

Baby Monitor

*3rd Year Project*

*Graham Healy - X00104195*

*Keith Doolan - X00105348*

Table of Contents

[1. Project Summary 10](#_Toc418268514)

[1.1. Background 10](#_Toc418268515)

[1.2. Objectives 12](#_Toc418268516)

[1.3. Prototyping & Testing 12](#_Toc418268517)

[Java 4 Kinect 14](#_Toc418268518)

[1.4. Technical Requirements 14](#_Toc418268519)

[Kinect Sensor 14](#_Toc418268520)

[Kinect API/SDK 14](#_Toc418268521)

[Programming Languages 15](#_Toc418268522)

[Development IDE’s/Tools 15](#_Toc418268523)

[Arduino Based Thermometer 15](#_Toc418268524)

[Computer/Laptop 16](#_Toc418268525)

[1.5. Risk Assessment 17](#_Toc418268526)

[Security 17](#_Toc418268527)

[Privacy 17](#_Toc418268528)

[Price 17](#_Toc418268529)

[User Acceptance 17](#_Toc418268530)

[Time Management 17](#_Toc418268531)

[2. Project Methodology 18](#_Toc418268532)

[2.1. Introduction 18](#_Toc418268533)

[2.2. Project Approach 18](#_Toc418268534)

[Project Plan 19](#_Toc418268535)

[2.3. Project Phases 19](#_Toc418268536)

[Phase 1: Project Framework /Skeleton 19](#_Toc418268537)

[Phase 2: Key functions/objectives 19](#_Toc418268538)

[Phase 3: Testing 1 19](#_Toc418268539)

[Phase 4: Graphic user interface 20](#_Toc418268540)

[Phase 5: Testing 2 20](#_Toc418268541)

[Phase 6: Additional Features 20](#_Toc418268542)

[Phase 7: Cross Platform Support 20](#_Toc418268543)

[Phase 8: Testing 3 20](#_Toc418268544)

[2.4. Deliverables and Milestones 21](#_Toc418268545)

[2.5. Gantt chart 22](#_Toc418268546)

[3. Conclusion 23](#_Toc418268547)

[4. Bibliography/ Reference 24](#_Toc418268548)

[Appendix A: 24](#_Toc418268549)

[Application Area 28](#_Toc418268550)

[Project Scope 28](#_Toc418268551)

[Similar Systems for Baby Monitoring 29](#_Toc418268552)

[Back-End 30](#_Toc418268553)

[Microsoft Kinect 30](#_Toc418268554)

[Arduino Powered Thermometer 31](#_Toc418268555)

[Programming Languages 31](#_Toc418268556)

[C++ 31](#_Toc418268557)

[Java 31](#_Toc418268558)

[C# 32](#_Toc418268559)

[How to interface with the Kinect 32](#_Toc418268560)

[Kinect SDK and API 32](#_Toc418268561)

[Front-End 33](#_Toc418268562)

[Play Framework 33](#_Toc418268563)

[ASP.NET 33](#_Toc418268564)

[UI Framework/Design 34](#_Toc418268565)

[Database Technology 35](#_Toc418268566)

[Oracle DB 35](#_Toc418268567)

[MySQL 35](#_Toc418268568)

[SQL Server 35](#_Toc418268569)

[Development Tools and Other Technology 36](#_Toc418268570)

[Visual Studio 36](#_Toc418268571)

[Redmine 36](#_Toc418268572)

[Server Software 36](#_Toc418268573)

[Azure 36](#_Toc418268574)

[Apache 37](#_Toc418268575)

[IIS (Internet Information Services) 37](#_Toc418268576)

[JSON/XML 37](#_Toc418268577)

[Special Requirements 37](#_Toc418268578)

[Computer/Laptop 37](#_Toc418268579)

[Minimum Hardware Requirements 37](#_Toc418268580)

[Drivers and Operating System 38](#_Toc418268581)

[Bibliography & References 38](#_Toc418268582)

[List of Figures 40](#_Toc418268583)

[Overview of Project and Target Audience 46](#_Toc418268584)

[Overview 46](#_Toc418268585)

[Audiences 46](#_Toc418268586)

[Parents/Guardians 46](#_Toc418268587)

[Health Services 46](#_Toc418268588)

[Childcare Services 47](#_Toc418268589)

[Other Potential Audiences 47](#_Toc418268590)

[Architecture Overview 48](#_Toc418268591)

[System Stack 48](#_Toc418268592)

[Web Client 48](#_Toc418268593)

[Database 48](#_Toc418268594)

[Web Application/Hardware Server 48](#_Toc418268595)

[Web Server 48](#_Toc418268596)

[Hardware Client 49](#_Toc418268597)

[Hardware 49](#_Toc418268598)

[System Architecture Diagram 50](#_Toc418268599)

[System Architecture Flow Diagram 51](#_Toc418268600)

[Identify and describe User Stores and main acceptance criteria 52](#_Toc418268601)

[Prioritise Stories 61](#_Toc418268602)

[Technical Options - Comparative Analysis 62](#_Toc418268603)

[Web Framework 62](#_Toc418268604)

[Programming Language 63](#_Toc418268605)

[Data Persistence - Database 64](#_Toc418268606)

[UI Design/Framework 65](#_Toc418268607)

[Twitter Bootstrap 65](#_Toc418268608)

[JavaScript 65](#_Toc418268609)

[Data Transfer 66](#_Toc418268610)

[Web Server 66](#_Toc418268611)

[Integrated Development Environment 67](#_Toc418268612)

[Source Control 67](#_Toc418268613)

[Git/GitHub 67](#_Toc418268614)

[Release Plan 68](#_Toc418268615)

[References 69](#_Toc418268616)

[1. Iteration Progress 72](#_Toc418268617)

[Kinect Tracking 72](#_Toc418268618)

[Arduino Temperature Sensor 73](#_Toc418268619)

[Component List: 73](#_Toc418268620)

[Circuit 74](#_Toc418268621)

[Hardware Client Program 76](#_Toc418268622)

[2. Problems Encountered 77](#_Toc418268623)

[3. Changes to overall project 78](#_Toc418268624)

[4. References 79](#_Toc418268625)

[1. Iteration Progress 82](#_Toc418268626)

[Kinect Tracking 82](#_Toc418268627)

[Arduino Temperature Sensor 82](#_Toc418268628)

[Data Model 82](#_Toc418268629)

[Web Application 83](#_Toc418268630)

[AJAX 83](#_Toc418268631)

[SignalR 83](#_Toc418268632)

[Notifications 83](#_Toc418268633)

[UI/UX 84](#_Toc418268634)

[Security 85](#_Toc418268635)

[2. Problems Encountered 86](#_Toc418268636)

[Incorrect Temperature reading: 86](#_Toc418268637)

[SignalR 86](#_Toc418268638)

[Using XML over JSON in some instances. 86](#_Toc418268639)

[Entity Framework gave errors with Code First models. 86](#_Toc418268640)

[3. Changes to overall project 87](#_Toc418268641)

[Dedicated mobile application to be abandoned 87](#_Toc418268642)

[Data-Model Update 87](#_Toc418268643)

[4. References 88](#_Toc418268644)

[Design 90](#_Toc418268645)

[Logo 90](#_Toc418268646)

[Google Maps 90](#_Toc418268647)

[Loading Spinner 90](#_Toc418268648)

[Connection Notification Bar 90](#_Toc418268649)

[Webpage Animations 90](#_Toc418268650)

[Colour scheme 90](#_Toc418268651)

[Charts & Timelines 91](#_Toc418268652)

[Widget 91](#_Toc418268653)

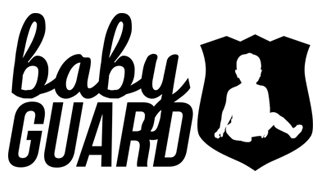
[3. Test Cases 92](#_Toc418268654)

[4. Problems Encountered 94](#_Toc418268655)

[5. Changes to overall project 94](#_Toc418268656)

[6. Database Diagram 94](#_Toc418268657)

[7. Class Diagram 94](#_Toc418268658)



Baby Monitor

*Detailed Project Proposal*

*Graham Healy - X00104195*

*Keith Doolan - X00105348*

Contents

[1. Project Summary 10](#_Toc402531629)

[1.1. Background 10](#_Toc402531630)

[1.2. Objectives 12](#_Toc402531631)

[1.3. Prototyping & Testing 12](#_Toc402531632)

[Java 4 Kinect 14](#_Toc402531633)

[1.4. Technical Requirements 14](#_Toc402531634)

[Kinect Sensor 14](#_Toc402531635)

[Kinect API/SDK 14](#_Toc402531636)

[Programming Languages 15](#_Toc402531637)

[Development IDE’s/Tools 15](#_Toc402531638)

[Arduino Based Thermometer 15](#_Toc402531639)

[Computer/Laptop 16](#_Toc402531640)

[1.5. Risk Assessment 17](#_Toc402531641)

[Security 17](#_Toc402531642)

[Privacy 17](#_Toc402531643)

[Price 17](#_Toc402531644)

[User Acceptance 17](#_Toc402531645)

[Time Management 17](#_Toc402531646)

[2. Project Methodology 18](#_Toc402531647)

[2.1. Introduction 18](#_Toc402531648)

[2.2. Project Approach 18](#_Toc402531649)

[Project Plan 19](#_Toc402531650)

[2.3. Project Phases 19](#_Toc402531651)

[Phase 1: Project Framework /Skeleton 19](#_Toc402531652)

[Phase 2: Key functions/objectives 19](#_Toc402531653)

[Phase 3: Testing 1 19](#_Toc402531654)

[Phase 4: Graphic user interface 20](#_Toc402531655)

[Phase 5: Testing 2 20](#_Toc402531656)

[Phase 6: Additional Features 20](#_Toc402531657)

[Phase 7: Cross Platform Support 20](#_Toc402531658)

[Phase 8: Testing 3 20](#_Toc402531659)

[2.4. Deliverables and Milestones 21](#_Toc402531660)

[2.5. Gantt chart 22](#_Toc402531661)

[3. Conclusion 23](#_Toc402531662)

[4. Bibliography/ Reference 24](#_Toc402531663)

[Appendix A: 24](#_Toc402531664)

# 1. Project Summary

The overall aim of this project is to build a quality application for monitoring the well-being of infants as they sleep taking advantage of a Microsoft Kinect. The application also aims to track sleeping behaviour and graph these patterns so it’s easy for an end-user to understand.

## Background

The proposed project that has been chosen is a baby/child monitor.

This project was picked because many parents/guardians worry about their child either when they’re at work or when they’re playing or sleeping at night.

For this project a program will be developed that runs with a Microsoft Kinect and also an optional self-engineered ‘smart’ thermometer.

The idea originally came from an initial proposal of an in-car baby monitor. According to Kids and Cars (kidsandcars.org, 2014) since 1999, nearly 700 children have been killed from being left in a car and suffering from heatstroke in the US alone.

This idea was to check if a child was left in a car in hot weather and would run an algorithm to detect temperature and sound an alarm. The team looked at many resources including web sources and academic sources. A past lecturer at the Institute of Technology Tallaght, Stephen Howell (2012) developed a program called Kinect2Scratch which allows users to send Kinect data to the Scratch programming language for kids.

After talking to project supervisors, the idea morphed into something on a larger scale. It was then decided to take it up a level and broaden the scope not only to a car, but even a child’s cot/bed or a playroom etc. It was found that this made it more end-user focused and that more features could be built in that would benefit the user.

A survey was conducted to find out the general public’s consensus of the idea of having a camera/thermometer to help notify them when their child may be under some form of discomfort.

Initial analysis of the survey shown that most people like the idea of being able to monitor their child as they sleep and not have to worry while other participants showed a slight concern about a camera that will be set up watching over a sleeping child.

When participants of the survey were asked what their issues were with the camera, they explained that they would not like the idea of a camera looking over them as they slept and how can they be sure that someone else is not potentially hacking into the camera to view it. There is possibly a flaw with how the survey was questioned or a lack of knowledge and explanation of what exactly the camera will do. From asking people, their perception was that the camera would be relaying direct video imagery to third party devices. This is an idea that was looked at and people seem to feel better if the camera would only collect data and not actual video imagery that would be relayed back to handheld devices.

The survey can be found here: <https://www.surveymonkey.com/s/XMQQXVH>

## Objectives

The main objectives for this project are as follows:

1. Create an application which runs on multiple platforms. (Web app, Windows, mobile etc.)
2. To detect a person/child in the room.
3. Detect if a child is sleeping or not.
4. Detect if a child is awake from sleep.
5. Detect unusual activity.
6. Use the thermometer to detect any high temperatures.
7. Notify user if child is awake (While sleeping).
8. Notify user if child has high temperature.
9. If a child is suffering from a high temperature, show the user the nearest hospital or load up a list of recommended phone numbers.
10. Set up custom alert with predefined temperature.
11. Map a child’s sleeping patterns.
12. Build profile of child
13. Create history of illnesses or sleeping patterns
14. Create graphs /charts to show a trend of sleeping patterns or a child’s sickness.

## Prototyping & Testing

To test the idea of motion tracking and skeletal recognition a few prototypes and proofs of concept were required.

According to Abhishek Kar. (2010) the Kinect can be used to accurately track skeletal movement. The idea of the Kinect tracking extreme movements like a fall has been proven by Gasparrini *et al*. (2014) where a Kinect sensor was placed on the ceiling and was able to detect a person falling and distinguish between an object and a person.

Firstly the Kinect SDK (Microsoft 2014) was downloaded which contained the drivers and API for the Kinect Sensor. Then there was set up of a Visual Studio C++ solution. The main core idea of this project is to detect out of ordinary movements.

The Kinect SDK comes with many sample programs and documentation so a small program was created to detect any changes in the skeleton movement. Basically it stores the information of the skeleton detected in an array. It compares the next frame to the last and if an anomaly is detected, it would simply print out a message. This proved that it was feasible to use this as a basis for the project.

### Java 4 Kinect

The team looked at a library for java by the University of Florida (University of Florida, 2014) called Java 4 Kinect which allows the Kinect API to be programmed in Java but the benefits lie more with using C++. This allows the program to have better performance and if it was then decided to purchase a Kinect for Windows, the API has a host of extra features to take advantage of.

## Technical Requirements

### Kinect Sensor

The main fundamental technical requirement of this project will be the Kinect Sensor. The Kinect is a powerful camera created by Microsoft, originally for the Xbox 360. The Kinect has many sensors which allow a program to capture colour frames, depth frames and track humans via skeletal tracking and facial recognition.

There are two versions of the Kinect sensor.

1. Kinect for XBOX 360: This version has lower specification parts then Kinect for Windows, but it still performs the key features required well. This sensor is also considerably cheaper than the one for Windows.

<http://www.xbox.com/en-ie/kinect>

1. Kinect for Windows: This is the dedicated sensor for Windows machines. It comes with upgraded cameras with a resolution of 1080p which can record at a steady 60 frames per second. Although better it is a lot more expensive.

<http://www.microsoft.com/en-us/kinectforwindows/>

### Kinect API/SDK

<http://www.microsoft.com/en-us/kinectforwindows/develop/downloads-docs.aspx>

The Kinect API contains all of the relevant methods and functions required to interface with the Kinect. This will be include in the project for ease of access to Kinect. These functions are absolutely key to the project because it allows us to get a frame from the camera and perform our own functions on it (e.g. depth frame analysis).

### Programming Languages

The project will be programmed In C++ or some variant. The team thought about programming in Java, but other languages have better performance benefits in relation to hardware.

### Development IDE’s/Tools

Since the project will be coded in C++/C#, the IDE used will be Microsoft’s visual studio.

The project will also be using the Kinect SDK application by Microsoft as it contains useful documentation and sample programs.

### Arduino Based Thermometer

There is Plans to engineer a small thermometer as an optional extra to use for monitoring. When putting the child to sleep, simply insert the thermometer in the child’s clothing and if the temperature is too hot. (Or higher than what a user specifies) then the users device will be sent a notification.

This thermometer can be achieved by using an Arduino UNO microcontroller board (Arduino 2014), in conjunction with a soldered on thermometer probe.

The Arduino comes with many libraries allowing the user to create powerful, ‘smart’ hardware. The Arduino language is based on C/C++ (Arduino, 2014) allowing us to potentially marry the Kinect and Arduino thermometer together.

### Computer/Laptop

#### Hardware specs

According to Microsoft (2014), the hardware requirements to run the Kinect device/SDK are as follows.

* 32-bit (x86) or 64-bit (x64) processors
* Dual-core, 2.66-GHz or faster processor
* USB 2.0 bus dedicated to the Kinect
* 2 GB of RAM
* Graphics card that supports DirectX 9.0c
* A Microsoft Kinect for Windows Sensor

#### Operating system

The minimum operating system required to run the Kinect SDK is Windows 7. Windows 8 is preferred.

The Kinect device requires appropriate drivers to interface with the operating system, programs. These drivers come with the Kinect SDK but can also be found separate (e.g. if you decide to go with OpenNI instead).

## Risk Assessment

### Security

There is a security risk involved in the fact that information on people may be stored online, depending on what decide to store. (E.g. names, addresses, telephone numbers etc.)

### Privacy

Also there are big privacy concerns. In the wake of the NSA spying scandal, many people are sceptical about technology and if there data is safe. It is a must that all information is secure and no information is being stored where it’s not needed.

### Price

If decisions are made to use the Kinect for Windows and its advanced features this will cost over €250. If it is decided that we use the Kinect for XBOX, this will only cost around €30 -50.

### User Acceptance

There is a stigma with cameras and especially in the wake of the NSA spying scandal many people would be sceptical to having a camera in their house watching their children. Efforts must be made to make it clear to the end users through on-screen messages and through a privacy policy that no camera information will be stored without the users consent.

### Time Management

This project will have to be managed efficiently, sticking within the tight time constraints. If goals were to fall behind on one aspect it would be very difficult to come back so it is imperative that objectives are met to keep on top of the project.

# Project Methodology

## Introduction

This project is a dynamic project with many different approaches. For programming purposes it is ideal to use Visual Studio, a powerful IDE. The Kinect SDK/API documentation and sample programs will be used as these will be very helpful in gaining an understanding of the system.

Once the Kinect is communicating with the program, the Arduino based thermometer will be introduced to the program. These two will then be used in conjunction with each other to help determine unusual activity/excessive temperatures.

For storing the data to be accessed by the program/web app the information will then have to be stored on a database online. It may be useful to store this data in an Object-Relational table. This allows us to encapsulate the data.

For creating the GUI a framework will be used, preferably .NET. There are many frameworks like Play that could also be used and will be researched further in the technical exploration report.

## Project Approach

User needs were identified by creating a survey and distributing it on social media sites like Facebook. This information was gathered and used to help make our objectives realistic and something that would benefit the end-users in some way.

A problem of storing the data was encountered. Luckily a Microsoft Azure account was given to students, this allows the creation of a database to store all of the data. This means that the native and web apps will be able to connect to the database.

Another problem facing the team is privacy concerns and security issues. Precautions must be made that the program and data is secure and all of the user’s data is private.

This also raises the issue of portability. According to statista.com (statista.com 2014), over 78.4% of smartphone market share is Android. A solution needs to be found to allow users on Android to access the data stored on the database. One solution may require writing a smaller Android program/application to pull data from the database.

Our initial solution to these problems involves creating a native app for Windows OS and then a web app for other users. This allows users to log in via their browsers (Chrome, Internet Explorer, Firefox etc.) and view a feed or track any events that have been uploaded online.

Another possibility is that a Java Native Interface could be used to wrap around the C++ code. According to Oracle (Oracle.com, 2014) the JNI to interface can be used with our C++ code. This in-theory means that it is possible to have a C++ app running on Android, allowing us to enable the notification objectives of the project on multiple platforms.

# Project Plan

## Project Phases

The project has been divided into eight phases. Each phase is in chronological order starting at one.

### Phase 1: Project Framework /Skeleton

This phase involves getting the core technologies working. At the end of this phase the Kinect Sensor and Arduino sensor should be communicating with the computer. Then the main framework (classes, methods etc.) for the program will be created to interface with these devices.

### Phase 2: Key functions/objectives

This phase involves implementing main key features into the project (Business Logic). These key objectives are the main underlying services that the application will provide.

### Phase 3: Testing 1

This phase will take place once phase one and phase two are complete. Although testing will take place as we program, this phase is a week of thoroughly testing all of the code created up to that point so the program is bug-free once the project moves onto the next phase.

### Phase 4: Graphic user interface

Phase 4 involves creating a graphical user interface to allow a user to interact with the program.

### Phase 5: Testing 2

Phase 5 is another week of testing the GUI to make sure all validation has been implemented and that quality of service is at the appropriate level.

### Phase 6: Additional Features

This phase is where all the additional features will be refined. Also any previously proposed features will be looked at to add depth to the project. All of these extra features will help improve the user experience.

### Phase 7: Cross Platform Support

This is where the Java Native Interface to get our C++ code to work on Android devices will be implemented.

### Phase 8: Testing 3

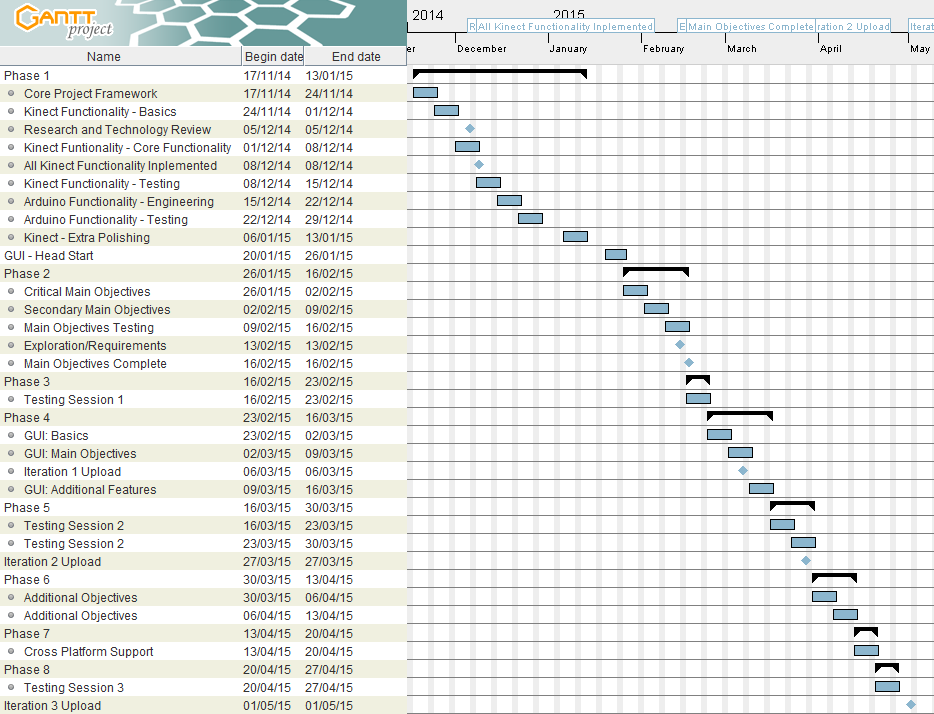
This is the last week of development where every aspect of the application will be thoroughly tested.

## Deliverables and Milestones

|  |  |  |
| --- | --- | --- |
| *Semester One* |  |  |
| Week | **Week Starting** | **Deliverable** |
| 1 | 15/09/2014 |  |
| 2 | 22/09/2014 |  |
| 3 | 29/09/2014 | **Initial Project Proposal Due Thursday 02/10/14** |
| 4 | 06/10/2014 |  |
| 5 | 13/10/2014 |  |
| 6 | 20/10/2014 |  |
| 7 | 27/10/2014 | **Detailed Project Proposal Due 31/10/14** |
| 8 | 03/11/2014 |  |
| 9 | 10/11/2014 |  |
| 10 | 17/11/2014 | **Phase 1: Project Framework /Skeleton** |
| 11 | 24/11/2014 | **Phase 1: Project Framework /Skeleton** |
| 12 | 01/12/2014 | **Research and Technology Review Due 05/12/14** |
| 13 | 08/12/2014 | **Phase 1: Project Framework /Skeleton** |
| *Semester Two* |  | **Phase 1: Project Framework /Skeleton** |
| 1 | 26/01/2015 | **Phase 2: Key functions/objectives** |
| 2 | 02/02/2015 | **Phase 2: Key functions/objectives** |
| 3 | 09/02/2015 | **Exploration/Requirements - 13/02/15** |
| 4 | 16/02/2015 | **Phase 3: Testing 1** |
| 5 | 23/02/2015 | **Phase 4: Graphic User Interface** |
| 6 | 02/03/2015 | **Iteration 1 Upload - 06/03/15** |
| 7 | 09/03/2015 | **Phase 4: Graphic User Interface** |
| 8 | 16/03/2015 | **Phase 5: Testing 2** |
| 9 | 23/03/2015 | **Iteration 2 Upload - 27/03/15** |
| Easter | 30/03/2015 | **Phase 6: Additional Features/Objectives** |
| Easter | 06/04/2015 | **Phase 6: Additional Features/Objectives** |
| 10 | 13/04/2015 | **Phase 7: Cross Platform Support** |
| 11 | 20/04/2015 | **Phase 8: Testing 3** |
| 12 | 27/04/2015 | **Iteration 3 Upload - 01/05/15** |

## Gantt chart

This Gantt chart describes in detail the key flow of the project. Each task is in a weekly segment and each phase is broken down into smaller tasks. Each upload and deliverable is marked with a milestone marker. This chart may be subject to slight change over the course of the project but it is imperative that the team sticks to this schedule.



# Conclusion

The team thinks that this project has a market (from market research). Tests and prototyping have been conducted and research into technology using the Microsoft Kinect show that it is definitely feasible to produce a program which can track unusual movement.

The team think that the project can bring some value in the sense of, depending on its application that it could prevent injury in young children who cannot be heard but still suffer. The team thinks that this technology could change someone’s life in a positive way and this is why this project is being undertaken.

# Bibliography/ Reference

Abhishek Kar, 2010. *Skeletal Tracking using Microsoft Kinect*. Available: <http://www.cs.berkeley.edu/~akar/cs397/Skeletal%20Tracking%20Using%20Microsoft%20Kinect.pdf>. Last Accessed: 16th Oct 2014

Arduino, 2014. *Arduino Uno Microcontroller Board*. Available: <http://arduino.cc/en/Main/ArduinoBoardUno>. Last Accessed: 26th Oct 2014

Arduino, 2014. *Arduino Language Reference Board*. Available: <http://arduino.cc/en/Reference/HomePage>. Last Accessed: 26th Oct 2014

Gasparrini, S, Cippitelli, E, Spinsante, S, & Gambi, E 2014, 'A Depth-Based Fall Detection System Using a Kinect® Sensor', Sensors (14248220), 14, 2, pp. 2756-2775, Academic Search Complete, EBSCOhost, viewed 30 October 2014.

Howell, S, 2012, *Kinect2Scratch (Version 2.5).* Available: <http://scratch.saorog.com/>. Last accessed: 16th Oct 2014

Kids and Cars, 2014, *Child Nontraffic Fatalities by type & year*. Available: <http://www.kidsandcars.org/statistics.html>. Last Accessed: 16th Oct 2014

Microsoft, 2014, *Kinect System Requirements.* Available: <http://msdn.microsoft.com/en-us/library/hh855359.aspx>. Last Accessed: 17th Oct 2014

Microsoft, 2014, *Kinect for Windows*. Available: <http://www.microsoft.com/en-us/kinectforwindows/>. Last Accessed: 20th Oct 2014

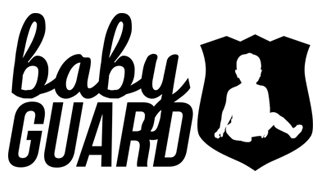
Open CV, 2014, *Open Source Computer Vision Library.* Available: <http://opencv.org/>. Last Accessed: 20th Oct 2014

Oracle, 2014, *Java Native Interface API*. Available: <http://docs.oracle.com/javase/7/docs/technotes/guides/jni/spec/intro.html#wp725>. Last Accessed: 29th Oct 2014

Statista.com, 2014, *Global market share held by smartphone operating systems from 2009 to 2014.* Available: <http://www.statista.com/statistics/263453/global-market-share-held-by-smartphone-operating-systems/>. Last Accessed: 29th Oct 2014

University of Florida, 2014, *Java 4 Kinect* Library. Available: <http://research.dwi.ufl.edu/ufdw/j4k/>. Last Accessed: 20th Oct 2014

# Appendix A:



Baby Monitor

*Research & Technology Review*

*Graham Healy - X00104195*

*Keith Doolan - X00105348*

Contents

[1. Project Summary 10](#_Toc418264884)

[1.1. Background 10](#_Toc418264885)

[1.2. Objectives 12](#_Toc418264886)

[1.3. Prototyping & Testing 12](#_Toc418264887)

[Java 4 Kinect 14](#_Toc418264888)

[1.4. Technical Requirements 14](#_Toc418264889)

[Kinect Sensor 14](#_Toc418264890)

[Kinect API/SDK 14](#_Toc418264891)

[Programming Languages 15](#_Toc418264892)

[Development IDE’s/Tools 15](#_Toc418264893)

[Arduino Based Thermometer 15](#_Toc418264894)

[Computer/Laptop 16](#_Toc418264895)

[1.5. Risk Assessment 17](#_Toc418264896)

[Security 17](#_Toc418264897)

[Privacy 17](#_Toc418264898)

[Price 17](#_Toc418264899)

[User Acceptance 17](#_Toc418264900)

[Time Management 17](#_Toc418264901)

[2. Project Methodology 18](#_Toc418264902)

[2.1. Introduction 18](#_Toc418264903)

[2.2. Project Approach 18](#_Toc418264904)

[Project Plan 19](#_Toc418264905)

[2.3. Project Phases 19](#_Toc418264906)

[Phase 1: Project Framework /Skeleton 19](#_Toc418264907)

[Phase 2: Key functions/objectives 19](#_Toc418264908)

[Phase 3: Testing 1 19](#_Toc418264909)

[Phase 4: Graphic user interface 20](#_Toc418264910)

[Phase 5: Testing 2 20](#_Toc418264911)

[Phase 6: Additional Features 20](#_Toc418264912)

[Phase 7: Cross Platform Support 20](#_Toc418264913)

[Phase 8: Testing 3 20](#_Toc418264914)

[2.4. Deliverables and Milestones 21](#_Toc418264915)

[2.5. Gantt chart 22](#_Toc418264916)

[3. Conclusion 23](#_Toc418264917)

[4. Bibliography/ Reference 24](#_Toc418264918)

[Appendix A: 24](#_Toc418264919)

[Application Area 28](#_Toc418264920)

[Project Scope 28](#_Toc418264921)

[Similar Systems for Baby Monitoring 29](#_Toc418264922)

[Back-End 30](#_Toc418264923)

[Microsoft Kinect 30](#_Toc418264924)

[Arduino Powered Thermometer 31](#_Toc418264925)

[Programming Languages 31](#_Toc418264926)

[C++ 31](#_Toc418264927)

[Java 31](#_Toc418264928)

[C# 32](#_Toc418264929)

[How to interface with the Kinect 32](#_Toc418264930)

[Kinect SDK and API 32](#_Toc418264931)

[Front-End 33](#_Toc418264932)

[Play Framework 33](#_Toc418264933)

[ASP.NET 33](#_Toc418264934)

[UI Framework/Design 34](#_Toc418264935)

[Database Technology 35](#_Toc418264936)

[Oracle DB 35](#_Toc418264937)

[MySQL 35](#_Toc418264938)

[SQL Server 35](#_Toc418264939)

[Development Tools and Other Technology 36](#_Toc418264940)

[Visual Studio 36](#_Toc418264941)

[Redmine 36](#_Toc418264942)

[Server Software 36](#_Toc418264943)

[Azure 36](#_Toc418264944)

[Apache 37](#_Toc418264945)

[IIS (Internet Information Services) 37](#_Toc418264946)

[JSON/XML 37](#_Toc418264947)

[Special Requirements 37](#_Toc418264948)

[Computer/Laptop 37](#_Toc418264949)

[Minimum Hardware Requirements 37](#_Toc418264950)

[Drivers and Operating System 38](#_Toc418264951)

[Bibliography & References 38](#_Toc418264952)

# Application Area

The area of this application is child monitoring. This is a big area with its own industry with products ranging from hardware devices like internet cameras with touch functionality and baby monitoring radios to simple software applications that listen for loud cries and noises. With more and more devices gaining access to the internet, devices like baby monitors are being transformed into baby nurseries that can detect and prevent injuries and fatalities from incidents like asphyxia and SIDS (Sudden Infant Death Syndrome).

# Project Scope

The scope of this project is to detect irregular sleeping movements indicating that a child may be awake and in some sort of discomfort. This will be done using a Microsoft Kinect and an Arduino based thermometer. The Kinect has an array of sensors and microphones with a large library and API. An algorithm will be created and if excessive movement is detected from the Kinect or the temperature is dangerously high, a notification will be sent to the user’s device. The user can then check on the child and dismiss the notification if needed. The notification will also be stored in a database to provide analytics for the user.

There will also be a web application for the user to log-on to. This will contain a history of sleeping cycles and provide some form of data analytics for the end user. This can then display graphs and charts based on the gathered data like the number of sleep disturbances and warnings. A program will run on the user’s machine with the Kinect and Arduino, log data and send it to the server running the web-app.

# Similar Systems for Baby Monitoring

There are other products similar to the project proposed. MimoBaby is a small smooth turtle shaped device that is attached to the baby’s garment at the waist. It will listen to the baby’s cries and know if the baby is moving or agitated. Other similar aspects of this product compared to the project is that the product graphs sleeping patterns and will alert parents when these patterns are irregular.

To add to this and make our product different, the project will also incorporate a thermometer to measure the temperature of the baby or even the room temperature to notify users if the child is too warm or the room is too cold or both. This is important as a baby in some cases if not all, would express their discomfort when it is too hot, this would help prevent rather that deal with the situation when it occurs. Unlike other traditional products, the project will be using Microsoft's Kinect camera to detect and track the babies motion using the built in depth imaging API's. Other products for detecting movement have been devices attached to the baby like the MimoBaby. This application we are developing is trying to eliminate any devices attached to a baby as this can cause some discomfort and safety hazards if not designed properly.

There are other products on the market that are not as complex as the MimoBaby. There are smart device applications like Cloud Baby Monitor on iOS devices and Baby Monitor on Android that use the sound sensors and cameras to alert parents by text or call when a baby's noise reaches certain decibels. This option was explored with just using a mobile device but it was found that it is impractical as two devices would be needed, one to run the application and listen to cries and another to receive alerts. The Microsoft Kinect comes with an array of HD quality microphones and an in-depth speech recognition API.

There are also other great features that have only been noticed when other applications and technologies have been assessed. Other products have information about tips and tricks or methods to soothe distraught babies. To add some depth to the application, this will be looked at further when the core functions have been implemented and refined. One idea is to play a lullaby to the baby while the parent is assisting the child or incorporate a function that will play a lullaby with a press of a button.

# Back-End

## Microsoft Kinect

The main hardware requirement in this project is the Microsoft Kinect. This is a motion sensing and tracking camera originally developed for the XBOX 360. The sensor contains an array of sensors including an infrared (IR) depth sensor and multiple microphones. This allows the sensor to collect image frames and advanced imagery such as image depth information and track multiple human skeletons.

The Kinect comes in two variations:

1. The Kinect for XBOX

This is the device originally developed for gaming purposes. It features lower specifications than the Kinect for Windows, but the Kinect SDK supports it to an extent.

1. The Kinect for Windows

This Kinect is primarily used for Windows development and deployment. It features higher specs including a higher frame rate and resolution. The Kinect SDK for this version also includes more support and features such as finger recognition.

## Arduino Powered Thermometer

The second hardware requirement is an Arduino based thermometer. This will be based off an Arduino UNO board, with temperature sensors added. It is then possible to connect via USB or WiFi to the system, which can then use the temperature information to make more accurate decisions about child temperature. Small devices are easy to build and program with most devices coming with its own library for programmers to implement. The Arduino language is based off C++/C# allowing for easy integration into applications.

## Programming Languages

### C++

C++ is a powerful programming language. It provides direct access to the underlying physical hardware and gives the developer and program detailed access to the physical memory. This language is very popular in applications such as video games where performance is the key requirement.

Since there is direct access to memory and hardware, C++ does not include an in-built garbage collection service. This means that poorly written and unmanaged code can lead to fatal errors and memory leaks. A significant portion of the project’s workflow will be making sure all memory access is managed correctly and this can take a significant amount of time and code (CPlusPlus.com, 2014).

### Java

Java is another programming language originally written by Sun Microsystems but is now managed by Oracle. One of the most popular languages due to its portability and ease-of-use. Java runs on top of a Java Virtual Machine (JVM) which is written in C++. Java also has garbage collection reducing the risk of memory leaks but since it runs on top of a JVM it is significantly slower than C++.

The Microsoft Kinect SDK is written in C++/C# and does not officially support Java, but the University of Florida have created a library called Java 4 Kinect (J4K) (University of Florida, 2014) which allows the Kinect to interface with Java applications.

### C#

C# or C-Sharp is Microsoft’s proprietary language. A derivative of C++, C# was developed to run on the .NET framework. C# is very similar to Java because it also runs on a virtual machine and has its own garbage collection facility for memory management but C# is more modern. C# contains features from C++ such as operator overloading, boxing and pointers.

Although all languages are Object Oriented Programming languages (OOP), the team decided to go with C++ and C# because of ASP.NET’s integration with C. Although there are various Java libraries to help the Kinect interface with Java, there is limited support and features compared to using C# or C++. The Kinect SDK is written in C++ and C#.

## How to interface with the Kinect

### Kinect SDK and API

The Kinect SDK comes with an advanced API to develop applications using the Kinect. It comes with functions to stop and start the Kinect, get depth, colour and skeleton frames and track movement. The Kinect API contains all of the relevant methods and functions required to interface with the Kinect. These functions are absolutely key to the project because it allows an application to get a frame from the camera and perform functions on it (e.g. depth frame analysis).

Microsoft also provide the required documentation and a programming guide. This allows the team to easily understand the way the Kinect works and all of the data formats and types needed to create and perform algorithms. Figure 1, 2 and 3 show the code required to connect, start and retrieve data from the Kinect.

There are other libraries such as OpenNI and OpenCV, but these are tedious to install and configure and are not as widely supported as Microsoft’s SDK. These libraries also do not have as many features built-in to the SDK such as facial recognition and speech recognition.

For a Java based application a third party library such as Java4Kinect would have to be used. This library allows a Java application to interface with the Kinect device. But since Java runs on a JVM and not officially supported by the Kinect there could be several interoperability issues. Figure 4 shows a small java snippet which uses J4K to interface with the Kinect.

# Front-End

For developing the user application there are some considerations. Scalability, ease of use and performance are some key considerations. Considering these options it was decided that the application would be best suited as a web application. Web based applications have extensive API’s and tutorials, they also have methods of controlling different aspects of development. There are many frameworks available like the Java Play framework and the ASP.NET framework.

## Play Framework

The Play framework is strictly Java only in regards to modelling the application. So from the offset this is not a good choice as the hardware that project requires needs to be programmed in C or a derivative language. The Play framework is still capable of completing the tasks that are needed but problems will occur with the programming language differences down the line.

## ASP.NET

ASP.NET is similar to Play framework but also offers extra features. The ASP.NET MVC framework is integrated well with Visual Studio and there is a large database of tutorials and API's available. The biggest draw to ASP.NET is the ease of use. During the set-up of a project user authentication can be integrated so user accounts like Google and Hotmail can be used. There is a lot of automation with ASP.NET, for example there is a scaffolding functionality that automatically generates a database table based on the model class that is made thus reducing the amount of SQL needed to be written. With this functionality Visual Studio will also generate pre-made views to add, update and delete data based on the model, giving your project basic CRUD (Create Read Update Delete) functionality.

In figures 5 and 6, the data context class is developed based on the model selected and the entities of this model will be the fields in the database table. This is done using the Entity framework which maps models to SQL. Figure 7 shows an automatically generated view, in comparison to the Play framework there is no functionality for this automation. It is possible to apply entities to the database model but it is not straight forward compared to ASP.NET’s implementation. With the play framework it takes longer to do.

Both frameworks have an MVC approach to development. MVC (Model-View-Controller) is an architectural pattern for developing Interfaces, logically separating the user-interface, core business logic and data storage, giving a developer more control over the application and makes maintenance easier allowing a developer to change the front-end without affecting the back-end. Both frameworks facilitate the creation of a RESTful interface and include features such as user authentication and object relational mapping.

# UI Framework/Design

For the web app the team decided to use a web framework to help create a responsive and visually pleasing UI dashboard that adheres to current trends and standards.

Twitter Bootstrap is a CSS framework that has various CSS and JavaScript components. Bootstrap allows a user to develop an intricate and responsive design with components such as drop down menus, a detailed grid system, colour context panels and various JavaScript plugins such an image slider. By default ASP.NET uses Twitter Bootstrap for its UI and Bootstrap is easily implemented with many demos and tutorials available. Figure 8 shows an auto-generated view using Bootstrap.

Gumby is also another CSS framework. Similar to Bootstrap, Gumby also offers a large amount of components for web developers to integrate into their designs. Gumby has advanced features compared to bootstrap like moving div containers automatically based on screen/size.

Pure.css is a lightweight framework, ideal for applications and dashboards. Each component is small in size meaning it will be fast for the browser to parse and load, making it fast and responsive.

In addition to CSS frameworks, various JavaScript plugins will be integrated into the views. Chart.js will be used for data analytics and displaying information in a way that is meaningful to the end user. Flot.js is also another powerful JavaScript plugin which can display real time information

# Database Technology

A database is required to store information such as user credentials, sleeping patterns and settings. The database will need to support object relational tables for encapsulation and database transactions to ensure data is correctly committed to the DB while avoiding conflict. One main benefit of using a web framework for development is that most framework’s offer various methods to connect to third party databases. Some frameworks can also map a web app’s model to a DB (Entity Framework, Ebean).

### Oracle DB

Oracle 12c is the latest version of Oracle’s closed-source database software. The world’s most popular DBMS (DB-Engines.com, 2014), Oracle 12c supports object relational tables, transaction locking and table partitioning. The Oracle DB is based on Java which means it is portable and stable but may cause performance issues depending on program structure and hardware requirements.

### MySQL

MySQL is an open source database software and the world's second most used DBMS (DB-Engines.com, 2014). Since MySQL is open source many third party versions or add-ons are available. These include performance plugins and GUI interfaces. MySQL is written in C++ and works on many operating systems. Many companies extend MySQL, creating their own DBMS software.

### SQL Server

SQL server is Microsoft’s database software and the third most popular DBMS in the world (DB-Engines.com, 2014). Compared to Oracle and MySQL, SQL Server is a proprietary software which requires a license to use. Written in C++, SQL Server is considered to be one of the fastest performing DBMS’s in the industry due to direct access to hardware and memory. This allows the DBMS software to dynamically manage memory while the program is in execution.

Each database vendor is supported by frameworks such as ASP.NET and Play framework. ASP.NET uses provider objects for each DBMS to allow communication between the DB and application while Play framework does not provide such implementation requiring the developer to include their own database connection drivers. The database vendors are similar in features, but the C++ based DBMS’s such as SQL Server and MySQL has faster processing speeds but SQL Server’s license fee may be a potential problem.

# Development Tools and Other Technology

## Visual Studio

Visual Studio 2013 is the chosen IDE for the project. Developed by Microsoft, VS has full integration for C# and C++ and uses intellisense to help identify potential programming errors and problems. Visual Studio also has built-in support for SCM protocols such as Ankh SVN, allowing the team to commit code to redmine from the IDE.

## Redmine

Redmine is the project management tool for the project. Redmine is an open source web application which allows users to manage their workflow and respond to mission critical tasks such as key milestones and deliverables. It also features a calendar and Gantt chart functionality allowing project managers and users to map out a project lifecycle.

Redmine also includes software configuration management or SCM. This software tracks changes in code over and allows developers and project managers to track progress of code issues. This also allows developers and managers to create issues on specific code tasks such as bugs and assign a priority and developer to fix the issue.

## Server Software

### Azure

Azure is Microsoft’s cloud platform. It will allow the team to deploy the project into the public cloud. It will host the web application and the database software (MySQL etc.). This means the project can scale automatically depending on load and guarantee an always-on service for the end-users. The visual studio IDE will allow Azure to run concurrently alongside the ASP.Net framework thus eliminating the need to ensure the Azure client is updating the Cloud database.

### Apache

Apache is an open-source HTTP web-server. It is the world’s most popular hosting software and has been run since 1996. Apache is commonly found in various hosting packages such as a LAMP (Linux, Apache, MySQL and PHP) or WAMP (Windows, Apache, MySQL and PHP) package like XAMP. Apache is a lightweight server and contains the basics for a HTTP web server to function.

### IIS (Internet Information Services)

IIS is a secure and flexible Windows server for all internet hosting needs, from web applications to databases. IIS can even deal with high load and multiple applications because of its scalable capabilities. IIS is not only a server for hosting, there are many administrative capabilities and functionalities that allow a user to manage the server. An example of these would be custom logging, event handling, worker login management and idle worker instance saving being the most recent additions.

### JSON/XML

The project will use JSON or XML for transfer from client to server. (So it can be recorded in DB) This will allow the client program to send information from the Kinect and Arduino to the server to be recorded in the database.

# Special Requirements

## Computer/Laptop

### Minimum Hardware Requirements

● 32-bit (x86) or 64-bit (x64) processors

● Dual-core, 2.66-GHz or faster processor

● USB 2.0 bus dedicated to the Kinect

● 2 GB of RAM

● Graphics card that supports DirectX 9.0c

● Microsoft Kinect for Windows Sensor

● Arduino UNO board and temperature probe.

### Drivers and Operating System

Windows 7/8/8.1 and Kinect SDK drivers are required to install the Kinect device.

# Bibliography & References

Arduino, 2014. *Arduino Uno Microcontroller Board*. Available: <http://arduino.cc/en/Main/ArduinoBoardUno>. Last Accessed: 10th Dec 2014

Arduino, 2014. *Arduino Language Reference Board*. Available:<http://arduino.cc/en/Reference/HomePage>. Last Accessed: 10th Dec 2014

Apache Foundation, (2014). *About Apache HTTP Server Project.* Available: <http://httpd.apache.org/ABOUT_APACHE.html>. Last Accessed: 18th Dec 2014

Chart, (2014). *Chart.js - Open source HTML5 Charts for your website.* Available: <http://www.chartjs.org/>. Last Accessed: 16th Dec 2014

CPlusPlus.com, (2014)*. A Brief Description - C++.* Available: [*http://www.cplusplus.com/info/description/*](http://www.cplusplus.com/info/description/). Last Accessed: 12th Dec 2014

DB-Engines.com, 2014. *DB-Engines Ranking.* Available: <http://db-engines.com/en/ranking>. Last Accessed: 12th Dec 2014

Flot, (2014). *Flot: Attractive JavaScript plotting for JQuery.* Available: <http://www.flotcharts.org/>. Last Accessed: 16th Dec 2014

Gumby, (2014). *Gumby 2.6 Framework*. Available: <http://gumbyframework.com/>. Last Accessed: 16th Dec 2014

Martin Man, (2014). *Cloud Baby Monitor*. Available: <https://itunes.apple.com/ie/app/cloud-baby-monitor-unlimited/id432791399?mt=8>. Last Accessed: 16th Dec 2014

Microsoft, (2014). *ASP.Net*. Available: <http://www.asp.net/>. Last Accessed: 10th Dec 2014

Microsoft Developer Network, (2014). *Getting and Displaying Depth Data in C#*. Available: <http://msdn.microsoft.com/en-us/library/jj131029.aspx>. Last Accessed: 10th Dec 2014

Microsoft, (2014). *Kinect System Requirements.* Available:<http://msdn.microsoft.com/en-us/library/hh855359.aspx>. Last Accessed: 10th Dec 2014

Microsoft, (2014). *Kinect for Windows*. Available:<http://www.microsoft.com/en-us/kinectforwindows/>. Last Accessed: 10th Dec 2014

Microsoft, (2014). *Explore SQL Server 2012-2014.* Available: <http://www.microsoft.com/en-us/server-cloud/products/sql-server/>. Last Accessed: 13th Dec 2014

Mimo, (2014). *Smart Baby Nursery.* Available: <http://mimobaby.com/>. Last Accessed: 16th Dec 2014

MySQL, (2014). *MySQL :: Why MySQL?* Available: [http://www.mysql.com/why-mysql/](http://www.mysql.com/why-mysql/l). Last Accessed: 13th Dec 2014

Open CV, (2014). *Open Source Computer Vision Library.* Available:<http://opencv.org/>. Last Accessed: 12th Dec 2014

Oracle, (2014). *Oracle Database 12c*. Available: <https://www.oracle.com/database/index.html>. Last Accessed: 12th Dec 2014

Play, (2014). *Play Framework.* Available: <https://www.playframework.com/>. Last Accessed: 10th Dec 2014

Pure.io, (2014). *Pure Framework.* Available: <http://purecss.io/>. Last Accessed: 16th Dec 2014

Redmine.org, (2014). *Overview - Redmine*. Available: <http://www.redmine.org/>.Last Accessed: 14th Dec 2014

Bootstrap, (2014). *Twitter* *Bootstrap.* Available: <http://getbootstrap.com/>. Last Accessed: 16th Dec 2014

University of Florida, 2014, *Java 4 Kinect Library*. Available:<http://research.dwi.ufl.edu/ufdw/j4k/>. Last Accessed: 10th Dec 2014

Bootswatch, (2014). *Bootswatch, Twitter Bootstrap Library. Available at:* [*http://bootswatch.com/*](http://bootswatch.com/). Last Accessed: 15th Dec 2014

YouTube, (2014). *Introducing ASP.NET MVC 5.* Available at: <https://www.youtube.com/watch?v=gxFtRdKr7CA>. Last Accessed: 16th Dec 2014

# List of Figures

Figure 1: Enumerating the Kinect device

private KinectSensor sensor;

private DepthImagePixel[] depthPixels;

private Skeleton[] skeletonData;

private byte[] colorPixels;

private WriteableBitmap colorBitmap;

private void enumer()

{

foreach (var potentialSensor in KinectSensor.KinectSensors)

{

this.sensor = potentialSensor;

break;

}

Console.Write("Enumerated Sensor " + sensor.Status);

}

Figure 2: Starting the Kinect’s sensors

private void start() {

if (this.sensor != null) {

this.sensor.ColorStream.Enable(ColorImageFormat.RgbResolution640x480Fps30);

this.sensor.DepthStream.Enable(DepthImageFormat.Resolution640x480Fps30);

this.sensor.SkeletonStream.Enable();

this.sensor.ColorStream.Enable(ColorImageFormat.InfraredResolution640x480Fps30);

this.skeletonData = new Skeleton[sensor.SkeletonStream.FrameSkeletonArrayLength];

this.sensor.Start();

Console.WriteLine("\nSensors Started");

this.depthPixels = new DepthImagePixel[this.sensor.DepthStream.FramePixelDataLength];

this.colorPixels = new byte[this.sensor.DepthStream.FramePixelDataLength \* sizeof(int)];

this.colorBitmap = new WriteableBitmap(this.sensor.DepthStream.FrameWidth, this.sensor.DepthStream.FrameHeight, 96.0, 96.0, PixelFormats.Bgr32, null);

if (this.sensor != null)

{

this.sensor.DepthFrameReady += this.SensorDepthFrameReady;

}

}

}

Figure 3: Getting depth information from the Kinect

private void SensorDepthFrameReady(object sender, DepthImageFrameReadyEventArgs e)

{

using (DepthImageFrame depthFrame = e.OpenDepthImageFrame())

{

if (depthFrame != null)

{

// Copy the pixel data from the image to a temporary array

depthFrame.CopyDepthImagePixelDataTo(this.depthPixels);

for (int i = 0; i < this.depthPixels.Length; ++i)

{

// Get the depth for this pixel

short depth = depthPixels[i].Depth;

if (depthPixels[i].PlayerIndex == 1) {

Console.Write("\r{0} ", depth);

}

}

}

}

}

Figure 4: Using Java4Kinect to interface with the Kinect

**public** **void** onDepthFrameEvent(**short**[] depth, **int**[] U, **int** V[]) {

**if**(viewer==**null** || label==**null**)**return**;

maps[0] = maps[1];

**float** a[]=getAccelerometerReading();

label.setText(((**int**)(a[0]\*100)/100f)+","+((**int**)(a[1]\*100)/100f)+","+((**int**)(a[2]\*100)/100f));

DepthMap map=**new** DepthMap(depthWidth(),depthHeight(),depth);

**if**(U!=**null** && V!=**null**) {

map.setUV(U,V,videoWidth(),videoHeight());

maps[1] = map;

};

**if**(mask\_players)map.maskPlayers();

viewer.map=map;

**if**(maps[0] != (maps[1])){

System.***out***.print("Not equal!");

}

}

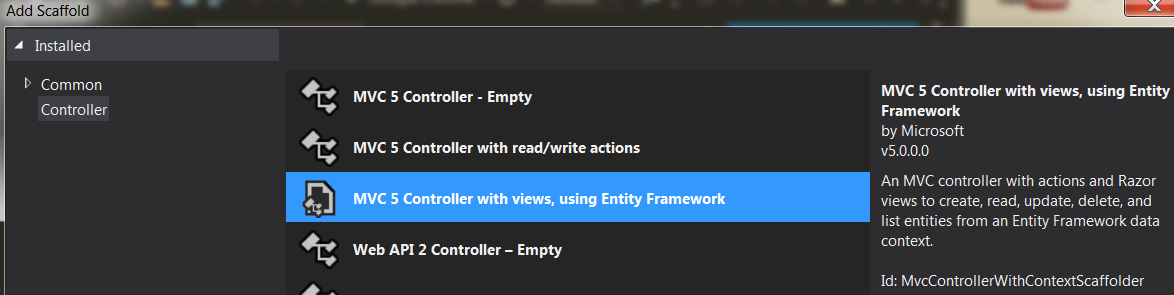
Figure 5: Adding a MVC scaffold in ASP.NET and Visual Studio 2013

Figure 6: Adding a controller in ASP.NET

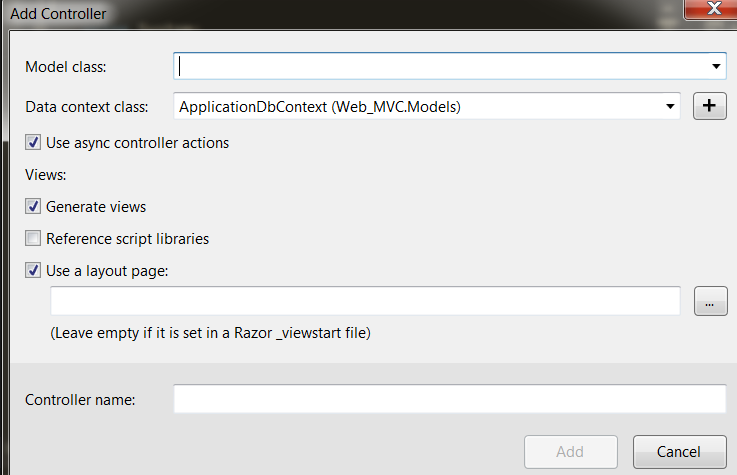
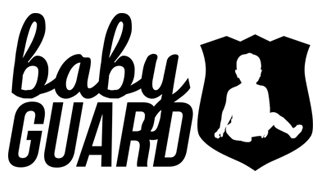


Figure 7: An auto-generated view using Twitter Bootstrap in ASP.NET.



Baby Monitor

*Exploration Requirements*

*Graham Healy - X00104195*

*Keith Doolan - X00105348*

Contents

[Overview of Project and Target Audience 46](#_Toc411612955)

[Overview 46](#_Toc411612956)

[Audiences 46](#_Toc411612957)

[Parents/Guardians 46](#_Toc411612958)

[Health Services 46](#_Toc411612959)

[Childcare Services 47](#_Toc411612960)

[Other Potential Audiences 47](#_Toc411612961)

[Architecture Overview 48](#_Toc411612962)

[System Stack 48](#_Toc411612963)

[Web Client 48](#_Toc411612964)

[Database 48](#_Toc411612965)

[Web Application/Hardware Server 48](#_Toc411612966)

[Web Server 48](#_Toc411612967)

[Hardware Client 49](#_Toc411612968)

[Hardware 49](#_Toc411612969)

[System Architecture Diagram 50](#_Toc411612970)

[System Architecture Flow Diagram 51](#_Toc411612971)

[Identify and describe User Stores and main acceptance criteria 52](#_Toc411612972)

[Prioritise Stories 61](#_Toc411612973)

[Technical Options - Comparative Analysis 62](#_Toc411612974)

[Web Framework 62](#_Toc411612975)

[Programming Language 63](#_Toc411612976)

[Data Persistence - Database 64](#_Toc411612977)

[UI Design/Framework 65](#_Toc411612978)

[Twitter Bootstrap 65](#_Toc411612979)

[JavaScript 65](#_Toc411612980)

[Data Transfer 66](#_Toc411612981)

[Web Server 66](#_Toc411612982)

[Integrated Development Environment 67](#_Toc411612983)

[Source Control 67](#_Toc411612984)

[Git/GitHub 67](#_Toc411612985)

[Release Plan 68](#_Toc411612986)

[References 69](#_Toc411612987)

# Overview of Project and Target Audience

### Overview

The system being developed is a smart baby tracker. This system is a combination of hardware and software to allow for the accurate detection of movement and warn the user of any potential hazards/dangers to an infant. This will allow parents/guardians of children to be warned via an application or web app if any movement has occurred or an environmental variable such as temperature has reached an unacceptable/unsafe level.

Data will be sent from hardware via a client program running on the users machine which is then sent over the internet securely to a server program making it persistent using a database. This information will be used in a web application and presented to the user in an easy to digest format.

# Audiences

### Parents/Guardians

The main audience for the system is parents/guardians of young children. This system is an alternative to traditional baby monitors available globally such as two-way-radios or simple camera solutions. Typically these devices offer little functionality and with the proliferation of Internet of Things (IoT) devices, a ‘smart’ baby monitor is a logical step forward. This offers a whole new depth of information to parents/guardians of young children and give parents/guardians peace of mind, knowing if something was to occur while they were asleep or not in sight of the infant they will be promptly alerted and given options to act upon such as call the emergency services or find the nearest health centre.

### Health Services

Another consideration for a possible audience is the health service. A version of the system can be developed to work and track vitals of new-born babies in maternity wards of hospitals potentially reducing the risk of suffocation, SIDS and hypothermia.

### Childcare Services

The system can also appeal to other industry sectors such as a child care service or crèche/Montessori. The system can assist child care staff in monitoring the large amount of children where incidents and accidents can be prevented with early detection.

### Other Potential Audiences

There is also the possibility of new markets. Due to the nature of the camera and the Arduino open source hardware foundation, the system can be repurposed to work in different scopes e.g. an intrusion detection system, detecting if elderly people have injured themselves at home, in car security etc. There is a lot of room for expansion and modularity.

# Architecture Overview

## System Stack

### Web Client

This is the application running on the top of the stack. This will display all the data taken by the hardware, convert it into information and present it in a meaningful way. The web application will offer the main functionality such as creating alerts and viewing historical data such as past tracking sessions. While initially this will be a web application, this can be expanded to include devices like Android, iOS and Windows Phone devices using a RESTful interface.

### Database

This is where any data collected by the system will be made persistent. The database will store data such as tracking data, configuration settings, preferences, user data etc. This will allow any client to retrieve and display historical information. The system will also take advantage of an Object Relational Mapping (ORM) tool called Entity Framework to facilitate the transfer of the system model to a database schema and provide CRUD functionality, allowing the system to scale easily when new features and modules are added.

### Web Application/Hardware Server

The server will provide functionality to allow the client to transmit data from the client machine/hardware by listening for traffic via sent via HTTP to the web application to be stored in the database and presented on the web application. This also is part of the Model View Controller (MVC) pattern so the server will be able to process and fetch information for the client. The server will implement a RESTful interface so other devices can also access the information processed by the application.

### Web Server

A web server/HTTP server is required to make the web application accessible to the internet. This software runs below the web software server that takes HTTP requests bound for the web application. The web server will handle HTTP requests (GET, POST, PUT, and DELETE) and fetch the requested content/page.

### Hardware Client

The layer builds on the hardware layer of the system. The client is responsible for the operation of the Kinect/Arduino and performing the required functions to detect motion and track temperature. The client also is required to transfer this data to the server so it can be presented in real-time and stored in a database.

### Hardware

#### Kinect

This is the last layer in the system architecture. The hardware is the core functionality of the project and every other layer of the system will make use of or represent data or functions of the hardware.

The Kinect is a camera developed by Microsoft which allows the accurate detection and tracking of human skeletons. The sensor also has an array of sensors such as depth and infrared sensors. The Kinect also has an array of microphones. This system will take advantage of the depth, infrared and microphone array initially. The data provided by these sensors will be accessed and manipulated by the client program running on the user’s machine. Microsoft provide an SDK and application program interface (API) to allow the system to access the data sent from the sensors.

#### Arduino

The Arduino (Arduino, 2015) is a microprocessor which provides the ability to create smart, Internet of Things (IoT) enabled devices. In this system the Arduino will be used to create a ‘smart’ thermometer. Using a thermometer probe connected to the Arduino’s serial port, sensor information will be sent to the hardware client to be processed. This data will then be sent to the web application.

## System Architecture Diagram



## System Architecture Flow Diagram



# Identify and describe User Stores and main acceptance criteria

For the User Stories the Priority will be set between 1 and 5.

* Lowest priority - Score of 5.
* Highest priority - Score of 1.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 1 | **Story Title:** Start Tracking | **Priority:** 1 | **Estimate:**  35 Hours |
| Start tracking service and all associated code. | | | |

**Confirmation:**

**Success** – All devices are started and data is sent to the server.

* Success message sent to server.
* Tracking data is sent to server and is displayed on front-end.

**Failure** – Devices could not start.

* Error message displayed.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 1 | **Story Title:** Stop Tracking | **Priority:** 1 | **Estimate:**  5 Hours |
| End tracking service and stop devices. | | | |

**Confirmation:**

**Success** – All devices are stopped, server connection is closed and data is no longer being sent to the server.

* Success message sent to server.
* Connection is closed and devices shut down.

**Failure** – Service could not be stopped.

* Error message displayed.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 3 | **Story Title:** Log in | **Priority:** 1 | **Estimate:**  10 Hours |
| Log in to access personal account. | | | |

**Confirmation:**

**Success** – User logged in successfully.

* Successful login message displayed (forefront).
* User home page displayed (background).

**Failure** – User login unsuccessful.

* Unsuccessful message displayed.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 4 | **Story Title:** Create Account | **Priority:** 1 | **Estimate:**  5 Hours |
| Make account to avail of the application. | | | |

**Confirmation:**

**Success** – Required details entered successfully.

* Access to personal home page.

**Failure** – User Account creation unsuccessful.

* Required fields message displayed.
* User account already in use.
* User name already saved to database.

|  |  |  |  |
| --- | --- | --- | --- |
| **IID:** 5 | **Story Title:** View Graphs | **Priority:** 2 | **Estimate:**  10 Hours |
| View the graphs tab that displays the sleep patterns, temperature, etc. | | | |

**Confirmation:**

**Success** – Graphs are displayed.

* View of multiple graphs.
* Correct data displayed.

**Failure** – Graphs do not appear.

* Not enough data gathered to display graphs successfully.
* Connection to database unsuccessful.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 6 | **Story Title:** Check Alerts | **Priority:** 2 | **Estimate:**  10 Hours |
| View the alerts page that will display if certain alerts have been triggered. | | | |

**Confirmation:**

**Success** – Alerts have been triggered.

* View alerts.
* Process data to graphs, etc.

**Failure** – Alerts missing or not available.

* Not data gathered to display alerts.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 7 | **Story Title:** User Alert Management | **Priority:** 4 | **Estimate:**  5 Hours |
| Manage alerts and give the user an option to set parameters of when alerts are triggered. | | | |

**Confirmation:**

**Success** – Alerts created.

* View alerts.
* Set parameters for alerts to trigger.

**Failure** – Cannot create alerts.

* Problems with connection to database.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 8 | **Story Title:** Delete Account | **Priority:** 3 | **Estimate:**  2 Hours |
| User account can be deleted thus removing all information from database. | | | |

**Confirmation:**

**Success** – User Account deleted.

* User logged out of the application.
* User returned to the login screen.

**Failure** – User Account cannot be deleted.

* Message displayed for unsuccessful deletion on account.
* Page refresh to eliminate any errors that may have occurred.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID:** 9 | **Story Title:** Display Local Health Services | **Priority:** 5 | **Estimate:**  10 Hours |
| Information about local health centres or emergency services. | | | |

**Confirmation:**

**Success** – Information displayed based on geographical location.

* View of local health services.
* Contact details of local health services.
* View of Google maps of user’s geographical location.

**Failure** – No information displayed.

* Cannot gather information based on location.
* Location entered on user account is incorrect.
* Cannot retrieve information from database.

# Prioritise Stories

Below is the list of the priorities and the order in which the project will be developed.

|  |
| --- |
| Iteration 1 |
| ID: 1 – Start Tracking |
| ID: 2 – Stop Tracking |

|  |
| --- |
| Iteration 2 |
| ID: 3 – Log In |
| ID: 4 – Create Account |
| ID: 5 – View Graphs |
| ID: 6 – Check Alerts |
| ID: 7 – User Alert Management |

|  |
| --- |
| Iteration 3 |
| ID: 8 – Delete Account |
| ID: 9 – Display Local Health Services |

# Technical Options - Comparative Analysis

## Web Framework

ASP.net 4.5 MVC has been selected as a web framework for this project. ASP.net is an enterprise grade web application framework built on C#. ASP.net follows the model view controller (MVC) design pattern. This allows for the logical and physical separation of the web application into a 3 tier architecture, separating the presentation layer from the core business logic layer which is also separated from the data access layer. MVC makes the project more maintainable and easier to manage. This allows the team to make modifications to the presentation layer without having to refactor massive amounts of code in the business logic layer. It also facilitates the swapping or changing different DBMSs without having to change either the presentation or business logic layer.

Another alternatives to ASP.net is Play framework. Play framework is an open source framework written in Java. Play framework is similar to ASP.net as it also follows the MVC pattern. But Play framework does not have the same level of support compared to ASP.net. As ASP.net is professionally supported by Microsoft there is a lot of documentation and tutorials on developing with the framework. Since Play framework is an open source framework, there is not as much learning material and literature available. Play framework also suffers from a higher level of bugs. ASP.net allows the team to create a flexible, scalable and modular application exploiting the use of an object oriented programming language and a large library of additional features such as object relational mapping and session authentication in a very short space of time.

There are also other framework alternatives like Java Server Faces (JSF) (Mojarra, 2015) and CppCMS but JSF would be incompatible with the Kinect and also lacks the large knowledge base that ASP.net includes. CppCMS (CppCMS, 2015) is also a viable alternative offering high performance and the ability to handle thousands asynchronous of HTTP requests but this framework has a steep learning curve and this could impact the development plan/timeline.

## Programming Language

C# is a proprietary programming language developed by Microsoft for the .NET framework. C# is an object oriented programming language (OO) based on C and derived from C++, offering OO features like inheritance and polymorphism, which the team is familiar with, this in turn can help speed up development time.

Java (Java, 2015) is a similar OO language developed by Sun Microsystems and Oracle. ASP.net can only be programmed in C# which rules out Java as a primary language. Java will still be required if an Android application is included in the project which makes use of a RESTful interface or web API.

C++ is a programming language also based on C. C++ is extremely fast and efficient as it does not offer a garbage collection service and can access hardware directly, but programming in C++ requires significant knowledge and can also take a significant amount of time. Although the performance benefit would be desired in the project, the lack of garbage collection and steep learning curve means C++ is not a viable option as a primary language. C# includes garbage collection services and memory management as standard and also includes some of the best features of C++. If required C++ code can be written and packaged as a dynamic link library (.DLL) file and imported into the C# project. The project team believes that programming in C# would be a positive challenge to help improve the team’s skill set and problem solving skills.

## Data Persistence - Database

For this project, MySQL will be used as the chosen database management software (DBMS). MySQL is a lightweight open source DBMS written in C/C++, making it very robust, fast and efficient. Since it is open source, various extensions/modules have been created adding extra functionality and improvements such as phpMyAdmin: a web based GUI for MySQL created with PHP. Oracle Express is a DBMS created by Oracle and written in Java. Although it is compatible, it is not as efficient and lightweight as MySQL. Since ASP’s MVC design pattern is being used it is straight forward to interchange and swap around DBMSs while only changing a small amount of code. Most DBMSs use structured query language (SQL) so there should not be any compatibility issues with a choice of DBMS.

ASP.net integrates a framework called Entity Framework (EF). According to Microsoft, 2015 (Microsoft, 2015): Entity Framework is an object-relational mapper that allows developers to work with relational data using domain-specific objects e.g. C# model. This reduces the need for developers to write time-consuming database access and object definition code in SQL. A developer can simply annotate a C# model class as an entity, mark a primary key and foreign key if applicable and the ORM mapper will create all of the required SQL. This is an extremely useful tool which allows developers to quickly make any new features or model code persistent, providing the developer with a more time to focus on more important aspects of the project. This feature is a big draw to web frameworks such as ASP.net and Play framework.

## 

## UI Design/Framework

### Twitter Bootstrap

This project will integrate Twitter Bootstrap (Bootstrap, 2015) as a foundation for the web application front-end. Bootstrap provides a uniform design template, with icons, HTML elements, JavaScript functionality and validation for the team to take advantage of and build a graphically pleasing user interface. Bootstrap also includes a responsive design, meaning a web application using Bootstrap can be viewed and resized dynamically depending on the device that the application was loaded on e.g.: the website will be mobile and tablet friendly out of the box. Bootstrap also includes cross browser compatibility out of the box which means the application will be displayed correctly across a broad range of browsers and conform to recognised international standards.

### JavaScript

JavaScript will be included in the front-end of the project to add an extra layer of functionality. Bootstrap will use JavaScript for validation of checkboxes and the JQuery, a JavaScript library will be used to implement graphical charting functionality such as Flot.js (Flot, 2014) and Morris.js (Morris.js, 2013). This will allow the front-end to represent the data from the server in a meaningful and graphically pleasing way. JavaScript and JQuery can also be used to implement design features like smooth scrolling, animation of elements, flashing warnings or errors, image carousels, light boxes and modals. Microsoft also provide a JavaScript library for displaying Kinect imagery on a HTML page so this can also be included as an extra feature.

## Data Transfer

One requirement of the system is sending data from a client to the server and vice versa. Using a RESTful interface which implements JSON this will enable the system to package data as strings and send it via HTTP to the server or client which can parse this information and display it in a meaningful way or rebuild the objects for further use. This will not only allow the hardware client to send information to the web server e.g. tracking session data, but it will also allow the team to add extra types of clients, regardless of platform or OS for example, allowing us to add an Android app or iOS app making it an ideal and crucial part of the system.

## Web Server

Microsoft Internet Information Services or IIS is the web server architecture this project will be implementing in its system stack. Developed by Microsoft (Microsoft, 2015), IIS is a GUI based web server developed with enterprise in mind. IIS incorporates security features like SSL/HTTPS and SQL injection blocking by default and the user interface makes it easy to manage and administrate.

Apache HTTP server is a web server alternative, developed by the Apache Software Foundation (Apache, 2015). Apache HTTP server is open source, process based and managed via a command line terminal compared to IIS which is managed via a GUI. To configure apache server, changes are made via configuration files whereas in IIS, changes are made via the GUI, making it an easier experience for a system administrator. IIS also includes built in integration with ASP.net where Apache HTTP server does not. A custom module called Mono (Mono, 2015) is required to be installed to ensure compatibility which may be limited and is not officially supported by Microsoft.

## Integrated Development Environment

Visual Studio has full integration with C# and ASP.net, making it an ideal environment for developing the project. Visual Studio has many features and includes advanced debugging and allows the team to easily include extra C# features like WebAPI etc., through the NuGet package manager. It is also possible to develop HTML, CSS and JavaScript in VS making it a perfect option for developing the web application. Visual Studio also integrates SVN and Git making it easy to push or pull any changes to the repository for source control.

## Source Control

### Git/GitHub

Git is a collaborative source control platform primarily for software development. Git allows software developers to share a repository where code can be stored locally or remotely on a server. GitHub is a service which allows developers to host their code remotely on a server via Git. GitHub includes features such as issue tracking tools that allows a collaborator to log issues and assign other members of the team to fix. GitHub also provides support for extensive documentation. Through GitHub Pages, it is possible to develop a small site from scratch or use pre-made templates and add wikis for knowledge bases.

# Release Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iteration** | **Stories** | **Duration** | **Start Date** | **End Date** |
| **1** | 1 - Start Tracking  2 - End Tracking | 40 Hours | 13th February  2015 | 28th February 2015 |
| **2** | 3 - Log in  4 – Create Account  5 - View Graphs  6 - Check alerts  7 - User Alert Management | 40 Hours | 28th February 2015 | 27th March  2015 |
| **3** | 8 – Delete Account  9 - Check Local Health Services | 40 Hours | 27th March 2015 | 1st May  2015 |

# References

Arduino, 2015. “Arduino Uno” Available: <http://arduino.cc/en/Main/ArduinoBoardUno>. Last Accessed: 12th Feb 2015

Apache, 2015. “Welcome! - The Apache HTTP Server Project”. Available: <http://httpd.apache.org/>. Last Accessed: 12th Feb 2015

CppCMS, 2015. “CppCMS -- High Performance C++ Web Framework”. Available: <http://cppcms.com/wikipp/en/page/main>. Last Accessed: 12th Feb 2015

Flot, 2014. “Flot: Attractive JavaScript plotting for jQuery”. Available: <http://www.flotcharts.org/>. Last Accessed: 12th Feb 2015

Java, 2015. “What is Java?” Available: <http://java.com/en/download/whatis_java.jsp>. Last Accessed: 12th Feb 2015

Microsoft, 2015. “ASP.net”. Available: <http://asp.net>. Last Accessed: 12th Feb 2015

Microsoft, 2015. “Entity Framework”. Available: <https://msdn.microsoft.com/en-ie/data/ef.aspx>. Last Accessed: 12th Feb 2015

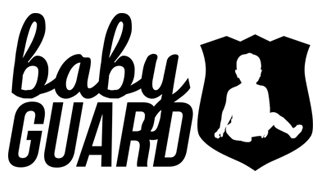
Microsoft, 2015. “Overview: The Official Microsoft IIS Site”. Available: <http://www.iis.net/overview>. Last Accessed: 12th Feb 2015

Mojarra, 2015. “Java Server Faces”. Available: <https://javaserverfaces.java.net/>. Last Accessed: 12th Feb 2015

Morris, 2013. “Morris.js”. Available: <http://morrisjs.github.io/morris.js/>. Last Accessed: 12th Feb 2015

MySQL, 2015. “About MySQL”. Available: <http://www.mysql.com/about/>. Last Accessed: 12th Feb 2015

Redmine 2015. “Redmine”. Available: <https://github.com/>. Last Accessed: 12th Feb 2015



Baby Monitor

*Iteration 1*

*Graham Healy - X00104195*

*Keith Doolan - X00105348*

Contents

[1. Iteration Progress 72](#_Toc413427089)

[Kinect Tracking 72](#_Toc413427090)

[Arduino Temperature Sensor 73](#_Toc413427091)

[Component List: 73](#_Toc413427092)

[Circuit 74](#_Toc413427093)

[Hardware Client Program 76](#_Toc413427094)

[2. Problems Encountered 77](#_Toc413427095)

[3. Changes to overall project 78](#_Toc413427096)

[4. References 79](#_Toc413427097)

# 1. Iteration Progress

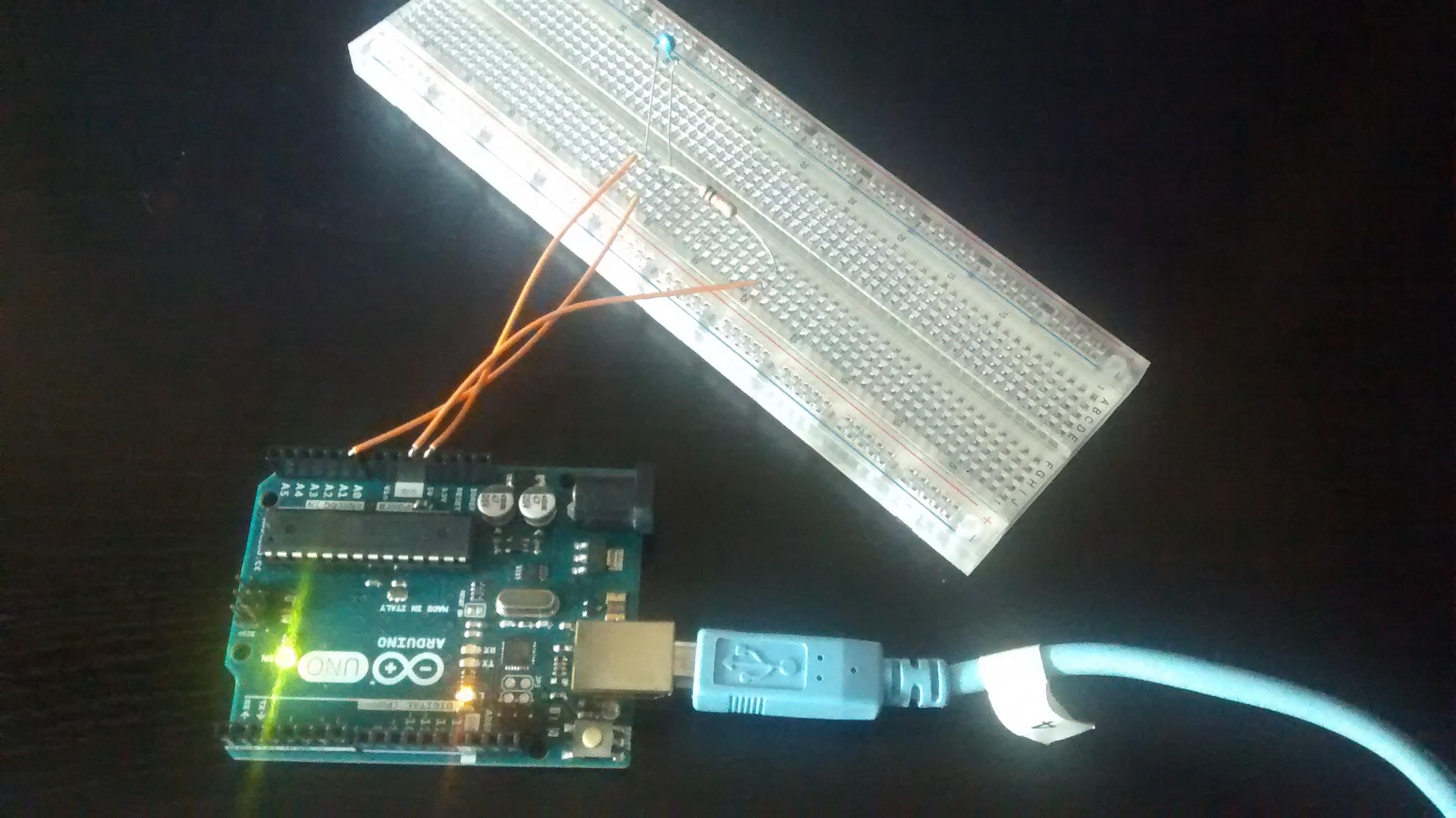
## Kinect Tracking

The Kinect tracking functionality is controlled by a class which is called Kinect. This class contains all the necessary functions and variables to allow the Kinect to detect motion.

The Kinect sensor is started and fires an event which returns a depth frame object. This object is then converted into an array of pixels which then can be used and manipulated. The depth is represented by a short and an additional player index is also included in depth information to help detect users (Only if skeleton tracking is present and enabled). On start up the first frame to the sensor is set as a reference frame. Every frame after that will be checked against this reference frame. The reference frame can then be set to update every x number of frames or after a certain time using a timer.

To detect motion, another array is created to hold the result of the frame comparison. The new frame fired by the event is subtracted from the reference frame. Since the pixel holds depth data, if the subtraction results in a 0, the depth has not changed. If there is a number greater than 0 then an integer counting the number of changed pixels is incremented. This number can then be converted into a percentage of the total size of the array, giving an adjustable threshold to fine-tune the granularity of the motion sensor and at what percent of changed pixels motion can be detected. This will also help to build in fault-tolerance and filter out any false-positives in the testing phase.

## Arduino Temperature Sensor

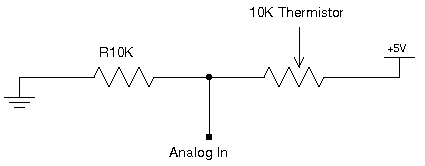


Above: The Arduino thermometer using a 10k resistor and thermistor.

### Component List:

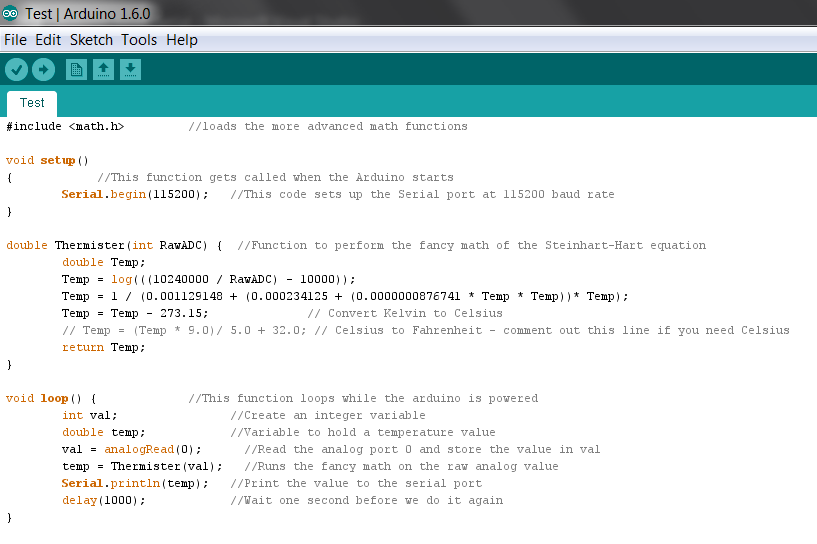
|  |  |
| --- | --- |
| Item | Quantity |
| Arduino UNO | 1 |
| USB cable type A/B | 1 |
| Breadboard | 1 |
| Resistor 10k ohm | 1 |
| Thermistor 10k ohm | 1 |
| Jumper Cables | 3 |

### Circuit



A thermistor is a cheap alternative to a thermometer and the idea was to get the Arduino functioning before more progress was made. This circuitry with its components converts heat from an electrical current to generate resistance. This resistance is then converted using the Steinhart-Hart equation to get a temperature. There is no direct correlation between voltage and heat that is accurate enough to give a temperature readings. So in this case the Steinhart-Hart equation measured against the resistance is more accurate with an error of less than 0.02%.

The code below is the source code used with the Arduino IDE. The syntax is similar to the C language or a derivative. There are two main methods or functions that are needed for an Arduino IDE program to work. The setup() function, which is the initial setup of the program and the Thermistor() function, which holds the main functionality.



Above: Thermometer code written in C++ in Arduino IDE.

The Arduino opens a connection via the USB port, simulating a serial COM port. This allows the Arduino to send in its resistance generated by the thermistor. This resistance is then converted with the Steinhart-Hart equation and returned. This can then be used by the hardware client to output to the windows form GUI and send the temperature to the service via a tracking state object.

## Hardware Client Program

A client program was created in C# to encompass all functionality for the motion detection and temperature logging. Based on Windows Forms, it has just one screen which displays if there has been any motion detected and the temperature. Since most user interaction will be with the web client this programs main responsibilities are transporting tracking session objects/details from the client’s machine/hardware to the system service, which will then store this information in a database and then display it on the client.

# 2. Problems Encountered

Since the Kinect does not officially support tracking for infants it was much harder to develop a motion detection algorithm. Instead of using skeletal data which would have made it much easier to track movement the solution was to develop a function from the ground up using raw depth data to detect motion. This allows for a more cohesive system where the Kinect class is solely responsible for tracking functionality, it also allows for extra fault tolerance and exception handling to be built in. There were some issues with the Arduino to begin with, but these related to incorrect components that gave incorrect readings. Once the correct resistors and thermistor was sourced an accurate reading was displayed.

During development some simple client/server functionality was implemented using TCP sockets to send and receive information. This is okay for primitive types such as strings and integers as it has to be converted from ASCII to bytes, but for large complex objects such as tracking state objects which hold data such as temperature and motion details this can be a problem. Also functions which call over the network could get complex such as CRUD (create, read, add, delete) functions. The solution chosen is to use a Windows Communication Foundation (WCF) service to run between the hardware client and other clients to act as a central hub or intermediary. The hardware client will send information to the service and the service is then responsible for storing and fetching this information. On top of that a RESTful approach can be implemented, simplifying the process even further. Using a RESTful approach HTTP verbs such as GET, POST, PUT and DELETE can be used to access and modify information in the service. This allows for cross-compatibility between operating systems, programming languages and devices as HTTP is one of the world’s most commonly used protocols. Using a restful service an Android app can request data from a C# service using a data interchange format such as JSON or XML. Real-time functionality can also be achieved in the web client using AJAX calls at certain intervals to get information.

# 3. Changes to overall project

Instead of using a server layer, a Windows Communication Foundation web service is going to act as an intermediary between the hardware client and other clients. It is going to be a RESTful web service (Roy Fielding, 2000) which will allow the program to use HTTP verbs such as GET, POST, PUT and delete to fetch and send information to be used/consumed by clients.

In Object Oriented Systems Analysis and Design (OOSAD2) the team learned about a design pattern called the factory class. The team feels that this would be an excellent addition to the system as it allows the hardware client to support multiple types of cameras. Using a factory class, the end user can select their type of camera from a drop-down menu and the factory class can return the type of camera object. This can be expanded to other types of hardware including the Arduino temperature sensor.

# 4. References

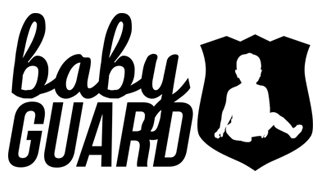
Arduino, 2015. “*Arduino UNO Microprocessor”.* Available: <http://www.arduino.cc/en/Main/arduinoBoardUno>. Last Accessed: 6th March 2015

Arduino, 2015. ”Thermistor: Example Schematic”. Available: <http://playground.arduino.cc/ComponentLib/Thermistor>. Last Accessed: 6th March 2015

Roy Fielding, 2000. “*Representational State Transfer”.* Available: <http://www.infoq.com/articles/rest-introduction>. Last Accessed: 6th March 2015

Microsoft, 2015. “What is Windows Communication Foundation?” Available: <https://msdn.microsoft.com/en-us/library/ms731082%28v=vs.110%29.aspx>. Last Accessed: 6th March 2015

Wikipedia, 2015. *“Thermistor: Steinhart-Hart equation”.* Available: <http://en.wikipedia.org/wiki/Thermistor#Steinhart.E2.80.93Hart_equation>. Last Accessed: 6th March 2015



Baby Monitor

*Iteration 2*

*Graham Healy - X00104195*

*Keith Doolan - X00105348*

Contents

[1. Iteration Progress 82](#_Toc416452011)

[Kinect Tracking 82](#_Toc416452012)

[Arduino Temperature Sensor 82](#_Toc416452013)

[Data Model 82](#_Toc416452014)

[Web Application 83](#_Toc416452015)

[AJAX 83](#_Toc416452016)

[SignalR 83](#_Toc416452017)

[Notifications 83](#_Toc416452018)

[UI/UX 84](#_Toc416452019)

[Security 85](#_Toc416452020)

[2. Problems Encountered 86](#_Toc416452021)

[Incorrect Temperature reading: 86](#_Toc416452022)

[SignalR 86](#_Toc416452023)

[Using XML over JSON in some instances. 86](#_Toc416452024)

[Entity Framework gave errors with Code First models. 86](#_Toc416452025)

[3. Changes to overall project 87](#_Toc416452026)

[Dedicated mobile application to be abandoned 87](#_Toc416452027)

[Data-Model Update 87](#_Toc416452028)

[4. References 88](#_Toc416452029)

# 1. Iteration Progress

## Kinect Tracking

The Kinect motion detection is now event driven. Initially the GUI would have to continually poll the Kinect class for any detection and temperature information. Instead the Kinect class will fire an event that will be handled by the GUI class. This means that any tracking info will be sent in real-time and ensures that there is no delay in the notification to the user.

## Arduino Temperature Sensor

The Arduino temperature sensor is now implemented correctly and is now gathering accurate temperature data in real time through the PC USB serial port. The data is then sent to the database and viewed on the web application.

## Data Model

A Session object has been introduced to the data model. This object contains a list of TrackingState objects. This allows for the logical grouping of tracking states thus, making it easier to manage and display for any client.



## Web Application

An ASP.NET web application has been added to the project. A user can log-in using OAUTH providers such as Facebook, Google, Twitter and LinkedIn. The user then has a profile that holds information such as tracking sessions and contact information.

### AJAX

A user can click a URL and be brought to a tracking page. This page will show all of the sessions for the user. This page uses JavaScript, jQuery and AJAX to make a request to the server which will return all of the sessions for the user. An AJAX request then runs at a specified interval to check if any new tracking information has been posted to the server. If an event has been sent to the server AJAX will add the new information to the page dynamically and in real-time without the user having to refresh the page.

### SignalR

SignalR is a library/package to add real-time functionality to .NET applications. SignalR has been implemented into the RESTful service and web application to provide real-time notifications to the user. When tracking information is posted to the service a SignalR notification is then fired to the appropriate user. This notification is then displayed on the users screen via JavaScript and HTML.

### Notifications

Push notifications have been implemented in this iteration. Notifications span over three different types.

#### Email

An Email helper class has been implemented to send email notifications. The Email is retrieved from the users profile in the database and a tracking alert email is then sent to the user containing information about the tracking event/incident.

#### SMS

Twilio has been integrated into the project. Twilio is a communications API for sending SMS and phone calls from an application. In the server when tracking information is posted the user’s phone number is retrieved from the database and an SMS message is sent to the user containing details about the tracking event/incident.

#### SignalR

SignalR has been implemented to send notifications to the web application from the RESTful service. Notifications sent by SignalR can be consumed with JavaScript/jQuery on the client side.

### UI/UX

The web application uses Bootstrap as a base framework. Bootstrap gives the application a professional and clean look and is easy to for users to navigate and read. Using Bootstrap also means that the application is responsive for all devices and screen sizes ensuring high compatibility between devices and web browsers.

The web application also users charts, these charts allow the user to view data gathered from the tracking information that has been triggered. The charts are currently implemented with morris.js. morris.js allows for more chart customisation, are responsive, aesthetically pleasing to the eye and will also be easier to implement with AJAX and JSON.

Google maps is also implemented into the application. The map is showing the city of Dublin and the emergency services that are available for children. The emergency services are currently hard-coded in using their longitude and latitude coordinates. This will be changed in time by using the google api to find the geolocation of the current user of the application. Then based on the users location nearby emergency services are displayed.

## Security

Security has also been implemented across the project. The RESTful service now uses HTTPS and SSL for transporting data instead of HTTP. The web application also now uses HTTPS and implements user authentication and authorization. Each user has a cookie which contains a unique token to authenticate users. This is then used on the pages that should not be accessed anonymously. This means that the page will only be shown if the user is authenticated.

## 2. Problems Encountered

### Incorrect Temperature reading:

The temperature was incorrect because a component of the circuitry was not of the same ohm as the other components and in turn the formula for calculating the temperature was incorrect. The issue was fixed when the right component was sourced from a local electrical components supplier

### SignalR

SignalR was difficult to implement on the RESTful service as the service had no way of managing connected clients. This meant that the service did not know where to send the notification when an event was posted. This was solved by overriding the methods when a user connects and disconnects. On the client side an additional parameter containing the user id was sent to the service and was stored in a dictionary/map, with the user id being the key and the connection id being the value. This meant that when a tracking state was posted, the service knew by the user id passed in where to send the notification.

### Using XML over JSON in some instances.

When making the AJAX calls on the web application, XML was difficult to strip down and retrieve information from. The code was complex so the solution was to use a JSON object instead. Using JSON objects it is simpler to extract information/data and efficient on the server.

### Entity Framework gave errors with Code First models.

The team also had problems with Entity Framework. When adding the Session object, Entity was having trouble with the retrieving the information from the database due to the Session object’s relation with the Tracking State object. The solution was found by modifying the code migration function’s order that occurs when changes to the data model are made. Once the order of the code migration was the changes were successfully committed to the database and the service was able to retrieve and return Session objects.

# 3. Changes to overall project

## Dedicated mobile application to be abandoned

With regard to the project deadline, the team agrees that the dedicated mobile application should be dropped in favour of the SMS and Email notifications currently implemented in the project. Using Email and SMS guarantees maximum compatibility and interoperability with user devices whereas a dedicated mobile application would be confined to a specific platform of users. Since the web application is responsive to mobile devices, the user would not lose any experience by not using a mobile application.

## Data-Model Update

Sessions have been added to the data model to allow for the logical-grouping of tracking information.

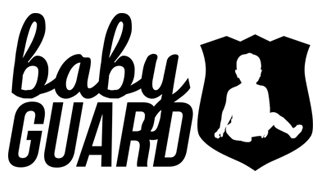
# 4. References

Google Developers, (2015). *Google Maps API*. Available: <https://developers.google.com/maps/>. Last Accessed: 22nd Oct 2014

Microsoft, (2015). *SmtpClient Class*. Available: <https://msdn.microsoft.com/en-us/library/system.net.mail.smtpclient%28v=vs.110%29.aspx>. Last Accessed: 5th April 2015

SignalR, (2015). *SignalR Asp.net* Available: <http://signalr.net/>. Last Accessed: 6th April 2015

Twilio, (2015). *Get started with voice and messaging.* Available: <https://www.twilio.com/user/account/voice-messaging/getting-started>. Last Accessed: 5th April 2015



Baby Monitor

*Iteration 3*

*Graham Healy - X00104195*

*Keith Doolan - X00105348*

Contents

[Design 91](#_Toc418274132)

[Logo 91](#_Toc418274133)

[Google Maps 91](#_Toc418274134)

[Loading Spinner 91](#_Toc418274135)

[Connection Notification Bar 91](#_Toc418274136)

[Webpage Animations 91](#_Toc418274137)

[Colour scheme 91](#_Toc418274138)

[Mobile First 92](#_Toc418274139)

[Charts & Timelines 92](#_Toc418274140)

[Widget 92](#_Toc418274141)

[3. Test Cases 93](#_Toc418274142)

[4. Problems Encountered 95](#_Toc418274143)

[5. Changes to overall project 95](#_Toc418274144)

[6. Database Diagram 95](#_Toc418274145)

## Design

### Logo

To make the end user feel secure that the application will succeed in its proposed objective of offering assistance in minding infants many logo names were circulated. After lengthy discussion Baby Guard was the name that was agreed upon and the Logo was then selected.

### Google Maps

Google Maps has now been implemented. The idea behind this was to allow the user to quickly find the location and name of the nearest emergency services. Using the Google map API with the extension Google services API it was straight forward in implementing the map into the web application. The Google Map API provided the tools to produce the map and geo location functionality while the Google services API provided the location of any services, in this case the local emergency services within a specified location based on the user's current device location. A radius based on the user's location was hardcoded into the application in a metric of meters to retrieve services within the specified radius.

### Loading Spinner

The loading spinner is implemented every time the user for functions of the application to complete processing. This is not a necessity of the application but it adds the illusion of real time responsiveness meaning the user does not think the application is unresponsive.

### Connection Notification Bar

In app notifications have been implemented using Noty. Noty is a JavaScript library for popping in and out toast notifications. Notifications have been implemented for the connection to the REST service and notifications will be fired when an event has occurred.

### Webpage Animations

There are web page element animations all around the web application. The animations vary from fading images to text sliding into place.

### Colour scheme

The colour scheme of the web application is based on Google Material Design. The design is attractive and easy to on the eye.

### Mobile First

The site is fully mobile responsive with web page elements and imagery changing size responsively to the size of the client device, e.g. mobile, tablet or desktop PC. A back to top button has also been added on mobile devices.

### Charts & Timelines

There are many charts and timelines found on the web application. The more events are triggered the more charts appear as the charts are produced per session. These have already been implemented but changes have been made to make them more responsive. An example of this would be the reading on the pie chart that increases when the pages is loaded.

### Widget

Another feature that adds some professionalism to the application is the twitter widget that is now under the email details of the developers on the contacts page. This is not so much a necessity but it adds a nice touch to the web application and it adds another method of contact the user can use to get in touch with the developers. This widgets can also be a source of information if users need to troubleshoot any problems they may be having.

## 3. Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Condition** | **Data** | **Expected Result(s)** | **Actual Result** |
| 1 | Select log in button  (scenario 1) | Name, E-mail etc. | Display restricted home page | User home page displayed |
| 2 | Log in using other services  (Google, Facebook) | Email and password for service | Display restricted home page | User home page displayed |
| 3 | Select tracking page | Retrieve temp and motion events per session | Display tracking page with sessions per user | Page displayed with sessions details |
| 4 | Select profile tab | N/A | Display manage account | Manage account displayed |
| 5 | Select, choose file button (upload Photo) | N/A | Windows Explorer opened to choose photo | Explorer opened |
| 6 | Upload button selected | User Image | User selected image displayed on profile | User image displayed |
| 7 | Select Baby Guard Logo | N/A | Initial home page displayed (before login) | Home page displayed |
| 8 | Select session | N/A | A breakdown of the session displayed | Session displayed |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Condition** | **Data** | **Expected Result(s)** | **Actual Result** |
| 9 | Select home page | N/A | View of home page | Home page displayed |
| 10 | View Google Map | Google Maps + Services API | Map with local emergency services displayed | emergency services displayed map |
| 11 | Select contact page | N/A | Developer contact details displayed | Details displayed |
| 12 | Select about page | N/A | Application details displayed | Details displayed |
| 13 | Select register account | N/A | Text fields displayed for user input | Text fields displayed |

## 4. Problems Encountered

The hardware client had problems with threading issues. The thread was registering the event handlers thus, notifications and alerts were fired twice. This was fixed by removing some event handlers.

## 5. Changes to overall project

The admin page has been dropped due to time constraints.

## 6. Database Diagram

7. Class Diagram



## 8. References

Needim, (2015). *Noty, jquery notification plugin*. Available: http://ned.im/noty/#/about. Last Accessed: 1st May 2015