**Software Requirements**

**Specifications**

**For**

**Patient Searching**

**Version 1.4**

**Prepared by**

**Ian Bell**

**Marilyse Gast**

**Peter Litster**

**Eric Raleigh**

**Chris Speirs**

**22-April-2013**

Table of Contents

[Part 1. Executive Summary 1](#_Toc354431180)

[Part 2. System Features 3](#_Toc354431181)

[Part 3. User Interface Screens 5](#_Toc354431182)

[Part 4. Entity Relationship Diagram 9](#_Toc354431183)

[Part 5. Sequence Diagrams 11](#_Toc354431184)

[Part 6. Project Structure 14](#_Toc354431185)

[Part 7. Proposed Technologies 16](#_Toc354431186)

Appendix A - Technical Problems and Solutions………………………………………………………………………..20

Appendix B - System Testing and jUnit Testing..………………………………………………………………………..21

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Peter Litster | 3/21/13 | Initial writing. | 1.0 |
| Ian Bell | 4/18/13 | Updated the Sequence Diagram and ERD | 1.1 |
| Marilyse Gast | 4/18/13 | Fixed Headings, Page Numbers, and Formatting | 1.2 |
| Peter Litster | 4/19/13 | Update TOC. Remove meeting minutes. Rename to System Design. | 1.3 |
| Ian Bell | 4/20/13 | Added Patient Details Sequence Diagram. Updated project structure image. Updated search page screen shot. | 1.4 |

# Part 1. Executive Summary

**Executive Summary**

**Purpose**

This document will identify all the requirements for the Patient Searching system. This document will describe the functionality of the product being developed. The signed parties agree to the specifications and features documented. Development of this project will be in accordance with the specification. If these specifications are met by the development team, the customer agrees to accept the final product as is.

**Project Scope**

This is a patient searching application that will, through user interaction, find patients within a database. The intent is to provide the searching capability only.

**Product Summary**

Create a System that will search a database of patient information. This will be a new application and will not be integrated with any other software. The application will be web based.

**Product Features**

The System will be used to search for patients using demographic information. The system will be robust enough to allow searching on any of the available fields.

# Part 2. System Features

**System Features**

**1. User System**

**1.1** Searching.

**1.1.1** Upon opening the patient search application, the user will be prompted to enter search criteria.

* + - 1. The results will be returned in a list format.
      2. Individual patients can be selected to show further information.

|  |  |
| --- | --- |
|  | PM initial Section sign off |
| Date | Notes |
|  |  |
|  |  |
|  |  |

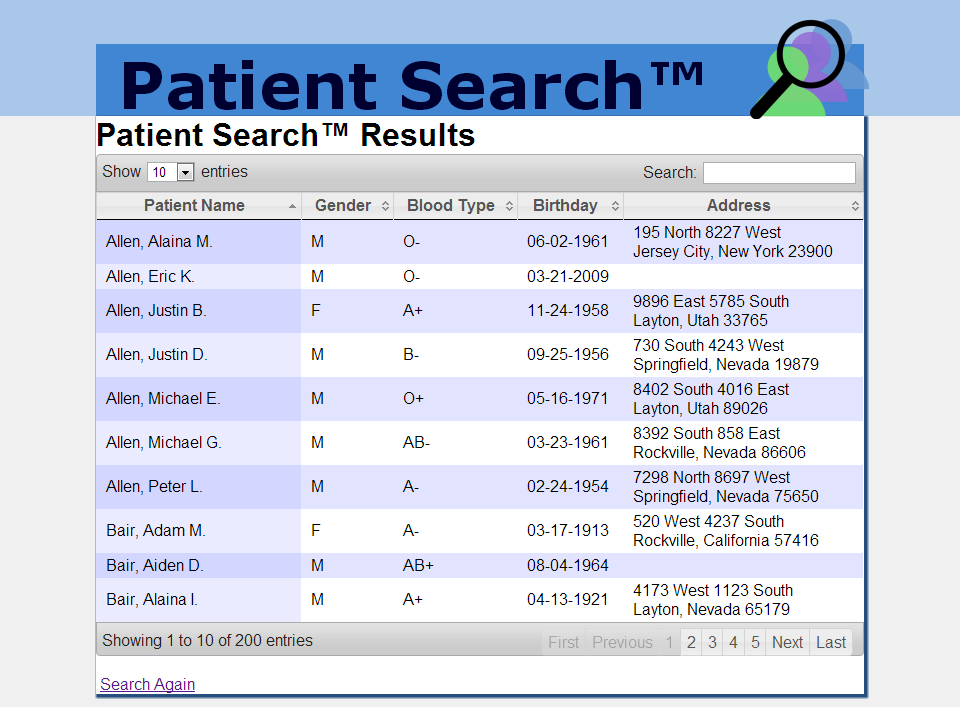
# Part 3. User Interface Screens

**User Interface Screens**

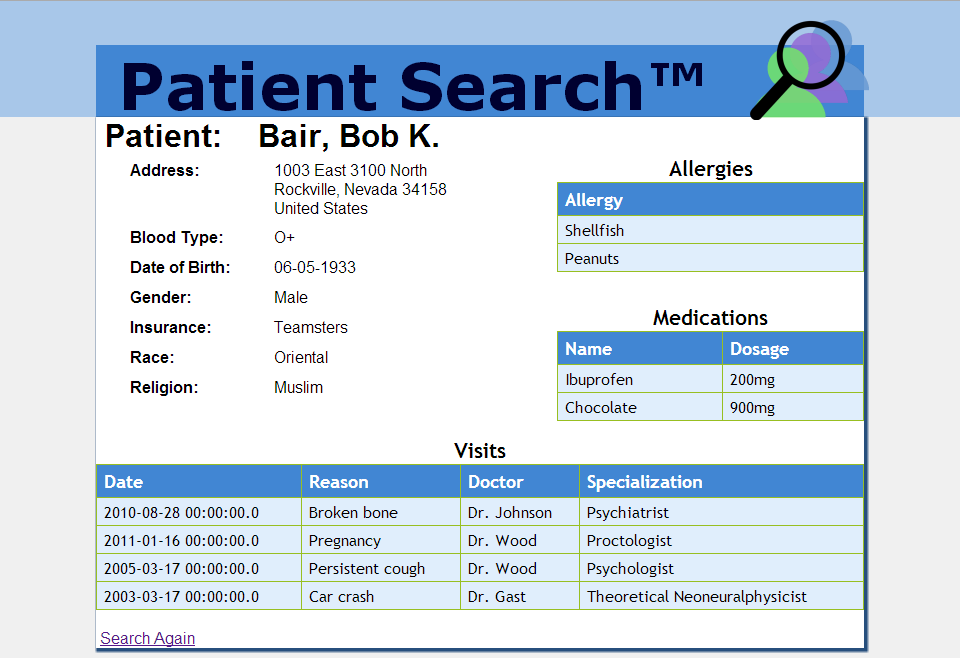
**Search Screen (index.jsp)**



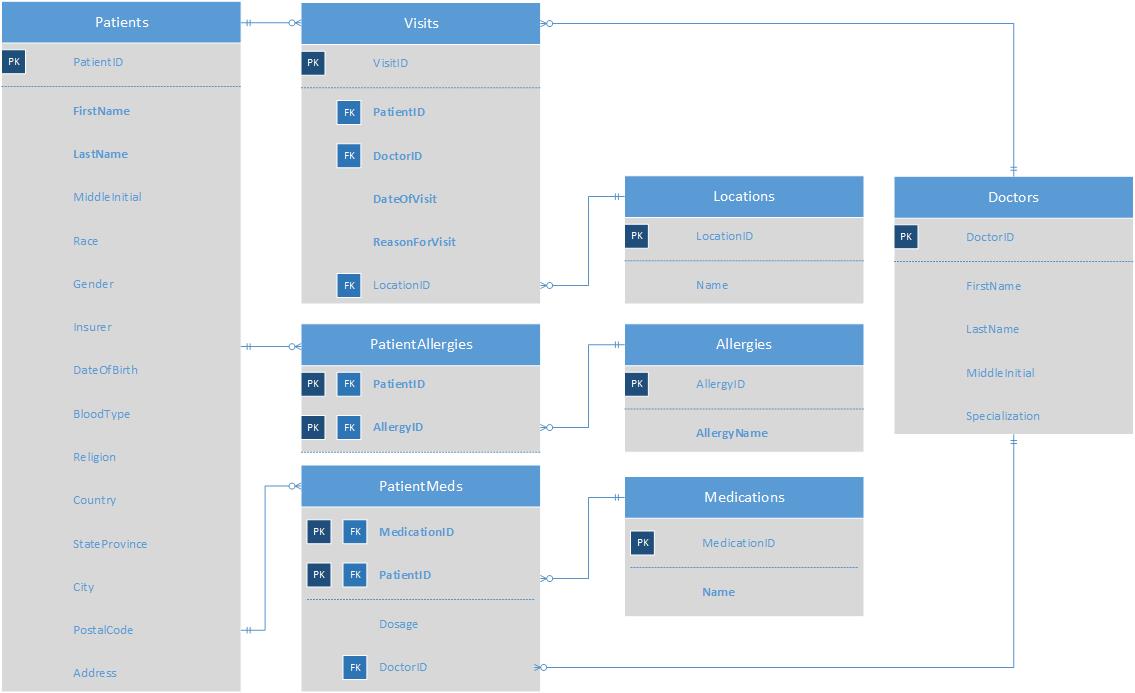
**Results List Screen (result.jsp)**



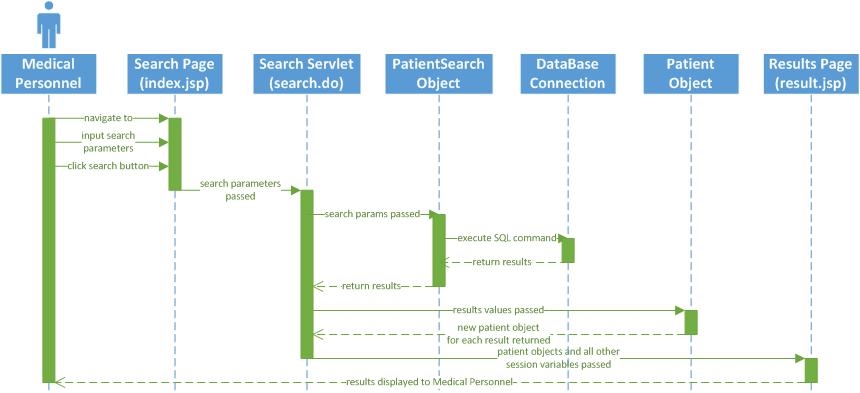
**Patient Details Screen (patient.jsp)**

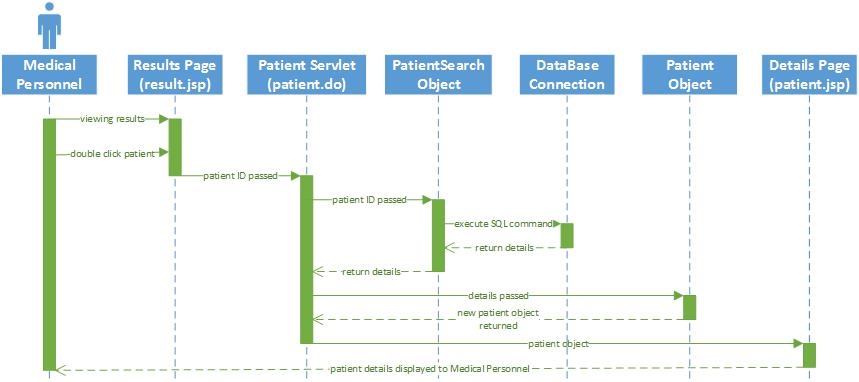


# Part 4. Entity Relationship Diagram

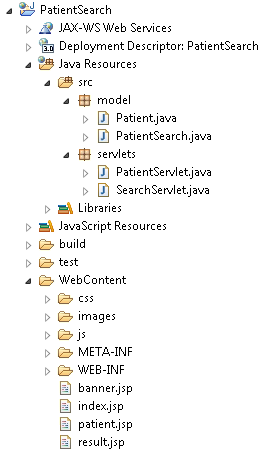


# Part 5. Sequence Diagrams





# Part 6. Project Structure



# Part 7. Proposed Technologies

**Proposed Technologies**

We have decided to use JSP with a servlet for this project. We determined that it would provide sufficient functionality while being something that had been covered sufficiently at the beginning of the project that we could implement it.

The database is running on a SQL Server that was obtained by one of the group members for use on a different project, but had sufficient space to house our testing area.

It was decided also to use Tomcat as our server environment. Again, this is because it was sufficient to perform the task and simple enough for us to implement in time. The project has also been deployed to GlassFish at various times during development.

**Appendix A - Technical Problems and Solutions**

**Establishing A Database Server**

We had some struggles establishing the database server we finally ended up using. Due to the student resource PHPMyadmin be not having a reliable connection early in the project, it took some time to establish a database. Peter set up his home computer to be a Glassfish server hosting a database. This proved futile due to certain circumstances. Finally we moved to using a Windows Azure powered SQL Server database.

**Database Table Creation Scripts Created Late**

Despite the extra time taken to find a place to host the actual database the scripts for the table creations were not prepared. Time having passed in the project beyond ideal for just having established a database, Chris became anxious and began writing the table creation scripts with assistance of other group members.

**Results Table**

With the results table being a main feature, the team focused resources on making it as user friendly as possible. Ian failed to find solution online that would provide the features that the results table needed to have (namely pagination and column sorting). After Ian had implemented a nearly satisfactory solution, Mari found a jQuery Table solution and implemented it. This proved to be the better solution by far because it used client-side code and didn’t require addition trips to the server.

**Appendix B – System Testing and jUnit Testing**

**Data Integrity**

The first step that was taken in system testing was to make sure that the data returned from searches matched the data returned from a query purely designed for retrieving data from the database directly. Through testing one field at a time initially there was a few errors found and corrected immediately. Through this testing we also determined that searching by visits will return all visits rather than “last visit” for simplicity.

Continued searching by multiple parameters has involved attempting to show that the query building in the POJO (Plain Old Java Object) is effective at retrieving exactly what was searched for when multiple items are searched. Because of the way the tables are related this generally includes pairing a single search parameter from the Patients table with up to two parameters from the PatientMeds and PatientAllergies tables and any one of the four visit/doctor information. Because of the many combinations this results in, testing on this level has not exhaustively accomplished during system testing. A more exhaustive test should be found in the jUnit testing section below.

**Aesthetics**

It has been determined that the form is very pleasing to the eye and intuitive in the way it is designed. It is easy to tell that the search form is designed for inputting searching parameters and that the search button is designed to search on the input items.

The results view is also aesthetically pleasing. It is nice that not too much information is displayed and cluttering the screen but that more information is acquirable through double click of any record in the table.

Patient detail view is likewise pleasing to look at and very intuitive in the way it is built. Information that is in list type is displayed in clearly labeled tables.

**jUnit Testing**

Four methods in the PatientSearch.java class underwent jUnit testing. The system was tested by adding information to the database, using the four functions to retrieve that information and verifying that it matches the information added, and then removing the added information when finished. Included is a test to verify that the database integrity has been maintained to ensure that the testing data was properly removed.

The most important test (as determined by our group) is for the search() method. This method has a lot of moving parts, and the test is designed to make sure that multiple variations of tests (determined randomly) are executed properly.