KV6002 - Team Project and Professionalism

Evaluation Report

Liam Brand

Contents

0.1 The System Produced

The overall project aim was to develop a prototype system that involves several sensors monitoring an environment, and then using the data acquired from this for analytics and visualization through an appropriate medium. This section aims to explore the implemented features with regards to how well they were implemented according to various criteria.

0.1.1 Fitness for Purpose

Subsystems were successfully implemented to meet the overall project goal outlined earlier. Two sensors were developed that posted data to a database, and a front end containing appropriate functionality for data visualization and analysis was created as well. The visualization and analysis subsystems were also made capable of communication with both devices, allowing the system to communicate with multiple environmental sensors. The physical demo was proof of the product's functionality.

As well, these elements have been implemented to be near real-time, as stored and processed data needs to be up to date for the system to be accurate. Analysing an hour old reading for example might not provide useful information as it's uncertain whether or not this reading represent the environment's current state. The sensors post their readings to the database every 15 seconds, allowing for the database to receive sensory data at a relatively quick pace. As well, XML http requests have been configured to fire every few seconds for relevant data visualization and data analysis methods, ensuring what is displayed or analysed is the most up to date information available and provides the most value to the user.

Adherence to industry standards was also achieved where possible. For example, the model features an ability to inform users of problematic temperature differences between sensors. When discussing temperature differences between rooms, IBACOS - a strategic partner of the US Department of Energy - notes that "Room-to-room temperature differences or floor-to-floor temperature differences should be no greater than 4 degrees Fahrenheit in the heating season"[4]. The model therefore used this as a guide for its temperature difference threshold, but unknowns such as the temperature location, potential for sensory noise and use of only one sensor rather than one for each room, so the threshold was increased to four degrees celsius to accommodate for these aspects.

0.1.2 Robustness

AWS Lambda is used, which features functionality to allow scaling[2] allowing the database to cope with additional device implementations. The devices themselves have features to cope with failures. It features two temperatures sensors with the second one being used if the first one fails, features in code to prevent

erroneous measurements. As well, the devices will automatically attempt to reestablish internet connection upon being disconnected, and they also feature a 32GB Micro-SD card as temporary storage for readings data should connection be lost. Web elements such as the model's code contains try/catch statements to handle exceptions such as empty readings from the database as well, and will print appropriate messages to the console to assist in troubleshooting should these issues occur.

0.1.3   Look and Feel

Petrie and Fraser[8] describe different issues that significantly impede users' ability to properly make use of a website. The primary offenders are complex pages, unclear navigation and poor colour contrast. The website's pages are simple, with large block elements for listed devices and clear headings denoting sections such as the model's device comparison results. Navigation is achieved via the use of a navbar, something seen often in websites and so it shouldn't be unfamiliar to users. The website's colour scheme is mainly dark blue and white, offering a good contrast between the background and elements like text. These aspects allow the system's interactive element to be familiar, easy to use and visually pleasing.

The device could look better however. It's primarily made up of breadboards with components wired into them. Whilst functional, it doesn't look very visually appealing. Should it be deployed industrially, a more aesthetically pleasing case to house the components should be developed.

0.1.4   Consistency

Subsystems were developed with other subsystems in mind. For example, the website was properly formatted to make space for the data visualization and intelligent model. These elements have clear areas of the web page reserved for them that allows them to blend into the rest of the website, and they were included in the media queries to ensure they scaled to different screen sizes. The database endpoints allow the developed sensors to post data to the database, and endpoints were also created to provide the visualization and model subsystems with the data that they needed to perform their functions. These things allowed subsystems to work together without anything feeling like it wasn't accommodated for.

0.1.5   Technical Evaluation

Employed technology was appropriate for each subsystem. The devices used sensors that gathered relevant data, the database allowed the creation of needed endpoints and included inbuilt scaling technology, and the website subsystem's usage of HTML and CSS allowed for a pleasant looking website. The use of JavaScript for the visualization and model subsystems allowed for the creation of automated HTTP requests that ran every few seconds. This allowed for the

regular retrieval of relevant data from the database which helped these systems be accurate as they always use the latest data available in the database, and as such the latest data from the sensor.

Code featured appropriate comments, indentations and method/variable names to maximize code readability. Proper functional decomposition was done where appropriate too. For example in the model, part of determining an ideal temperature was to look at temperature readings from the same time for previous days. For this, timestamps needed to be created so readings made around these times could be searched for and retrieved. Creation of these timestamps was initially done in the *checkTemperature* method, but it was later turned into a *getPreviousTimestamps* helper function to maximize readability and group together relevant code.

0.1.6   Non-Functional Requirements

Primary non-functional requirements concerned the website's usability and the rate at which data was retrieved from the sensors and processed. These aspects have already been covered in previous sections. There is also a requirement for the development of an economic model to help cost the intelligent model, which can be viewed in Appendix B.

0.2   Project Management, Process and Personal Achievement

0.2.1   Terms of Reference

The description of subsystem requirements aided development by providing an understanding of the scope, complexity and methods/technologies that were used for their implementation. For example, the model's specification helped bring up some early questions regarding its possible behavior (e.g. what to do if it's too hot) and this allowed the developer to know what needed to be understood before development could begin. Discussion of the project's testing could have used more depth however, and things like unit testing (and perhaps the respective testing frameworks for the different technologies being used) could have been mentioned.

0.2.2   Requirements and Design Documentation

For requirements, a questionnaire was used with the client, shown in Appendix E, as well as numerous client meetings to fine tune requirements outlined from initial talks and the questionnaire. This consistent communication allowed for a high level view of the requirements to be understood, followed by understanding the requirements' finer details. Other products on the market were also

reviewed, such as the ecobee4[5] smart thermostat. This allowed an understanding of what the market already provided, helping determine some possible features to make the product more competitive on the marketplace.

For design, wireframes for the website were created, shown in Appendix C. These helped in understanding how the website would look which helped development, and also allowed the data visualization and model subsystems to understand how their respective information should be output so that it can fit into the created website (e.g. how do we tell the user it's too hot in a way that displays well on the designed page?).

0.2.3 Time Management

Most time objectives were met, shown by the meeting minutes' satisfaction of a live end to end product by March. Development primarily finished mid April however rather than the start of April, so a more thorough understanding of task complexity and required development time is needed.

0.2.4 Configuration Management and Integration Strategies

Configuration and integration was primarily managed through the use of version control. A global repository was created with each group member added as a collaborator, and changes were committed and pushed to their own branches where appropriate. These branches were reviewed to determine their compatibility with existing code and merged into the master branch after. This ensured that all written code fit together without breaking other elements, and also provided a global repository accessible by each group member for the project's source code. Appendix A shows an excerpt of a git log, showing some of the commits, pull requests, and merges done to build the website.

Integration of subsystems was also discussed during meetings, for example the meeting on the 25th of March features an end to end prototype utilizing basic subsystem functionalities together. This enabled smaller iterative integrations rather than connecting everything all at once at the end and risking subsystems interfering with and breaking eachother.

0.2.5 Testing Strategies

Use of the project during code merges on git ensured that code updates didn't break the project, and a live end to end demo was conducted prior to the demonstration to ensure all features worked. A more thorough style of integration testing with a sophisticated testing plan would have been better however, and unit tests for specific code elements would have helped ensure system stability.

0.2.6   Group Leadership

No leader was elected among the group, this had issues in determining solid internal deadlines on different pieces of functionality. Likely this is one of the reasons that the project's time plan couldn't be properly adhered to.

0.2.7   Quality Planning

Group members inspected each others developed code to look for faults overlooked during initial development. These code reviews helped prevent tunnel vision where issues are repeatedly overlooked. To increase efficiency, notes could be made from these code reviews and published in a group folder for others to see, so they can check their own code during development for similar mistakes.

0.2.8   Potential Additional Functionality

As of right now the visualization and analysis subsystems are hard coded to only retrieve data from the two created devices, moving forward this should be made more dynamic to accommodate for additional created devices to allow these subsystems to function properly as more devices are added. In addition, the data visualization could be scaled up to display data from longer periods of time (e.g. months or years). Also, the data analysis is limited in what it does with the information it learns. For example if gas or an extremely high temperature is detected in a reading this manifests through website alerts, but a fully realised product could notify the appropriate services automatically to help the user. Important notifications could also be harder to miss, sending them to the users phone as a regular mobile notification. These aspects may have been achievable with better time management.

0.2.9   Problems Encountered, Their Solutions, and Lessons Learned

Better time management could have been achieved from sticking to the project plan, in future more attention will be paid to these time plans to prevent falling behind too much. Also, a better idea of whether or not to have a leader will allow more discipline in sticking to self imposed deadlined, so in future it would be better to explicitly discuss whether or not the group will have a leader and if so, who it is.

0.3   Professional Issues

The Code of Conduct was adhered to with good collaboration between teammates, as each member showed up to the group meetings and gave fair warning if other commitments would cause them to be late. Members were polite whilst still offering fair criticism. Moving forward these standards would need to be strictly adhered to, as insufficient communication between team members could

cause project issues that directly affect clients causing a loss of faith in the company and as such the developed product.

0.4   Legal Issues

Data breaches against stored data were considered during the TOR. The usage of AWS helped mitigate this, with their databases featuring implemented security measured[1] such as encryption and private keys required for endpoint access helping ensure the stored data is private. The repository containing the project's source code was also private with read and write privileges only given to group members, to prevent source code leaks. Terms of use such as copyright conditions were also adhered to, for example the AWS service terms[3] were reviewed against proposed project functionality to ensure what we were using them for was acceptable by Amazon. These terms included things like ensuring the only data stored on the database was lawfully obtained by us, and these terms would need to be kept in mind once other users come into the picture should the product be deployed industrially.

As well, the Data Protection Act[7] will need to be considered should the project be deployed. In particular, the Act states that a person has the right to find out what data relating to a person is stored by an organisation. Facilities to provide customers with this data if they request it should be set up. It also states data shouldn't be kept for longer than is needed, so it should be determined how long sensory data is required for the system's ideal function and then functionality should be in place to ensure data that is no longer needed is properly disposed of.

0.5   Social Issues

The major social concern with this product noted in the TOR is the perception of it from the public. Devices inside of a person's house sending information about its status can make many uncomfortable. During their study into the reasoning of users behind using or not using smart sensors and their opinions on them, Lau et al[6] have a participant who says "with smart speakers...I can avoid having it, so I'd really not, I don't want another thing that could possibly violate my privacy". This was difficult to mitigate during development as industrial deployment of the product never happened, so it was impossible to determine how users felt about it being in their home.

Should the product be deployed, the company needs to have complete transparency in how the sensors behave, what they store about people's homes, how this data is stored and how it is processed to provide system functionality such as the short term decision making. This could be in the form of an FAQ on the website, as well as a complete willingness to answer more specific question should users contact the company. Being open will help dissuade any negative

perceptions of the sensors as customers will fully understand what the product is doing and why.

0.6 Ethical Issues

The main ethical concern identified for the project was misuse of sensor date. This was mitigated during development as to retrieve sensory data special keys are required, and these keys were only given to developers who needed certain endpoints for database access. This meant everyone only accessed the endpoints (and as such, the data) that they needed. With regards to the database itself, access was mainly restricted to the subsystem's developer.

Should the product be deployed this policy will need to continue to be adhered to so that only those who actually need data will have access to it. If a larger company was established, data privilege levels should be assigned to employees that dictate what they have access to. These levels will be based on what data the employee will need access to in order to perform their job.

Bibliography

[1] Amazon. Aws database documentation, . Available at: `https://aws.amazon.com/products/databases/` (Accessed 02/05/19).

[2] Amazon. Aws lambda documentation, . URL: `https://docs.aws.amazon.com/lambda/latest/dg/scaling.html`.

[3] Amazon. Aws service terms, . Available at: `https://aws.amazon.com/service-terms/` (Accessed 02/05/19).

[4] Arlan Burdick. Advanced strategy guideline. air distribution basics and duct design. Technical report, IBACOS, 2011.

[5] ecobee. ecobee4 smart thermostat. Viewable at: `https://www.ecobee.com/ecobee4/` (Accessed 02/05/19).

[6] Josephine Lau, Benjamin Zimmerman, and Florian Schaub. Alexa, are you listening?: Privacy perceptions, concerns and privacy-seeking behaviors with smart speakers. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):102, 2018.

[7] Government of the United Kingdom. Data protection act 2018, 2018. Viewable at: `https://www.gov.uk/data-protection` (Accessed 02/05/19).

[8] Helen Petrie, Fraser Hamilton, and Neil King. Tension, what tension?: Website accessibility and visual design. In *Proceedings of the 2004 international cross-disciplinary workshop on Web accessibility (W4A)*, pages 13–18. ACM, 2004.

Appendix A

Git Log Excerpt

```
commit 5043823dd50f80c7d9fae0940121880f289ab523
Author: liambrand <liam.brand@northumbria.ac.uk>
Date:   Wed Apr 10 13:49:59 2019 +0100

    Basic model working

commit c56a98297993f9fac9de92603913f2297fae98ba
Merge: 764706f 998f3a8
Author: SnowTurtle96 <jg.clarke@hotmail.co.uk>
Date:   Wed Apr 10 11:47:13 2019 +0100

    Merge pull request #3 from SnowTurtle96/branch-apicall

    Layout change, api-call issue

commit 998f3a861d61bdf0673fb79dfe88b60de9aba922
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date:   Wed Apr 10 11:44:01 2019 +0100

    Layout change, api-call issue

commit 764706f3c5270b6e65170fef13e52c1e7ef763cc
Author: liambrand <liam.brand@northumbria.ac.uk>
Date:   Wed Apr 10 11:14:55 2019 +0100

    Moved script includes to header.php

commit 92d46488d5d63eb4eabcee60a25bfc7a70f5b99c
Author: SARABIRDS <under863@gmail.com>
Date:   Wed Apr 10 02:11:27 2019 +0100
```

initial commit

commit 0c81167490a0035269786dc66ed1e99e419476a1
Merge: 34bc207 419ab61
Author: SnowTurtle96 <jg.clarke@hotmail.co.uk>
Date: Wed Apr 10 01:00:43 2019 +0100

    Merge pull request #2 from SnowTurtle96/branch-apicall

    Branch apicall

commit 34bc2076cd62293fc3cf750ff1549d793d300cf5
Merge: 3945797 6d37c6d
Author: SnowTurtle96 <jg.clarke@hotmail.co.uk>
Date: Wed Apr 10 01:00:23 2019 +0100

    Merge pull request #1 from SnowTurtle96/model

    Model

commit 3945797b0f5b89bcec6f22fdc6583e70e81a8e4f
Author: Jamie <jg.clarke@hotmai.co.uk>
Date: Wed Apr 10 00:54:36 2019 +0100

    lambdas

commit 419ab619b006aa7cdc4f28f93a820933239e8722
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date: Wed Apr 10 00:45:11 2019 +0100

    Start of user device calling - broken

commit 9e45e746014855001083f4980e4e755c9fc9a594
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date: Mon Apr 8 23:11:35 2019 +0100

    Device entry fix

commit 60be8dbbde2454c9fadb8a122b8a7d6d0df18abf
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date: Mon Apr 8 22:52:44 2019 +0100

    Device creation fix

commit 22bf7c646c9953e108ee8325a36155b95dcd0fbe
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>

```
Date:   Mon Apr 8 22:21:17 2019 +0100

    Login functionality complete

commit d2b285a4ff59f965f7c57ff3b9190d2a461e52b1
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date:   Mon Apr 8 01:28:28 2019 +0100

    Login validation check

    To fix

commit c3017e18eb7d6c34b03f8bcd5abd9120969e1676
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date:   Sun Apr 7 23:52:26 2019 +0100

    API for new device, plus form added

commit 5ac1c169450f8bba7e68e3e18cab66737b355d79
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date:   Sun Apr 7 18:44:06 2019 +0100

    API fix

commit a4e14c146c1efb4b8464c6196feec3b3f6ad4d2d
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date:   Sun Apr 7 15:01:08 2019 +0100

    apiCall function updates, addition of login GET

    To fix

commit 0a73ec96aca445665d81f01cd49b336a97813bdf
Author: chris-mordue-nu <chris.mordue@northumbria.ac.uk>
Date:   Fri Apr 5 03:24:21 2019 +0100

    API-Call helper file

    Almost complete helper file for API call, data passing other than login/return.

    issue:
    fname=test&lname=test&email=tst%40test.c&pass=ttt&login=&return=

commit d71ea224d9d4af0cfe0cecd879a990bd9e85f832
Author: ANewby <adam_newby1@hotmail.co.uk>
Date:   Thu Apr 4 23:46:24 2019 +0100
```

Charts added with two dropdowns

commit 6d37c6d5f8d3655aad54eee13da8ecd118e41cfb
Author: liambrand <liam.brand@northumbria.ac.uk>
Date: Thu Apr 4 20:30:35 2019 +0100

        Merged model.js into device-model for now

commit 25798897952dd58808026fabb2bc4fefb46db44e
Author: liambrand <liam.brand@northumbria.ac.uk>
Date: Thu Apr 4 20:16:08 2019 +0100

        Added model files

commit 093b2270a54113ea4514dd034c4c7f5c165e7ddf
Author: Jamie <jg.clarke@hotmai.co.uk>
Date: Thu Apr 4 13:29:10 2019 +0100

        new folder structure, Chris's web work

commit 6d4b051a8121a5e77554fbf1ce1710b5cdcb58ca
Author: Jamie <jg.clarke@hotmai.co.uk>
Date: Tue Apr 2 15:14:19 2019 +0100

        Initial Commit

Appendix B

Economic Model

Economic Breakdown for the Short-Term Model

As part of the model's development, the system's implementation needs to be financially evaluated to determine the cost of its deployment on an industrial level. The Terms of Reference outlines the project's main finances, but here will be a more in-depth exploration of the financial aspects behind the model specifically. There will also be a brief look at the potential finances behind developing a more sophisticated model.

Current Model

If the model was to continue being used in its current form, this would involve it continuing to retrieve device data from the system's database via the use of an endpoint. This endpoint takes the user's id then returns all associated device information, allowing it to be searched for relevant information that can be used for the short-term decision-making process.

The primary cost consideration here would be the continued and increasing usage of AWS Lambda, an Amazon service that automatically runs associated code for various events. In the case of our system, the event is when an http endpoint is triggered. Usage of Lambda remained free during the prototype stage thanks to the relatively low amount of requests and data use. Moving forward however, possible costs will need to be considered. Lambda's usage is based on two things, requests and compute time. Requests pertains to simple requests that call Lambda code, and compute time is the time it takes for the code to execute and terminate. The costs (shown below in dollars) can be viewed on AWS' Lambda page.

Free Tier	Requests	Duration
1M REQUESTS	**1M REQUESTS FREE**	**400,000 GB-SECONDS PER MONTH FREE**
per month	*First 1M requests per month are free.*	*First 400,000 GB-seconds per month, up to 3.2M seconds of compute time, are free.*
400,000 GB-SECONDS	**$0.20 PER 1M REQUESTS**	**$0.00001667 FOR EVERY GB-SECOND USED THEREAFTER**
of compute time per month.	*$0.0000002 per request.*	*The price depends on the amount of memory you allocate to your function.*
The Lambda free tier does not automatically expire at the end of your 12 month AWS Free Tier term, but is available to both existing and new AWS customers indefinitely.		

Requests

Lambda offers the first million requests each month for free. First how many users the system in its current implementation can support whilst remaining in this free tier will be calculated, then a rough cost per user in request fees will be determined.

The model sends a request to the endpoint that returns device data approximately every four seconds. To be completely prepared for financial ramifications, the most intensive scenario will be assumed where the model runs constantly and is at no point deactivated for maintenance or updates.

With 86400 seconds being in a day, scheduled retrieval of a single user's data will use 21600 (86400/4) requests each day. That means over the course of 30 days, a single user's model will use 648000 requests. Dividing a million by this figure yields a little over 1.5, meaning in its current state

the system can only support one user full time without incurring fees. Beyond this point, requests would need to be paid for.

Lambda offers 1 million requests for $0.20 (AWS Lambda, 2018), which is equivalent to £0.15 at the time of writing. We can work out the cost of a single request by diving this cost by 1 million, then by multiplying this by the number of requests needed for a single user's model we can determine how much in requests a single user will cost. By doing this, we arrive at the figure of £0.0972 per user. So, for each user that uses the product that the current model needs to accommodate for, an additional £0.0972 per month will need to be spent on requests from AWS Lambda.

Compute Time

Lambda offers 400,000 gb-seconds per month free per month. First it will be determined how many users can be supporter by the system in its current implementation without incurring costs. Then a cost per user in compute time fees will be determined.

The screenshot below shows the Lambda details for the endpoint that returns user device data once supplied with a user ID.

Summary	
Code SHA-256	Request ID
L/Crb7QNEnnkRqOrh9YgOIzK8blEJloHoZ3DUkOkjBU=	abbd03b6-f521-48ef-9fb5-f04a4b12afaa
Duration	Billed duration
25.19 ms	100 ms
Resources configured	Max memory used
128 MB	55 MB

The memory used by the endpoint doesn't always equal 55mb, but to ensure that the worst-case scenario has been prepared for it will be assumed that every request uses this much data. This amount of data can increase if more device data is returned but this increase is ultimately negligible, nearly all the computing time used is from processing the respective scripts and code that is used for the endpoint's function rather than the actual bandwidth used returning the data.

400,000 GB-seconds per month equates to around 400000000 megabytes. This is enough to process around 7272727 requests. It was earlier established that a single user will (in a worst-case scenario) require 648000 requests to retrieve up to date model information. This means that from a free perspective, the current system's usage of Lambda can support around 11 users (7272727/648000 is about 11) before costs are incurred from the usage of the endpoints. After this point, a single user's model requests will use around 35640000 megabytes of compute-time, equating to around $0.59 in monthly costs. At the time of writing, this translates into around £0.45.

In summary, the free tier will be exceeded by just two users. Even though Lambda's memory allowance offers support for many more, the sheer number of requests required to keep the model up to date burns through the free allowance quite quickly. Only the respective free allowances have been exceeded, each user will cost £0.0972 in request fees and £0.45 in compute-time fees, giving a total per-user cost of roughly £0.54.

Improved Model

Rather than continuing with the current model's implementation, a more sophisticated model could be created that takes advantage of true deep learning technologies. This will allow its behaviour to be more accurate, as it will be capable of picking up on small nuances in user behaviour and distinguishing them from real problems. For example if a user frequently has parties at certain times

of the year, the current implementation might flag this activity as a high temperature whereas a true machine learned model might recognise it and understand that it isn't a problem.

AWS

AWS offers technologies capable of deep learning. Whilst its lack of prior use in the project makes it difficult to estimate the cost per user as was done with the current model's implementation, instance pricing can still be reviewed.

The three main rentable instances in order of power are P2, P3 and G3 (AWS Deep Learning Developer Guide, 2018). To begin with, it would be best to start with the cheapest option. This would prevent money being wasted should this foray into deep learning prove to not be useful, but should more power be required it will be easy to scale up compared to wasting money and realizing what was purchased was overkill.

P2 instances are categorised by their specifications, with more expensive instances containing additional GPUs, vCPUs and more RAM, among other things. The closest P2 instances that can be obtained are based in Ireland, below is a pricing breakdown taken from the EC2 instance documentation of how much these instances would cost.

p2.xlarge	4	12	61 GiB	EBS Only	$0.972 per Hour
p2.8xlarge	32	94	488 GiB	EBS Only	$7.776 per Hour
p2.16xlarge	64	188	768 GiB	EBS Only	$15.552 per Hour

How much this will cost is something that can only be properly determined once they have been used, as how many computing hours are needed to train the model is unknown. As well, the model may not be usable during its training, but frequent training will likely be needed in order to keep up with user habits as they change and nuances occur. Two different instances could be obtained as a work around for this. As one model trains, the other performs the necessary short term decision making. Once this first model is done training it takes over and the second model begins training. This could be repeated frequently to allow for the models to both stay up to date and current, but alternating deep learning instances would mean this could happen with any service interruption.

Sources

- Amazon. AWS Lambda Documentation. Available at: https://aws.amazon.com/lambda/ (Accessed (03/05/19).
- Amazon. Recommended Deep Learning Instances. Available at: https://docs.aws.amazon.com/dlami/latest/devguide/gpu.html (Accessed (03/05/19).
- Amazon. EC2 Pricing. Available at: https://aws.amazon.com/ec2/pricing/ (Accessed 03/05/19).
- Amazon. P2 Instance Details. Available at: https://aws.amazon.com/ec2/instance-types/p2/ (Accessed 03/05/19).

Appendix C

Wireframes

http://hermann

First Name:

First name

Email:

email@email.com

Password:

****

Re-enter Password:

****

☑ I'm not a robot

Continue

Already registered?

http://hermann

Devices   Summary   Sensors   Support   Account   Logout

My Account

Save
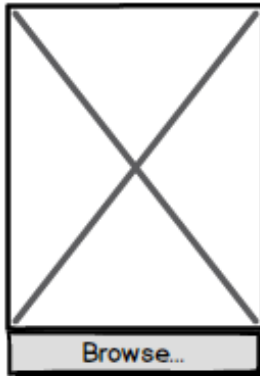
My account details

Email: email@email.com

Current password: **********

New password: **********

My custom alert values

Temperature   ##   ✓

Humidity   ##   ✓

Light Level   ##   ✓

☐ Detection of CO2

☐ Detection of Dangerous Gas

Browse...

Email Preferences

Email me with:

☐ daily reports   ☐ when a reading is above recommended values   ☐ when a reading is above my custom alert value

☐ weekly reports   ☐ when a reading is below recommended values   ☐ when a reading is below my custom alert value

☐ monthly reports

Devices   Summary   Sensors   Support   Account   Logout

Configure your device

Add
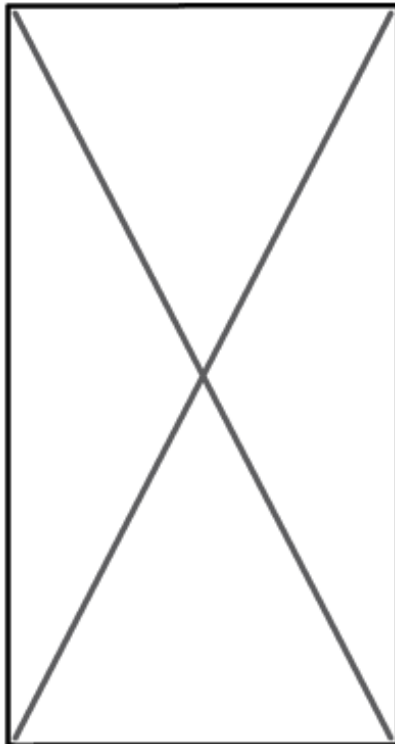
Select your device:

Device#0000 ▼

Provide a name:

Where will this be located?:

Location list ▼

Alert me when:

☐ the temperature is above my preferred level

☐ the temperature is below my preferred level

☐ carbon dioxide has been detected

☐ dangerous gases have been detected

Provide a report:

☐ daily

☐ weekly

☐ monthly

☐ never

http://hermann

Devices   Summary   Sensors   Support   Account   Logout

My Devices

Configure Device

Device name
Location

View

Device name
Location

View

Device name
Location

View

http://hermann

Devices   Summary   Sensors   Support   Account   Logout

Your sensor readings

Filters (by device or sensor) ▼ ▼

TEMPERATURE

LIGHT LEVEL
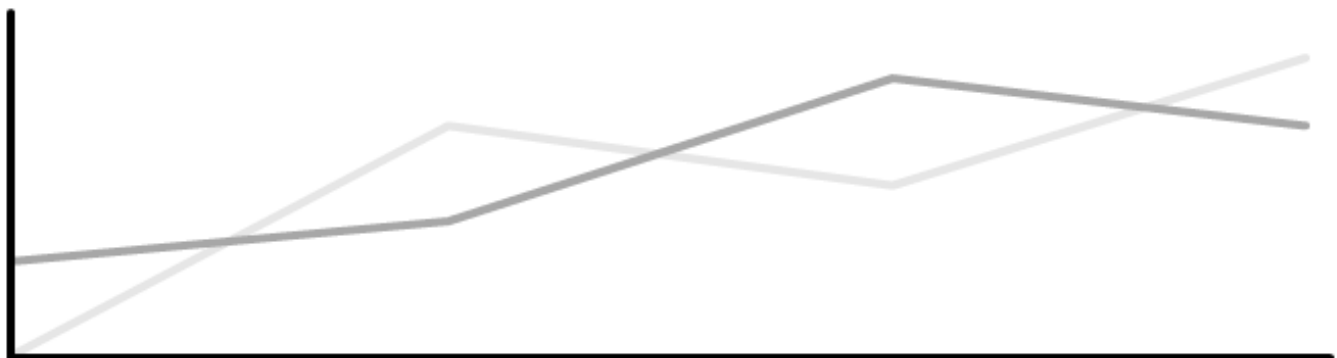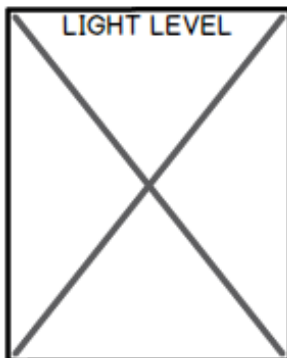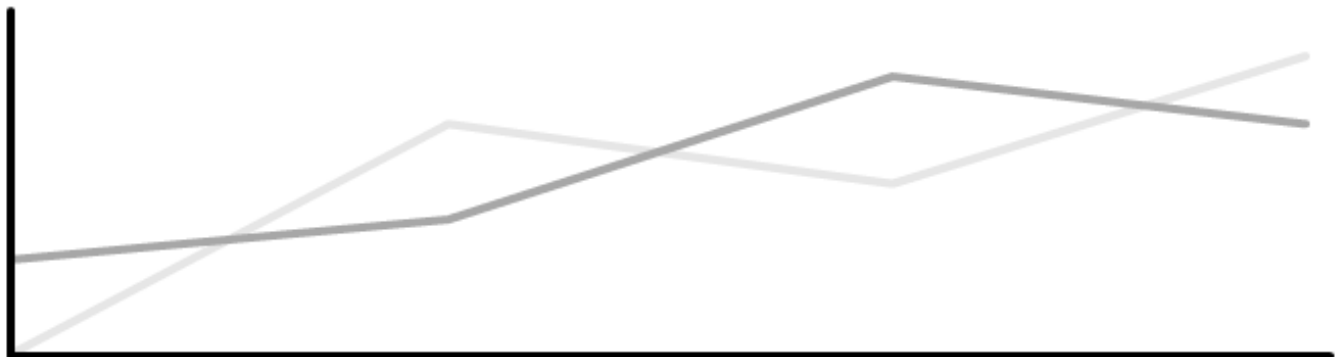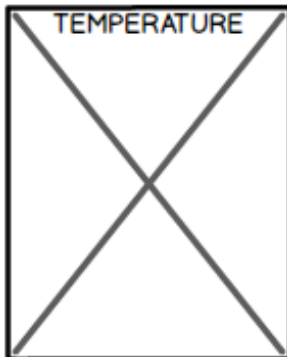
http://hermann

Devices   Summary   Sensors   Support   Account   Logout

Summary View: Hermann1   **Export Summary**

Readings: | Historic Views | ▼ |

Device Configuration

Device name:
[   ]

Device location:
| Location list   ▼ |

Provide a report:
☐ daily
☐ weekly
☐ monthly
☐ never

Alert me when:
☐ the temperature is above my preferred level
☐ the temperature is below my preferred level
☐ carbon dioxide has been detected
☐ dangerous gases have been detected

Save

‹ Back

Devices   Summary   Sensors   Support   Account   Logout

Product Support

Enter your search term...

🔍 search

I can't find my device	View

I can't view my reports	View

I can't view my device readings	View

http://hermann

email@email.com

****

☑ **I'm not a robot**

Login

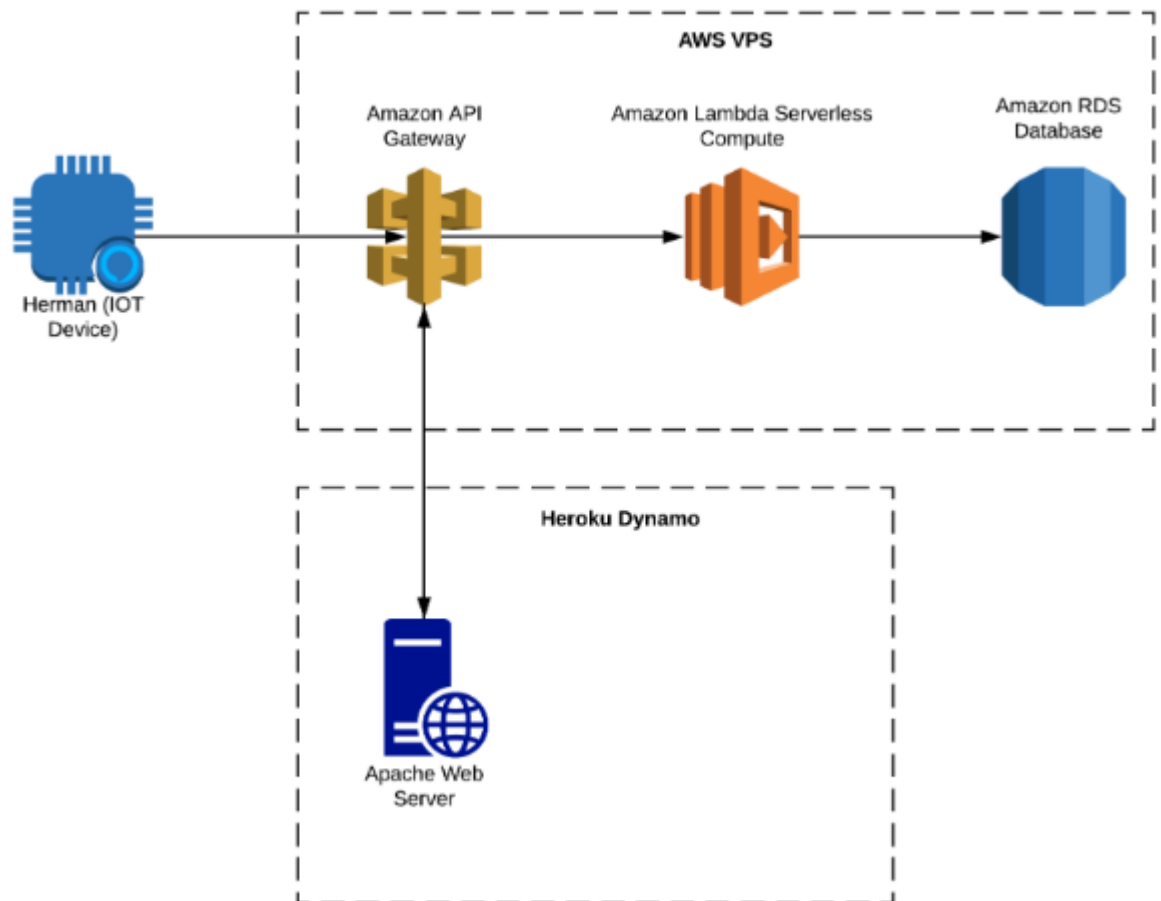Need to register?

Appendix D

Infrastructure Diagram

Appendix E

Client Questionnaire

Hermann Requirements Gathering

Sub-system 1 (IoT Sensors)**:**

1. What data would you expect to be recorded by the sensor? Prioritise the following suggestions for device data recording, with 1 being the most important data, to 6 as the least important data.

Sensor Data Recording	Priority (1-6)
Humidity	4
Temperature	2
Dangerous Gases (Carbon Monoxide and Flammable Gases)	1
Carbon Dioxide	1
Infrared	?
Light	6

Sub-system 2 (Webserver/DB):

1. What level of scalability do you require for the platform?

☐ HIGH (x)   ☐ MED   ☐ LOW

2. What level of availability do you require for the platform?

☐ HIGH (X)   ☐ MED   ☐ LOW

Sub-system 3 (Web Application):

1. What functionality do you anticipate using in a web-application for this product? Prioritise the following suggestions in order of importance where 1 is the most important and 6 is the least important to you.

Functionality	Priority (1-6)
Varying levels of privileges for device management and viewing?	4
Quick summary view reporting for devices?	5
Account management functionality?	3
Export functionality for reports to Word, Excel or other?	6
IoT device management?	2
Accessible from a range of devices?	1

Sub-system 4 (Data Representation):

1. How would you like the data to be visualised?

☐ Line Graphs (x)   Other _____

2. Apart from seeing timed events such as daily, weekly, monthly and annually, what else would you like to see?

A   summary   over   certain   time   frames

_____

_____

_____

Sub-system 5 (Short-term Decision Making):

1. How, if at all, would you like to be notified of notable sensor events? (select all that apply)
   ☐ I wouldn't   ☐ Email   ☐ Website Popup   ☐ Phone Notification(x)   ☐ Other

   If other, please specify: _____

Appendix F

Terms of Reference

TERMS OF REFERENCE

KV6002

Project Tutor: Dr Huseyin Seker
NORTHUMBRIA UNIVERSITY

JAMIE CLARKE, ADAM NEWBY, LIAM BRAND, ZAK ATKINSON, CHRIS MORDUE

Contents

Vision of the Project

Scope

Our project is the development of an IoT system used for the intelligent monitoring of an environment. The monitoring aspect will be handled by sensors relative to what we want to monitor in the environment (e.g. temperature) and the intelligent aspect will come from the system's ability to make intelligent short-term decisions in order to benefit the environment and the people within it (e.g. raise the heating.

Background

The ability to remotely monitor home and work environments is becoming increasingly common, and not without reason. Health and safety are improved massively due to the ability to see when certain elements of the environment could cause issues with people. The system would also produce financial benefits, as short-term decision making may allow elements such as temperature to be properly maintained to help reduce heat waste and save money. Given that the system uses IoT for multiple sensors, it may also be able to show where certain aspects of the environment are performing poorly. For example, one room might have a fair bit lower temperature than the others which could point to poor insulation.

Objectives

To help create the system and achieve these benefits the system has been broken down into a series of objectives.

Group Responsibility

As a group we need to ensure we deliver a system composed of multiple devices, able to measure multiple factors of an environment using sensors as well as being able to communicate with each other. This data should be stored in a database, where it can be retrieved either for use in predicting behavior and making intelligent short-term decisions or for visualizing and displaying to the client(s). We must also consider financial, legal, ethical and social issues with the system. Finally, it must be tested to ensure it works.

Individual Responsibilities

- **Selection and Deployment of IoT Sensors – Zachary Atkinson**
  This involves research into the system's core hardware requirements and a selection of appropriate and affordable sensors relevant to these requirements. Sensors should be properly deployed so that they are capable of measurements and they should also be able to communicate with relevant software to allow for the storage/visualization of the measurements they take.

- **Development of Webserver and Database – Jamie Clarke**
  A webserver capable of communicating with the sensors, as well as collecting and storing their data needs to be developed. Factors such as what type of database to use, big data stream, HCI and UI issues as well as legal/ethical concerns such as data accessibility, data privacy and data protection also need to be considered. Ideally the webserver and database should be able to accommodate for multiple sensors and multiple clients.

- **Development of Web Application – Chris Mordue**
  The web application needs to be capable of communication with the database. It must display necessary information and text to the client. HCI and UI issues should be considered, as well as factors relating to data accessibility, data privacy and data protection.

- **Data Representation and Visualisation – Adam Newby**
  The appropriate data analytics method for data representation and visualization needs to be identified, with this method being developed to display continuous sensor measurement data so that the client is informed of device activities. HCI and UI issues should be considered, as well as factors relating to data accessibility, data privacy and data protection. Ideally comparisons could be made between different sets of data in different rooms (e.g. room temperatures over time to see the temperature differences in different rooms). This data could also be displayed as daily, weekly, monthly or annual summaries of sensor data (e.g. temperature over the course of a year with weekly intervals).

- **Development of Intelligent Method for Short-Term Decision Making – Liam Brand**
  The appropriate method needs to be identified for gathering data from the sensors, with this data being used to make a predictive model for short-term decision making by learning users' activities from a single sensor. Ideally it would learn from users based on multiple sensors rather than a single one, and potentially an economic model could be drawn up to help cost this digital system.

Team System Specification

Source of Requirements

To gather the requirements for the common elements we will look at doing interviews and questionnaires to our clients. We have identified our clients as ourselves; Zachary Atkinson, Liam Brand, Jamie Clarke, Christopher Mordue and Adam Newby. Another client we will have is the Project Supervisor; Huseyin Seker. As Clients we will hope to represent an unbiased view and be able to make informed decisions. The clients we have chosen represent the generic stakeholders who this system will be aimed at. The clients have already agreed to participate in this study.

As mentioned before the stakeholders will help us gather the initial requirements. Secondly to further understand the industry which our product will be entering and get a grasp of existing products currently on the market, we will research into these existing products. We will be looking at IoT products, so the connection between multiple home devices as sharing of data is common between our product and this type. Secondly, we will be looking at Hive. Hive is a service that provides a range of smart home devices, these devices include: lights, thermostat and many other sensors. all of these devices are controlled and monitored through mobile application and their website. Hive is a direct competitor to our product as it provides an exact service, that we wish to offer.

The group will analyse the above to further identify any further set of requirements which we can deem useful to our stakeholders. This will include research into finding out what the existing customer base likes and dislikes of this service and looking at reviews online and elsewhere.

Requirements specification for common elements

Remote monitoring of home and work environments are becoming more common, and useful in several aspects to lead smart homes and cities. Systems can improve health and safety, reduce energy use and waste amongst many other benefits, all made possible through the deployment of thousands of sensors, which is called Internet of Things. Systems that are developed are becoming increasingly complex due to the wide range of sensors in our environments. Intelligent data analytics tools are developed to support sensors to support short/long-term decision making.

The aim of this project is to develop a prototype, deploying several sensors, to evaluate the feasibility of using a compact IoT system in a home or work environment.

Below is a list of the 5 high level top priority requirements for the group:

- Selection and deployment of IoT sensors
- Development of Webserver and Database
- Development of Web Application
- Development of data analytics: Data representation and visualisation
- Development of intelligent method for short-term decision making

Subsystem Requirements

Subsystem 1: Selection and Deployment of IoT Sensors (Zachary Atkinson)

Scope

The scope for sub-system one is the selection and deployment of IOT sensors. The sensors will capture a range of data to record the rooms environment. The data to be collected by the sensor needs to be researched. It will need to be data useful to most everyday users. This means only capturing data non-scientific users will know about like carbon dioxide and temperature.

The sensor device needs to be as compact and small as possible. This will make the device more appealing to users as it will be less noticeable. This will also reduce the cost, a very important factor as it will encourage the user to buy our product and increases the chance of the user buying more units to place in more rooms of the environment.

Technology

This sub-system will be implemented via a raspberry pi. I researched using an ATMEGA chip however after adding basics such as SD card reader, power, networking and EPROM, it would be a lot more efficient for the time scale to use a raspberry pi of some kind. Though the raspberry pi 3 has the most computing power, the raspberry pi zero W has all the functionality needed as well as adequate computing power for the requirements. This also reduces the form factor and overall cost quite considerably. This also removes the need to design and create a PCB.

The software to run the sensor and communicate with the server will either be written in python or Java. It is far easier to interface with the hardware on and connected to the raspberry pi via python. However, Java is a more common language and make interfacing with the server easier. Overall, I'm more versed in Java than python so on that basis I will be proceeding with Java.

The sensors used will be –
        MQ-9 Sensor – Detects flammable and carbon monoxide gas.
        MQ-135 Sensor – Records carbon dioxide gas.
        DHT11 Module – Records the temperature and humidity of the air.
        LDR – Records light level.

Requirements

<u>Temperature</u> – The sensor should be able to take the temperature of the surrounding air. This will be useful in many ways for the user. It will be useful at night to see if a room isn't heated adequately. Or potentially if the user is out of the environment, they can check the sensor online and see they left the heating on, costing them money. Another major benefit to the user would be room comparisons. The user can have a sensor in two or more rooms then compare the temperature difference. This can highlight to a user an open window, an open door or even poor insulation.

<u>Humidity</u> – Humidity is also a very useful piece of data to collect on an environment. Living or working in a high humidity environment runs the risk of growing mould and bacteria, the former being very expensive to remove from an environment. Generally, causes stuffy and uncomfortable conditions. Whereas if the humidity is to low, you run the risk of catching a cold, dry itchy skin and damages wood and painted walls.

Carbon Dioxide – Every household should have a carbon dioxide (CO2) detector. By UK law all landlords must have one in their properties. There is CO2 in the air, it's created from cars, animals and even us breathing. However, a sudden rapid rise in an environment is likely due to the fire nearby. CO2 and temperature sensors can their fore help detect a fire in progress or forming. This is extremely important as it allows people in the environment to be alerted to the fire, so it can be put out with less risk to life and reduce overall damage.

Dangerous Gases – The main dangerous gasses that can be found in a home are carbon dioxide, carbon monoxide and flammable gas. Carbon dioxide is usually caused by a fire and can impede escape by blocking line of sight, hurting eyes and making it harder to breath. Carbon monoxide usually caused by faulty boilers is an odourless, invisible gas. This gas is exposed to enough can kill. Lastly flammable gasses, if ignited by a hot enough heat source or a flame can ignite causing a fire or explosion. Carbon dioxide is already being captured by a CO2 sensor. Carbon monoxide and flammable gasses can be detected by a MQ-9 sensor.

IOT – The sensor needs to be able to upload the data gathered using the internet to a server. To aid the development and future updates to the product, it would be best if the communication software was written in a modern language. This means it will be compatible with most modern servers that we will likely be using for the product as well as any updates we make to the server.

Must –
- The hardware must have a sensor to record room temperature.
- The hardware must have a sensor to record humidity.
- The hardware must have a sensor to record CO2.
- The hardware must be able to access the internet to upload recorded data.

Should –
- The hardware should have a sensor to record dangerous gases.

Could –
- The hardware could have a sensor to detect Infrared radiation.
- The hardware could have a buzzer to act as an alarm.
- The hardware could have an RGB LED to convey the sensor status.

Subsystem 2: Development of Webserver and Database (Jamie Clarke)

This subcomponent is concerned with developing and deploying the backend infrastructure for the IOT project. The first aim of this subcomponent is deploying a webserver that can be accessible by other group members allowing them to deploy the code written for the website in subcomponent three. Requirements for this will be mainly be gathered from other group members on the development team using interviews and a questionnaire querying what functionality will be required. The development team will be the main drivers of requirements regarding the web server due to it not being business facing and having a smaller impact on the client. The client will however be polled using interviews to decide the level of scalability they desire for the system, this will determine whether the system will be hosted on scalable cloud-based infrastructure such as AWS or onsite servers provided by the university in the case of no scalability being required. The questionnaire given to clients will be aimed at gathering information on topics such as the desired level of scalability for the website and analytics tools along with polling the level of concurrent users that should be able to be supported. Requirements will also be gathered by looking at similar products on the market and understanding what sort of infrastructure they use looking at the pros and cons of self-hosted versus cloud-based infrastructure. There will also be research done into the different types of webservers available and a determination made on which one will be most suitable to the needs of the client.

The database is the second part of the subsystem. The job of the database will be to receive information from many IOT devices simulations and store that information for use and retrieval by website. The database will be the heart of the system storing all user credentials and IOT device data. Requirements will be gathered for the database on topics such as data retention, specifically how long sensor data should be retained for use within the analytics and predictive subsystems. Options will be provided to clients along with the costs associated with them. Along with data retention database security will be covered asking the client which if any fields within the data model should be encrypted and which data within the system should be treated as sensitive. A secondary way requirement will be gathered for this area is a literature review on the different types of databases available both open source and licensed.

The final section of this subsystem is the API gateway. The API gateway allows database methods to be exposed to the outside world indirectly. This will allow the IOT devices to connect to our central database without having to provide each user of the system with a plaintext password to the groups infrastructure which would be a massive security risk. The API gateway will be used by the IOT devices to connect to the database and send the readings that they have gathered. The API gateway will also allow the webserver/website to read data from the database that has been posted by the IOT devices. These requests for data will be what drives subsystems 3, 4 and 5. Requirements for the API gateway will be gathered indirectly from client requirements from the website, data analytics and the model for short term decision making. For example, if a user wanted the data analytics to show data from a longer time period then this would be an indirect requirement for the API gateway as a new database query would need to be generated.

Technology
After reviewing available hosting platforms on the market along with in house infrastructure offered by Northumbria it made sense to opt for an elastic cloud solution due to our big data needs. Opting to host with a provider such as AWS will allow us infinite scalability and the power of the cloud behind our big data analytics and our predictive modelling. As customers increase our infrastructure will be able to elastically scale to meet demand.

The technologies that will be used are as follows.

- AWS – Lambda
  "AWS Lambda lets you run code without provisioning or managing servers. You pay only for the compute time you consume - there is no charge when your code is not running" (Amazon Web Services, Inc., 2019).

- AWS – RDS (MySQL)
  "RDS makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups" (Amazon Web Services, Inc., 2019).

- AWS – API Gateway (Python)
  "Fully managed service that makes it easy for developers to publish, maintain, monitor, secure, and operate APIs at any scale. It's a pay-as-you-go service that takes care of all of the undifferentiated heavy lifting involved in securely and reliably running APIs at scale" (Amazon Web Services, Inc., 2019).

In 'Above the Clouds: A Berkeley View of Cloud Computing' fox states "The illusion of infinite computing resources available on demand, thereby eliminating the need for Cloud Computing users to plan far ahead for provisioning" along with "The elimination of an up-front commitment by Cloud users, thereby allowing companies to start small and increase hardware resources only when there is an increase in their needs". (Fox, 2009). As we are developing a project from infancy the ability to increase hardware resources only as our needs increase will be invaluable and keep project costs down.

High Level Requirements
This subcomponent has been broken down into the following requirements in order of importance.

- The webserver must be able to communicate with the sensors also being able to effectively collect and monitor the data they give out.
- The webserver must be capable of deploying the (developed) data analytics methods.
- The database must be capable of storing the data given by the imbedded device, it must also be able to label each sensor and client.
- Big data, HCI and UI must be considered during the development of both the web server and database.
- Data accessibility privacy and protection must be considered during design and development.
- The system should be developed so that it can accommodate both multiple sensors per embedded device.
- The system could be developed so that it supports multiple clients within the database

Subsystem 3: Development of Web Application (Chris Mordue)

Project Scope

The scope of work to deliver the third sub-system as part of the development a compact IoT device delivers a web application to the client for device management and the display of sensor data, which can be accessed from any device. This development will provide a central location view and monitor all sensor data that is being recorded by the device. Products that are currently available on the market, such as Hive and Nest, offer a similar solution to providing their customers with mobile access to sensor data. The client should be able to create an account to manage the device and view sensor data, however, it should also be possible for multiple accounts to be registered, so that the device can be expanded on to other clients that want to make use of the sensor. It should also be possible to add more than one sensor to the web application, and, have the option to view data from each device individually.

To develop a product that meets the requirements of the client, the sub-system must deliver upon several objectives, including:

- To meet the client expectations of the system, requirements have been gathered and documented below, through the project vision identified by the client and research in existing products.
- To meet client requirements, design documentation including wireframes and use case diagrams will be created.
- A critical self-evaluation of the sub-system will be produced, which will review what could be done differently for future developments and offer suggestions for how the sub-system could be expanded on in future versions of the web application.

It is becoming more common to have the ability to monitor a home or work environment with a sensors. By delivering a web application as part of the product, it will bring several benefits to the usability and experience of the product for the customer, including:

- Access to the sensor data with a clear and concise representation of information
- Access to the sensor data from any location, on any device
- Configurability of the sensor settings provided
- Improved client experience when using the device

The sub-system will contribute to the overall success of the product, if it delivers a solution which is cost-effective for the client and can be scalable to larger audiences in the future. The product should consider all other sub-systems which form the complete device and be developed with future expansion of the product in mind.

Technology

The sub-system will be implemented using commonly found technologies which are specifically in place to develop dynamic web applications. Standard HTML and CSS will be used be used to develop all web pages across the website.

Requirements Gathering

Several requirements have been gathered to create a web application which meets the client and industry expectation for compact IoT devices. A set of requirements have been pre-defined in the vision of the project documentation, which was created with the client at the beginning of the project.

To further understand industries that produce sensors, research into existing products has taken place to identify a further set of requirements from consumers of IoT products.

The Internet of Things is a connection between various types of devices, such as smartphones and computers, which communicate between each other as well as people. Internet of Things is commonly found in home automation, but, other examples can also be found in healthcare, transportation and infrastructure. It is the connection of *things* and the sharing of data which creates the Internet of Things.

Hive is a service that provides a range of smart home devices, including; thermostats, lights, smart plugs, motion sensors, window sensors and door sensors. All devices are accessible through a mobile application and the Hive website – which provides configuration options for controlling all smart devices. The Hive application has also been expanded to the Apple Watch device, now offering easier access to managing smart devices in the customers home. Nest is a service which provides a similar range of products to Hive. Nest offers a range of varied products, which include; thermostats, smoke detectors and security systems (such as smart doorbells and locks). Similarly, Nest offers a mobile application and web-solution for customers to access and manage their smart home devices.

Formal documentation of the requirements for this system can be found below, which have been split into three separate *epic* pieces of functionality. An *epic* is large parts of functionality that can be developed as smaller, more manageable items and delivered as part of a sprint of work to the client for review. Within each *epic* is a list of **MUST**, **SHOULD** and **COULD** requirements. The requirements have been prioritised using the MoSCoW prioritisation technique, which groups requirements by rating. Generally, to deliver a minimum viable product, all **must** requirements are satisfied as part of the build of the web application.

Epics

1. Access, registration and login functionality
2. IoT device management (add/remove/configure)
3. View current/historic device readings

Functional and Non-functional Requirements

Epic 1: Access, registration, login functionality

- The web application **must** provide account registration/login facilities to view device readings.
- The web application **should** provide two-factor authentication when logging in.
- The web application **should** be accessible from a mobile device.

Epic 2: IoT device management (add/remove/configure)

- The web application **must** provide options for adding or removing a device in a local area.
- The web application **could** provide options for device configuration, such as disabling a sensor.

Epic 3: View current/historic device readings

- The web application **must** show current readings from the device.
- The web application **must** show historic readings from the device.
- The web application **should** provide reading comparisons between current and historic data.

The web application **could** provide advanced alerting, in the form of email or SMS, for readings which are above recommended levels.

Subsystem 4: Data Representation and Visualisation (Adam Newby)

Project Scope

The subcomponent which will be my main focus, will be the Development of data analytics and how best to represent such data in a visual manner which will be appropriate for the client and the needs of the application. This main objective will be split into 5 sub objectives which all must be considered and combated appropriately. The 5 sub objectives which will be discussed are:

- The appropriate data analytics method best suited for data representation and visualisation
- Development of appropriate data analytics method to display continuous measurements from sensors
- Considerations of big data streams, Human Computer Interaction (HCI) and User Interface (UI) while also looking at data accessibility, privacy and protection.
- The development of data analytics methods to be able to compare and display the outcome of multiple sensors.
- Development of summary data to display daily, weekly, monthly and annual summary of data.

The way the requirements will be gathered for this subcomponent will be through a questionnaires and interviews with the clients. The following two strategies will provide the necessary information to build an extensive overview of requirements which will cater the client.

The questions that will be asked to the client will be catered around the 5 sub-objectives mentioned before. An example of this will be asking the client about what would be their preferred way of to view their data, what graph would be most suitable. Also how often would they like updates from the data and for their history as well, would they like to see daily, weekly, monthly and annual summaries while also wanting to see outside sources for example temperature the over the last 3 months inside then the outside temperature for that season so the user could possibly see a correlation.

Once the clients have been interviewed we will collate all of our results and see if you we can find must, could and should requirements from them. Another way of find requirements will be by looking at similar IOT applications on the market and the current industry standard for data representation. This will outline what products in the market have in common and what the team and I have to do to make a successful product.

Technology

The technology we will be using for this, will be using: HTML, CSS and JavaScript. The HTML and CSS will be for the design and the shell that will host the graphs. The JavaScript will be for the actual display of data, this will be done by using a library such as ChartistJS or Google Charts. The data will be sent from the database in a JSON format. APIs will also be used to gather information from third party sources for example the MET office to get the outside temperature.

Must Requirements

- The development of data analytics methods to be able to compare and display the outcome of multiple sensors.
- Development of summary data to display daily, weekly, monthly and annual summary of data.
- Development of appropriate data analytics method to display continuous measurements from sensors

Should Requirements

- The appropriate data analytics method best suited for data representation and visualisation

Could Requirements

- Considerations of big data streams, Human Computer Interaction (HCI) and User Interface (UI) while also looking at data accessibility, privacy and protection.

Subsystem 5: Intelligent Short-term Decision Making (Liam Brand)

The subsystem I am most involved in is one that allows our product to have intelligent methods that allow it to make short-term problem-solving decisions.

For example, if the sensors are monitoring temperature, they could encounter a few of these different scenarios. If the room is too cold or too warm, how can our product deal with this? How can it deal with erratic behaviour? If the sensor is constantly spiking up and down there could be a few different problems with this. It may be that the sensor is in a part of the room causing this or it could be a hardware fault with the sensor itself. Even if the temperature is within a safe limit, what if there is a sudden massive increase or decrease in the temperature? This could be caused by something innocent such as a door being left open, or an emergency such as a fire. These are just a few problems with possible causes, but the idea is that our product can interpret when some of these events happen and act accordingly. In the case of the fire detection for example it may try to alert the emergency services, or in the example of temperature being too low it may automatically turn the heating up. Similar devices already exist, one of the most prominent smart thermostats is a device called the ecobee4 by ecobee (ACM). This device can monitor temperatures and respond to voice commands to perform actions such as changing the temperature, however these devices do not feature the additional sensors that our device uses, nor are they capable of intelligent short-term decision making. This subsystem will allow our product to not only monitor useful information for the customer, but also to help them deal with it.

Technology

Relevant sensor information will be stored in a web server which will be accessible using HTTP requests. Following the retrieval of this information, it will be processed by a context-aware program which will take appropriate action. Context-awareness here refers to the ability for the application to adapt itself to the context of its user(s) , in this case the context refers to environmental statuses such as temperature and humidity (Huebscher, M.C. and McCann, J.A., 2004). The program in question will likely take the form of some simple scripting on JavaScript accompanying the front end, which will look at data retrieved from the web server and act accordingly based on it. There is also the possibility of a script acting as middleware between the web server and the front end (possibly in Python) that takes care of this processing and simply passes it to the front end.

Gaining Requirements

The requirements will be gained from the use of client interviews and potentially questionnaires, as well as being impacted by the sensor itself. What the sensor measures will affect what kind of values decisions need to be made about, so if temperature is measured we need to make sure we take the appropriate actions for different temperature scenarios. The interview and questions will primarily concern the reactions to the various environmental conditions that the sensor will detect. To refer to the previously stated potential scenarios, what would customers want to happen if it was too cold or

warm? If there's a risk of a fire do they instantly want the system to alert the emergency authorities or do they want a local alert only to the residents first? If it's too cold do they want the heating automatically turned up or should the product simply prompt the user as a suggestion? A client interview would help to answer these questions and ascertain the desired behaviour of the product. These methods will attempt to understand what it is that might be desired from the product we are attempting to make. From this initial understanding we will look at what we could viably implement. Some requirements and potential features that we gain through these methods could be very difficult to create, so even if they sound like a good addition we need to be sure we can properly implement them within the scope of our constraints (mainly time and cost).

Following this there will be a decision made on what seems like a good idea to implement. After this decision is made the requirements of the subsystem could be further fine-tuned by some additional client interviews. In them it would be ensured that what has been decided on as the features to implement is satisfactory to the client and perhaps any slight tweaks, changes or caveats they wish to discuss would be noted down to keep in mind for when development begins.

Subcomponent Specification – Requirement Specification

Requirements for the subcomponents have been broken down into being either a 'must', a 'should' or a 'could' based on a few different factors such as difficulty, time constraints and how ultimately important a component is for the overall function of the subsystem.

- **Must** search and identify appropriate predictive method for sensor data mining
- **Must** consider data stream mining
- **Must** develop a predictive model for short-term decision making based on a single sensor by learning from users' activities
- **Should** develop a predictive model for short-term decision making based on multiple sensors by learning from users' activities and past sensor data
- **Could** develop an economic model to help cost this digital system

The first three requirements are deemed the highest priority because they are vital for the functioning of the subcomponent. For our product to be capable of short-term decision making it needs to appropriately gather information from a sensor, use it to learn something and then act on what it has learned. The ability for it to do this on a larger scale through learning from multiple sensors would be a very good addition to the system but is ultimately not vital to get a form of short-term decision making. The economic model doesn't contribute to system functionality at all, hence why it is of the least importance in the list.

Project Tasks and Deliverables

This section discusses the project management approach to successfully complete the build of the final product, as well as project management techniques such as project planning and risk management. Resources for the project have also been identified, such as the hardware requirements and technologies used for development. A commentary has been provided on each project phase has also been included to provide clarity on how each stage of the project will be approached.

The agreed project management methodology for the project is an agile approach to the hardware and software development. This focuses on continuous review and feedback of the project cost, time and scope, as well as the overall quality of the product after each sprint. The aim is that this iterative approach to the development cycle will help develop a product which is not only fit for purpose but is also of high build quality. The benefit of using this approach is that sub-systems identified earlier in this document can be developed in parallel, with large pieces of functionality being complete for client review.

A project risk log has been created to track and monitor all identified risks to the project. The risk log will be reviewed on a weekly basis and updated as a team throughout the project during meetings. It is important that risks are tracked so that the appropriate mitigation can be applied to reduce the impact or likelihood of a risk to the project occurring. Additionally, a project plan has been created to ensure the project remains on track for completion in April 2019. As with the project risk log, this will be reviewed at each meeting and development stage to ensure that all tasks are being completed in line with project timescales.

Early identification of project resource requirements is a crucial step in ensuring the successful development of the product. This early identification of requirements has ensured that all hardware components will be in place for the build of the product, and that the project team work cohesively across a range of sub-systems which should form one complete product. A range of technologies will be used to develop the product, including:

- Python
- Java
- HTML, CSS, PHP
- JavaScript
- SQL

The table below lists all hardware and software resources required for the project.

Hardware	Software
- PI - Power cable - Temperature/Humidity Sensor - LDR - MQ-9/MQ-135 - TRI LED - Buzzer - PIR - Case - Cables - Micro SD Card	- Brackets - GitHub Desktop - Office Applications - BlueJ - Eclipse - Visual Studio

As mentioned, a list of project phases has been included below, as well as the artefacts that should be delivered as the outcome of each project stage.

- **Requirements Capture**
  Requirements capture will take place with the client, as well as between group members to identify the requirements for each sub-system. The initial client requirements were gathered during the project vision stage, identifying a set of high-level requirements per sub-system that was to be developed. Interviews between each group member, where a sub-system relies upon another to function, will be conducted to capture unique functional sub-system requirements.

- **Analysis & Design**
  Through the analysis and design section, research into existing products will take place and several documents will be produced. Deliverables in this phase will all be agreed upon by team members and the client before proceeding into build.

- **Systems Build**
  The main section of work to be completed is the system build, separated into five unique sub-components of work. Each sub-component should be built with integration in mind, utilising the same (or similar) technologies which are suitable for the development of the project, and in line with other technologies used in other sub-systems. Following the mid-point review, a code review of each sub-system will take place and integration will commence to tie all sub-systems together.

- **Testing/Configuration/Delivery**
  Following successful integration, the product will undergo performance and user testing. Performance testing will ensure that data from the sensor is being sent to the web-server and displayed within good time, showing accurate results to the product user. User testing will take place with the client for the project, ensuring that it meets the requirements specification documented within this terms of reference.

Project Plan

Project Plan Gantt chart / timeline

Start — 21 Jan

✔ Code of Conduct — 31 Jan

✔ Project Vision — 31 Jan

Terms of Reference — 11 Feb - 25 Feb

Today — 24 Feb '19

Terms of Reference — 25 Feb

Sub-component 1 - Hardware — 25 Feb - 01 Apr

Sub-component 2 - Webserver and Database — 25 Feb - 22 Mar

Sub-component 3 - Web Application — 25 Feb - 01 Apr

Sub-component 4 - Data Representation and Visualisation — 25 Feb - 25 Mar

Sub-component 5 - Intelligent Short-Term Decision Making — 25 Feb - 12 Apr

Sub-component 2 MVP — 12 Mar

Sub-component 4 MVP — 12 Mar

Sub-component 3 MVP — 14 Mar

Sub-component 1 MVP — 18 Mar

Mid-point Review — 18 Mar

Sub-component 5 MVP — 22 Mar

Integration — 01 Apr - 05 Apr

Testing — 08 Apr - 12 Apr

Project Submission — 25 Apr

Finish — 29 Apr

Timeline axis: 27 Jan '19, 03 Feb '19, 10 Feb '19, 17 Feb '19, 24 Feb '19, 03 Mar '19, 10 Mar '19, 17 Mar '19, 24 Mar '19, 31 Mar '19, 07 Apr '19, 14 Apr '19, 21 Apr '19, 28 Apr '19

16

Legal, Social, Ethical and Professional Considerations

Below is a document that identifies potential dilemma within legal, social, ethical and professional dimension. The dilemma has been separated into the categories on the left-hand side of the table and then an impact rating and probability rating of that dilemma occurring and what effect that would have.

	Impact	Probability
Ethical		
Misuse of User Data	H	L
Legal		
Data Breaches	H	L
Data handling (GDPR)	H	L
Trademark violations	H	L
Software licensing violations	M	L
Social		
In-house IOT concerns	H	H
Professional		
Loss of team member	H	L
Overbudget	L	M
Timescale overran	N/A	L
Inability to deliver client requirements	L	M

Ethical

The first category where concerns can be identified is ethical. Two concerns have been listed in the table above with the impact they may have on the project and the likeliness that this dilemma will happen. The only concern identified here is the misuse of user data. Due to the nature of the project and sensor reading and data being collected from within a user's home there is an ethical dilemma that this data will not be misused in any way. Levels of access to data should be restricted from employees depending on the sensitivity of it with some data only being accessible by the user who owns the device and the sensors.

Legal

Data breaches are a primary legal concern to the project. This is a concern for any project or company that retains user data. This is exacerbated by the fact that this project deals with big data and thus legally the stakes are a lot higher than other projects. The problem of data breaches will also have to be treat even more seriously since we are dealing with big data from IOT sensors within people's houses. The impact rating for this issue is high due to the product becoming unsellable if the company's reputation is damaged. In 'The effect of data breaches on shareholder wealth' Gatzlaff states "the frequency of such incidents has been increasing over time and can result in significant costs for the affected firm". This statement demonstrates the importance of keeping user data safe is a company aims to be financially successful and maintain good public opinion (Gatzlaff, 2010).

Trademark violations are a legal concern that the team will need to address but has a low probability of being an issue. When doing our market research, we will need to ensure that the system that we are creating does not infringe upon competitor's intellectual property and that we do not violate any trademarks that other companies have. Although this is a low probability dilemma the impact if this was to occur would be high as it could affect both timeline and budget along with redesigning our applications. Software licensing violations is the final dilemma that we may encounter under the umbrella of legal problems. Software licensing issues may occur if we don't not take the time to plan what software will be needed and what sorts of costs need to be set aside to pay for this software depending on the plan and license, we require.

Social

The only social problem identified that may affect the problem negatively is the public perception of the technologies involved with the problem. Technologies such as IOT and big data especially when implemented into an in-home solution will be met with concerns from the public. The company/project will have to be established in such a way that it is demonstrated to customers that the company can be trusted with user data and has a good track record of maintaining user privacy. The impact of this dilemma was rated as high due to the fact that selling units will be impossible in the current climate if the public does not trust the device and also high probability due to recent breaches with other big data companies. Although this is the highest rated concern it should be noted that competitor's devices are still selling well. An example of a competitor's device that ran into this social issue if Amazons popular Alexa range. Lau makes the following statement in her paper Privacy Perceptions, Concerns and Privacy-seeking Behaviours with Smart Speakers, "As expected, privacy emerged as a major deterring factor for smart speaker adoption". Lau also made the following statement "They believed in the sanctity of their home and expected their private home affairs to remain private". In turn smart device adoption may be hindered within the home due to this issue of public mistrust (Zimmerman, 2018).

Professional

The professional category was a main source of potential issues for the project. The first identified was loss of a team member. This could take form in many ways such as illness or unforeseen circumstances that do not allow a team member to continue working on the project. This issue has largely been mitigated by dividing the project into standalone subsystems that can be presented independently. Developing the project in this manner will allow us to suffer the loss of a team member without a great effect on the final project, if for example the person developing the web server was indisposed, we could host the other sub systems locally and make use of dummy data.

The project running overbudget was a second area identified as a professional risk. This may occur if the requirements are too broad and not attainable or tasks were under estimated. To mitigate this risk, we have created Gantt charts that map our time accurately and allow for contingency time if certain tasks overrun. The chance of this occurring was rated as moderate due to the use of must, should and could. It's very likely that objectives listed only as 'could' will not be completed due to time constraints. Timescale being overrun has the same mitigations being taken as the project going overbudget. This will be avoided by strict project planning using Gannt charts and breaking tasks down into digestible objectives using the terms must should and could. This will set expectations for what the client can expect from an early date. The final professional dilemma that may arise is lack of technical ability for the project team to deliver the set objectives. This has been given an impact rating of low due use of project management tools and planning time for each task including time for research if skills set need development. Along with this client expectations have been manged using must, should and could objectives.

Project Costing

Serviced office space

We have looked at serviced offices in the Newcastle area that are good value but most importantly are equipped for our business needs. The office found that fulfils the businesses needs at the best price was *"Waterloo Square" (Flexioffices.co.uk, 2019).* Waterloo square has all the usual serviced office facility's such as a reception, meetings rooms, office space, 24-hour access and a telephone system. However, it has many added benefits for our IT company like fibre internet, air conditioning, disabled access, CAT 6 cabling and more. Prices start from £145 per person meaning renting an office for 5 people per month will be £725 (£145 X 5). If we divide this by the average amount of days in a month (30), we find the cost to hire the office for a day which is (£24.16).

Business insurance

It has been very difficult to get a specific quote for business insurance. Instead I have searched for averages that other people have paid. While doing so I came across "Boughtbymany" who have done the leg work for me.

[Public liability prices

It's difficult to provide examples of public liability costs for different jobs because there are so many factors that influence quotes. But here are some starting prices and cover levels:

- Axa says 10% of customers paid £59 or less. It offers up to £10m of cover.

- Direct Line says 10% of customers paid £54 or less. It offers up to £10m of cover.

- Hiscox says the starting price of its insurance is £55, based on 10% of their customers. It offers up to £10m of cover.

- PolicyBee does not provide starting prices but a quote for a fashion consultant was £43.80 a year. It offers up to £5m of cover.**]**
(Bodenham, 2018)

Since being a fashion consultant company is vastly different from IT, we can remove the PolicyBee quote. This leaves us with Axa, Direct Line and Hiscox all offering £10m of cover. We can find the average of the quotes (£59, £54, £55) which gives us an average of (£56) per month. We can multiple the monthly cost by 12 then divide by 365 to get the daily cost which is (£1.84).

Total

The total running costs per day for the business is £26. We got to this total by adding the daily office rental cost of £24.16 then adding the daily insurance cost of £1.84.

On-Cost Employee Wages

Living Wage

To find out how the on-cost of an employee we first need to figure out the hourly rate for that employee. All employees will be paid the same rate. This is because no member of the team is more important over another. We will also pay employees a real living wage. This is a wage that is based on what <u>people</u> need to live in the UK. *"The current real living wage is £9.00 an hour (not including London)" (Livingwage.org.uk, 2019).*

National Insurance

All our employees are national insurance category "A" and since all our employees work full time and earn £9.00 an hour, they are all national insurance class "1".

National Insurance class	Who pays
Class 1	Employees earning more than £162 a week and under State Pension age - they're automatically deducted by your employer
Class 1A or 1B	Employers pay these directly on their employee's expenses or benefits
Class 2	Self-employed people - you do not have to pay if you earn less than £6,205 a year (but you can choose to pay voluntary contributions)
Class 3	Voluntary contributions - you can pay them to fill or avoid gaps in your National Insurance record
Class 4	Self-employed people earning profits over £8,424 a year

Category letter	Employee group
A	All employees apart from those in groups B, C, J, H, M and Z in this table
B	Married women and widows entitled to pay reduced National Insurance
C	Employees over the State Pension age
J	Employees who can defer National Insurance because they're already paying it in another job
H	Apprentice under 25
M	Employees under 21
Z	Employees under 21 who can defer National Insurance because they're already paying it in another job

(GOV.UK, 2019)   **(GOV.UK, 2019)**

We can use the employee national insurance rates table to calculate the amount of national insurance the employer and employee needs to pay. Below is a copy of the table for the employee and employer.

Category letter	£116 to £162 a week (£503 to £702 a month)	£162.01 to £892 a week (£702.01 to £3,863 a month)	Over £892 a week (£3,863 a month)
A	0%	12%	2%
B	0%	5.85%	2%
C	N/A	N/A	N/A
H	0%	12%	2%
J	0%	2%	2%
M	0%	12%	2%
Z	0%	2%	2%

Category letter	£116 to £162 a week (£503 to £702 a month)	£162.01 to £892 a week (£702.01 to £3,863 a month)	Over £892 a week (£3,863 a month)
A	0%	13.8%	13.8%
B	0%	13.8%	13.8%
C	0%	13.8%	13.8%
H	0%	0%	13.8%
J	0%	13.8%	13.8%
M	0%	0%	13.8%
Z	0%	0%	13.8%

(GOV.UK, 2019)   **(GOV.UK, 2019)**

Working full time means working 35 hours or more in a single week. So, working 9am to 5pm (8 hours) during weekdays is a total of 40 hours. By including a 1-hour lunch break (7 work hours) that will be a 35-work hour week. We multiply their living wage by the number of hours (£9.00 X 35) to get the employees weekly pay (£315).

Employee – The employer will deduct the tax from the employees pay and make their payment on their behalf. The employee pays 0% tax on the first £162. The next £730 will be taxed by 12%, so the remaining £153 is taxed at 12% (£18.36) to leave £134.64. So, the un taxed £162 plus the before 12% taxed £153 now £134.64 after tax is equal to £296.64. So, the employee's weekly wage after national insurance is £296.64 with them being taxed £18.36. We can divide this (£18.36) by 5 to get the daily tax (£3.67).

Before tax – £315 weekly pay
After tax –   £296.64 weekly pay
Paid –   £18.36 weekly tax
Paid –   £3.67 daily tax

<u>Employer</u> – The employer will also pay towards national insurance depending on how much each employee earns. The employer pays 0% tax on the first £162. The next £730 will be taxed by 13.8%, so the remaining £153 is taxed at 13.8% (£21.11). This means the employer will pay a total of £21.11 in national insurance for the employee. We can divide this (£21.11) by 5 to get the daily tax (£4.22).

- Paid - £21.11 weekly tax
- Paid - £4.22 daily tax

Total

Working full time means working 35 hours or more in a single week. So, working 9am to 5pm (8 hours) during weekdays is a total of 40 hours. By including a 1-hour lunch break (7 work hours) that will be a 35-work hour week. We then take how many hours an employee will be working in a single day (7), Multiply that by the real living wage (£9.00) to find the cost per employee day before tax (£63). We don't need to do anything with the employee tax since that comes out of their wages. However, we do need to add the national insurance tax we calculated daily, this makes the on-cost wages for an employee per day (£67.22).

Day rate for each employee

We calculate the day rate for each employee using the following calculation *"Day Rate = (running costs per day/number of employees) + on-cost wages for an employee per day" (Conniss, 2019).* The running costs per day is £26. We divide this by the number of employees which is 5. This is equal to £5.20 which we add to the on-cost wages for an employee per day. So, we take the £5.20 and add the £67.22 on-cost wages of an employee to get £72.42. Since we all work the same number of hours and have the same pay this result will be the same for each employee.

Summary

Daily Rates
Employee: £72.42
Number of staff members: 5
Number of days required: 45
Total cost for staff time: £15,124.50

Project-specific costs

Electronic Components

Component	Unit Cost	Quantity	Total Cost	Best Source
Raspberry pi W zero	£12.29	3	£36.87	The PI hut
Power Cable	£3.99	3	£11.97	Amazon
Temperature & Humidity Sensor	£0.99	3	£2.97	Amazon
Light Resistor	£0.05	3	£0.15	Amazon
CO2 Sensor	£1.20	3	£3.60	Amazon
Dangerous Gas Sensor	£1.20	3	£3.60	Amazon
RGB LED	£0.10	3	£0.30	Amazon
Buzzer	£0.70	3	£2.10	Amazon
PIR	£0.97	3	£2.91	Amazon
Acrylic Project Box	£1.91	3	£5.73	B&Q
Solder Wires (20 pcs)	£2.00	3	£6.00	Amazon
Micro SD card 8GB (A)	£4.39	3	£13.17	Amazon
			£89.37	

Hardware

Hardware	Cost	Best Source
Soldering Iron	£59.99	Amazon
Solder	£2.71	Amazon
Dremel	£19.94	Amazon
Voltage Meter	£8.00	Amazon
Helping Hands & Magnifying Glass	£5.50	Amazon
	£96.14	

Software

Software	Type	Cost
Raspbian	OS	£0
Eclipse	IDE	£0
IDLE	IDE	£0
Putty	SSH Client	£0
FileZilla	FTP Client	£0
Java	JDE & JDK	£0
		£0

Other

Name	Type	Notes	Cost	Best Source
Domain Registration	Networking		£16.10	GoDaddy – Easier to restore lost domain and cheapest
Amazon RDS	Web Hosting	Free 10,000 requests per month	£0	Amazon
Amazon API Gateway	Web Hosting	Free 1,000,000 API calls per month	£0	Amazon
Amazon Lambda	Web Hosting	Free 1,000,000 API requests per month	£0	Amazon
			£16.10	

Overall project specific costs: £201.61

Cost per sensor: £29.79

<u>**Scalability –**</u> The cost per prototype sensor according to the unit cost per sensor component is **£29.79**. An important part of being able to scale is cost, once the design is finalised, is that we will be able to get components cheaper by buying in bulk. Another way to reduce cost would be to replace the raspberry PI zero running the sensor. We believe this is possible by using an AT mega chip and other components like a wireless module, EPROM and an SD card reader. This will require more programming and prototyping but could dramatically reduce the products cost and reliance on third parties.

We are using Amazon web service products to host the website, database and to manage calls and requests. Amazon is one of the cheapest and most scalable web services providers. Unfortunately, we won't know the exact costs to use these web services as we scale up how many clients and sensors we have, until we begin prototyping the sensors. We already have ideas to reduce the amount of calls and requests the sensors make by having a sensor in the client's house act as a master sensor. The master sensor will receive calls from all the slave sensors around the house and will send all that data at once to the database.

Contingency cost

Contingency cost: £1696.41

Our contingency cost is 10% of the overall project cost. This will provide at least 5 additional working days for all members of staff as well as some funds to replace any sensor components should they not work.

Total for the project

Office and Business Insurance (45 days + 18 weekend days) = £1638
Total staff cost for 45 days   **= £15,124.50**
Overall Project specific costs   **= £201.61**

Overall Total   **= £16,964.11**

Appendix and References

Attachment of supporting terms of reference documentation.

Ref	Name of document	Attachment
A	Project Plan	Project Plan 0.1.mpp
B	Project Risk Log	Risk Log.xlsx
C		

Complete list of references to material used throughout the terms of reference, including individual sections.

- (Chris Mordue) Pavithra, D. and Balakrishnan, R. (2015). IoT based Monitoring and Control System for Home Automation. Global Conference on Communication Technologies, pp.169, 170, 171, 172, 173. https://www.bernardmarr.com/default.asp?contentID=704

- (Chris Mordue) Nest. (n.d.). Meet the Nest app. [online] Available at: https://nest.com/uk/app/ [Accessed 14 Feb. 2019].

- (Chris Mordue) Hivehome.com. (n.d.). Hive | A British Gas Innovation | Start Your Connected Home. [online] Available at: https://www.hivehome.com/ [Accessed 14 Feb. 2019].

- (Chris Mordue) en.wikipedia.org. (n.d.). Internet of things. [online] Available at: https://en.wikipedia.org/wiki/Internet_of_things [Accessed 17 Jan. 2019].

- (Jamie Clarke) Lau, J., Zimmerman, B. and Schaub, F., 2018. Alexa, Are You Listening?: Privacy Perceptions, Concerns and Privacy-seeking Behaviours with Smart Speakers. Proceedings of the ACM on Human-Computer Interaction, 2(CSCW), p.102.

- (Jamie Clarke) Gatzlaff, K.M. and McCullough, K.A., 2010. The effect of data breaches on shareholder wealth. Risk Management and Insurance Review, 13(1), pp.61-83.

- (Jamie Clarke) Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A. and Stoica, I., 2009. Above the clouds: A berkeley view of cloud computing. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28(13), p.2009.

- (Jamie Clarke) Amazon Web Services, Inc. (2019). Amazon Relational Database Service (RDS) – AWS. [online] Available at: https://aws.amazon.com/rds/ [Accessed 24 Feb. 2019].

- (Jamie Clarke) Amazon Web Services, Inc. (2019). Amazon API Gateway. [online] Available at: https://aws.amazon.com/api-gateway/ [Accessed 24 Feb. 2019].

- (Jamie Clarke) Amazon Web Services, Inc. (2019). AWS Lambda – Serverless Compute - Amazon Web Services. [online] Available at: https://aws.amazon.com/lambda/ [Accessed 24 Feb. 2019].

- (Zachary Atkinson) Bodenham, D. (2018). How much does public liability insurance cost?. [online] Bought by Many. Available at: https://boughtbymany.com/news/article/how-much-does-public-liability-insurance-cost/ [Accessed 19 Feb. 2019].

- (Zachary Atkinson) GOV.UK. (2019). National Insurance. [online] Available at: https://www.gov.uk/national-insurance/national-insurance-classes [Accessed 19 Feb. 2019].

- (Zachary Atkinson) GOV.UK. (2019). National Insurance rates and categories. [online] Available at: https://www.gov.uk/national-insurance-rates-letters/category-letters [Accessed 19 Feb. 2019].

- (Zachary Atkinson) GOV.UK. (2019). National Insurance rates and categories. [online] Available at: https://www.gov.uk/national-insurance-rates-letters [Accessed 19 Feb. 2019].

- (Zachary Atkinson) Flexioffices.co.uk. (2019). Serviced Office Waterloo Square - NE1, Waterloo Square, Newcastle upon Tyne, NE1 4DN | Flexioffices. [online] Available at: https://www.flexioffices.co.uk/tyne-and-wear/newcastle-upon-tyne/waterloo-square_ne1_id7269 [Accessed 21 Feb. 2019].

- (Zachary Atkinson) Conniss, L. (2019). Project Costings. 1st ed. [ebook] Northumbria: Northumbria University, p.16. Available at: https://elp.northumbria.ac.uk/ [Accessed 19 Feb. 2019].

- (Liam Brand) ACM. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.58.8059&rep=rep1&type=pdf Accessed: 19th February 2019

- (Liam Brand) Huebscher, M.C. and McCann, J.A., 2004, October. Adaptive middleware for context-aware applications in smart-homes. In Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing (pp. 111-116).

Appendix G

Meeting Minutes

KV6002 MEETING MINUTES

Location: CIS

Date: 2nd March 2019

Time: 11:15

Agenda Items

1. It was agreed that the design and development of sub-systems should begin from next week, and designs for each sub-system are to be agreed by all team members before development begins.

2. The sending of device readings was discussed as a group, and it was agreed that data would not be updated in intervals and would only update if the reading value increased or decreased by a specified amount (such as a change of value by 0.5).

3. Only dummy data will be used whilst testing each sub-system for the overall product to ensure that it functions as expected, until readings can be received accurately.

4. It is important that the web application developed is mobile friendly, as this is replacing a mobile application.

5. The questionnaire should include a question for determining the purpose of the website, whether it has a corporate front, with abilities to log in and manage devices, or, is solely for the management of a device.

Action Items	Owner(s)	Deadline	Status
Develop and agree product design diagrams, including; (1) **product wireframes**, (2) **ERD**	JC, AN, CM	04/03/19	New
Create file structure for web application	CM	04/03/19	New
List a minimum of three questions based on literature and product reviews for each sub-system to gather requirements.	All	06/03/19	New
Completion of a single device build	ZA	06/03/19	New

KV6002 MEETING MINUTES

Location: CIS

Date: 11th March 2019

Time: 10:00

Agenda Items

1. Continue with the development of the individual subsystems.

2. Gave brief overview of each other's statuses and how far along everyone is.

Action Items	Owner(s)	Deadline	Status
Finish an MVP product of each subsystem	All	18/03/19	New
Implement the answers from questionnaire with client	All	18/03/19	New

KV6002 MEETING MINUTES

Location: CIS

Date: 18th March 2019

Time: 10:00

Agenda Items

1. MVP product progress is to be determined.

2. Completed MVP subsystems should be explained so that the rest of the group can understand them, which will aid in joining subsystems together to form the whole product.

3. For subsystems not at the MVP stage, blockers will be discussed to overcome issues.

Action Items	Owner(s)	Deadline	Status
Get end to end demo	All	30/03/19	New
Implement live data for subsystems that require it	All	25/03/19	New

KV6002 MEETING MINUTES

Location: CIS

Date: 25th March 2019

Time: 10:00

Agenda Items

1. Product progress is to be determined.

2. Individual subsystem updates from each group member

3. Comparison between what each subsystem is and to the objectives that were outlined in the TOR

Action Items	Owner(s)	Deadline	Status
Get end to end demo with live data	All	03/04/19	Ongoing

KV6002 MEETING MINUTES

Location: CIS Ground Floor

Date: 5th April 2019

Time: 12:00

Agenda Items

1. Continue progressing with sub-systems, taking note that we will need time to test.

2. Update all group members on current progress, and what each hope to achieve.

Action Items	Owner(s)	Deadline	Status
Progress with individual sub-system	All	12/03/19	New
Keep group members updated on status and progress	All	12/03/19	New