Adam Stock

The University of the West of England Faculty of Environment & Technology Frenchay  

Discrete internet Communicator -An ultra-low power, low cost network connected haptic feedback wearable device.

Contents

[Abstract 3](#_Toc34316256)

[Acknowledgements 3](#_Toc34316257)

[Introduction 3](#_Toc34316258)

[Background 3](#_Toc34316259)

[Direction 3](#_Toc34316260)

[Intended audience 3](#_Toc34316261)

[Objectives 4](#_Toc34316262)

[Research 4](#_Toc34316263)

[Does it exist? 4](#_Toc34316264)

[Hey Bracelet 4](#_Toc34316265)

[Sony Wena 5](#_Toc34316266)

[Generic low-cost smartwatch 6](#_Toc34316267)

[Comparison 6](#_Toc34316268)

[Energy harvesting 7](#_Toc34316269)

[Mechanical harvesting and automatic watches 7](#_Toc34316270)

[Piezoelectric 8](#_Toc34316271)

[Seebeck/Peltier 8](#_Toc34316272)

[Solar 8](#_Toc34316273)

[Conclusion 9](#_Toc34316274)

[Suitable systems 10](#_Toc34316275)

[CC2640R2F 10](#_Toc34316276)

[NXP-QN908X 10](#_Toc34316277)

[NXP-QN902X 11](#_Toc34316278)

[Comparison 11](#_Toc34316279)

[Hardware decision 12](#_Toc34316280)

[Requirements 12](#_Toc34316281)

[Functional 12](#_Toc34316282)

[Non-Functional 13](#_Toc34316283)

[Methodology 13](#_Toc34316284)

[Design 14](#_Toc34316285)

[High level design 14](#_Toc34316286)

[Architecture 14](#_Toc34316287)

[Implementation 17](#_Toc34316288)

[Prerequisite’s 17](#_Toc34316289)

[Virtual machine 17](#_Toc34316290)

[Software tools and development environment 18](#_Toc34316291)

[First steps, setup and hardware testing 18](#_Toc34316292)

[App creation and CRUD 24](#_Toc34316293)

# Abstract

Words go here

# Acknowledgements

Acknowledgements go here

# Introduction

## Background

This project was inspired by my housemate during my second year. She was experiencing some relationship issues with her partner at the time and after a long teary chat with her the primary cause of the issue was revealed – she felt that with her partner working long hours with his independent business and being out of contact with him for much of this time, that he would not think about her (it is worth noting that this is at time in history when most people have smart devices and are rarely unreachable).

This got me to thinking that a discreate device without the distraction abilities of a smartphone or computer could fill this void in the technological landscape while also being able to provide secondary useful functions to a user.

## Direction

Thinking on this subject further drew me to the concept of a simple low-cost device that can provide notification and potentially response to a user about an extremely limited number of matters without the user needing to actively seek out this information or be concerned with the devices upkeep or setup.

From this I noticed the ubiquitous appearance over the last decade or two of rubber wristbands worn by people to show support for various causes such as Cancer Research and Help for Heroes. These bands were inexpensive and did not require maintenance from the wearer as per other forms of jewelry.  
With this in mind, I wondered if my notification idea could be integrated into a band like this in such a way that the band maintains its key plus points – low cost, zero maintenance and robustness.   
I have an interest in mechanical watches and wondered if that kinetic (often described as ‘automatic’) mechanisms used to power many high-end watches such as those by Rolex, Omega and Patek Philippe could be adapted to provide power to the device allowing the user to never have to perform maintenance for the life of the device.

This is all in first person but as it’s your idea this is explaining how you came up with it I think it probable has to remain so.

## Intended audience

* Youth (>18)
  + ‘best friends’
  + Boyfriend/Girlfriend
  + Parents summoning children
* Adults
  + Parents of the above
  + Healthcare workers (if embedded in rubber as charity wristbands hygiene should be simple)
  + Hospital staff as a method of mass notification – as a replacement to a pager
  + Events staff as a method to notify groups of people
  + Lovers
  + Those in need of discreate notification

## Objectives

you need to have the written intro into these bullet points.

* Low cost
* Low power requirements
* Short range communication with a more powerful device
* Haptic feedback

# Research

## Does it exist?

Seeing as the technologies required for this project all exist already (energy harvesting, short range radio, haptic feedback) I began research by seeing if similar products already exist on the market and if so, do they meet the objectives I have set out for the device? in short no, many devices exist that fulfill one or more of the objective but they fail to meet all in a reasonable manor. This must now be in third person

I will outline some of these devices below – again third person

### Hey Bracelet

(House of Haptics, 2018) – ‘The revolutionary bracelet that sends touch over distance’

A picture containing spectacles

Description automatically generated

The Hey Bracelet consists of a pair of bracelets and a smartphone companion app. These items align with my first idea closely with their tag line of ‘The revolutionary bracelet that sends touch over distance' they seem to be aimed squarely at lovers with distance separating them. The bracelet consists of a motor that allows the band to tighten around the wearer’s wrist and a touch sensor on the outside to instigate this action in another user. However, these devices diverge from my idea in two key areas. Firstly cost, at $114.99 at the time of writing (18/11/18) it is difficult to consider these a low-cost item, and secondly power, they require charging via USB.

### Sony Wena

(SONY EUROPE LIMITED, 2018)- ‘The watch. Reborn.’

A picture containing sky

Description automatically generated

The Sony Wena is not a watch but in fact a strap that should fit many watch faces and is available in multiple sizes and designs, including several Sony made face designs. This device is positioned as a more all-round smart device with GPS (GPS, GLONASS, QZSS and SBAS compatible), Bluetooth 4.2 low energy, heartbeat and accelerations sensors, OLED display and NFC like (‘Wena pay’) hardware. Its features include receiving notifications from a smart phone, contactless payment and fitness tracking as well as being water resistant to enable to user to wear the device in the shower or humid conditions. The device has an expected battery life of approximately 1 week with a charging time of 1.5 hours. So, in many ways this device would cover the functions of our product with two key differences, firstly it requires charging (albeit far less frequently than other described devices) and secondly its price tag. Starting at £349 the device is clearly aimed at the premium end of the market and can certainly not be considered disposable or suitable for a younger user.

### Generic low-cost smartwatch

There are many such devices on the market as of 2019 and it would be beyond the scope of this work to review all of them, so in this instance I will focus on a single model the ‘V8 Smartwatch’ this device is made by several companies and often sold under the label’s ‘GONOKER’ , ‘Leegoal’ and ‘DreaT’. The person.



This device consists of the basic functionality of a feature phone in the form factor of a watch. It includes a MediaTek 6261s processor which is based around an ARM Cortex M4 or M7, this is assumed based on information from MediaTek’s website and a datasheet stating it uses the ARM7-E instruction set (Huang, 2012; MediaTek Inc, 2019) . This variant of the chip uses the J-S extensions to enable acceleration of Java based applications. It contains an inbuilt logic for Bluetooth 3.0, GSM/GPRS/EDGE, FM radio, AAC audio (with amplifier), I2C, SPI, UART, USB and SDIO and can be debugged using a JTAG interface. These are accompanied by 128MB of RAM, a 1.51” 240x240 display and a 0.3MP camera. This high amount of integration enables the device to run for a day or more on a single 280mAh battery.

The device can be set to take notifications from an accompanying Bluetooth enabled smart phone using its provided Android or IOS app and has a price of under £10 meaning that it is likely the closest to meeting the goals of our project.

### Comparison

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Product | Notification | Battery life | Additional features | Cost |
| Hey Bracelet | (Single notification based on proprietary app) | 1 week | Waterproof | £86.82 ($115 USD 7/03/2019) |
| Sony Wena | (Multiple based on a proprietary app) | 1 week | Waterproof, OLED display, Wena Pay | Starting at £349 |
| Generic Smartwatch | (Multiple based on a proprietary app) | 1-3 days | Phone features, Apps, Camera, SD card, Colour LCD display | > £10 |

## Energy harvesting

A potential solution to power a device such as is being created in this project would be energy harvesting. This is the process of using some of the energy the human body naturally expends during its normal function and converting/storing this for use by a device, this is often transparent to the end user.

Methods for doing this have existing in many forms for hundreds of years but have only recently become viable to power wearable electronics. This is due to efficiency improvements in both the harvesting methods and in the devices they are intended to power. Below I will outline the details of several of these methods and their suitability for our purposes.

### Mechanical harvesting and automatic watches

This is likely the oldest form of energy harvesting done by humanity with origins known to be dating back at least as far as Daniel Schwenter (1585-1636) and his idea to use respiration to wind a mechanical watch (Richard Watkins, 2016) however this approach is not known to have been used practically.

Commonly this method involves a weight connected to a shaft in such a manner that when attached to the human body this weight will cause the shaft to partially rotate due to the common movement of the person. This shaft would then be used with a ratcheting mechanism to wind a spiral torsion spring storing the collected energy for slow and constant release via a lever escapement (WF&CO, 2019).

The amount of energy stored by mechanisms as described above can power a mechanical watch for upwards of 50 days from a full charge (Hublot, 2019) however at a cost of £262,000 a mechanical device like this is far beyond the funding and scope of this project. Research has been completed into the efficiency of these mechanisms and this is thought to be approximately 45%.(Longhan Xie, Carmen G. Menet, Ho Ching, Ruxu Du, 2009) This however does not give us power in a form that can be used to drive electronics.

This problem has to a certain extent been solved by companies starting with Seiko with their ‘Kinetic’ range beginning in 1988 (Seiko, 2018) followed by various other companies including Citizen and Swatch. These watches contain a mechanical harvesting system as described earlier but instead of transferring energy to the escapement of a watch, they power small generators and in turn a battery or capacitor used to run a quartz clock movement. As these technologies are proprietary the actual power values are not disclosed to the public however upon research some have theorized the production to be within an order of magnitude of 1 joule every 2 days or 0.000277778 Watt hours (Wardell, 2016). At the 1.8v minimum we are likely to need for our microcontrollers this can be converted to 0.15432 mAh using the formulae

Or in our case

Or 154.32 Microamps. This is likely enough for our electronics but is unlikely to be enough to drive a haptic feedback device like a vibration motor.

### Piezoelectric

This method works on the concept that many materials (often crystalline) generate small quantities of electricity when put under mechanical stress, this may be from compression or deformation (flexing) this is often used in industry to detect vibration or sound. In order to use this energy circuitry is required to normalize the energy produced.

When put under stress the piezo element will output a voltage in a range based on the amount of mechanical stress inputted. This can typically range from 20/30 V + down to zero hence the requirement of a buck regulator or similar, to keep the output voltage at a level that is safe to use without burning out energy storage components or the microcontroller itself.

Piezoelectric energy is a reversable process so care must be taken that the energy harvested cannot be conducted back into the piezo element as this would result in deformation of the element and in turn loss of the potential energy.

The amount of energy created with this method is quite low with high voltage outputs and low current (ampere) output. It is not uncommon for elements to put out voltages in a range up to 100 volts or more if under extreme load (such as being hit by a hammer) but the ampere output is usually less than 100 microamps (Elhalwagy. et al, 2017; Hickman, 2017). With outputs like this, losses involved in rectifying and converting the energy output may result in very low efficiency.

### Seebeck/Peltier

The Seebeck/Peltier effect is the effect that when a temperature gradient is applied to a conductor the energy will excite negatively charged particles (electrons) at the hot end of the conductor and cause them to move towards the cold side producing a tiny voltage. Due to the tiny amount of energy produced connecting these in series as would be done with a battery results in the series connections losing the energy generated via resistance and the effects of the temperature gradient. To counter this a second material is used to provide the reverse connection, they are doped in such a way that positively charged particles move from the hot side to the cold thus continuing the flow, these are the same concepts as in semiconductors and are called p-type and n-type conductors.

The output of a thermoelectric generator (TEG) such as this is affected by several factors. Starting with materials used, we are looking for materials with a high electrical conductivity and a low thermal conductivity and at room temperature, in our case the likely choice would be bismuth telluride (Bi2Te3) as it provides these properties at room temperature. Note other materials exist but either exhibit these properties in the wrong temperature ranges or require use of complex and expensive nanotechnologies bringing costs above the already expensive bismuth telluride.

Research has already been conducted into the viability of this form of energy production (Melissa Hyland, Haywood Hunter, Jie Liu, Elena Veety, Daryoosh Vashaee, 2016) . They conclude that in a best-case scenario we can expect production of around 20 μW/cm2 with a more realistic output of 2-8 μW/cm2. Given that our device electronics will likely only have a footprint of a few cm2 additional TEG’s would need to be located in the wristband giving additional cost and complexity to the design.

### Solar

Solar cells have been relatively common in portable electronics since pocket calculators popularized them in the late 1970s (Tout, 2019) and with recent pressure for renewable energy sources have become popular to power everything from cars to homes. These cells, also known as photovoltaic cells receive energy in the form of light and use this to excite electrons in a layer of a semiconducting material to produce an electrical output. Being a relatively mature technology, these devices can be found in many form factors, the main one I will look at here is the BPW34 silicone pin photodiode (Vishay.com, 2008). This device measures 5.4 x 4.3 x 3.2 mm and has a surface area of 0.23 cm2. It can produce around 50 µA under good conditions. For our purposes this means several of them may be required to meet our power needs even under good near ideal conditions. Given that our device is going to be wrist mounted it is unlikely that these conditions will be met due to clothing covering cells, shallow light angles from the user having mobility of their hands as well as factors like being indoors. This means in order to meet our power requirements multiple cells would be required as well as a method to store the energy for continued usage, this would add complexity and cost to the project. Third person.

### Conclusion

Below is a table summarizing the above methods –

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Technology | Cost | Complexity | Size/Practicality | Power potential |
| Mechanical | High (Commonly <£100) | High as no pre made units are available | Acceptable as often already installed in wearables | Viable as already used to power similar wearables |
| Piezoelectric | Low (£1-5 for device and associated electronics) | Minimal as units already exist | Large to produce required power output | Low power density |
| Seebeck | Low(£1-5 for device and associated storage) | Minimal as units already exist | Large to produce required power output | Low power density |
| Solar | Low (£1.50 per cell as of 15/03/2019) | Minimal | Many cells required to negate physical limitations | Viable as multiple cells can be used |

A characteristic of the aforementioned energy harvesting methods is that they provide extremely low amounts of current. Given proper storage using a capacitor or even a small rechargeable battery the amount of energy collected even while being intermittent in delivery should be enough for the relatively low power but frequent draw of a microcontroller. However the use of a vibration motor will cause a high current draw for a relatively long time and is likely to overwhelm most of the above methods if used frequently – this should be taken as a consideration during the design process.

Given that a wearable device will often have a lifespan of less than 5 years, the complexity and cost of the above methods and the low cost and ubiquity of high capacity low volume batteries such as the CR-2477 by Panasonic (Mouser, 2019). The work involved in producing a product with a harvesting method like this is likely not worthwhile.

The aforementioned battery has an output of 3v meaning it can power the device directly, a capacity of 1000mAh meaning it can potentially power a low power device for several years. As well as having a large operating range in heat, solid state that is unlikely to be effected by movement and no reliance on light meaning it will likely provide a more reliable power source for the lifetime of the device.

I think you need to increase the size of your literature review. You need to add searches on haptic devices. Ideally you need at least another 10 to 15 more references. This though can take the similar structure to the energy harvesting you’ve already done.

## Suitable systems

this section is looking at the processors might be better placed in the design section or better the implementation section.

Below are several chips and developments boards considered for the project plus a short description of their pro’s and con’s and suitability for the task at hand.

Note prices are taken from <https://uk.farnell.com> on the 23/2/2019 unless otherwise specified.

### CC2640R2F

This is an ultra-low power microcontroller (cortex m3) and RF system on a single chip, it is designed specifically to be capable of running on a coin cell battery or via an energy harvesting device (Texas Instruments, 2019). The wireless capabilities include Bluetooth 4.3 and Bluetooth 5le connected to an ultra-low power sensor controller (based on a cortex m0) this is designed to run independently of the main cortex m3 processor allowing for extremely low power consumption while the main processor is in sleep states.  
The cost of this chip is approximately £4.07 per chip and the development kit (Launchpad) is £23.55 and is available immediately.

* Wide Supply Voltage Range
  + Normal Operation: 1.8 to 3.8 V
  + External Regulator Mode: 1.7 to 1.95 V
* Active-Mode RX: 5.9 mA
* Active-Mode TX at 0 dBm: 6.1 mA
* Active-Mode TX at +5 dBm: 9.1 mA
* Active-Mode MCU: 61 µA/MHz
* Active-Mode MCU: 48.5 CoreMark/mA
* Active-Mode Sensor Controller: 0.4mA + 8.2 µA/MHz
* Standby: 1.1 µA (RTC Running and RAM/CPU Retention)
* Shutdown: 100 nA (Wake Up on External Events)

### NXP-QN908X

QN908x is an ultra-low-power, high-performance and highly integrated Bluetooth Low Energy solution for Bluetooth® Smart applications such as sports and fitness, human interface devices, and app-enabled smart accessories. It is specially designed for wearable electronics with a small capacity battery.(NXP products, 2019). This chip is based around the Cortex®-M4F microcontroller and is capable of providing floating point operations as denoted by the ‘F’ postfix and provides Bluetooth Low Energy support.   
This chip is based on a single CPU design and as such requires its main CPU to be active in order to use its Bluetooth functionality.  
The cost of this chip is approximately £4.52 and the development platform (QN9080-DK) is £88.93 but is currently out of stock.

* Single 1.8 V ~ 3.6 V power supply
* 1 µA deep sleep mode
* 2 µA sleep mode (32-kHz OSC/RTC on)
* 3.5 mA RX current with DC-DC at 3 V supply
* 3.5 mA TX current @0dBm Tx power with DC-DC at 3 V supply

### NXP-QN902X

QN902x is an ultra-low power, high-performance and highly integrated Bluetooth LE solution. It is used in Bluetooth Smart applications such as sports and fitness, human interface devices, and app-enabled smart accessories. It is specially designed for wearable electronics and can run on a small capacity battery such as a coin cell battery(NXP products, 2019).   
This MCU is designed around Cortex® M0 design and also provides Bluetooth Low Energy support.   
The cost of this chip is approximately £3.95 and the development platform (QN9020DKUL) is £37.40 but is again unavailable at the time of research.

* Single power supply of 2.4 V to 3.6 V for QN9020/1
* Single power supply of 1.8 V to 3.6 V for QN9022
* Integrated DC-to-DC converter and LDO
* 2 μA deep sleep mode
* 3 μA sleep mode (32 kHz RC oscillator on)
* 9.25 mA RX current with DC-to-DC converter
* 8.8 mA TX current @0 dBm TX power with DC-to-DC converter

### Comparison

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Chip | Architecture | Power Consumption | RF support | Cost for chip | Cost for development |
| CC2640R2F | Cortex® M3 + Cortex® M0 components |  | Bluetooth 4.3, Bluetooth 5 low energy | £4.07 | £23.55 |
| NXP-QN908X | Cortex®-M4F |  | Bluetooth 5 low energy | £4.52 | £88.93 |
| NXP-QN902X | Cortex®-M0 |  | Bluetooth 5 low energy | £3.95 | £37.40 |

### Hardware decision

The decision for what MCU to use was almost made for us due to time/availability constraints, with the only device both available and at a reasonable cost being the TI CC2640R2F. In actuality the NXP-QN902X may have been the better choice as its simpler design based on a Cortex M0 is likely to yield minor power efficiency advantages however the larger costs of the development kit and at the time of research being unavailable made the CC2640R2F the better choice. The NXP-QN908X was largely ruled out due to its very high development cost, also being based on a Cortex M4 means its computational power is fay beyond what is required – its inclusion of a floating-point math coprocessor as denoted by the ‘F’ part of Cortex M4F is also an addition that would be unused in this project.  
An advantage of using the TI CC2640R2F allows for usage of Texas Instruments own Eclipse based ‘Code Composer Studio’ development environment which we already have some experience with.

# Requirements

This section of the document outlines the overall requirements for the project and uses MoSCoW (Must have, Should have, Could have, Won’t have ~~this time~~) prioritization. They are split into two sections functional and non-functional.

Functional

|  |  |  |
| --- | --- | --- |
| Requirement # | Requirement description | Category |
| 1.1 | System powers on | Must have |
| 1.2 | System pairs with Bluetooth device | Must have |
| 1.3 | System retries previous Bluetooth connection upon disconnection | Must have |
| 1.4 | App should boot in android need to make clear this is a mobile app. In fact it might be better to split this into two projects won the embedded system and the other the mobile app and have separate requirements for both. See what Craig thinks. | Must have |
| 1.5 | App should transmit notification to device | Must have |
| 1.6 | Device should receive notification from app | Must have |
| 1.7 | Device should vibrate if notification dictates | Must have |
| 1.8 | App should initiate notification message manually | Should have |
| 1.9 | App should initiate notification message from local source (e.g. alarm) | Should have |
| 1.11 | App should initiate notification from remote source (database lookup) | Must have |
| 1.12 | App should be able to send notification to a remote user (database insertion) | Must have |
| 1.13 | The product should provide differentiation between sources of notification (intensity and/or duration of pulse) | Should have |
| 1.14 | Device should have a way to silence notification locally (button/motion) | Could have |
| 1.15 | Device should be able to display reason for message visually (RGB or display) | could have |

Non-Functional

|  |  |  |
| --- | --- | --- |
| Requirement # | Requirement description | Category |
| 2.1 | Device must provide notification only to the user (discretion) |  |
| 2.1 | Device must provide long service life without user input |  |
| 2.3 | Device must have physical attributes to enable comfortable wearing/use |  |
| 2.4 | Device should provide usage without power input (charging) |  |
| 2.5 | The product must provide notifications in a timely manor (>10s) |  |
| 2.6 | The product should be of near disposable cost (>£10) |  |
| 2.7 | The device must work over a reasonable range between it and internet enabled device |  |
| 2.8 | The device should provide a widespread method to connect to an internet enabled device (Bluetooth) |  |

# Methodology

The methodology being used for this project is the Agile method. This will allow us to change features and design considerations throughout the process.

Two methodologies were considered for the project, Waterfall and Agile. Though for a single person project with limited scope the waterfall method may work, it does not allow for additions or revisions to the end product without significant time delays. It also introduces the potential for scope creep turning the project into an impossible task as features are proposed during any resets in the design process.

Using an Agile methodology allows us to expand on features continuously throughout the project while not suffering from the scope creep issues of Waterfall – this is accomplished using the previously mentioned MoSCoW method. This allows new features to be proposed and logged without requiring them to be in the final product while also giving a list for any new iterations or revisions to follow – saving on future development time.

# Design

Right you need a high level overview which shows to distant mobile phones and each of the mobile phones talking to your haptic device. This introduces the idea that you are developing a mobile device application and an embedded system which provides haptic feedback.

I think you should then do the two designed separately and document them separately.

She then have a picture of the mobile phone communicating with the haptic device; this gives you three components the mobile device, haptic device and the communication systems between them.

The haptic device should have an ACP diagram and then property some state code diagrams and then maybe some pseudocode.

The mobile device should have some kind of high-level diagram, some class diagrams, maybe some data structures and pseudocode.

The communications protocol between the two devices should be stated should be given.

I think this will make it a lot clearer as to what you’re making.

## High level design

#### Outline

The device being created is a small low-cost wearable for personal notifications.

Its purpose is to provide discreate notifications to a user with minimal setup and configuration.

It is intended to be used by children and adults with access to a smart phone but without any prior experience with such devices.

#### Design goals

* Be easy to use
* Require no prior knowledge to use
* Provide discreate notification for local or remote matters using a vibrate motor
* Be energy efficient to run long term from an embedded battery

### Architecture

For this project research has been performed as above into potential competitor’s devices with the aim to show any pitfalls in similar designs and to allow the final product to outperform other similar devices on the market.

#### Hardware

The hardware will be based around the CC2640R2FRSMT MCU and developed on the relevant ‘Launchpad’ development board for this chip and using technical data from the ‘CC13x0, CC26x0 SimpleLink™ Wireless MCU Technical Reference Manual’ (Texas Instruments, 2017). Once development is completed using the Launchpad development device a cost reduction will be performed. This will involve installing the same MCU onto a bespoke designed PCB that will be considerably smaller than the launchpad and contain no extraneous devices for programming and debugging as are found on development boards.

The final hardware should consist of –

* The MCU
* PCB
* Ancillary components inc –
  + Resistors
  + Capacitors
  + 24Mhz Crystal

Note these components are found on page 43 of the Texas Instruments datasheet ‘swrs204b.pdf’ as found on their website.

Further attention will need to be paid as some items in this document are described as components when they are in fact resistance/capacitance of tracks on the PCB, specifically those related to the Bluetooth antenna.

#### Data

This product will need to store some customer data and as such some considerations must be made as to what should be stored, where it should be stored and how that information is secured.

##### What is stored

Data to be stored should be kept to a minimum, the minimum that can be stored for this system to work would consist of the following –

* Users name

This will be taken when the user signs up and stored entirely on the database making it harder for an attacker to acquire.

(in future iterations this would be secured with a one-way hash function so that the data is never stored on either device or database in an easily readable way)

* Users unique identification

This should be a unique number generated by the database and given to the user, this will allow multiple people of the same name to use the system and cut down on data being sent and received.

* The wristband status

This is a Boolean value that shows if the user’s device is/should be triggered.

In further and expanded iterations of this project more details could be stored in further fields of the database for the purpose of creating an interface from other systems – such as a web interface or a method to mass notify people at an event or similar.

##### How it is stored

The physical device will store no user information.   
The only unique information on the device will be the BD\_ADDR, this is a 48bit address commonly displayed as a 12-digit hexadecimal number. The first 24bits of this address is a organization unique identifier (OUI), which identifies the manufacturer and the last 24bits are a unique number(JIMBLOM, 2020). This information should not be modifiable.

The android app stores only the user’s unique identification number as this is what it would use to look up its respective band’s status. As this number alone gives no sensitive information its protection is not essential. However, the number could be used to intercept another user’s notification and so some form of encryption may be used. The app also contains the password for connection to the database and this should be protected so that an attacker cannot read the contents of the database.

The database stores user’s names, unique identifiers and the statuses of their respective bands and as such care must be made to make this inaccessible outside of the confines of the app. This may be achieved using password security built into MySQL but may be further strengthened by other methods.

#### Communication

This project will use two instances of communication to function, the first being between the device and the smart phone, and the second between the smartphone and the database.

##### Device – smartphone

This will be handled by Bluetooth low energy and should rely on the GATT service feature (Generic ATTributes). These are a collection of ‘services’ and ‘characteristics’ that allow communication between devices (Bluetooth SIG, 2020).

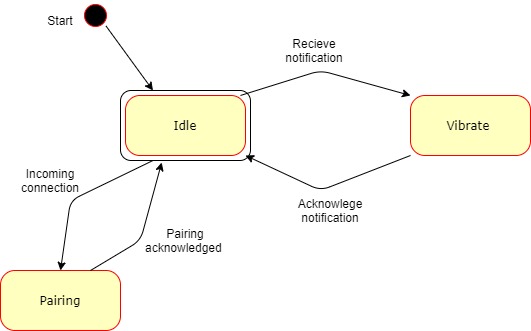
##### Smartphone – database

This will be handled by android using either Wi-Fi or mobile internet to connect to the internet. Messages will be sent via HTTP invoking PHP scripts on the database server. These scripts will be what directly communicates with the database.

#### Finite state machine

This FSM diagram provides a high-level visual representation of the states of a functioning system and the transitions between those states.

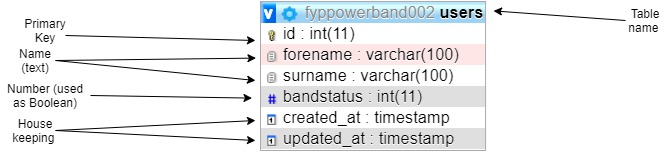
##### The band –



##### The app –

INSERT IMAGE HERE X

#### Database diagram



#### Modules

The app is comprised of various system modules with specific functions, this choice has been made to make updates and upgrades easier in the future as a component can be replaced without needing to rewrite the entire application.

* + - Main application
      * Shows menus and functions of the app to the end user
      * Handles user input via android
    - JSON parser
      * Handles compilation of data for database requests
      * Compiles HTTP requests
      * Sends commands via HTTP
    - Notification handler
      * Handles Bluetooth requests
      * Deals with GATT services

# Implementation

again if you think of this is to projects and then each sprint or phase could this the changes you’re making to each of those two projects to get them working together.

## Prerequisite’s

Before attempting this project, multiple systems were required to be in place, many of these systems relied on each other to function. In order to accomplish this, I did the following – third person.

### Virtual machine

The CC2640R2F launchpad development board by Texas Instruments is recommended to be programmed via the ‘Code Composer Studio’ IDE. I was already using this with another project so as to stop cross contamination between projects and to enable me to work from multiple places I created a Windows 10 virtual machine on an external SSD. The SSD running over USB 3.0 enabled the Virtual machine to run at near native performance compared to a USB 2.0 Hard drive or Memory stick. Third person.

The virtual environment used is a blank version of Windows 10 Home (the 128GB ram limit on home versions is not an issue with the hardware available) running on VMware Workstation Pro. This environment was then used on 3 machines and forwarded the following resources –

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Machine | Cpu available | Cpu allocated | Ram available | Ram allocated |
| CLEVO N850HK1 | Intel I7 7700HQ  2.8-3.8 ghz  4 cores  8 threads | 4 virtual cores | 8GB | 4GB |
| Custom Desktop | AMD 3900x  3.8-4.6 ghz  12 cores  24 threads | 16 virtual cores | 16GB | 12GB |
| UWE HP Desktop | Intel I5 7500  3.4-3.8 ghz  4 cores  4 threads | 4 virtual cores | 16GB | 8GB |

The usage of more Ram provided a better experience while programming – faster switching between applications, however more cores allowed for considerably faster compilation of larger projects.

### Software tools and development environment

Inside the virtual machine various tools were installed for development, the main tools used are listed as follows –

* Notepad++
* CodeComposerStudio 9.0.1 (CCS)
* Android studio 3.6.1
* SimpleLink™ CC2640R2 SDK - Bluetooth® low energy v3.30

Additionally, outside of the virtual machine, either on another machine on the same network (for ease of security) or occasionally in the host machine for the VM the following tool was used to provide a remote MySQL database –

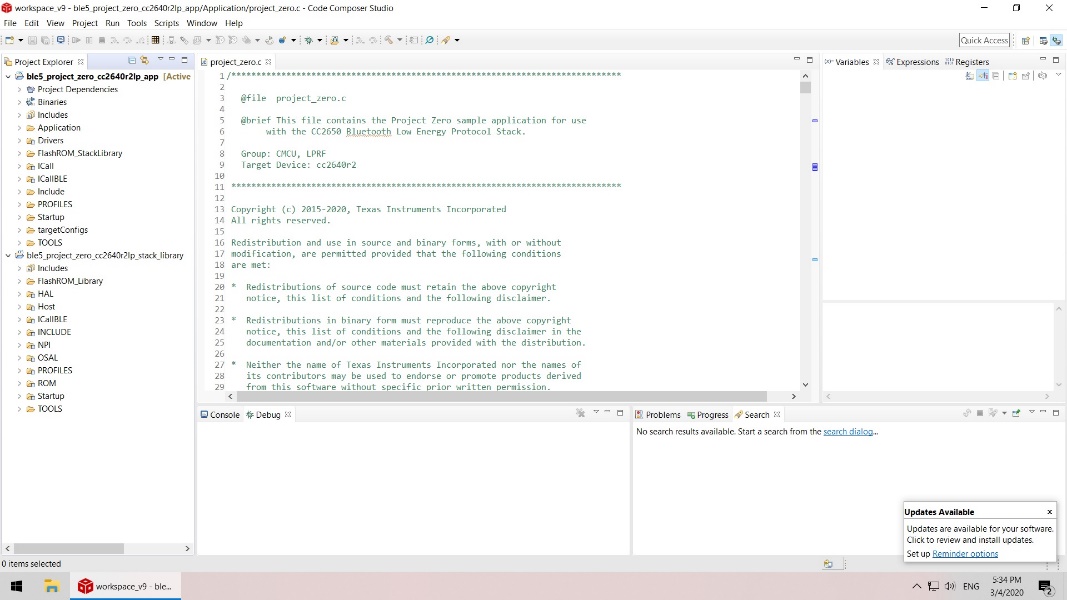
* WampServer 3.2.0

During the development cycle several of these tools were updated as will be discussed further on.

### First steps, setup and hardware testing

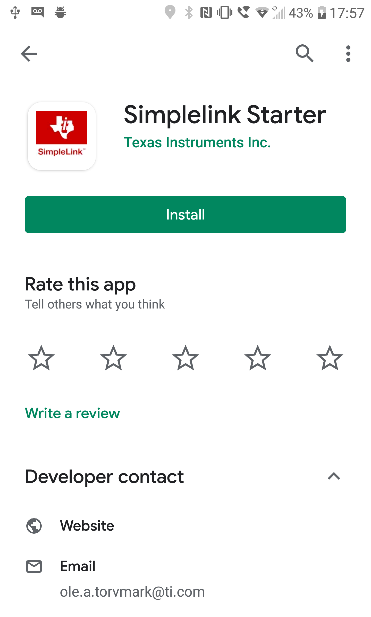
Before starting development in earnest some testing was done to check that the hardware was working correctly so that testing of my own code and general troubleshooting would not be futile. This involved downloading the demo application from the CC2640R2 SDK on to the development board and connecting too it with the TI Simplelink starter application (available [here](https://play.google.com/store/apps/details?id=com.ti.ble.simplelinkstarter&hl=en_GB)).

Upon attempting to download this as per the ‘Getting started’ section of the TI website we came upon the first hurdle, the links in the ‘Resource Explorer’ section of CCS (this is essentially a web browser linked directly to TI’s support pages) no longer linked correctly to its web servers and gave various connection/404 errors. To circumvent this a regular browser could be used as well as a combination of Google searches and using the search form on the TI website, however this would often serve archived results that were no longer relevant to this version of the SDK or CCS or serve results relevant to other development boards – frequently the MSP430. This meant great care had to be used when vetting the usefulness of these results. Originally the demo software was available as a separate download, but this has since been integrated into the SDK. After acquiring the correct information several dependencies needed to be loaded into CCS in order to make Project Zero compile, these were mostly related to the BluetoothLowEnergyStackHandler and TIRTOS, the latter of which had updates giving issues with its integration, luckily an update was released for the SDK containing all the required components and dependencies giving a project that looked like the image below.



After resolving this issue, the project could be loaded up in CCS easily complete with its single dependency ble5\_project\_zero\_cc2640r2lp\_stack\_library that is included in the SDK files.

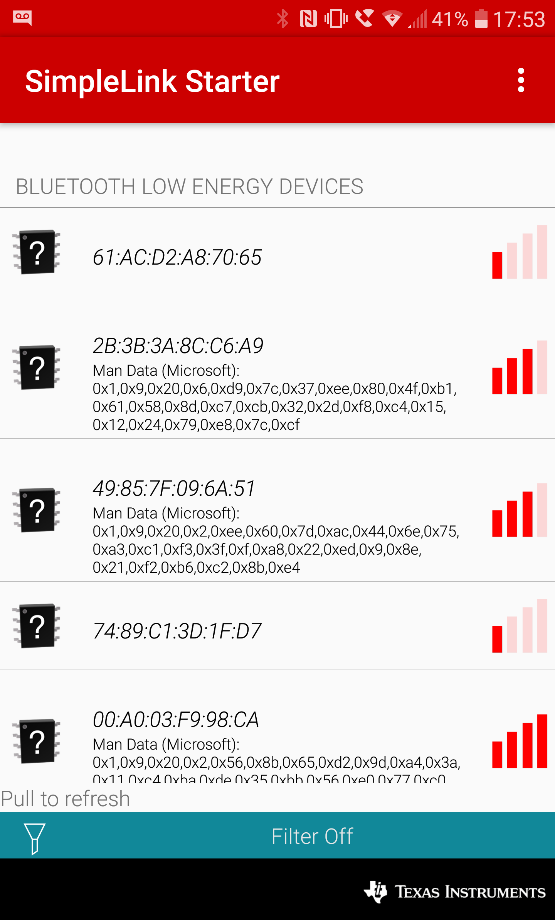
Next on an Android device with access to the play store the Simplelink software should be downloaded and installed giving the results as per the following images –



Installation of this will provide an entry into the main app drawer looking like the following –



Opening this application will give you the following provided that Bluetooth is enabled on the device–



Upon deploying projectzero to the board using the deployment buttons at the top of the IDE the board will start broadcasting its name, this can be scanned for in the simplelink app.

INSERT SCREENSHOT

Once connected the advertised GATT services can be seen for the LED’s and Button’s.

INSERT SCREENSHOT

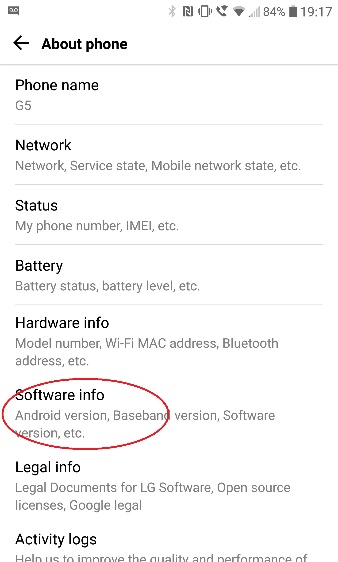
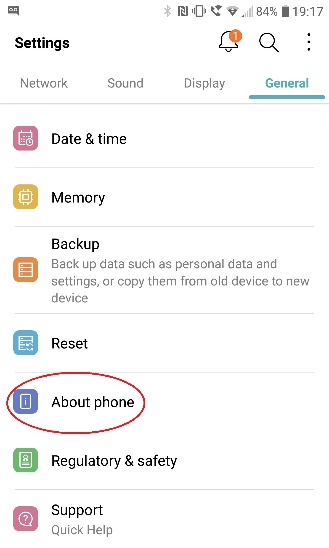
The status of these items can either be see physically on the board as with the LED’s or read in the app with the buttons.

INSERT DOUBLE SCREENSHOT HERE

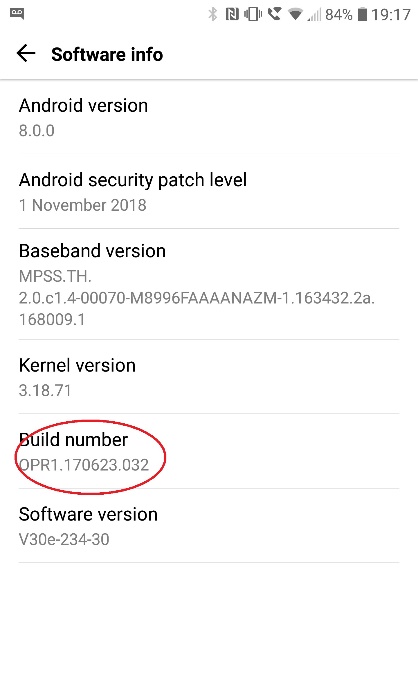
Next a basic application was written in Android studio to test its ability to be deployed to a device and that its debugging tools are available for use. In this instance a tutorial on database usage was followed in order to achieve multiple targets in one application(Tamada, 2017).

In order to make the debugging tools work a development enabled version of android must be used. These features can be enabled in most android devices.

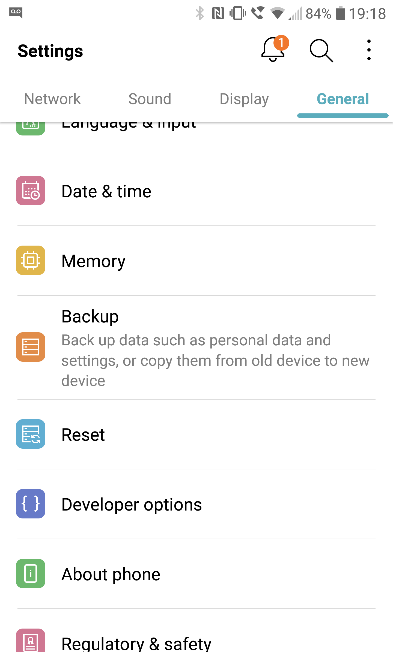
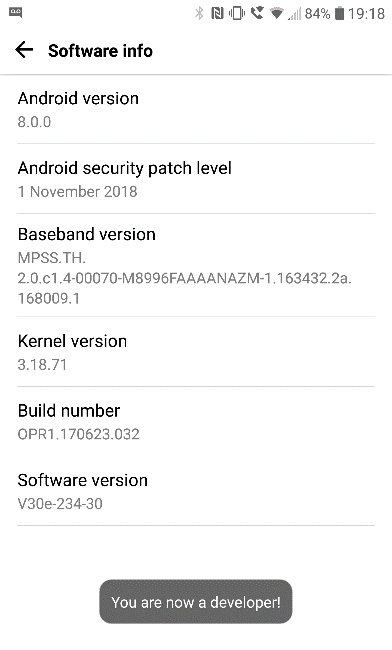
This is accomplished by going into settings – about device – and to software info



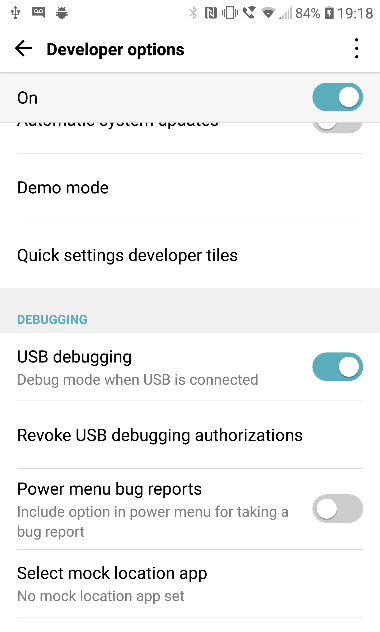
And pressing repeatedly on the Build number

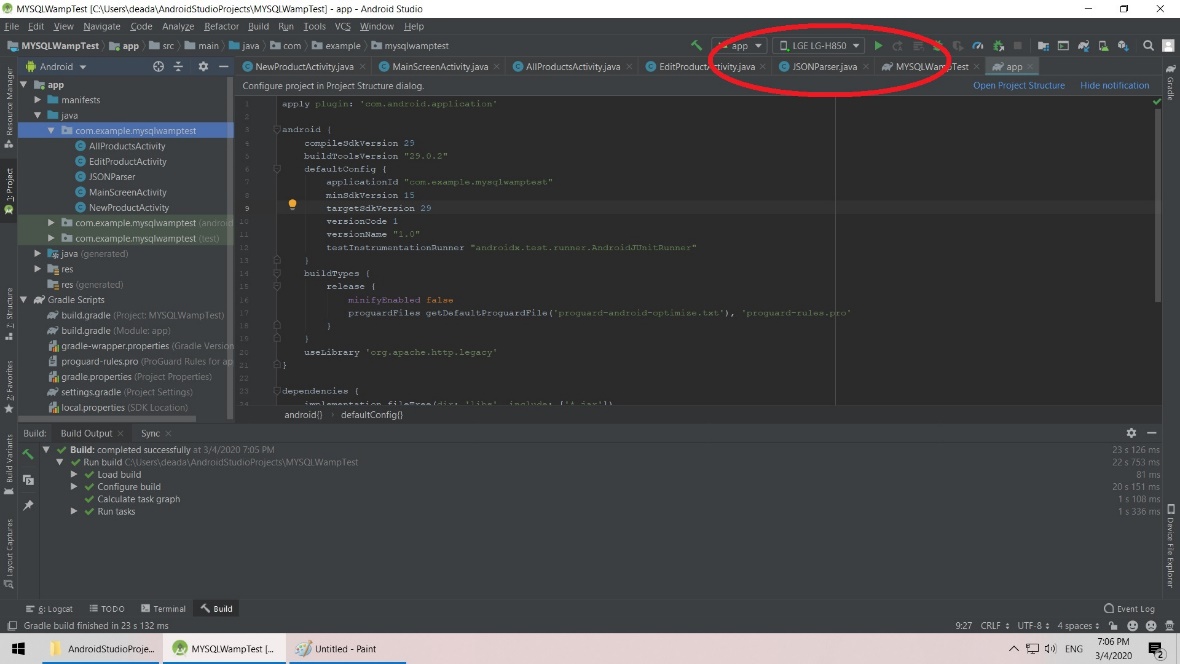


If performed correctly a countdown should begin with each press resulting in a message confirming you to be a developer and giving a new subcategory in settings for development tools.



In the Developer options menu, the function ‘USB debugging’ needs to be enabled

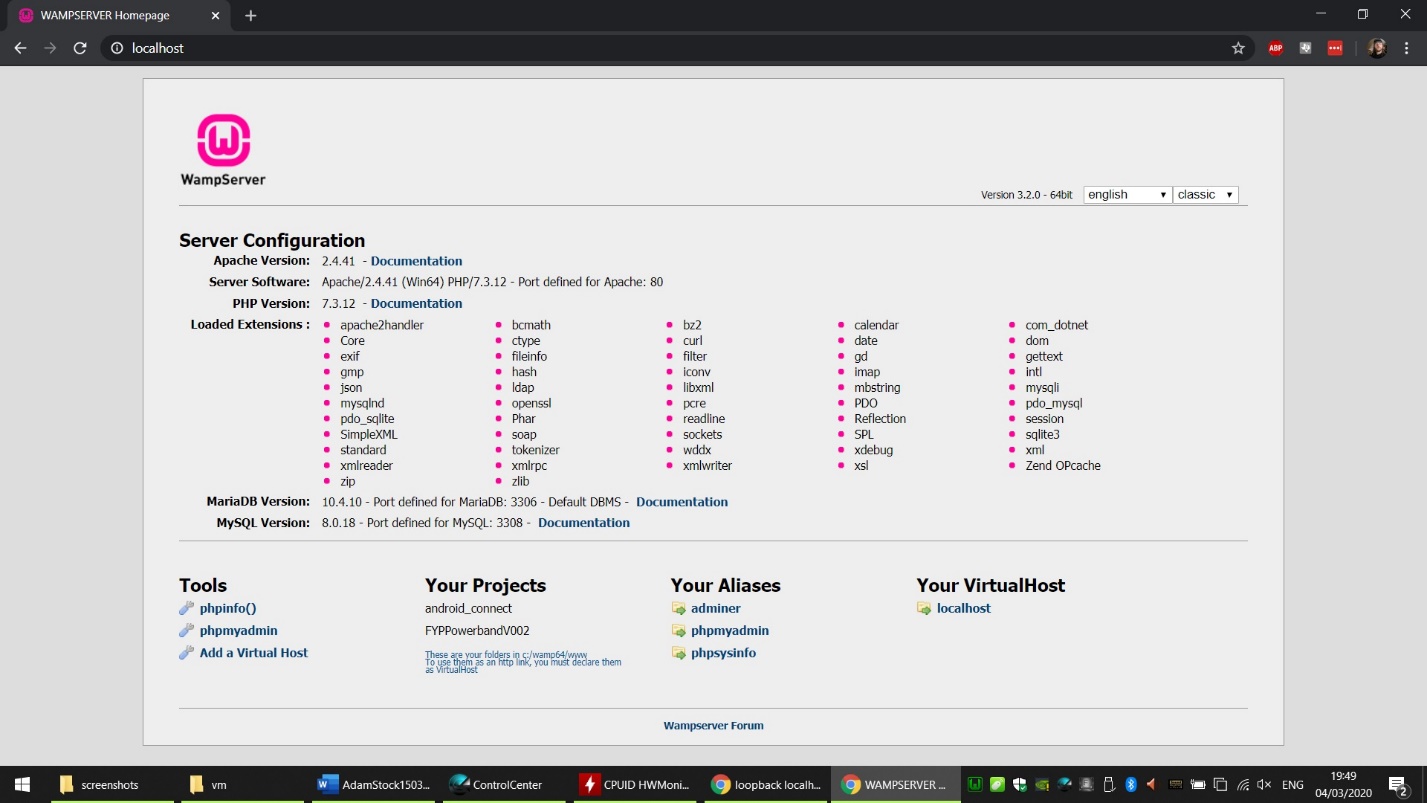


Once this is done the device can be connected to a machine running Android Studio and should appear as a deployable test device. 

For purposes of testing the IDE and its debugging tools this worked with the application being deployed to a test device (LG G5 mobile phone).

INSERT SCREENSHOT OF PROGRAM ON DEVICE

Next a MySQL server was created using the WampServer software stack available [here](http://www.wampserver.com/en/). Do you really need to supply a link to where you get it from? In order to achieve this the software should be downloaded onto a PC on the same network as you. Once installed opening a browser and directing it to the Localhost via the loopback function – commonly found by using the IP address 127.0.0.1 will bring you to the main controls for Wamp.



From this the phpMyAdmin link was pressed, this asked us to log in and select what type of database is to be used, in our case we will use MySQL rather than MariaDB and the default username and password combination. At this point it is worth checking that both the computer and mobile device being used are on the same subnet and to edit the httpd.conf file in Wamp to enable connections from other IP addresses. This is accomplished by editing the listen lines (in this case line 70-71) to the following –

Listen 0.0.0.0:80

Listen [::0]:80

This will allow connections from port 80 from other IP addresses. Doing this at this stage will significantly cut down troubleshooting later on.

After doing this the Wampserver should be restarted before proceeding.

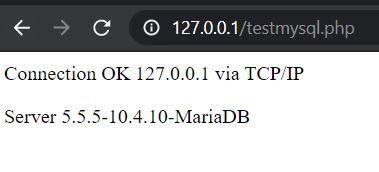
From phpMyAdmin the ‘databases’ link should be clicked and a new database created. For this project a database called fyppowerbandXXX will be used (the XXX to be replaced by a version number). In this database a table called users is created with the fields –

|  |  |
| --- | --- |
| ID | This is the primary key and as such is also the unique number for each user |
| Forename | User forename |
| Surname | User surname |
| Bandstatus | This is a number to be used as a Boolean to enable/disable the band |
| Created\_at | For testing |
| Updated\_at | For testing |

Once this is created the ability to connect to it should be tested, to do this a php file included with Wamp is used. This is accomplished by typing the following into a web browser–

IP ADDRESS OF WAMP/testmysql.php

And should provide this output –



Note the database type is not relevant here as this just shows the network is allowing communication.

### App creation and CRUD

Now that the background components are tested and working the initial iteration of the Android application can be created. From following the aforementioned tutorial, it was found that it contained various issues related to components being deprecated