JustHealth
CO600 Technical Report

Charlotte Hutchinson, Richard Logan, Ben McGregor, and Stephen Tate

***Abstract*—The JustHealth project has produced a platform designed to facilitate the connection between patients with long-term health conditions and their carers.**

The four key results from the JustHealth project are:

- **Secure Backend Platform/API Server**
- **Web Application**
- **Android Application**
- **Comprehensive Documentation**

Resolute project management and extensive software engineering skills have enabled the team to successfully achieve all initial aims. All implemented functionality has been repeatedly and thoroughly tested to ensure it works to create a full user experience. Fulfilling all the requirements classified as 'must' and 'should' using the MoSCoW method allows us to consider the completed project a success. JustHealth integrates with existing Android calendar and notification functionality to ensure patients and carers are reminded to take medication and attend appointments.

To facilitate interaction with the entire JustHealth platform and as project evidence, a comprehensive amount of documentation has been produced; including initial research of the healthcare domain, technical and user documentation of all functionality, and a vast library of planning, testing and project management reports.

Detailed in this report are specifics as to the background research and planning concerning both the project itself and the domain it operates in, details of the technical implementation of the project's features, as well as the conclusions that the authors have come to.

CONTENTS

LIST OF FIGURES

I. Introduction

Prescribed medication is often critical to a patient's health, but its also vital to the practice of healthcare and guaranteeing that the correct drugs are recommended for conditions. The objective of JustHealth is to be a unique platform that facilitates the relationship between patients and their carers in full or part time care. The system works seamlessly across both web and mobile (Android) systems. JustHealth allows patients and carers to track their prescriptions and appointments, complete with alerts and reminders.

This report details the technical output of the project, as well as the numerous processes, tools and considerations that JustHealth have used or made throughout the course of the project.

II. Background

A. Interest in the field

As a group we have a combined interest in the medical field. Stephen spent a year in industry working for GSK, considered the one of the largest pharmaceutical companies in the world. Charlotte's personal health experiences offered a different perspective, she considered the concept of the application to have huge impact potential and be extremely useful to users. The JustHealth team enrolled for the module eHealth (CO816), which we envisioned would compensate each other throughout the project.

B. Research

JustHealth committed to thoroughly researching the healthcare domain and application needs before gathering requirements. There were two primary areas of research during this time:

- Market Research
- Target Audience Research

Market Research was an essential component of the project as it gave us a substantial view of the current market situation, including the positive and negative features of competition. We took notes of the feedback and reviews available which then assisted with listing the initial requirements. Our choice to develop an application for the healthcare industry was based primarily on personal experiences; Stephen was in the unique position to gather primary research from people with experience in the development process of making a similar healthcare application. Having worked for a year in the industry at GlaxoSmithKline (GSK) who produce bespoke mobile applications for some of their products, Stephen was able to contact the product owners and gain access to some applications that had been retired from the Android and iOS App Stores. As well as this, Stephen was able to contact a friend who had experience deploying a mobile system to local district nurses working for the NHS.

This primary research enabled us to be able to gather requirements from an improved technical standpoint, combined with nurses that have been using a similar application to the one that JustHealth were looking to develop. In conjunction with the market research, we looked into other applications that were on the market as well as asking previously identified potential users to complete questionnaires, enabling us to ascertain the ideals that patients and carers would want from such an application.

The Target Audience research was another important aspect that enabled JustHealth to determine aims for an application that would be able to reach and positively impact the largest market sectors. Ultimately, this enabled us to decide on building our mobile application on the Android platform over others, such as iOS.

Additionally, we carried out a proportion of research into the type of methodology that we would use throughout the project. This was important in ensuring that the team was able to understand how we would work throughout the project and ensuring that what we had selected was fit for purpose for this project. We decided that an iterative development process would be best suited to the aims of the project.

III. Aims

A. Problem

The health industry is being revolutionised by technology, it currently has the fastest growth in development of new applications and more of the general public are looking for the latest assertive technology that can work with their health care plan. Personal experiences allowed us to analyse certain areas of health care, concluding that a significant problem existed with patients that are required to take a high quantity of medication for long-term conditions. There are many patients that forget to take medication, which accounts for a significant number of conditions being repeatedly misdiagnosed and prescriptions wrongly assigned to patients (Brown & Bussell, 2011). When patients have carers that are responsible for their well-being, this becomes a larger issue.

B. Solution

JustHealth anticipated that a bespoke prescription tracking application could be the solution to this issue. After completing research of the competition and target market, we outlined the functionality that would be important to measuring the success of the application. The primary aim of JustHealth was develop a platform that facilitated the relationship between a carer and their patients. The key functions to be implemented were prescription and appointment tracking, with personalised reminders and notifications for each user. This also included a patient management area for the carers and a user management portal for the JustHealth product owners or specified administrators.

C. Functionality

Through following an iterative development methodology, we applied some of the tools that are used specifically within Agile (Agile, n.d.). A key feature for outlining the functionality to be implemented is the MoSCoW method, which allowed us to categorise each element of functionality. The four categories are:

- MUST: This is essentially the functionality that has to work in order for the project to be considered a success.
- SHOULD: This is the functionality that isn't critical to the operation of the application, but may be required to meet some of the initial project aims.
- COULD: Any requirement that is considered a desirable but not necessary functionality, will be implemented if time and other resources permit.
- WON'T: Any requirements that we have identified as not critically important and wouldn't add value to the final product. May be requirements that are unrealistic due to time, cost or resource constraints.

JustHealth used this prioritisation method after separating our initial requirements into different groups of functional, non-functional and domain. Examples of our initial efforts can be found in Appendix A.

D. Target Audience

Using our research as a foundation for the decision, we identified that we do not need to generalise a specific target audience for the application. The main requirement being that the patient has an ongoing health condition and has a carer visit them in their home regularly. We have recognised that our target audience is not patients that are already in a care home, it would be too difficult to monitor usage and would not make best use of the JustHealth functionality. As a group we have decided that providing the application user interface (UI) is simple and easy to use, there is no need to restrict the target audience.
We have made the following assumptions before implementation of the application:

- A relationship between a carer and a patient already exists
- A Carer could have more than one patient
- When a patient ticks to say they have taken the medication, it has physically been taken.

IV. Technical Details

A. Introduction

This section of the report outlines the technical details of the JustHealth platform as well as the development process used to produce it. The architecture of the platform consists of a core API server connected to a database that handles all functionality. Clients can provide requests and send inputs to the server in order to interact with the offered functionality. As part of this project, a web and mobile Android application were also produced, in order to present and showcase the type of functionality that can be accessed, the different ways that data can be displayed or utilised by a user, and what strengths a specific type of client may have.

For a more detailed view of the platform's architecture, please see Appendix B.

B. API

One of the key focuses of JustHealth was to design the system with extensibility and existing industry standards in mind. As our idea of JustHealth was always as a **platform** that was able to support multiple client applications, it was logical to attempt to build an API that would allow access to functionality by any client application, leaving the platform available for third party development. We decided to implement this using the universal and well-documented HTTP format. This meant that we could build the Web and Android application without duplicating every piece of functionality, and also ensuring any other application, such as one for iOS, could be built easily and quickly by anyone, assuming that they would have access to the API listings / documentation and could make HTTP requests.

From a technical standpoint, the API server was built using the Flask framework. This is a lightweight Python framework for building web applications allowed us to easily specify URLs and routing, as well as how to restrict the different API calls to specific HTTP methods, such as POST or GET. Despite only one group member having used this before and our relative inexperience, the simplicity and ease-of-use of the framework really showed when the other group members quickly became proficient with its use. Many of our API functions were essentially interfaces for writing queries to our database, such as selecting, creating, updating or deleting prescriptions/appointments for the user. Results were provided primarily as either JSON, representing objects, or simply as short text strings when denoting success or error messages. When possible, error messages were as specific as they could be, but we utilised standard HTTP error codes where this was not appropriate.

The database connection was facilitated by the use of an Object-Relational Mapping (ORM) tool. We selected peewee, a Python ORM that integrates perfectly with Flask, and again is exceptionally simple to use. This allowed us to treat our database as another part of our Python application, with tables becoming models, and individual rows in the database represented as objects. Abstracting the database like this allowed us to work easily and created a malleable interface as well as providing a host of other benefits such as removing SQL which helped to simplify complex queries and allow for better security.

C. Web Application

Whilst the web application can be viewed as a separate client, just as the Android client can, from a technical perspective it is integrated with the API server and a part of the Flask application. It accesses the API methods internally, rather than via HTTP requests. However, this distinction is maintained in order to provide the emphasis that the API server is the core aspect of the platform which the client connects too.

A large proportion of the functionality within the JustHealth application, particularly notifications and reminders, lend themselves especially well to always-connected mobile devices, yet there remains a definite case for a web application. JustHealth has aimed to be as easy-to-use and unobtrusive as possible, but there potentially remains a fair amount of data entry required for prescriptions and appointments, specifically by the carer. The web application provides simple forms in order to do this, and mirrors all of the functionality available on the mobile application.

Another primary reason for creating a web application is its unparalleled user reach. The vast majority of devices have a web browser and that is part of the reason why the JustHealth website was designed to be as responsive as possible from the beginning. In turn, this allows JustHealth's platform to be automatically compatible with any device that has an available web browser, without needing to explicitly design a native application for a specific OS/device.

D. Android Application

JustHealth decided to build a mobile application on the Android platform as opposed to any other based on initial research. It showed that Android existed as the most widely used system and therefore, would allow us to be able to target as many users as possible. Initially, this seemed to be one of the most daunting parts of the project despite our combined experiences of Java programming, as none of us had any previous experience with using the language within an Android environment.

Initially, the learning curve was steep although we were able to quickly pick it up through using the official Android documentation and quickly learning from our mistakes. Our task from the outset was to implement the same level of functionality on the Android platform that we had implemented for the website. This was with an aim to ensure that a user would have a seamless experience irrelevant to the type of device being used (mobile, tablet or laptop).

The Android application relies heavily on the API that has been developed. As a result, standard practice for development required the functionality on the Android application to be implemented in the second week of any given iteration. This gave us chance to implement the API methods that the mobile application would use then POST request in order to perform the desired task.

Our biggest achievements in regards to the Android application were:

1) Push Notifications using the Google Cloud Messaging Service (GCM)
2) Integration with the Android Native Calendar
3) Asynchronous Processing

Firstly, the functionality for push notifications was extremely important and a considerable achievement. It required us to implement methods on both the JustHealth server and the Android application. The building of this functionality was very much an iterative process, it was initially important to ensure that we were able to retrieve the Registration ID of any given device as this was a unique ID associated to the specific application and device. In order to do this, it required us to implement methods on the Android application. Once this had been completed, we began to try and build an example notification using the python command line interface. Once a success message was being consistently received from the GCM service, we were confident that the phone was successfully receiving notifications but we were simply unable to see them due to a lack of methods implemented to display them.

Our next step was to implement methods that would show the notification on the phone. We built the GCM python file, which was similar to the API python file, but dedicated to dealing with the Android push notifications for simplicity purposes. In the GCM file, we implemented methods that both stored and deleted the Registration ID of a user's device, determined the title and the content of a notification and could finally push the notification to the relevant device. Lastly, we customised the methods for the Android application to ensure that the Registration ID was sent with the user details on the log-in submission, and removed upon user log-out. This all enabled the notifications to be displayed and for the user to be alerted correctly.

Another considerable achievement of the JustHealth application was the integration of the JustHealth appointment system with the native Android calendar. When a user adds an appointment on the Android application they have the opportunity to also add it to their native Android calendar. If they choose to do so, a method in the API automatically adds the appointment to their calendar. The ID of the appointment is then added to the JustHealth database to ensure that it can be referred to at a later date, should the user want to view, update or delete the appointment from the application.

Asynchronous processing was key to ensuring that the application was as quick as possible and it was also recommended for development standard practice in the Android documentation. As this standard practice wasn't something that we had ensured from the outset, it took time to revisit code and alter it successfully. However, it gave us the ability to be able to use a loading dialog to show the user that the application was working as expected and hadn't crashed in the process. We could ensure that areas containing multiple POST requests worked efficiently. For example, where the

log-in procedure makes two at the same time, there was a noticeable improvement in response time.

E. Security

Security has been a fundamental part of the JustHealth project and it is something that we have taken seriously throughout development, especially as the application exists in the healthcare industry. It is a significant area that we have implemented in the foundations of the application. We have treated the security of the application holistically and have used a combination of security techniques in order to ensure the security of the platform and confidentiality of user data.

Firstly, we have implemented strong password security, it has been used throughout the application to ensure that no plaintext passwords are stored in the JustHealth database or the mobile application. This has been done through the use of standard python encryption and hashing methods.

As well as this, we recognised the importance of using SSL/TLS in order to protect the confidentiality of data when transmitted. Due to the fact that we are running our own web server on raptor, we were unable to used signed certificates. In addition to this, Android would throw exceptions upon the use of self-signed certificates. Therefore, the decision was made not to implement SSL/TLS, although evidence can be found in the technical documentation which shows exactly how we would have implemented this had the application been hosted on our own server and not raptor.

HTTP Basic security was used on the JustHealth API, ensuring that only authenticated users are able to query the API. This was implemented on all publicly accessible methods except those that can be accessed without needing to be logged in, such as the registration process. Therefore, all other methods require the username and encrypted password of a registered user to be sent with any POST request to an API method.

Additionally, we apply further checks in order to ensure that the user that is querying the API is only able to perform requests with regards to their own accounts. If they are a carer, they are also allowed to perform requests with regards to their connected patients.

The JustHealth application utilises an Object Relational Mapper (Peewee) and therefore, does not use any SQL. This mitigates the risk of SQL injection completely. If data is sent to the database in the wrong format, it will simply be rejected by the ORM.

F. Testing

Testing was fundamental in ensuring that JustHealth was able to develop fully functioning web and mobile applications as well as a comprehensive back-end API. JustHealth adopted a test driven development approach and therefore ensured that test cases were written at the start of every iteration, covering every piece of functionality that was scheduled to be implemented. Following this, development would begin and each of the test cases were run once implementation was complete.

Initially, test cases for the first two iterations had been written in a Microsoft Word document template that was designed at the start of the project *(Refer to Appendix C)*. However, this became unmanageable for the team especially when it came to re-running tests that had failed. The formatting of the document would change unexpectedly, resulting in frustration and subsequently would be time consuming to fix. To overcome this, alternatives to this method were researched online. Upon discovering industry standard project management tools, we found that they were either made for a larger project or they were costly.

The decision was taken to build a testing portal in python using a backend PostgreSQL database. The team used the testing portal to record, analyse and run tests which worked effectively. Details of the testing portal and how it was utilised are detailed in the latest version of the JustHealth Test Plan.

JustHealth tested components implemented within each iteration using both automated testing and functional test cases. The whole application was tested again before the project fair to ensure that no functionality had broken since last tested. The process of automated testing adopted the use of an automated unit-testing package available within python. These were mainly used to test the API and enabled the team to ensure that errors were being caught, many that would not have been visible on the website or mobile application. The automated testing was very useful as it was much quicker to run than the functional tests when development changes were merged to the main project repository. Thus enabling us to run them more frequently and allowing a heavier reliance on the process.

G. Design

Fig. 1. JustHealth Final Logo

1) Branding: There are presently over 31,000 applications on the market that fall into the health & fitness bracket; an industry that is estimated to be worth over 600 million dollars. Although these are unregulated and can be giving misleading information, there is an obvious need to stand out from the market and be unique. JustHealth allocated Ben as the design manager and after a few hours with a sketchpad, discussing aesthetics of logos and designing an image to brand JustHealth with, we had an initial design to build on. We decided it was important to have a logo that included the company name, but

containing an image to separate from the main logo for future brand identification.

Whilst researching branding, it was noticeable that there were a number of factors to take into account; these included colours, fonts, size, capitalisation and letter spacing. Some factors are more crucial than others, but essentially branding is important to the success of the product and potentially gaining worldwide recognition.

User feedback regarding the overall image of JustHealth has been positive and we are proud of the brand that we have built thus far.

2) Application Design: The design process for the JustHealth application was continual throughout development, using our initial ideas to build on but careful consideration of user feedback. The primary reason for designing the way we have is our belief that in order to produce a high quality application, fulfilling the requirements outlined, it's important to have designs that give an idea of what an appropriate final result might look like.

Using wire framing design software, Balsamiq (https://balsamiq.com/), we were easily able to produce mock-ups to envisage a potential end product. The tool enabled us to adapt the designs for each iteration, and create new mock-ups when required, annotating them to provide extra information when implementing functionality. Producing designs this way saved valuable time and resources that we were able to allocate to other areas of development.

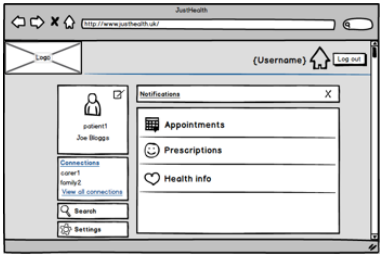

Fig. 2. Web Layout Design

H. Project Management

During the process of this project we followed an iterative development approach. Each iteration was a two-week cycle with an aim to have a 'finished' product at the end. Each iteration began with a planning meeting to outline the processes, create test cases to facilitate test driven development and create designs for the features to implement.

In the Autumn term, we aimed to meet at least twice, the first to plan, review documentation, discuss any previous work and reflect on the previous iteration. The second meeting would strategically be towards the end of the week, after our supervisor meeting to discuss the outcome and debrief. In addition to the two regular meetings, we would often meet to ensure that the iterative development pair programming standards were met.

As a result of a process review in the Spring term, and to ensure that we were following the iterative methodology process in detail. We regulated our meetings, establishing a mindset of attending shorter 'stand-up' meetings, with meticulous project deliverables.

All tasks for each week were put on Trello, a collaboration tool that organised the project into boards based on each iteration. Trello allowed us to track the statuses of assigned work. Whilst micro-managing wasn't necessarily needed, it allowed team members to view what was completed, in progress, assigned members and the deadline.

Microsoft Excel was used initially to create our Gantt chart, however, the initial stages of project management allowed us to quickly analyse that this was not an effective method of visualising individual project tasks. Through researching other tools to make better use of time and resources, the decision was made to use the free project management tool, ProjectLibre. As a management tool, it proved to work more effectively than Microsoft Excel, despite causing compatibility problems during the project. If resources had been available, it would have been preferable to use a more established tool like Microsoft Project as this would have solved our compatibility issues.

The entire project was controlled using GitHub; this is a web-based repository providing document version control and a source code management system that enabled effective collaboration amongst the JustHealth team. To provide an example of its use, through the process of implementing a feature, a branch would be created off the master repository. Once completed and fully tested, a pull request would be created to merge the branch back to the master repository. GitHub allowed us to effectively create and track issues, ensuring project bugs were monitored closely; easily prioritise them, assign group

Fig. 3. Android Layout Design

members and set realistic deadlines for fixes.

Below is outlined the 'ideal' process that we aimed to follow for every piece of functionality. We didn't manage to follow this 100% for every little piece of work, but for the most part it worked well.

1) Identify a piece of functionality to work on
2) Write a card on Trello to document this feature, who is assigned, when it is due and what needs to be done
3) Create a git branch from either master, or an existing parent branch for a wider piece of functionality
4) Write general test cases for this test. Here we would select the inputs and outputs we would expect from a piece of functionality
5) A developer, or pair, would work on this feature, committing work to the git repository
6) Once completed a pull request would be created
7) All tests would be run and the team would review the request
8) On completion of all tests, the pull request would be merged to the parent branch

Our goal with this development process is to ensure that anything on the master branch is deploy-able at any time. Through the running of tests and reviews on any pull request, we could attempt to ensure that master was kept clean of any bugs or issues. If an issue was found, it would be logged on the GitHub issues page, categorised and assigned as appropriate.

I. Performance Measures

As a team we tried to measure our performance continually throughout the project. At the end of each iteration we reviewed our overall performance and judged ourselves on the aims we set at the start of the two weeks. Over the Winter break we also held a review meeting where we deeply reflected on the term and looked back at the requirements we had set and how well we had met our goals. Our main area of performance assessment was looking back at our MoSCoW requirements. By the end of the project we aimed to complete all the Must and Should requirements and as many as possible Could requirements.

During the project we had two releases: Release 1 was over the Winter Break, and the applications were given to a number of 'alpha' testers that could give us some feedback on initial functionality and design. The response to this mainly resulted in a shift towards focusing more on design in order to provide a easier experience. Release 2 was prior to the Project Fair, once all core functionality was produced. Again, we asked people to try out the application and answer some basic questions in order to act as another performance measure, although responses to this could no longer influence our development.

J. Limitations

Our project has one main limitation; this is that we are unable to actually check that the patient physically takes the medication. As mentioned previously, we assume that if the patient confirms they have taken their medication then we can only assume this is truthful. Other limitations for our project are that certain areas of our target audience may struggle to use the technology we created, however in order to mitigate this as fully as possible we have tried to create both the web application and the mobile application with a strong focus on simple and clear user interfaces.

K. Future Developments

With our project there is a large number of opportunities for future developments. The way the platform has been created, with a focus of extensibility, makes it simple to create any native applications for any situation, such as an iOS application or for the support of wear-ables and smart watches. Other possible software developments include:

- Linking a patient to a pharmacy and informing them if they are low on medication stocks
- Specific lightweight API calls to better support accessories- enabling patients to take their own heart rate / blood pressure
- Text to speech functionality
- Emergency alarm button for patients to alert their carers
- A family and friends interface so such users are able to talk with the carer and track their relative's prescriptions/appointments
- Primary / Secondary Carers to account for holidays/time off
- Wheelchair request accessibility alerting (such as integration with TFL)
- Live chat service - a patient could be able to call or instant message their relative or carer

V. Evaluation

All of the market research that was completed before beginning the project allowed us to identify a gap in the market and consider what healthcare tracking products have already been developed and are available. JustHealth has been designed to be a unique product which combines a variety of different features that may be available elsewhere, but not as a single solution.

During the course of the project we tried to use third party libraries or tools that had already been built for specific purposes, in order to reinvent the wheel. Listed below are the major tools/libraries we used. A complete listing can be found in the Technical Documentation.

GitHub
: Version control and issue tracking

Trello
: Task tracking and project management

unittest
: Python library for automated unit testing

Flask
: Python framework for building web application

Android Libraries
: Numerous libraries, mainly for aiding presentation

GCM
: Google's Cloud Messaging Service to support Push Notifications to Android devices

BootStrap
: CSS framework for easy responsive design and consistency

moment.js
: JavaScript library for elegantly handling dates/times

In assessment of our project management technique, it could be argued that we 'reinvented the wheel' by creating the testing portal. During the course of the project we decided to change our method of testing from a text-based to data-based format. However, by this time, it was too late to change to using a test management tool such as Rally (https://www.rallydev.com/). As a result, we looked into a tool that provided an independent testing facility. As our research was not particularly fruitful, we decided to create a bespoke web application for our needs. This proved to be very effective and ensured all of our testing was in a simple format.

JustHealth solves the problem of carers having extremely busy schedules and too little time available in order to monitor their patients physically; JustHealth provides a virtual solution to this. The platform also ensures patients and carers are able to see and track both prescriptions and appointments, ensuring the previously mentioned compliance issues are not arising.

As previously mentioned, there is plenty of scope for JustHealth to continue to develop and expand in the future with many more features possible to implement, such as support for wearable devices and automated data entry.

Guidance we would give to others undertaking a similar project would be to research the target audience better and to further study the limitations of the health care domain.

VI. Conclusion

Our aim at the start of the project was to create a prescription and appointment tracking system for patients and their carers. Looking back at our initial requirements, we have met all of our Must and Should requirements and five of our Could requirements. The most important pieces of functionality beyond the core tracking systems are the notification and reminder systems that alert users in real time to what needs their attention. Along with accessibility features and providing a secure and safe environment patients are able to track and record their medication and appointments with their carers. All functionality within the project has been created to a high standard with a focus on efficiency and with the production environment in mind. Another successful factor of the project was the third party tools and libraries we used throughout the project.

A failure of the project was our failings in the area of test driven development. Despite our greatest efforts, at times we failed to follow the development method of writing all test cases before starting on development. Due to lack of time on certain iterations, we had to start development before finishing all the test cases, especially when there were issues with getting test cases to work correctly. Another failure on our project has been front-end web validation, which has been an ongoing issue almost from the very beginning. Despite a large amount of time being spent on this, we failed to conquer the cross browser issues that came into play between the different browser rendering engines.

Ultimately, we are all extremely proud and happy with the project we have produced. Having excellent feedback from the project fair, with the majority of people questioning whether we would take the project further and go into business really helped affirm our feeling. Both lecturers and students approved of the designs and functionality, believing it was presented in a simple and effective way.

Acknowledgment

The authors would like to thank Dr Yang He for her support and supervision throughout the project. As well as, the select group of people that provided user feedback which enabled us to significantly improve our application and the overall user experience.

References

Brown, M. T. & Bussell, J. K., 2011. *Medication Adherence: WHO Cares?* [Online]
Available at: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3068890/ [Accessed 10 February 2015].

Unknown, n.d. *Agile Methodology.* [Online]
Available at: http://agilemethodology.org/ [Accessed 15 February 2015].