow and more pressures are put open an already overworked health system, early diagnosis of issues found in elderly patients can lead to better rehabilitation and medication. This can then in turn decrease the levels of pressure placed upon the health system.

The 4-Stage Balance Test

The 4-Stage Balance test is a tool that can be used to assess a patient's mobility and risk of falls, based on their ability to hold four progressively more challenging positions. The test is used to evaluate the static balance of elderly individuals. The test was developed by Physiopedia, a non-profit organization, who have built a physiotherapy community to educate and advocate for the physiotherapy profession all over the world.

The balance positions to be held by the patient include;

1. Standing with their feet side-by-side
2. Instep Stance – placing the instep of one foot so that it is touching the big toe of the other.
3. Tandem Stance – place one foot in front of the other where the big toe of the back foot is touching the heel of the front foot.
4. Standing on one foot.

The patient will carry out each activity for 10 seconds, before on successfully completing an activity proceeding to the next activity. Should the patient not be able to hold the position for the duration of the time and lose balance the test should be stopped.

Not being able to carry out specific balance positions can give an indication that the patient is at an increased risk of falling. Monitoring the patient's performance can provide a medical practitioner with early diagnosis of any risks and will allow them to recommend gait and balance exercises or "an evidence-based fall prevention program, such as Otago balance program, Tai Chi" (Physiopedia, 2021).

Bibliography

Ainone, 2020. How do you measure balance?
[Online] Available at: Ainone Balance – Ainone [Accessed 30 October 2021].

ALMA.care, 2020. ALMA.care
[Online] Available at: ALMA.care [Accessed 30 October 2021].

Berg, K., 1989. Balance and its measure in the elderly: a review. *Physiotherapy Canada*, *41*(5), pp.240-246.

BuiltIn, 2021. What is wearable Technology?
[Online] Available at: What is Wearable Technology? Examples of Wearables. | Built In [Accessed 09 October 2021].

Clarkson Consulting, 2021. The Impact of Wearables and Digital Health Technologies
[Online] Available at: The Impact of Wearables and Digital Health Technologies | Clarkston Consulting [Accessed 21 February 2022].

Evans, C., Ingerson, B. and Ben-Kiki, O., 2001. YAML Ain't Markup Language (YAML™) Version 1.1.

ErisnTech, 2021. 8 Most Used Android IDE For Developing Android Apps
[Online] Available at: https://www.erisn.com/8-most-used-android-ides-for-developing-android-app[Accessed 22 December 2021].

Faust, K., 2016. *CMAKE* (Doctoral dissertation, University of Rijeka. Department of Informatics).

Firebase, 2021. Firebase Documentation.
[Online] Available at: Build Documentation | Firebase Documentation (google.com)Accessed 02 December 2022].

Fitbit, 2021. Fitbit – Charge 5.
[Online] Available at: Introducing the Revolution in the Fitbit Charge Family: Charge 5 - Fitbit Blog [Accessed 02 November 2021].

Full Stack Python, 2021. Flask.
[Online] Available at: Flask - Full Stack Python [Accessed 02 January 2022].

GlobalData, 2020. Wearable technology is a key driver in the growth of mobile health
[Online] Available at: Wearable technology is a key driver in the growth of mobile health, says GlobalData (cxotoday.com) [Accessed 10 October 2021].

Gomez, C., Oller, J. and Paradells, J., 2012. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors*, *12*(9), pp.11734-11753.

Helge Omre, A. Bluetooth low energy: Wireless connectivity for medical monitoring. *J. Diabetes Sci. Technol.* 2010, *4*, 457–463.

Insights, 2021 Developer Survey
[Online] Available at: Stack Overflow Developer Survey 2021 [Accessed 29 December 2021].

IBM Cloud Education, 2021. Docker
[Online] Available at: What is Docker? | IBM [Accessed 26 October 2021].

Khan, M.W. and Abbasi, E., 2015. Differentiating Parameters for Selecting Simple Object Access Protocol (SOAP) vs. Representational State Transfer (REST) Based Architecture. *Journal of Advances in Computer Networks*, *3*(1), pp.63-6.

Khasnis, A. and Gokula, R.M., 2003. Romberg's test. *Journal of postgraduate medicine*, *49*(2), p.169.

LinkedIn, 2021. Top Market Trends for Wearable Technology in 2021
[Online] Available at: Top Market Trends for Wearable Technology in 2021 | LinkedIn [Accessed 10 October 2021].

Li, Y. and Manoharan, S., 2013, August. A performance comparison of SQL and NoSQL databases. In 2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM) (pp. 15-19). IEEE.

Medium.com, 2021. Kotlin vs Java: Which is the Best Choice for Android App Development?
[Online] Available at: https://medium.com/javarevisited/kotlin-vs-java-which-is-the-best-choice-for-android-app-development [Accessed 22 December 2021].

Michigan State University, 2020. Meeting the Healthcare Needs of an Aging Population [Online] Available at: https://www.michiganstateuniversityonline.com/resources/healthcare-management/meeting-healthcare-needs-of-aging-population [Accessed 01 November 2021].

Movesense, 2021. Open Wearable Tech Platform.
[Online] Available at: Movesense – Open Wearable Tech Platform
[Accessed 09 October 2021].

Mukhopadhyay, S.C., 2014. Wearable sensors for human activity monitoring: A review. *IEEE sensors journal*, *15*(3), pp.1321-1330.

NodeJS, 2021, Introduction to Node.js.
[Online] Available at: Introduction to Node.js (nodejs.dev) [Accessed 29 December 2021].

0'RIELLY, 2020. Open Wearable Tech Platform.
[Online] Available at: Introduction - Getting Started with Bluetooth Low Energy [Book] (oreilly.com)
[Accessed 25 October 2021].

Physiopedia, 2021. The 4-Stage Balance Test.
[Online] Available at: The 4-Stage Balance Test - Physiopedia (physio-pedia.com) [Accessed 21 February 2022].

Rad, B.B., Bhatti, H.J. and Ahmadi, M., 2017. An introduction to docker and analysis of its performance. *International Journal of Computer Science and Network Security (IJCSNS)*, *17*(3), p.228.

Rutherford, J.J., 2010. Wearable technology. *IEEE Engineering in Medicine and Biology Magazine*, *29*(3), pp.19-24.

Silberschatz, A., Galvin, P.B. and Gagne, G. 2018. *Operating system concepts*, 10th edition John Wiley & Sons.

SQLite, 2021. What is SQLite?
[Online] Available at: SQLite Home Page [Accessed 02 January 2022].

Tao, X. (2005). Wearable Electronics and Photonics. (X. Tao, Ed.) (1st ed., pp. 1–244). Cambridge: Woodhead Publishing Limited.

Tong, R. ed., 2018. *Wearable technology in medicine and health care*. Academic Press.

U.S. Food & Drug Administration. What is Digital Health?
[Online] Available at: What is Digital Health? | FDA [Accessed 21 February 2022].

Vashist, S.K. and Luong, J.H. eds., 2018. *Handbook of immunoassay technologies: approaches, performances, and applications*. Academic Press.

Wikipedia, 2021. IntelliJ IDEA.
[Online] Available at: IntelliJ IDEA - Wikipedia [Accessed 27 November 2021].

Wright, R. and Keith, L., 2014. Wearable technology: If the tech fits, wear it. *Journal of Electronic Resources in Medical Libraries*, *11*(4), pp.204-216.

Zhang, X., Wen, Z., Wu, Y. and Zou, J., 2011, May. The implementation and application of the internet of things platform based on the REST architecture. In *2011 International Conference on Business Management and Electronic Information* (Vol. 2, pp. 43-45). IEEE.