BSc in Software Development

Year 3

COMP07030 Software Design Project

*<MediDB>*

**

*<G00308668>*

*<Gary McHugh>*

Contents

[Introduction 3](#_Toc449298099)

[Architecture of the solution 4](#_Toc449298100)

[Data Model 5](#_Toc449298101)

[Technologies used 6](#_Toc449298102)

[Problems Encountered/Solved 8](#_Toc449298103)

[Conclusions 8](#_Toc449298104)

[Recommendations 9](#_Toc449298105)

Student Number: G00308668

Student Name: Gary McHugh

Supervisor: Dr John Healy

GitHub Link:

<https://github.com/GaryMcHugh/Final-Year-Project-3rd-Year->

YouTube Video Link: <https://www.youtube.com/watch?v=XL0zDU0YU9k>

# Introduction

MediDB is a medical application that allows users to add, delete and view patients and doctors through easy to use interfaces. It allows the user to view each patient/doctor individually, as well as view them all at once in a table view. They can do this using HTML forms that I designed and created for this application. I also created a logo for the application which can be seen in fig1.0. This application was designed to be used in a hospital or medical centre. It allows the user to store information like the patient’s name, gender, date of birth and injury. It also lets the user specify the doctor’s name that worked with the patient. They can then store information about that doctor like name, date of birth and their respective job title.



fig1.0: MediDB Logo

**Aims:**

* To create a medical based client server application that uses a database to store, add and delete records using graphic user interfaces.

**Objectives:**

* Learn about how client server applications work.
* Improve my knowledge of MySQL.
* Learn how to use JSP and learn JSP syntax.
* Create concise interfaces using HTML and Bootstrap CSS.

**Scope:**

This application will be able to:

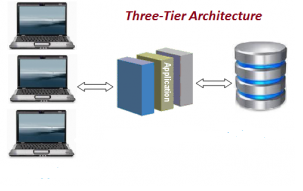
* Insert data into a database using an interface.
* Delete data from a database using an interface.
* Display data from the database in a table.

# Architecture of the solution

MediDB is a 3-tier client server application:

* The first tier of the application contains html forms which act as the interface for users to interact with.
* The second tier contains the JSP code which is run on an Apache Tomcat server when the user submits the form.
* The third and final tier contains a MySQL database that contains the tables and records used by the user in the application.

Below is a graphic representation of the 3-tier client server application:



MySQL

Apache Tomcat

Html form’s

The three interface designers that I researched were Swing, WindowBuilder and HTML forms.

Swing is a java API for creating interfaces provided by Oracle’s Java Foundation Classes (JFC).

WindowBuilder is an Eclipse Plug-in composed of Swing and SWT.

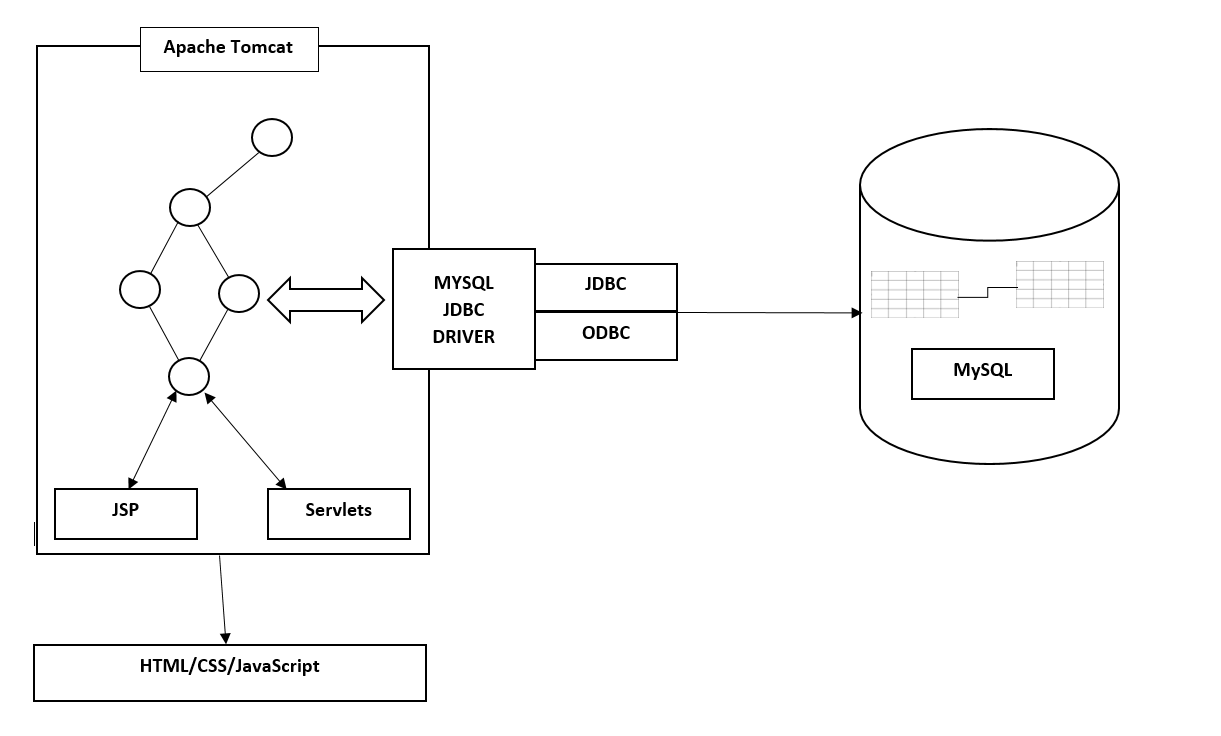
My reason for choosing HTML over Swing and WindowBuilder was that I could use Bootstrap CSS with HTML Form’s. Bootstrap CSS allowed me to make better looking interfaces controls such as buttons of various sizes and colours and better looking tables. WindowBuilder or Swing would not have allowed me to do this as I would have had to use plain buttons and tables.

I used Apache Tomcat over an alternative like Jetty as Apache Tomcat was more widely used. I also found that there was a lot of useful tutorials on how to configure Apache Tomcat than there were on Jetty. Due to ease of use and the wide variety of information available, I chose Apache Tomcat.

I used MySQL over MongoDB or Derby as I have a lot of experience with it from previous modules that I have taken throughout this course. Most recently Database Management Systems which I took this semester (year 3 semester 2). As a result of this I chose to use a MySQL database as part of my project.

# Data Model

Below is a Data Model of my MediDB application:



I designed the application so that the user would interact with the HTML form’s, this would act as their interface for the application. Upon submitting the form, a HTTP POST method would be executed to the relevant form. This executes the Java code that is in the form, depending on what form the user is on. The Java code opens a connection to the MySQL database using JDBC and ODBC. An SQL statement is then executed from the code to the database. The change is then reflected in the database. The user can then view this change in the application through the tables.

I designed the application in this way as it is an efficient and common way to achieve this kind of functionality in an application. This design also allows me to easily swap out a component such as the MySQL database. I could easily replace this with a Derby database and retain the functionality, I may however have to change the MySQL statements within the JSP code, this is a minor change.

# Technologies used

I used the following technologies:

* Eclipse Java EE
* Apache Tomcat
* Connector/J
* HTML and Bootstrap CSS
* MySQL
* JSP

**Eclipse Java EE:**

I decided to use Eclipse Java EE as I have experience with using Eclipse to develop Java applications from my 2nd year Java module. However, I did not have any experience in developing JSP applications using Java EE so I also gained the opportunity to learn how to use the Java EE version of eclipse.

**Apache Tomcat:**

I used Apache Tomcat 8.0 as it was the latest version of Apache Tomcat that was compatible with Eclipse Java EE. This allowed me to run my JSP applications that I created on an Apache Tomcat server through Eclipse. I found that Apache Tomcat was the most commonly used server for these types of applications which influenced my decision to use this technology. I had not hosted an application on a server locally before so this allowed me to gain knowledge in using this type of technology.

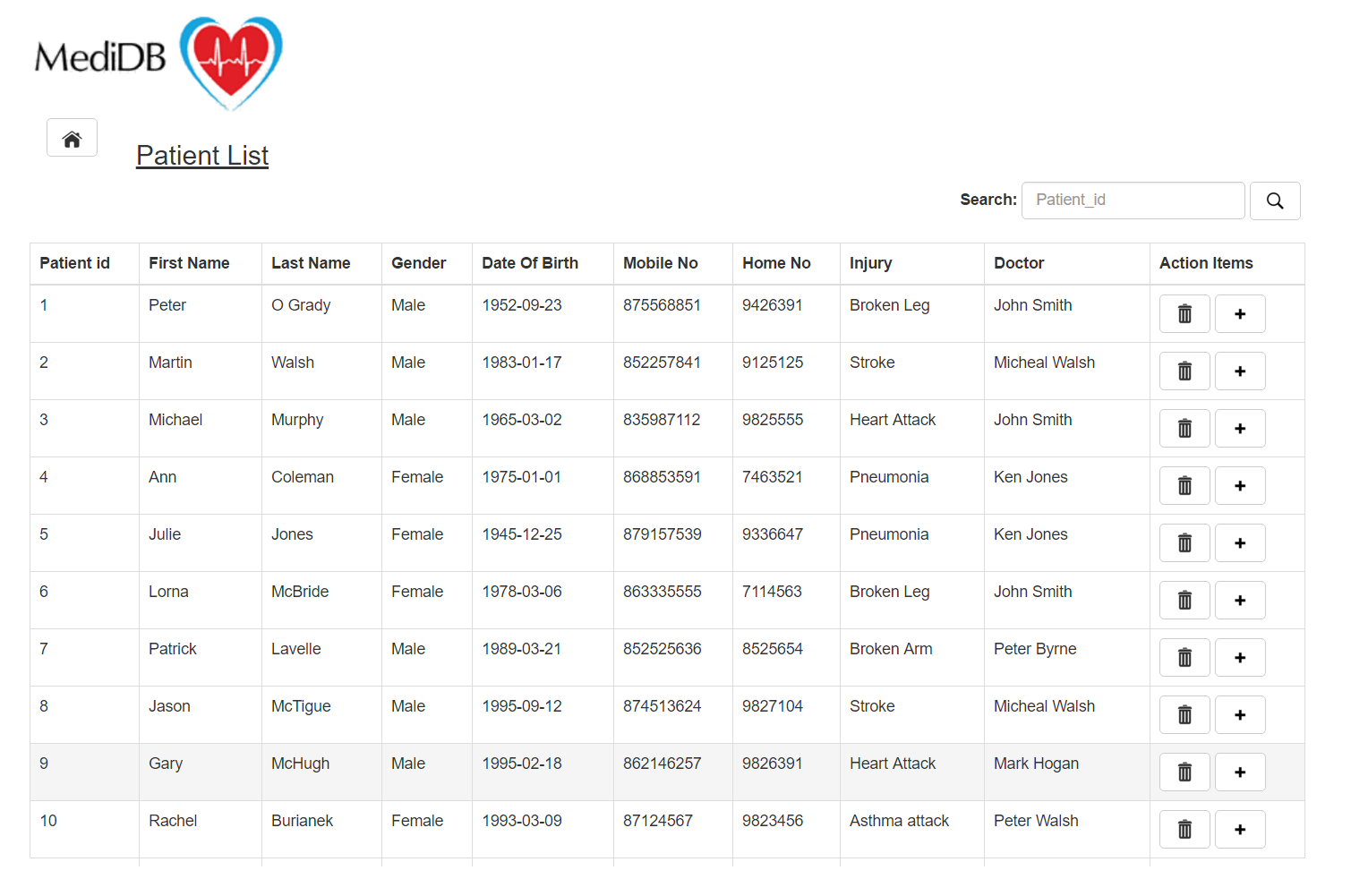
**Connector/J:**

Through my research I found out that Connector/J is required by tomcat to communicate with a MySQL database. The jar file contained in the download (mysql-connector-java-5.1.38-bin.jar) had imported into the eclipse project and placed in the lib folder contained in the WEB-INF folder within the project. This allowed my application to communicate with the MySQL database.

**HTML and Bootstrap CSS:**

I used HTML forms as the interface for the application, through my research I found that Bootstrap CSS gave the forms a more appealing look to the forms used by the user. Bootstrap CSS takes regular HTML controls like buttons and improves the look of them, this is also true for tables.

An example of an appealing interface that uses Bootstrap CSS can be seen below:



**MySQL:**

I decided to use a MySQL database as I was most familiar with this kind of database from previous modules in college. MySQL is also the most popular relational database used in Industry. I already had a good knowledge of the statements needed by the application to insert, delete and return data from the database. Therefore, I choose the MySQL database.

**JSP:**

I decided to use JSP as the syntax was easy to learn when compared to Servlets or PHP. I could also execute Java code when using JSP, this was an advantage as Java is my strongest programming language. I had no previous experience with JSP before so this allowed me to gain knowledge of a new technology.

# Problems Encountered/Solved

The first issue that I encountered in the development of this software was in relation to the compatibility of Apache Tomcat and Eclipse Java EE. I had initially downloaded and configured Apache Tomcat 9.0 as it was the latest version of Apache Tomcat available from the Apache Tomcat website. I then created a JSP application and tried to run it on the Apache Tomcat 9.0 server. I received an error message in Eclipse to indicate that the latest version of Apache Tomcat that was compatible with Eclipse was Apache Tomcat 8.0. To rectify this issue, I un-installed Apache Tomcat 9.0 and Installed and configured an older version of Apache Tomcat (Apache Tomcat 8.0).

The second issue that I encountered was in relation to the Update functionality that I tried to add to the project. I had successfully added the insert, delete and view functionality to my project as I had planned to do in my objectives. I then tried to add some additional functionality to my project in the form of an Update. This would have allowed users to change patient’s and doctor’s information without having to delete and re-create them. Unfortunately, I was unable to implement this functionality due to time restrictions. As a result, I had to remove the Update forms that I had created and also remove the code that I had put into the project in an attempt to get the Update working.

The final issue that I encountered during this project was attempting to navigate from a submit button. I had attempted to navigate to the ‘TableView’ form after the user had added a record to the database. However, through my research I found that you cannot navigate from a submit button as it preforms the HTTP POST to the form. I then had to re-design the navigation by adding a home button in the top left corner of the form. This allowed the user to easily navigate to the main menu after adding a patient or doctor. I then added a home button to all of the forms for easy navigation.

# Conclusions

I believe that this project was a good learning experience for me as I have improved my problem solving skills, research skills and built confidence in my ability to work as an individual on a project. I have also gained knowledge in how a client server application works and the technologies necessary to create an application like this. I have also improved my knowledge of MySQL through creating a medical database with auto-incrementing fields within tables. I have learned how to use JSP to take information from a user through controls like text boxes and drop down lists and use this in a database.

I have also learned how to create appealing interfaces using HTML and Bootstrap CSS. I believe that the skills that I have gained through the development of this project will benefit me in the working world. The research and problem solving skills that I have gained through the development of this project will benefit me on future college projects and assignments. I have also improved my communication skills through the weekly meetings that attended with my supervisor. This also improved my teamwork skills which are a vital part of software development. Overall this project was a great learning experience for me and improved various skills which I will use in future projects and assignments.

I could also build on this client server application for my Forth Year Project by adding a document database like Mongo instead of a MySQL database. I could also add Spring to the project instead of using JDBC and ODBC

# Recommendations

I would recommend that an Update page with Update functionality would be added to the project. This would allow users to easily change data about doctor’s and patient’s without having to delete them and add them again. I would also recommend adding Spring to the project instead of using JDBC and ODBC. Spring is more widely used out in industry and would improve this client server application. I would also re-design the delete form as there are a lot of controls on it that are un-used. Most of the controls on the page are set to read only. I would also make it so that the delete button on the table view forms would delete the record that they were selected on. This would mean that if the delete button was pressed on the record with patient number two, this patient would be deleted instead of the button bringing you to the delete form.

To summarise:

* Add Update Functionality
* Use Spring instead of JDBC and ODBC
* Re-design Delete form
* Change Delete button functionality on table view forms