

**Blood Pressure Monitor Desktop Application**

**Software Workshop Team Project**

Group Raleigh

Abdullah ALI 1848984 [axa1303@student.bham.ac.uk](mailto:axa1303@student.bham.ac.uk)

Shukri ALI 1083327 [saa727@student.bham.ac.uk](mailto:saa727@student.bham.ac.uk)

Thomas CUNNINGHAM 1864252 [txc754@student.bham.ac.uk](mailto:txc754@student.bham.ac.uk)

Xumin DING 1854515 [xxd715@student.bham.ac.uk](mailto:xxd715@student.bham.ac.uk)

Xinyi SUN 1786228 [xxs728@student.bham.ac.uk](mailto:xxs728@student.bham.ac.uk)

19th March 2018

School of Computer Science

University of Birmingham

Birmingham, B15 2TT

**Abstract**

This report covers the development of a multi-threaded Client-Server-Database architecture system for managing submissions of the blood pressure readings from patients to doctors.

This system allows for patients to report their blood pressure readings from their home computer so that their doctor can review them whenever necessary, rather than having to wait for a patient to deliver a form to their local surgery. This system updates the doctor in real time as readings come in and organises the data in a manner that is easy to understand for both doctor and patient. It also attempts to ensure that patients take their readings properly by providing helpful information on how to do so.

The main features of this system include: Remote submission of blood pressure readings to a database that can be easily accessed by the doctor, real-time notifications of submitted readings in the doctor’s client application, AES encrypted password transmission, user friendly organisation of data (graphing of submitted readings) and PDF exporting of information submitted.

**Statement of Contribution**

All team members have agreed with the following contribution on the software workshop team project that they worked on with:

Abdullah ALI: 20 %

Shukri ALI: 20 %

Thomas CUNNINGHAM: 20 %

Xumin DING: 20 %

Xinyi SUN: 20 %

Signatures: Abdullah ALI, Shukri ALI, Thomas CUNNINGHAM, Xumin Ding, Xinyi SUN

**Table of Contents**

[1. Overview 2](#_Toc509230396)

[1.1 System Description 2](#_Toc509230397)

[1.2 Background Research 3](#_Toc509230398)

[1.3 Requirements 6](#_Toc509230399)

[2. System Design 8](#_Toc509230400)

[2.1 Whole system 8](#_Toc509230401)

[2.2 Database 12](#_Toc509230402)

[2.3 Network 13](#_Toc509230403)

[2.5 Graphical User Interface (GUI) 15](#_Toc509230404)

[3. Testing 17](#_Toc509230405)

[3.1 Functional Tests 17](#_Toc509230406)

[3.2 Junit Tests 20](#_Toc509230407)

[4. Team Organisation 21](#_Toc509230408)

[5. Evaluation 22](#_Toc509230409)

[Reference 23](#_Toc509230410)

[Appendix 24](#_Toc509230411)

[Meeting Minutes – Raleigh 24](#_Toc509230412)

[Product Review 34](#_Toc509230413)

[View (windows) 38](#_Toc509230414)

[Diagrams 41](#_Toc509230415)

# 1. Overview

# 1.1 System Description

An innovative medical application that connects the doctors to their patients and enables doctors to receive effortlessly, their patient’s blood pressure readings. Currently, the way the blood pressure reading is obtained is a long process and requires staff to focus on paper work instead of more essential clinical work. Having a medical student in the team that has personal experience in this field made us realise the need for the app in the market now.

Blood pressure monitoring is very important and for some patients can mean live or dead situation and so the more efficient and accurate it can be done the better it is for all parties involved.

We assume the patient is provided with equipment that can monitor the blood pressure. Once the equipment has completed the reading, the patient is able to obtain three blood pressure readings and register that into the app with time and date recorded.

From the patient perspective, the patient is able to login into the application with the provided unique username and password. Once logged in, the patient has number of options. The patient can submit the reading by pressing a button, the reading will then be uploaded into the database. To ensure that patient is aware of the progress made over time they can view some indicators in numbers and graph displayed in their homepage. In addition, the patience is able to find instruction to perform the blood pressure reading and if needed can watch a video of how to obtain these readings.

From the doctor perspective, once logged in the doctor can search for patients based on their name or prefix. In addition, the doctor is able to see a list of patients that have completed their readings. The doctor should be able to select a patient, view their details and if necessary export a pdf file.

We will create an app that exists of three-tier architecture, which are the Client, a Server and a Database. This architecture will enable the system to have separate business logic, data storage and user interface increase the performance and scalability.

# 1.2 Background Research

Blood pressure is represented by two values. The systolic blood pressure, which is the pressure that the blood is forced through the blood vessels during a heartbeat, and the diastolic blood pressure, which is the pressure in blood vessels between heartbeats. Hypertension, also commonly known as high blood pressure is when either one, or both numbers are abnormally high. Estimates show that around 40% of the world's population has hypertension[[1]](#footnote-1), which is an enormous amount. Hypertension is a precursor to many life-limiting conditions like heart failure and kidney failure, so it is important to screen people regularly for this condition and to manage them appropriately.

Blood pressure measurements are difficult to record accurately in a clinical setting, as the anxiety and stress of the experience tends to give inaccurately high readings[[2]](#footnote-2). Therefore, to diagnose and manage hypertension, patients are required to monitor their readings at home for a given period; the average of which is used to confirm the diagnosis of hypertension.

Current guidelines suggest that a patient should record at least 2 consecutive readings, twice a day, for ideally 7 days (a total of 28 readings)[[3]](#footnote-3). Currently, most patients are given a paper diary, on which they record their blood pressure readings. The paper is then handed to a clinician, whom then manually uses those readings to calculate the average and determine whether the patient meets the criteria for confirmation of a hypertension diagnosis. As one can imagine, this is not only wasteful of resources like paper and printing but is also wasteful of the clinician and patient's time. By digitalising this process, we can save the time taken for the patient to manually take the readings back to the clinician, and for the clinician to manually calculate the average. Not to mention eliminating the risk of the patient losing their diary.

Existing Products Review

In this section, our aim is to analyse products and services currently available, for home BP monitoring. A point to be noted is that none of the products were found to solve the main problem we identified, which was of being able to transfer the readings directly to the clinician.

Product 1: Paper Diary[[4]](#footnote-4)

Please see all the images for the product reviews in the appendix in the section for product review.

Paper Diary is created by a leading Hypertension authority and most clinicians today use this or a variation on this style of paper diary for home BP recording.

Pros:

* Instructions on how to record the readings, written clearly; by experts in the field.
* Easy to use, with clear indication of how exactly to fill in the form, with addition of a comments section for any extra comments about that reading.
* Secure (can't be hacked)

Cons:

* Requires readings to be manually inputted and the average to be manually calculated
* Is not able to show visually, the patient's blood pressure trends.

Product 2: Blood Pressure Log – MyDiary (Android App)[[5]](#footnote-5)

This is an android app which can be used as a diary. This allows us to record blood pressure readings, which can then be displayed numerically, as well as graphically.

Pros:

* Patient can easily enter their readings
* Readings are represented visually as graphs of trends over time
* Extra information like location, posture, weight, and comments can also be added.

Cons:

* Very cluttered interface
* No way of transferring the readings across to the clinician

Product 3: Cardio Journal – Blood Pressure Log (Android App)[[6]](#footnote-6)

Pros:

* Very simple and visually pleasant graphical user interface
* numerical display for inputting readings, to prevent entering of invalid characters or in an invalid format.

Cons:

* This app also does not provide a feature for sending the readings to the clinician automatically.
* The app provides no information on how to take readings appropriately as well as when to take them.

In summary, research into existing systems has been very informative in highlighting features which will be potentially very useful in our project. It highlighted that the unique feature of our app will be the connectivity with the clinician, so this should be our main focus. We also realised that while the app should be clear and informative, it should also have a clean GUI which is easy to use and navigate.

# 1.3 Requirements

Functional

1.3.1 User login

* For login purposes, the system shall require the user to input his/her username and correct password.
* The system shall notice the user to try again account, when they wrongly input username or password, and show “wrong username or password!”.
* The system should check if the user has signed up, if not, show “user does not exist” message.
* The system should distinguish users from doctor and patient. The doctor should log into the doctor’s home page, while the patient should log into the patient’s home page.
* The system will encrypt the user password.

1.3.2 Patient

* The app should enable the patient to input 3 blood pressure readings, with date and time it was recorded.
* The app should enable patients to write comments with their BP and upload it.
* The database must be able to store patient blood pressure readings.
* The app should record all readings of patient and present them in a graph to show a progress.
* The app should enable automatic calculation of average value of all blood pressure readings they have uploaded and present it both in text and chart.
* The app should also calculate and indicate the highest and lowest BP recorded.
* The app should enable automatically update reading left once the patient upload his/her readings and present it in text.
* The app should update the graph instantly after the reading is submitted.

1.3.3 Doctor

* The app should allow doctors to create new patient account that contains all their information.
* The doctor should be able to set the blood pressure target for the patient.
* The database must be able to store basic patient information.
* The system should generate username and password of patients automatically when the doctor adds a new patient.
* The app should enable doctors to search a patient by name or prefix, and show their name, date of birth, and address in a patient list.
* The app should enable doctor to choose a patient showed in both lists by double click, and there should be a new window contains information (name, readings, etc.) of this specified patient.
* The app should show the patients that completed their readings in the completed list.
* The app should enable to generate PDF of patient information showed in patient view.
* The app should show in the patientView page a graph indicating patient progress.
* The should show the basic statistics of the patient BP progress in the patient view page.

Non-functional

* (Efficiency) The application shall provide a response to user input within 1 second.
* (Security) The application shall ensure the all the information (including users’ personal information, patients’ readings etc.) can only be accessed by the patients themselves and their corresponding doctors.
* (Availability) The application should be able to access at any time as long as users need.
* (Capacity) The system must enable concurrently support for multi users at a time.
* (Scalability) The system should allow as many doctors and patients to use the system.

# 2. System Design

# 2.1 Whole system

Our blood pressure submission system exists in a three-tier architecture. This architecture is composed from a Client, a Server and a Database, please see the component diagram. The client is responsible for making user driven requests of the server, which depending on the nature of the request, it will then query the database for the required information to return to the client application.

We constructed this system in distinct parts with specified functionality and interactions between each of these parts. With these specifications agreed it was therefore possible to work concurrently providing that each of us achieved the goals that we had set out, as we could be sure that upon completion each of the parts should produce a correctly functioning program. This also aided the testing of each components. We could more easily divide functionality into smaller units and test those, as well as testing for correct interaction between components in isolation.

Typical use of the system is as follows: Any user of the system is presented with a login screen as the view of the client application. The user will then input their login details, which will identify whether the user is stored as a doctor or a patient. The password which the user types is encrypted using AES and then wrapped in a serialised “TransmissionObject” which contains and carries the login information across the network to the server application. Once it has arrived at the server, the credentials are unencrypted and used to verify the user by querying the database for those credentials. If verified, the client will be returned either a “Patient” or “Doctor” type of the abstract “User” class, otherwise a null object is returned. Based on this information, the user is presented with a new window that is relevant to either a patient or doctor. A patient is then able to submit new readings and review the ones that they have recorded so far, as well as other minor functionality such as watching instructional videos on blood pressure readings. A doctor however can review all the patients that are currently stored in the database and the readings they have submitted. They are able to search for patients by searching for their name, and are notified in real time as patients complete their readings.

The communication between client and server is via a serialised class called “TransmissionObject”. This object manages all communication between the two, where it specifies what request/information is being provided via instance variables and encapsulates objects that contain information such as subclasses of the “User” object.

Due to the nature of the information that is moved throughout the system, computation of the information is primarily handled on the server, whilst the organisation of how it is to be presented is handled within the model. This ensures that information is stored and handled efficiently whilst contained within the database, but also that when the appropriate information is apprehended at the client it is presented to the user of the system in the most suitable and accessible way possible.

Use Case Diagram

A use case diagram was drawn up to visualise the whole system that described above, which can be found in Appendix. Five actors have been found, which are patient, doctor, server, database and BP application. The tasks and relationship between actors has also been connected.

Use Case Analysis

Two non-trivial scenarios were picked for the use case analysis:

* Track blood pressure readings
* Export blood pressure report

*Preconditions*

* The doctor has downloaded the BP application.
* The doctor has been registered and stored in the database.
* The patient has been added by the doctor and stored in the database.
* The patient has been given the username and password to log in the BP application.
* The patient has downloaded the BP application.
* The blood pressure monitor equipment has been provided.

*Flow of Events*

1. The patient goes to see the doctor talked about his physical status.
2. The doctor asks patient for his personal details and attempts to register the patient on the BP application to track the changes of blood pressure in the next 2 weeks.
3. The doctor gives the patients his username and password for login and urges him complete the BP readings.
4. On the next day, the patient starts submitting his BP readings twice a day whilst each submit contain 3 BP readings and recorded date and time.
5. The patient leaves some comments.
6. After submission, the blood pressure trend graph is plotted. The average BP reading and remaining reading no. will be printed out.
7. The doctor check daily to see the volatility of patient’ blood pressure over time.
8. The patient stops measuring blood pressure when the remaining BP readings no. counts down to zero.
9. The doctor generates a blood pressure report for the patient.

*Postconditions*

* All the BP reading statistics sends to the database.
* The patient who has completed the BP readings will be listed in the complete reading list on the doctor homepage.
* The doctor is able to access the patient information including the latest BP readings.

*Actors*

Actors involved in this use case are the patient, doctor, server, database and the BP application. The application is like the intermediary between patient and doctor. The doctor adds new patient on the application so that the patient can login and start track his blood pressure readings overtime.

*Scenario 1 - Track blood pressure readings*

Patient dummy has been with doctor from health check and has been advised to provide 2-week blood pressure statistics. The measure equipment has been given therefore patient dummy does not have to go to see the doctor everyday as he can measure the blood pressure by himself. Patient dummy wanted to start his first blood pressure recording, however, this was the first time for he to use blood pressure measuring equipment and it was hard to understand the instruction that come along with the equipment. Hence, he logged into his patient homepage using the username and password that given by the doctor and started watching the instruction video. He found it was very easy to follow and finish the first 3 BP readings. According to the doctor’s advice, he needed to measure twice a day. In Day 1 patient dummy submitted twice BP readings and 2 weeks later patient has completed all BP readings.

*Scenario 2 - Export blood pressure report*

Concerning about the health condition of patient dummy, doctor doc decided to introduce patient dummy using the BP monitor application to track his blood pressure in the next 14 days. He added patient dummy on add patient page by entering patient personal information. After the registration successful, the username and password for patient dummy were generated. (Precondition) Since the patient dummy started BP tracking, the doctor doc could see his patient dummy’s blood pressure on a daily base. He logged into his doctor homepage and started search patient name “dummy”. The patient dummy appears in the result list. He double clicked on the dummy, which open a window that shows all the blood pressure readings from patient dummy, where the statistics including average, highest, lowest BP readings etc. The doctor also viewed the comments the dummy left. 2 weeks later when doctor doc logged into his homepage, he saw dummy appeared in the complete reading list, meant that the patient dummy has taken his advice to finish the BP measurements. The doctor then clicked on the generate PDF button to export a blood pressure report for patient dummy.

**Prototype**

The initial prototype was built up to visualise the concept of the blood pressure monitor desktop application, based on the requirements which were proposed at the early stage of the whole project. The prototype of this application would include a user log in page that requires users, in this case would be either patients or doctors, to log in the application using their own username and password. (See Appendix)

Once patients/doctors log in successfully, another window will be opened to display their homepages, which varies from the type of users. The registered doctors will see the doctor homepage and the registered patients will be given the patient homepage.

On the patient home page, a relevant video and readable guidance are provided to help patient understand how to use the blood pressure monitor. The BP target that was prescribed by the doctors was printed on the screen. After measurements, the patient allows to input and submit the blood pressure readings, along with the recorded time and date, to track the BP trend on a daily base. The BP trend graph will be presented on the top of the page as a real-time update. (See Appendix)

Doctors are permitted to access the registered patients’ details through the doctor home page. From the prototype of doctor home page, it was given the functionality to search existed patients and add new patients. To add a new patient, the doctor needs to click on the “add patient” button, which opens a new window that the doctors can input all the details required for creating a new patient. To read the patient up-to-date blood pressure readings, the doctor can click on the patient he searched, which pops up a new window that displays the most important patient information on the screen, such as patient’s personal details, blood pressure target and date of the last reading etc. The BP reading trend graph that generated on the patient home page was also plotted on this page, sending an idea to the doctor that how the patients’ blood pressure keeps changing over time. Doctors can also view a list of patients who have completed the prescribed reading amounts on the doctor home page. (See Appendix)

# 2.2 Database

While deciding on how the database should be constructed, a significant decision was regarding whether we should be storing User data as whole objects, or as individual data. We decided to choose the latter as this allowed us to be able to manipulate specific user fields and query certain users via SQL, as opposed to loading the objects to memory, making the changes in JAVA and then storing the object back into the database. This allowed our system to be able to query data very quickly.

Our database contains two tables, please the ER diagram in the appendix. A users table, which stores all the users which have been created, as well as a readings table, which stores all the readings which have been submitted. A common user id field, which is present in both tables is then used to determine which reading belongs to which user.

Another consideration was whether to add the constraints at the database level for invalid data entry. We eventually decided that it is best for invalid data entry to be stopped at the point of entry in the client GUI, rather than sending it to the server and waiting for the server to respond with an error. Therefore, the database class methods require certain pre-conditions regarding the validity of the parameters, when data is being entered or retrieved from the database.

In order to link our java system to our PostGreSQL database, we were required to implement the JDBC API. Using this, we created a dedicated Db class which contains methods for any and all queries that the client may wish to make. Each time a request is received from the client, it is passed through from the sockets architecture, and calls the relevant method in the Db class. The methods in this class then execute a PreparedStatement query and process the result to return any required information. Prepared Statements are used for security purposes as they cannot be altered while being executed.

# 2.3 Network

The networking functionality of the application is divided amongst several classes that are all responsible for facilitating essential aspects of it. The system is divided into a three-tiered architecture of Client, Server and Database; however, this piece of the report is for the first two, with the Database being addressed in its own section.

Client

Much of the networking in the client is divided between the ‘Communicate’ and ‘Client’ classes. The ‘Client’ being actually responsible for what is sent and received, whilst the ‘Communicate’ class acts as a wrapper for easing the process of networking for the model classes.

The class ‘Client’ handles the process of connecting with the server via Java’s sockets functionality, making requests of it, and ensuring that protocol is kept to. If the model needs to make a request of the database or server, it calls on a method in ‘Communicate’. This method creates a ‘TransmissionObject’, which is a serialised object containing the information that needs to be sent to the server, as well as an instance variable specifying the nature of the request it. Furthermore, if the object contains a user’s password, it encrypts the password using AES with a preshared key between the server and client. Once the object is created in the correct format it is passed to the ‘Client’ class to be sent to the server.

The client contains a further peripheral piece of networking functionality that is responsible for keeping it abreast of updates in the server. When the ‘startUpdater()’ method is called from within ‘Communicate’ class and given an observable ‘Update’ object as an argument, it runs a separate thread that updates the ‘Update’ class whenever a change has been made. Being that this class is observable, it drives notification of the model whenever an update is found on the server, ensuring the GUI is displaying the most up to date information.

Server

The ‘Server’ class is responsible for setting up initial networking, and then producing threads for handling incoming requests. It also keeps in memory any changes that have occurred that clients should be aware of.

Upon first run, the server will attempt a port on the host machine that is read from a text file containing environment variables. If this port is not available for any reason it will iterate through five more ports to see if they can be used, if they cannot it will inform the user.

When a free port is found the sever begins listening for client connections. When it accepts a client, it assigns them a socket, and an ObjectDataStream for input and output, and then hands this to a thread to perform the request.

The thread is a runnable task that is managed by Java’s ‘Executor’ API. The task is submitted to a cached thread pool which will run the task until the request has been completed. These requests typically are for querying the database for information, and then returning the information to the client in a ‘TransmissionObject’.

The Server stores readings that are added to the database in a class named ‘Changes’. When the client checks for changes it provides an offset value to ensure that not all changes are sent, but instead the appropriate number.

# 2.5 Graphical User Interface (GUI)

The GUI is one of the crucial parts of the application and without it, it would be near impossible to have the patient and doctor interact.

As we are implementing the three-tier multilayer architecture, the GUI is the part that initiated and completes the requests of the application process. It is located on the client side and to ensure the maintainability of the application we have separated the model from view. To do this we have constructed a class called model which performs like a link between the business logic and the view classes. This will ensure that changes done to the view classes will not change the working of the logic and databases. This way, the view requests and receives data but will never influence the structure of the business logic of the application.

Patient and doctor both have the same login page and once they type in their username and password and press the login button then action is performed to check whether those details are correct, if not then a popup message appears stating that those details are incorrect. If they are correct the login page will be disposed and then the model will generate the correct user object to allowing the appropriate homepage to appear.

In the doctor home page, when the doctor searches for a patient details, we use the JList to show the result in a ScrollPane. After the doctor types the name or a prefix and presses the search button which activated the actionPerformed, we implement the setViewPortView method to show the result in the ScrollPane. In order for the doctor to view the selected patient details we have addMouseListener to allow for double click on the patient name then to display ViewPatient page with the corresponding patient details. The ViewPatient page mostly exists of JLabels that show the patient details which are mainly obtained through the patient object getters, except the blood pressure indicators which are obtained through the methods from the model class that calculate the numbers such as the overall average. In addition it shows the same line graph that we had on the patient home page. Additionally, when the doctor a new patient, the new page appears to create the patient account. We have placed number of constraints to ensure that the details put in by the doctor are aligned with the expected values. Once the submit button is pressed the details are added to the database, message appears that it is successfully submitted and then goes back to the doctor homepage.

In the patient home page, to keep the patient informed we have placed a line graph that shows the blood pressure progress over time with the additional line of the target blood pressure making the patient constant aware of it. Also, this will be in real time, when the patient submits their readings it will automatically update the graph.

We thought it was very important to implement a graph to make the application visually appealing. At first, we used JavaFX as it is part of the JDK, and allowed for creating very elegant graphs. However, after setting up the graph, we realised that JavaFX does not run on CentOS system, on which we will be demonstrating the project. For this reason, we had to completely redesign the graphs on an external API ‘JFreeChart’. The Graph displays the normal ranges, the trends of all blood pressures recorded, as well as the average blood pressure; making it easy to get an overview of the blood pressure.

In addition the page also displays the numbers of the overall average blood pressure, the target blood pressure and the number of readings that are left to be submitted by the patient. These information is obtained by requesting the details from the specific patient object and we use the getters in the patient class.

To add the readings, the patient types in the date/time, 3 blood pressures and comments, once the button is pressed it will call the actionPerformed method and perform a task. This then will call number of methods from the class constrains to check whether all the text field are filled and follow the constrains and if not, the patient will see a message asking to try again. If the constraints are satisfied, then it calls the addReading method in the model class from the Communicate class which is a networking wrapper for adding a reading to the databases. Then an update method is called that ensures that all methods that call for these data in the model class are notified.

# 3. Testing

# 3.1 Functional Tests

|  |  |  |
| --- | --- | --- |
| **Test** | **Action** | **Sent requests and expected Response** |
| 1 | On the log in page, click the “login” button with correct username and password. | A user request is sent to server to check the database; the input matches and a response is sent to login into the either a patient home page or a doctor home page. |
| 2 | On the log in page, click the “login” button with incorrect username or password. | A user request is sent to server to check the database; the input does not match, and a response is sent to inform the user that the login has failed with a reason explained in the pop-up window. |
| 3 | On the patient home page, click the “play” button. | A patient request is sent to server, a response is sent back link to the web site where the video is stored. |
| 4 | On the patient home page, click the “Read Instruction” button. | A patient request is sent to server, a response is sent back link to the web site where the instruction is stored. |
| 5 | On the patient home page, select a future date and time that recorded the blood pressure, along with correct inputs in the blood pressure reading fields and click “submit” button. | A patient request is sent to server, a response is sent to warn the patient the submit failed, with a reason of inappropriate date inputs, explained on the pop-up window. |
| 6 | On the patient home page, type non-numerical inputs into the blood pressure readings fields and click “submit” button. | A patient request is sent to server, a response is sent to warn the patient the submit failed, with a reason of wrong format of inputs, explained on the pop-up window. |
| 7 | On the patient home page, type numerical inputs into the blood pressure readings fields, with some empty fields left and click “submit” button. | A patient request is sent to server, a response is sent to warn the patient the submit failed, with a reason of “the field cannot be empty”, explained on the pop-up window. |
| 8 | On the patient home page, type numerical inputs into all blood pressure readings fields, where the systolic is over 300 or less 30 and click “submit” button. | A patient request is sent to server, a response is sent to warn the patient the submit failed, with a reason of wrong inputs, explained on the pop-up window. |
| 9 | On the patient home page, type numerical inputs into all blood pressure readings fields, where the diastolic is over 200 or less 20 and click “submit” button. | A patient request is sent to server, a response is sent to warn the patient the submit failed, with a reason of wrong inputs, explained on the pop-up window. |
| 10 | On the patient home page, input all the blood pressure readings to the corresponding textbox in correct format, along with the recorded date, time and any comments(optional), click “submit” button. | A patient request is sent to server, a response is sent back a “submission success” message in the pop-up window and plot the inputted blood pressure readings on the graph accurately. A latest average blood pressure reading is computed. The screen prints out remaining blood pressure readings for the patient. |
| 11 | When the remaining BP reading says “1” left, input the last readings data into all the fields and click “submit” button. | A patient request is sent to server, a response is sent back a “you have completed all your readings” message in the pop-up window and plot the inputted blood pressure readings on the graph accurately. A latest average blood pressure reading is computed. The screen prints out “0” remaining blood pressure readings for the patient. |
| 12 | When the remaining BP reading says “0” left, input the last readings data into all the fields and click “submit” button. | A patient request is sent to server, a response is sent back a “you already finished your readings” message in the pop-up window. |
| 13 | On the doctor home page, search the patient by typing in an unregistered patient name in the search bar, click “search” button. | A doctor request is sent to server, a response is sent to inform the doctor that the inputted name does not exist in the database in a pop-up window. |
| 14 | On the doctor home page, search the patient by typing in a registered patient name in the search bar, click “search” button. | A doctor request is sent to server, a response is sent to list all the possible search results in the textbox below. |
| 15 | Double click on a registered patient displayed in the result list that the doctor is looking for. | A doctor request is sent to server, a response is sent to open the new window – view patient page, which presents the chosen patient’s personal information collected from the database, including name, DOB, address and NHS number. Each BP reading that patient has entered is listed in the right bottom box in detail. The target, average BP, highest BP, lowest BP and remaining BP readings will be computed and displayed on the left bottom. An up-to-date patient BP trend graph is plotted in the middle of the page. |
| 16 | On the view patient page, click “generate PDF” button. |  |
| 17 | On the doctor home page, click the “add patient” button. | A doctor request is sent to server, a response is sent to open add new patient page in a new window. |
| 18 | On the add patient page, input the information with some empty fields left and click “submit” button. | A doctor request is sent to server, a response is sent to warn the doctor that the submission is failed, with a reason of “the field cannot be empty”, explained on the pop-up window. |
| 19 | On the add patient page, input the first name of the new patient in illegal format whilst all the other information is correctly entered, click “submit” button. | A doctor request is sent to server, a response is sent to warn the doctor that the submission is failed, with a reason of “the first name must be between 1 to 30 characters, explained on the pop-up window. |
| 20 | On the add patient page, input the last name of the new patient in illegal format whilst all the other information is correctly entered, click “submit” button. | A doctor request is sent to server, a response is sent to warn the doctor that the submission is failed, with a reason of “the last name must be between 1 to 30 characters”, explained on the pop-up window. |
| 21 | On the add patient page, input the number of readings left as “0” whilst all the other information is correctly entered, click “submit” button. | A doctor request is sent to server, a response is sent to warn the doctor that the submission is failed, with a reason of “the number of readings left must be between 1 to 30”, explained on the pop-up window. |
| 22 | On the add patient page, input the systolic of BP target as “400” whilst all the other information is correctly entered, click “submit” button. | A doctor request is sent to server, a response is sent to warn the doctor that the submission is failed, with a reason of “the systolic must between 30 to 300”, explained on the pop-up window. |
| 23 | On the add patient page, input the diastolic of BP target as “300” whilst all the other information is correctly entered, click “submit” button. | A doctor request is sent to server, a response is sent to warn the doctor that the submission is failed, with a reason of “the systolic must between 20 to 200”, explained on the pop-up window. |
| 24 | On the add patient page, input all the patient information to each corresponding textbox in correct formats, click “submit” button. | A doctor request is sent to server, a response is sent to generate a new patient, in the database and the new patient allows to log in to the application with correct username and password. |

# 3.2 Junit Tests

For JUnit test, please see our codes.

# 4. Team Organisation

Our project is split into four parts: database, server, client, GUI, each team member took a main part as following:

* Thomas: the networking between server and client.
* Abdullah: database and Collections
* Shukri: GUI
* Xinyi: actions performed in doctor part(model).
* Xumin: actions performed in patient part(model).

When combining each section, there was several modifications in client part and GUI part as following:

Abdullah: Add graph and PDF generating function in GUI.

Shukri: Separate GUI containing all actions into model-view-controller architecture.

To cover risk of team members absence, we have made sure that we shadow each other’s work. So, we have assigned Abdullah and Tom to work closely and Shukri, Xinyi and Xumin to work closely as their work was related.

The report organization is as following:

* Thomas: Whole System Design
* Abdullah: Background Research/Database
* Shukri: Introduction/GUI/Evaluation/Compiling the report
* Xinyi: Requirements/Team Organization
* Xumin: Prototype/Use case/Testing/Diagrams

Regular meeting twice a week on Monday and Tuesday. During the last week it became daily meeting. The meeting/project dairy can be seen in the appendix. In addition, the team constantly communicated through Facebook messenger.

# 5. Evaluation

We believe that we successfully implemented all functionality we planned at the start of the process and the app can be used in real life. Given more time, we would like to add additional functions such as messenger for the doctor and patients and total encryption of the object, but time did not permit us to do so. Creating this app was definitely a learning experience and an opportunity for personal growth as we were working with people with different personalities and at the same time implementing what we have learned through the year.

On the whole, the team worked well together, we were able to share knowledge with each other and were able to delegate work in such a way that could maximise the strengths of every member.

Time management could have been improved to allow for early finish, but we did manage to finish on time and we are satisfied with the final product.

# Reference

WHO. *Raised blood pressure. http://www.who.int/gho/ncd/risk\_factors/blood\_pressure\_prevalence\_text/en/ (accessed 26 February 2018).*

Beevers G, Lip GYH, O’Brien E. Blood pressure measurement : Part I—Sphygmomanometry: factors common to all techniques. *BMJ : British Medical Journal* 2001; 322(7292).

* NICE. *Hypertension in adults: diagnosis and management. https://www.nice.org.uk/guidance/CG127/chapter/1-Guidance#diagnosing-hypertension-2 (accessed 26 February 2018).*

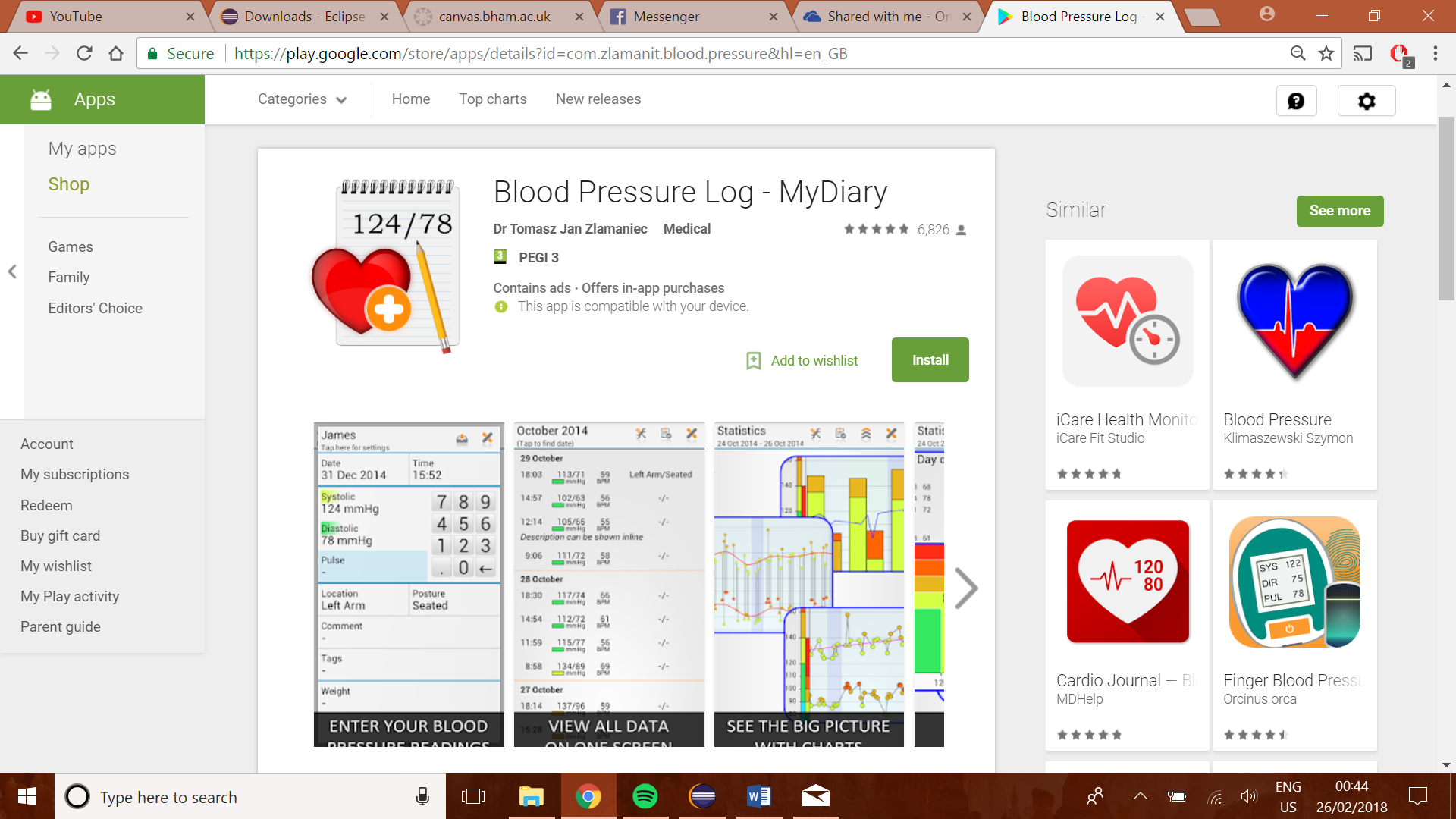
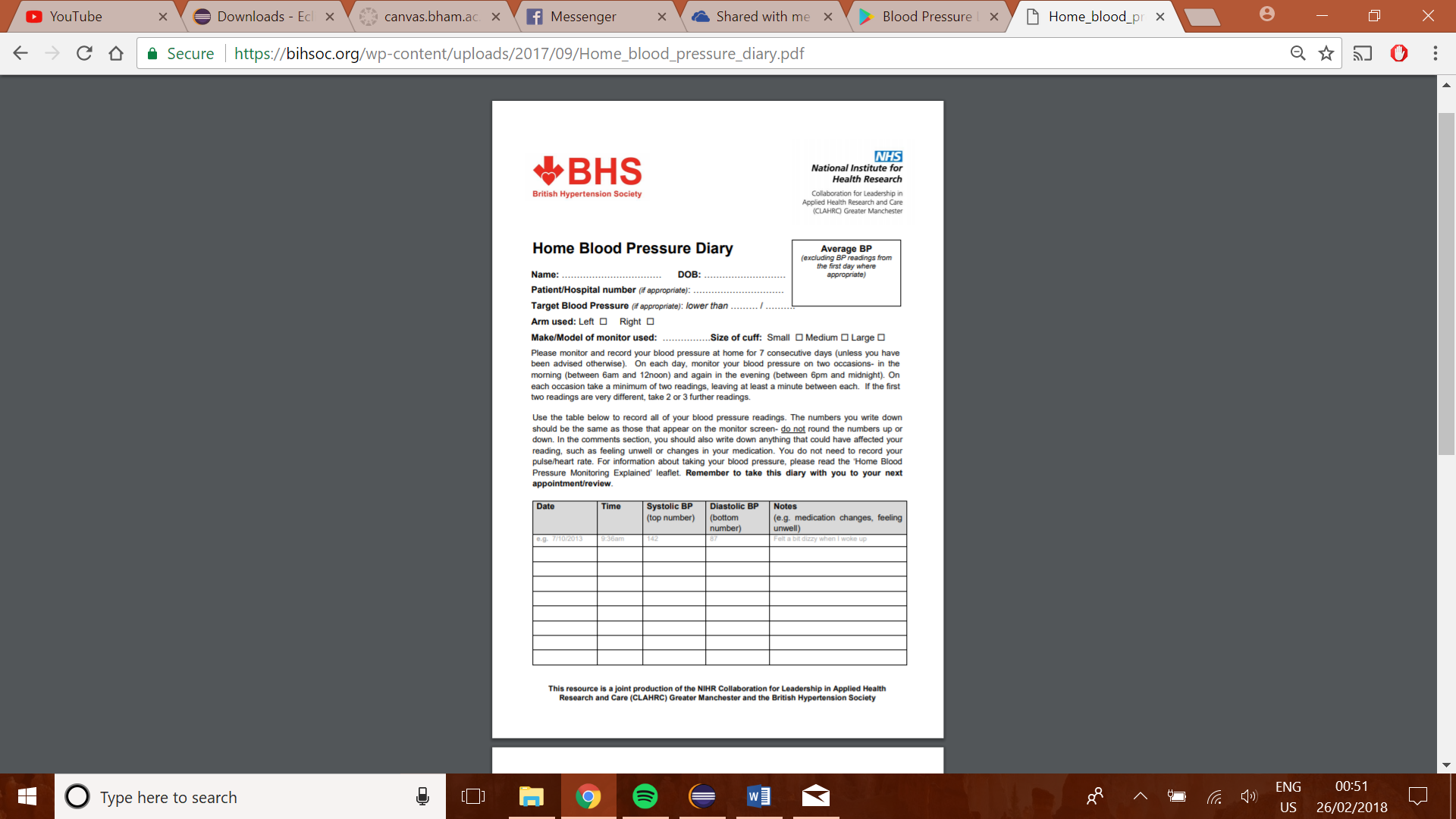
# Appendix

## Meeting Minutes – Raleigh

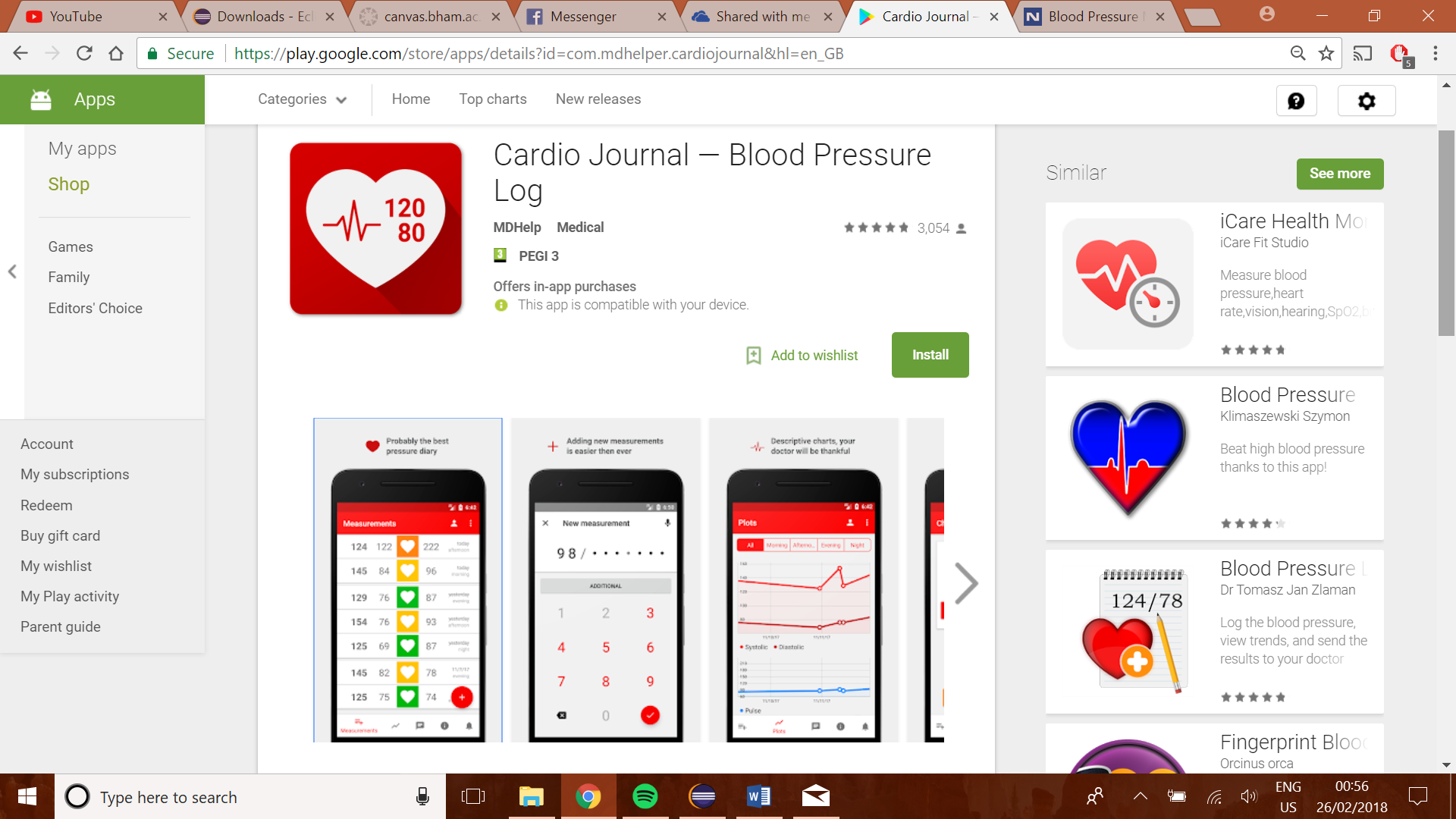
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date** | **participants** | **Actions taken or agreed to be taken** | **Progress** | **Next steps** | **Next meeting date** |
| 19/02/2018  16:00-17:00 | Abdullah, Shukri, Thomas, Xinyi, Xumin | * The team discussed what everyone’s background is and their skills. * Abdullah and Thomas are stronger in coding, so will take the lead on the application component but every will be contributing to application somehow. Xumin and Xinyi will be focussing on the research and analysis. Shukri volunteered to be the secretary and wants to work on the GUI. * Number of project ideas were discussed but the group decided that the best idea was a medical application where the patience is able to register their blood pressure online and the doctor can receive it efficiently instead of processing through paper. Our back up plan is the Client-Server Messenger Project. * Everyone needs to look more into the part they want to take in programming of the app * Research application architecture and other similar ideas in the market. * Ask the tutor about the server part of the project. | N/A | * Present to tutor the ideas for the project * Set up plan and sub plans for the project with mini deadlines. | 20/02/2018  15:00-16:00 |
| 21/02/2018  14:00-15:30 | Abdullah, Shukri, Thomas, Xinyi, Xumin | * We briefly explained our project idea to Cory and he thought it was good idea to proceed with it. * Thomas will be the point of contact. * The group discussed the different components of the application, such as class user, class patient, class doctor, class reading, databases, GUI, Server, sockets, threads and the graphics. * We divided the work and decided that Xumin will work on class reading and the diagrams for the application. Xinyi will work on User classes (patient and doctor) and the requirements. Shukri will work on the GUI and oversee the report. Thomas will be working on the database part. Abdullah will be working on the server and background research. | -We finalised project idea.  -We roughly divided the work and now everyone has a task to work.  -we have better overview of how the project looks like. | -on next meeting we will be checking the progress made  -research needs to be completed on the project, server and databases. | 26/02/2018  16:00 |
| 26/2/2018  15:00-16:00 | Abdullah, Shukri, Thomas, Xinyi, Xumin | * Xumin done the use case diagram based on the points we have discussed and after seeing the diagram we only had to make minor change of the notification part of the app, which we all agreed we will leave that part out. * Abdullah discussed the background research he done about other apps available in the market. He found 2 mobile apps and showed the group. This gave us a better few of the features we should focus on for the app. * We spoke about the important features of the apps, which we decided on that the app should be a desktop app, should have the function for the patient to enter BP readings, follow the progress of the BP level over time, instructions of how to do the BP reading, the doctor side should have the function to look for patients, add patients with required number of readings, exporting patient records pdf. * Patients details should exist of name, DOB, ID, username, address, medical staff assigned, BP reading, BP target, optional NHS number. This part includes some encryption work. * The app should start with a login page that recognises if the user is a patient or a doctor and the loads the right page with the correct details. * The reading should be entered as three readings each existing of **systolic and diastolic and the date and time it was recorded.** * We draw out in details how GUI part should look like to incorporate the features we discussed. We have agreed that when searching for patient it should have a dropdown list. * Shukri wrote brief specification of the project. She said she done the login page, including the part that connects with the DB and checks the login details. However the group said that it was Abdullah that is doing the DB side of the app and Shukri should create the GUI without a function for now. * Thomas is still working on the networks part and trying to figure out how the sockets works. * Xinyi is still working on the requirements and will finish on Monday. | -We have a clear picture of all the features that have to be included in the app  -we have reviewed other similar apps or ways.  -app specification  -use case | -Abdullah is going to complete the background research  -Xinyi will complete the requirements  -meeting with Cory  -Xumin will work on the design of the prototype on paper | 27/02/2018  14:30 |
| 27/02/2018  14:30-14:50 | Abdullah, Shukri, Thomas, Xinyi, Xumin | Cory mention few points for us to consider,   * We need sort of interaction between clients * Then wait for someone to connect to the socket * You have threads to various sockets * Cory explained a way we can get concurrency. First, we have an open socket and that sockets waits for someone to connect. So, when a thread connects to that socket, automatically another socket opens and waits. * For the DB, all have different connections * For IP address, you need to be able to change it * The group asked about the SVN. Corry explained we can login with our own CS login and all work should be saved there even text. * We asked about the work limit and he said the project specification should be no more than 2 pages and the rest should be about the app documentation. * Cory mentioned that to pay attention to the real time interaction, for example multiple doctors reading report from the same patient. * We asked about the complexity of the app and adding encryption. Corry said that would be good aim. | -We have better understanding for the network side. |  | 27/02/2018  15:00 |
| 27/02/2018  15:00-16:00 | Abdullah, Shukri, Thomas, Xinyi, Xumin | * After our meeting with Cory, we have decided to continue with our meeting. * We wrote down all the classes we need, such as the user, patient, docter, reading, etc. We discussed the methods we need for all the classes to work together, like getUser(), getReading(). This will enable Xumin to create the class diagram. We made sure we discussed how the classes interact with each other. * Thomas managed to connect to the server * We have decided that Abdullah will create a SampleObject class to test the methods and database without the Server, while Tom is working on the networking part. | -Abdullah completed the background research.  -Xinyi has completed the requirements.  -Xumin has completed the design of the app on paper.  -we have clarified all the methods that are needed for all the different components, like the getUser() method.  -we have agreed that everyone should have their part of the code done on Monday, so that we can integrate all. | -we will integrate the codes and see if the base features work.  -Abdullah will have the base of the SampleObject class completed | 5/3/2018  15:00 |
| 5/3/2018  15:00-16:00 | Abdullah, Shukri, Thomas, Xinyi | -Abdullah started on the database and created some the base classes for the user, patient, doctor, etc.  -Tom has started on the networking part of the application and is now able to send and receive objects, using threads. Also, he enforced communication protocol and objects will be automatically serialised.  -Xinyi started working on the methods for the login page on the GUI side when the button is pressed. There was some misunderstanding, so Xinyi has been working on the network functionality which was Tom’s part.  -Shukri has completed the most the GUI but the group made suggestions to improve the GUI.  -We have decided to work from the GUI components and have the methods there and later divide in to MVC structure.  Xinyi and Abdullah worked on number of methods such as the searchUser(), addUser(), etc.  -We discussed how we can have the patient details be updated in real time. Number of possible solution were mentioned, having an interval update or we can research the chat app to see how that works. | -the base for the GUI is complete.  -the base for the network has been completed.  -the work on the database has started  -Xumin has created the initial draft of the class diagram  -Abdullah has established the SampleObject class | -Xinyi will complete the necessary codes for the login page.  -Shukri will implement the new changes to the GUI  -Abdullah will continue building the database and create more methods for the model to access data  -Tom will work on the Comminicate class which process the model classes | 6/3/2018  14:00 |
| 6/3/2018  14:00-16:00 | Abdullah, Shukri, Xinyi, Xumin | -Xumin has worked on a frame that contains a line graph that can show the diastolic and systolic progress over time. However, it did not connect to the data and she was not sure how to do it. Abdullah offered to have a look and see if he can sort out that.  -we have found out the GUI looks different in in different systems and so Shukri will be working on this.  -The doctor side of the GUI needs additional page to view patient details.  - also needs to add patient page to create for the patient.  -Xinyi has to add a JList for when the doctor searches patient and also for the patients that have completed their readings.  -Xumin noticed that most of the constraints are missing in the textFields, this could cause problems in the database and so she will be working on that.  -Abdullah suggested that we should also display the average BP, highest level and lowest level. Shukri will add this to the GUI and Xinyi will work on the model side to create the methods.  -Everyone is having issues with SVN and finding it difficult to work on. Abdullah will help everyone with their SVN issues. | -Xinyi has completed the login page.  -Shukri made the changes to the GIU  -Abdullah added more methods in the SampleObject  -Tom continues working on the client side networking | -Shukri will make the gui more consistent over different OS  -Shukri will create additional GUI to add new patient and to view patient details. Also, will include the BP statistics to display to patients  -Xumin will create the constraint class  -Tom continues working on the client  side networking | 7/3/2018  12:00 |
| 7/3/2018  12:00-14:00 | Xinye, Shukri , Abdullah | -Shukri has implemented all the functionality to the patient homepage to the database and the model. She has also added additional methods.  -The plan was to try and see if the basics of the application is working through the server but unfortunately the university is having internet connectivity issue. After hours of waiting, we have tried to login but we found that there was server issue.  -As Tom was available over the phone, we informed him and he said he would look in to it. We have tried again connect to the database through our sampleObject class that acts like server. This was successful, and the basic parts (login, doctor homepage and patient homepage) of the app worked as expected.  -Abdullah informed us that it would be difficult to make Xumins BP graph work with the database and will attempt to construct a different graph. | -Shukri continues improving the GUI and has implemented some methods.  -Xumin created the constraint class  -Tom the basic functionality for the networking side on order to test if the app works. | - Tom is going to work on the server side of the app and try to locate the issue.  -Shukri and Xinyi will be working on the other page to get their functionality work.  -Abdullah will be working on the graph. | 9/03/2018  10:00 |
| 9/03/2018  10:00-12:00 | Xinyi, Shukri, Abdullah | -Tom informed the group that he fixed server. The team was able to successfully connect as number of different clients with the server on lab machines.  -we have tried to see if all the functionality of the app worked and found that some of the constraints and not entirely implemented. Xinyi said she will work on this.  - group agreed that adding JCalander and JSpinner to select the DOB, time and date will be the app user friendly. | -Tom fixed the Server  -the app mostly works  -All the pages are complete and can be tested  - Abdullah is making progress on the graph using javafx on swing  -Xinyi is going to add additional constraints. | -Abdullah will complete the graph  -Shukri will be working adding the JCalander and JSpinner to the Patient home page and NewPatient.  -Xinyi and Xumin will test all the constraints and add new ones if necessary. | 13/03/2018  14:00 |
| 13/03/2018  14:00-16:00 | Abdullah, Shukri, Thomas, Xinyi, Xumin | -Xinyi completed the doctor homepage with the patientView class and addPatient class.  -Abdullah will create another method for getCompletePatient with readings.  -Tom has all the methods that are in the SampleObject in client classes.  -Tom will be working on the update method that ensures that the once the patient submits the readings it instantly changes the view on the page. He will also work on the update class  -Shukri will implement the observer in the model. She has also created a document with the group of the report is structured and the work is divided in terms of the report. | -Abdullah Graph is the correct reading  -Shukri completed the JCalander and JSpinner but time is not sending the correct format  -Xinyi and Xumin have updated the constraints. | -Abdullah will try to format the JCalander in the correct way.  -Tom will be working on the update method for the real time interaction.  -Shukri will implement the observer in the model. | 14/03/2018  13:00 |
| 14/03/2018  13:00 – 16:00 | Abdullah, Shukri, Thomas, Xinyi, Xumin | -Xumin worked on the diagrams for the database and the server.  -Abdul Has added the pdf class but the pdf functionality needs to be added. He also found an issue with the graph that it does not work on the lab machines.  -Tom said it would be difficult to have with the timeframe everything encrypted and so will be using encryption for the password.  -Shukri has completed the new GUI look without functionality and so Shukri and Xinyi will be working to add the functionalities from the old classes.  - | -Abdullah fixed the JCalander  -Tom has created the update class  -Shukri implemented the update class but its working  -Abdullah added a pdf class | -Tom will fix the update class  -Shukri attempt the pdf  - Xinyi and Xumin will work on the new GUI design  -Abdullah will fix the graph issue | 15/03/2018  13:00 |
| 15/03/2018  13:00 – 16:00 | Abdullah, Shukri, Thomas, Xinyi | Tom completed the encryption part of the project.  Shukri has worked on the GUI, tried to implement the business logic but Abdul will take over this part. She has also added some method to the gui to complement the new components. Xinyi added also more methods to doctor side to reflect the new gui design. | -Shukri can create the pdf file but can’t get the bp readings into a table  -Tom has now implemented the encryption for the password and fixed the update class  -Xinyi and Xumin have made the new GUI functioning | -Abdullah will attempt to finish of the PDF  -Junit Testing will be done. | 17/03/2018  13:00 |
| 17/03/2018  13:00 – 16:00 | Abdullah, Shukri, Xumin, Xinyi | -Everyone has now completed their part of the report and send it to Shukri to compile the report.  -The testing has now been completed  - the application is working without error. |  |  |  |

## Product Review

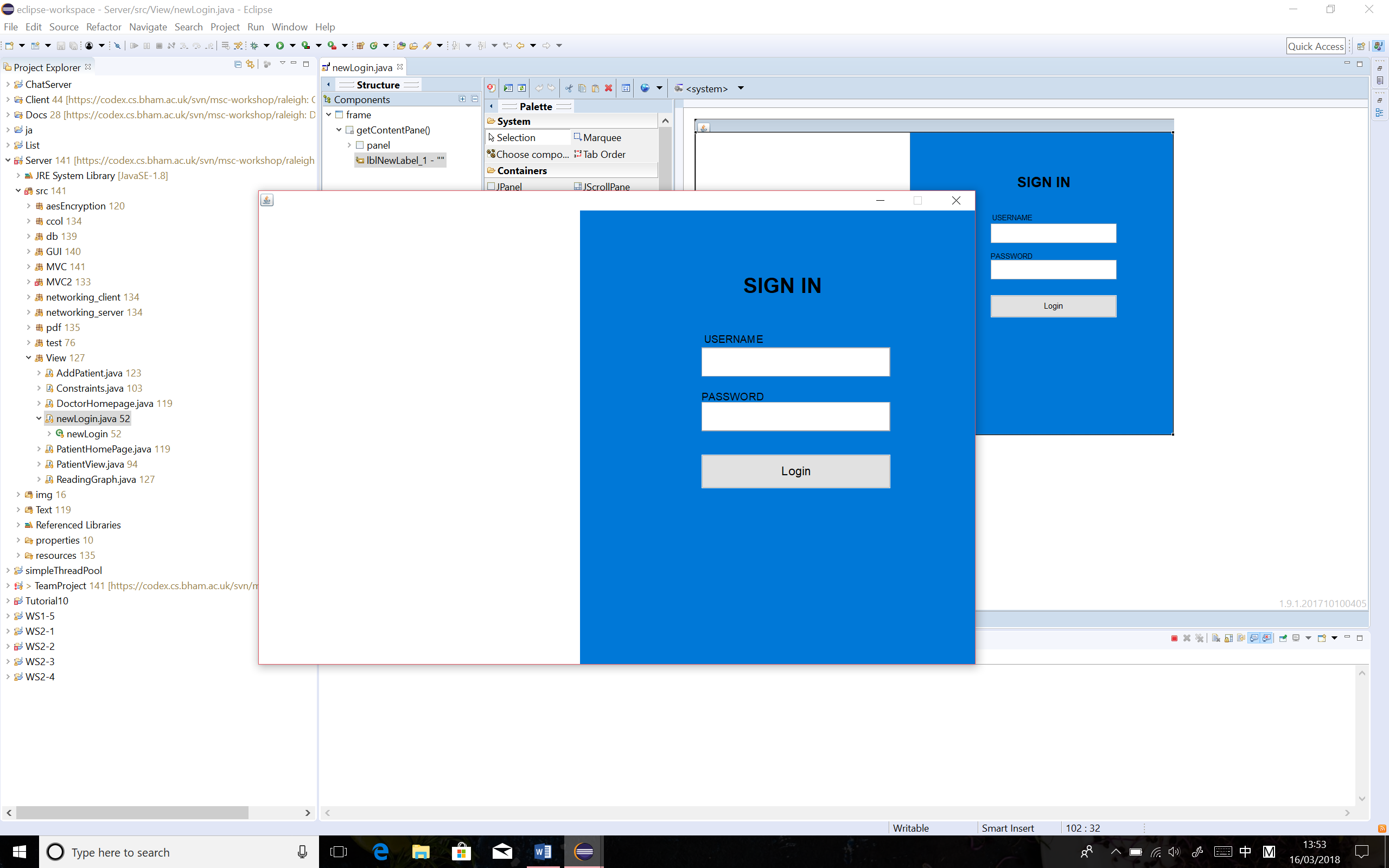
Paper Diary MyDiary (Android App)

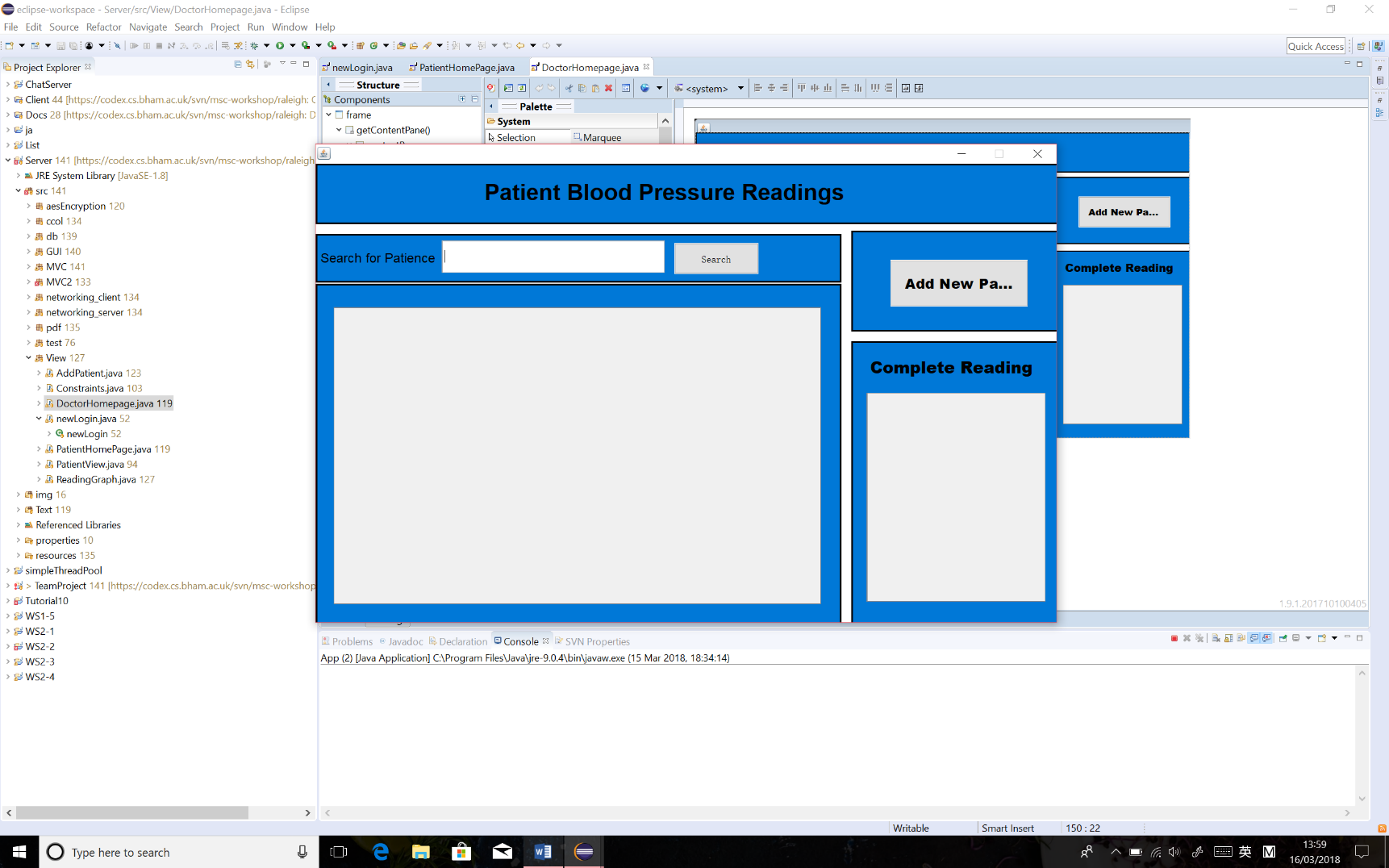
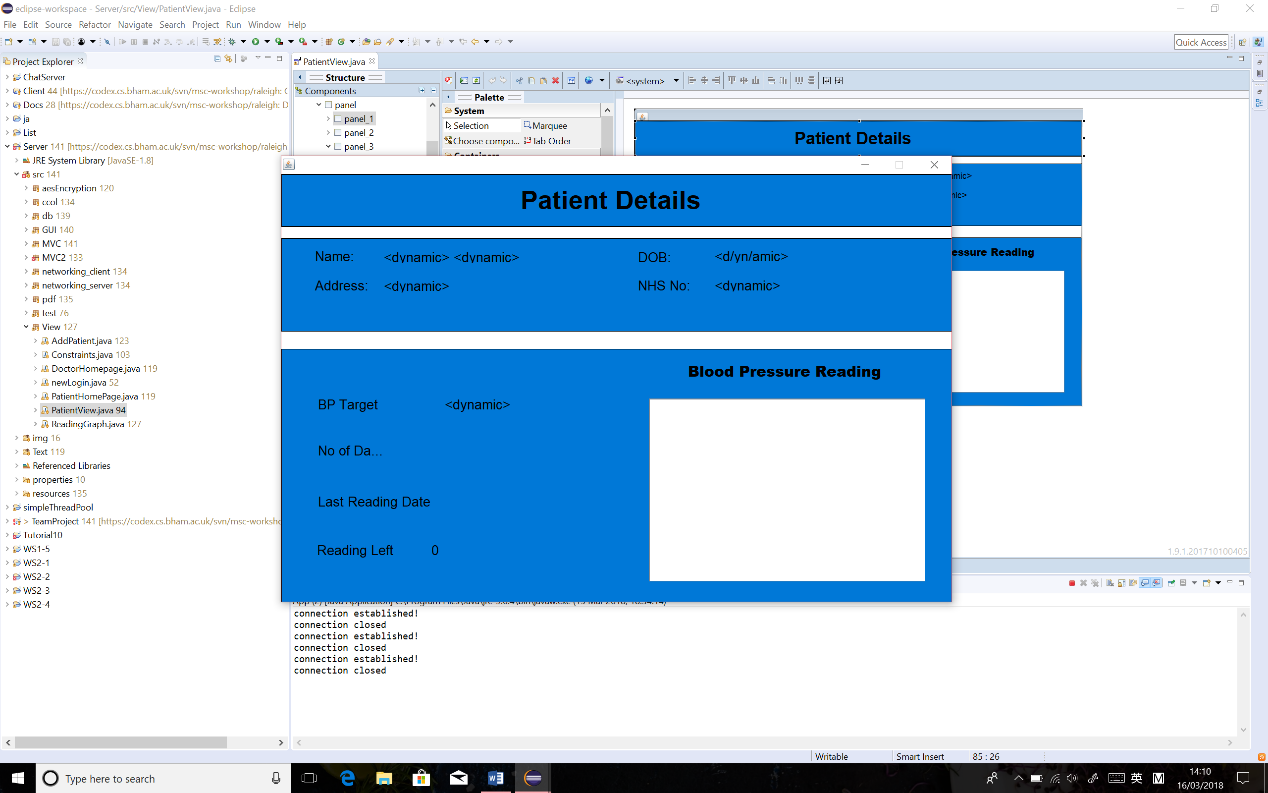
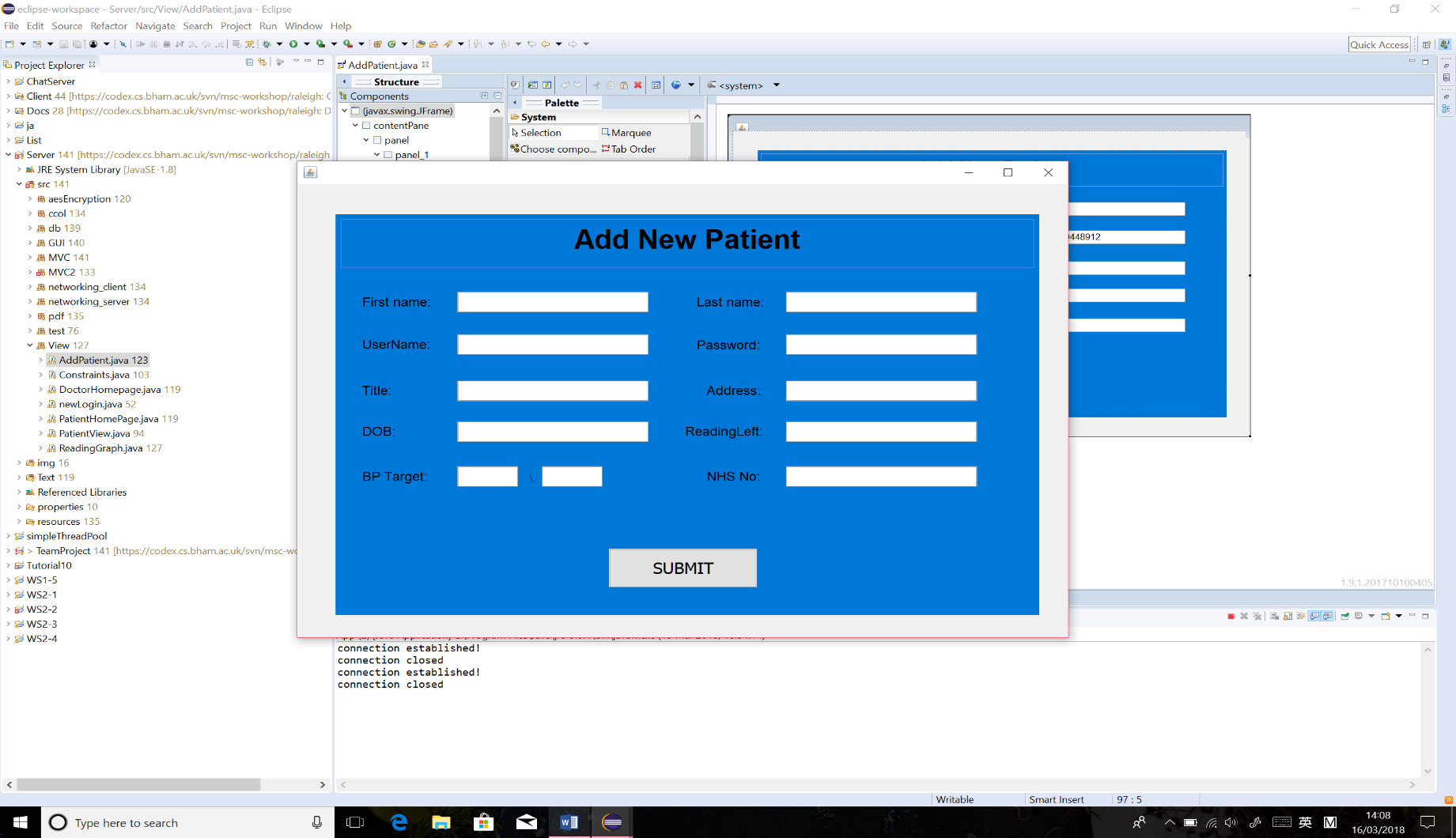
 

Cardio Journal – Blood Pressure Log (Android App)

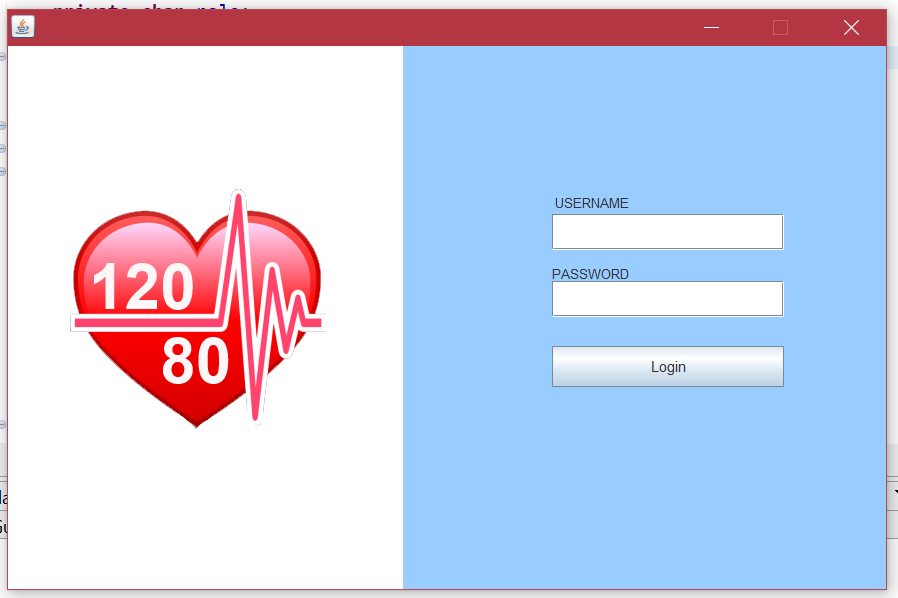


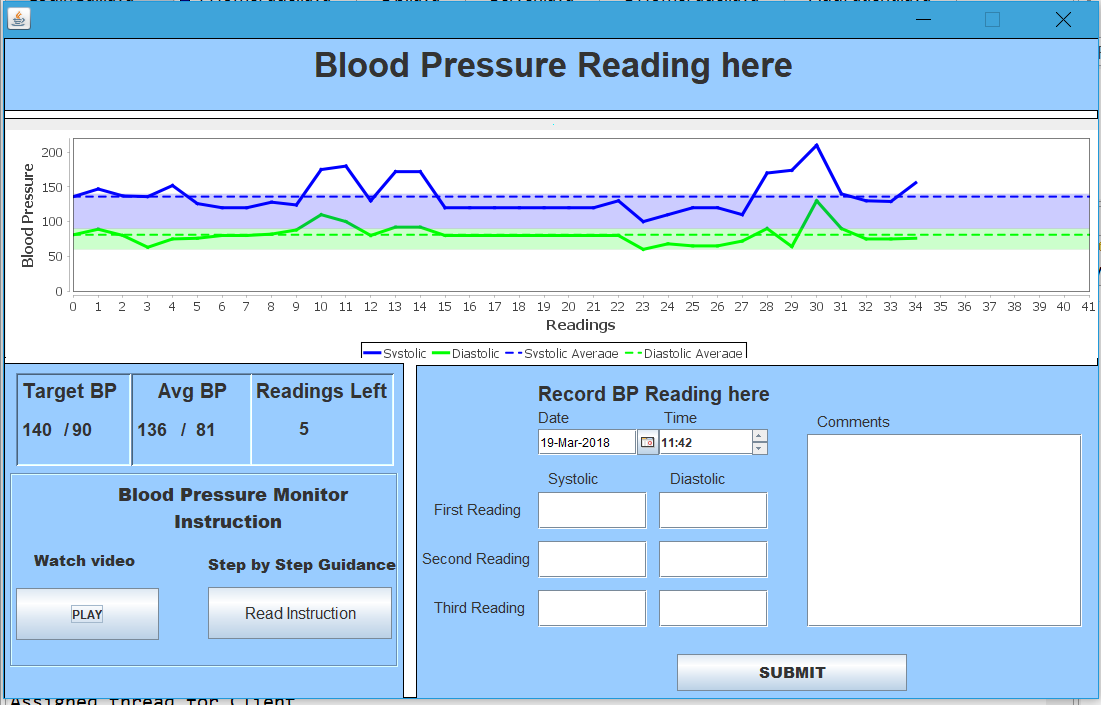
Login (Homepage) Patient (Homepage)

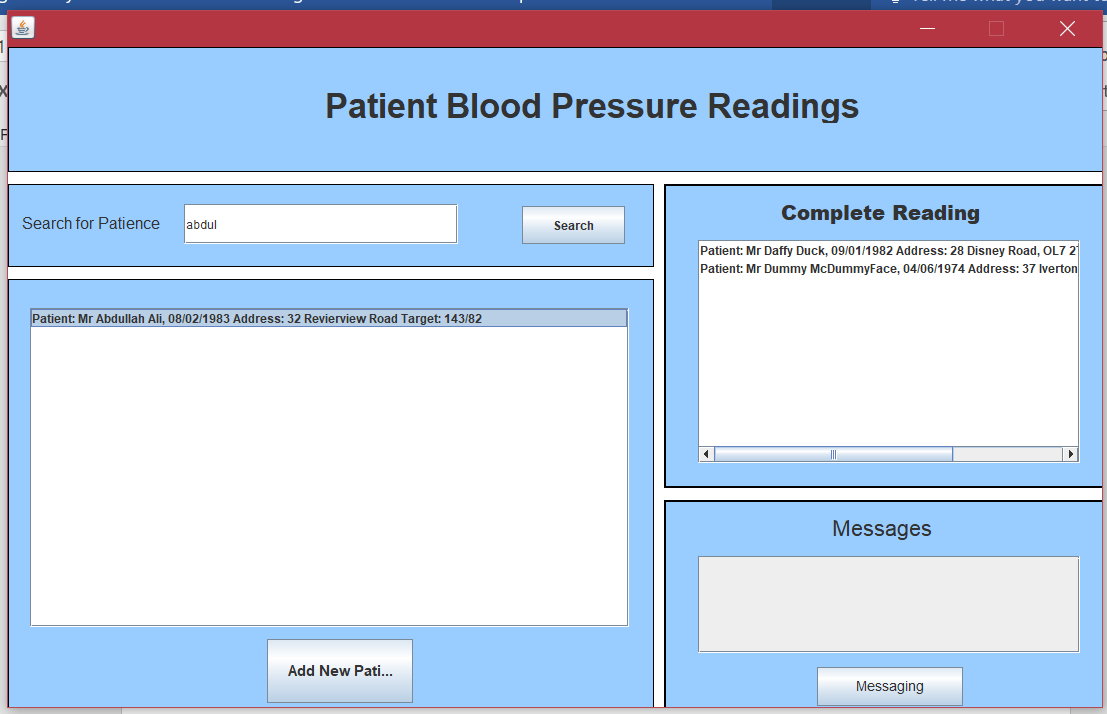


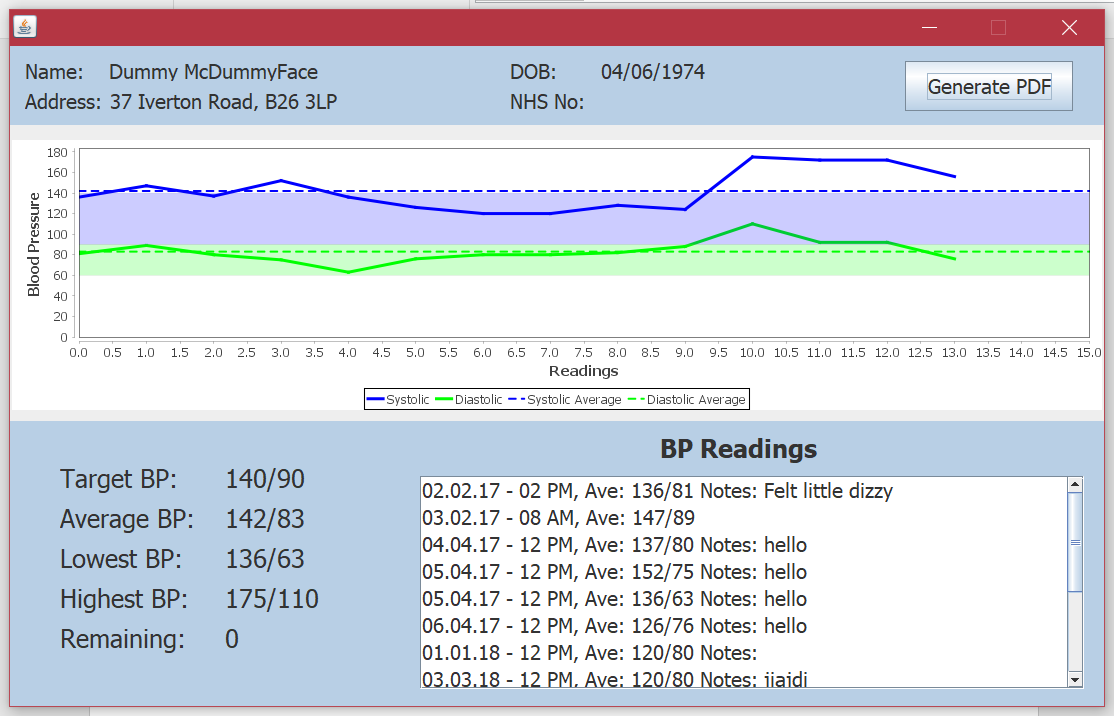
Doctor (Homepage, Patient Details Page, Add Patient page)

## View (windows)





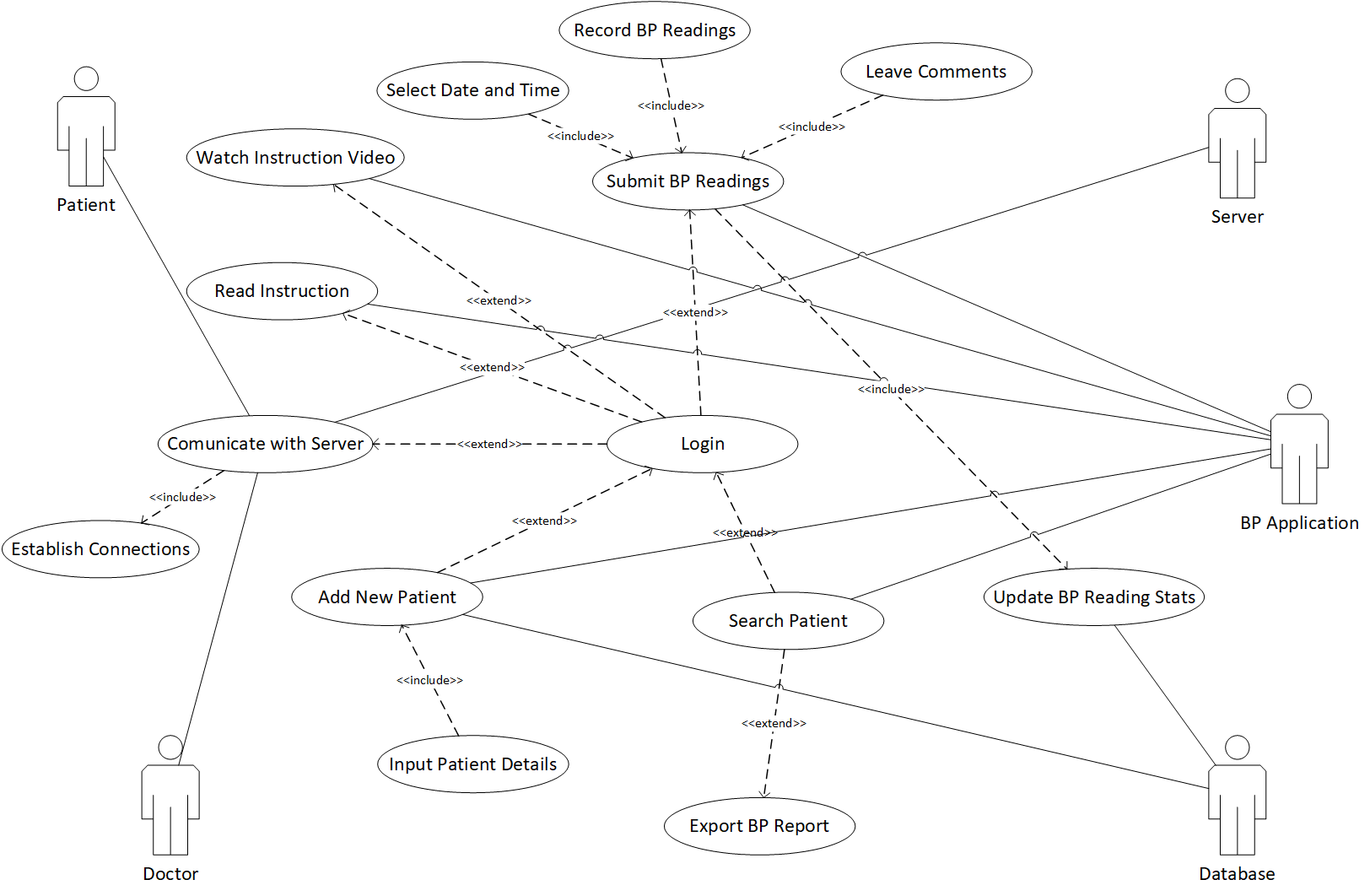






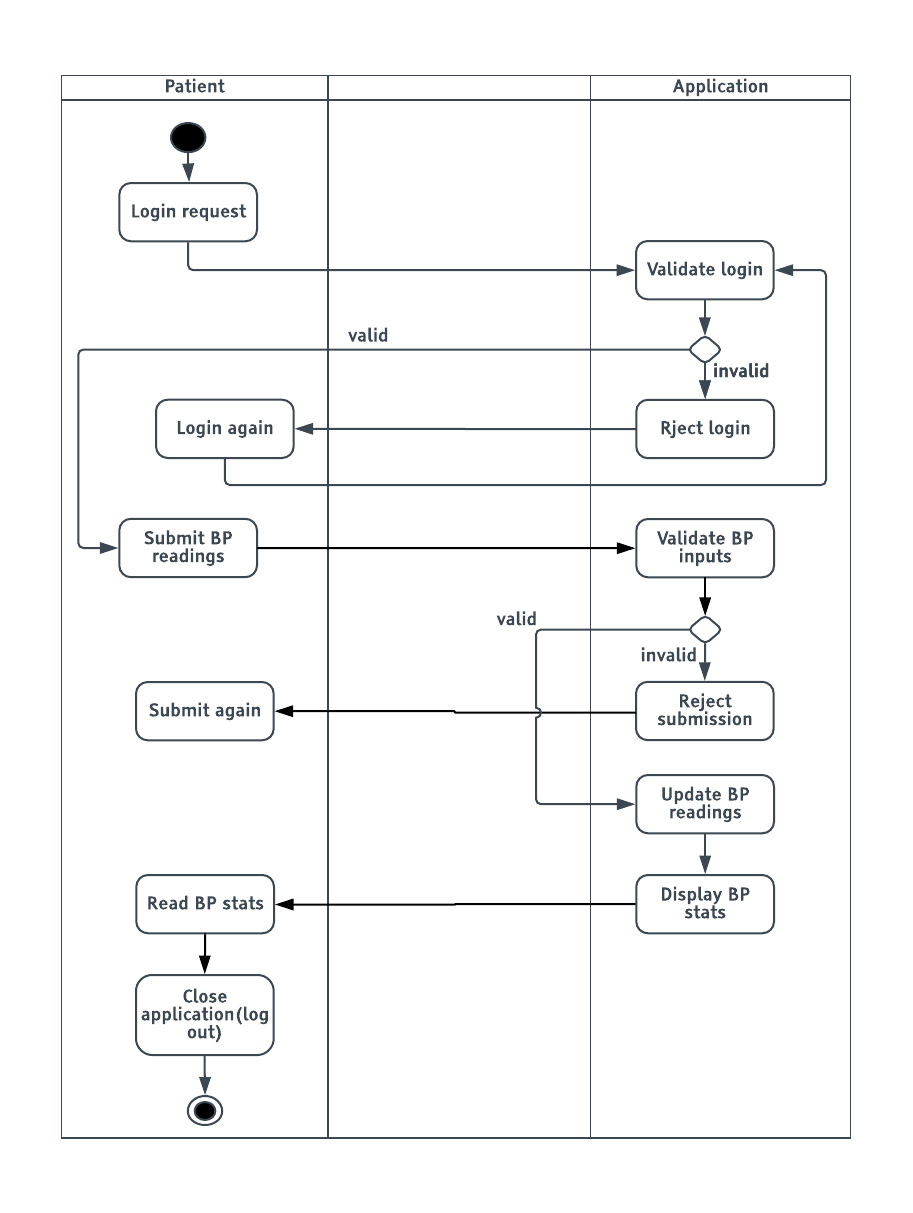
## Diagrams

Use Case Diagram

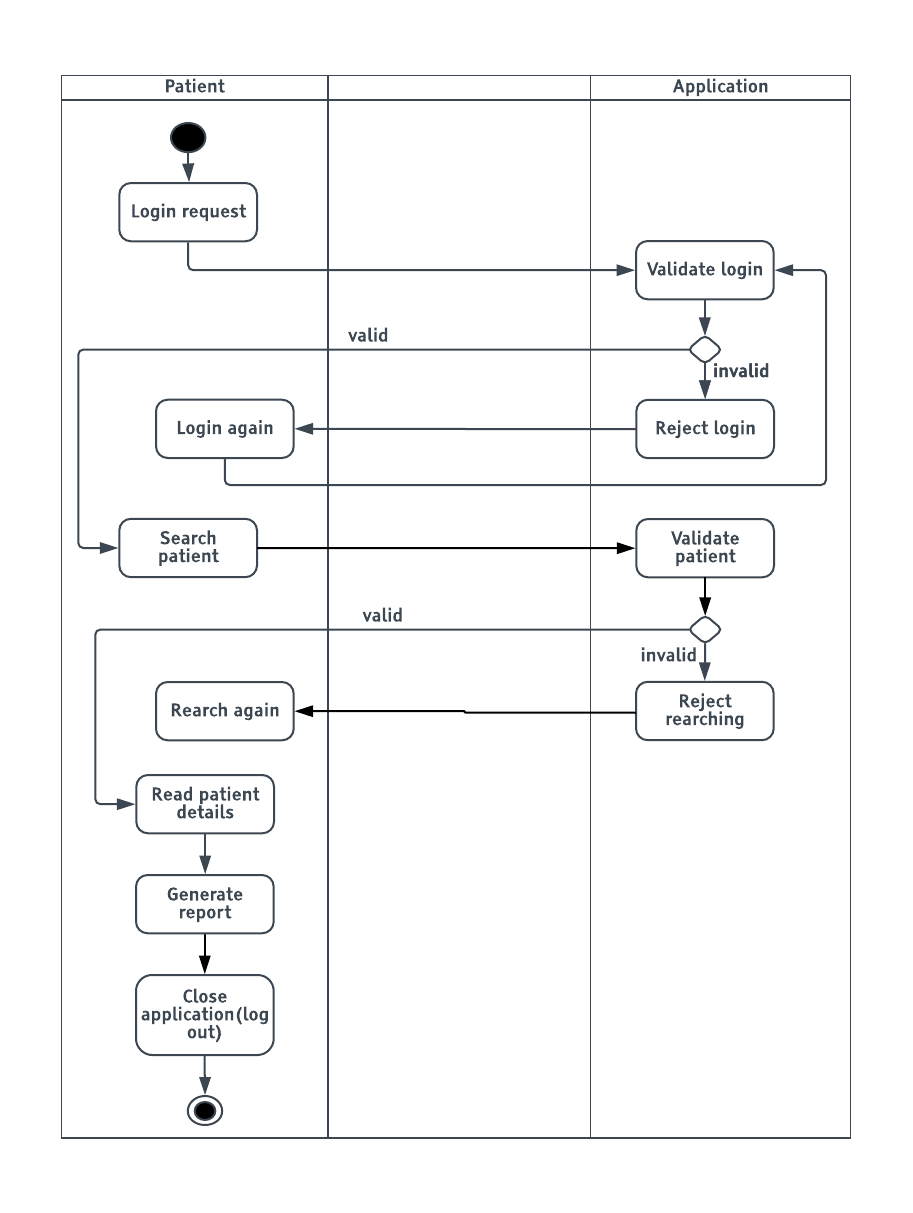


Activity Diagram

Scenario 1



Scenario 2



Component Diagram

ER Diagram



1. http://www.who.int/gho/ncd/risk\_factors/blood\_pressure\_prevalence\_text/en/ [↑](#footnote-ref-1)
2. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1120141/ [↑](#footnote-ref-2)
3. https://www.nice.org.uk/guidance/CG127/chapter/1-Guidance#diagnosing-hypertension-2 [↑](#footnote-ref-3)
4. https://bihsoc.org/wp-content/uploads/2017/09/Home\_blood\_pressure\_diary.pdf [↑](#footnote-ref-4)
5. https://play.google.com/store/apps/details?id=com.zlamanit.blood.pressure&hl=en\_GB [↑](#footnote-ref-5)
6. https://play.google.com/store/apps/details?id=com.mdhelper.cardiojournal&hl=en\_GB [↑](#footnote-ref-6)