NANYANG TECHNOLOGICAL UNIVERSITY

School of Computer Engineering

SCE11-0353

WEB APPLICATION FOR RADIOLOGY INFORMATION SYSTEM

Submitted in Fulfillment of the Requirements for the

Degree of Bachelor of Computer Engineering of the

Nanyang Technological University

Author: Desmond Poh Lik Meng (U0922923K)

Supervisor: Assoc Prof Lin Feng

2012

Abstract

Telemedicine is emerging as a fast growing medical technology in the health care industry. It helps to eliminate distance barriers and can improve access to medical services that would often not be usually available in distant rural communities. Created from a convergence of telecommunication and information technologies, telemedicine permits communications between the patient and healthcare provider with fidelity and convenience.

The most popular use of telemedicine is via teleradiology, which is the ability to send medical images from one location to another. Traditionally, it requires a sending station, a receiving station, and a transmission network. However, with today's high-speed broadband Internet, healthcare professionals and patients alike can view medical images stored on remote servers; only a standard Personal Computer and DSL/Cable Internet connection is needed.

In this project, we will be developing a web based Radiology Information System (RIS), in an attempt to migrate the information systems used in radiology departments online. It aims to let healthcare professionals access patient medical data at any time, while letting patients themselves monitor their health.

The development process, from its inspiration to implementation to wrap up, is documented in this report. Special attention is given to discussing the application design, difficulties encountered during design and development, the strengths and weaknesses of the application, and areas that can be improved upon in future.

Acknowledgement

I would like to express my deepest gratitude to the following individuals who have provided me valuable insights, help and guidance for completing my Final Year Project.

- Dr. Lin Feng, my project supervisor. His guidance and encouragement has proved invaluable at all stages of this project. Were it not for him, this project would not have even be half completed.

- The staff members at the Radiology Department of Raffles Medical Group. Their paid introductory tour of a typical workflow process of their department gave many insights into the potential challenges this project would be facing.

- The team at BitBucket, a source version control vendor. Their graciousness to upgrade the code repository at no cost meant that critical mistakes made during development can be rolled back without any down time.

- All my friends and family who have supported and helped me during the time which this project was going through a difficult development period.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Telemedicine refers to provision of medical information and services via telecommunication devices. Created from a convergence of telecommunication and information technologies, it not only permits long distance communication between healthcare providers, but also with patients, via today's broadband Internet infrastructure.

Telemedicine is part of a larger movement towards electronic based healthcare (eHealth). Compared to traditional healthcare, the concept of eHealth is based upon the central idea of storing all medical information, records and images electronically. Although adoption of eHealth varies widely, both internationally and within Singapore itself, most medical institutions have already migrated much of their paper-based processes to Healthcare Information Systems (HIS).

Complementing a HIS is a specialized subsystem called Radiology Information System (RIS). As its name implies, a RIS is used almost exclusively by the radiology department of a medical institution. In layman's terms, a RIS is a computerized database for storing, manipulating and distributing patient radiological data and imagery.

1.2 Objective

The objectives of this project are as follows:

- To allow radiologists to work using the RIS system, and its underlying database
- To allow radiologists to retrieve imaging orders for a patient referred to them
- To allow radiologists to store medical image(s) taken
- To allow radiologists to enter data relating to the medical image
- To access the Outpatient Information System (OPIS) database, which is being developed concurrently in another project, in future

1.3 Scope

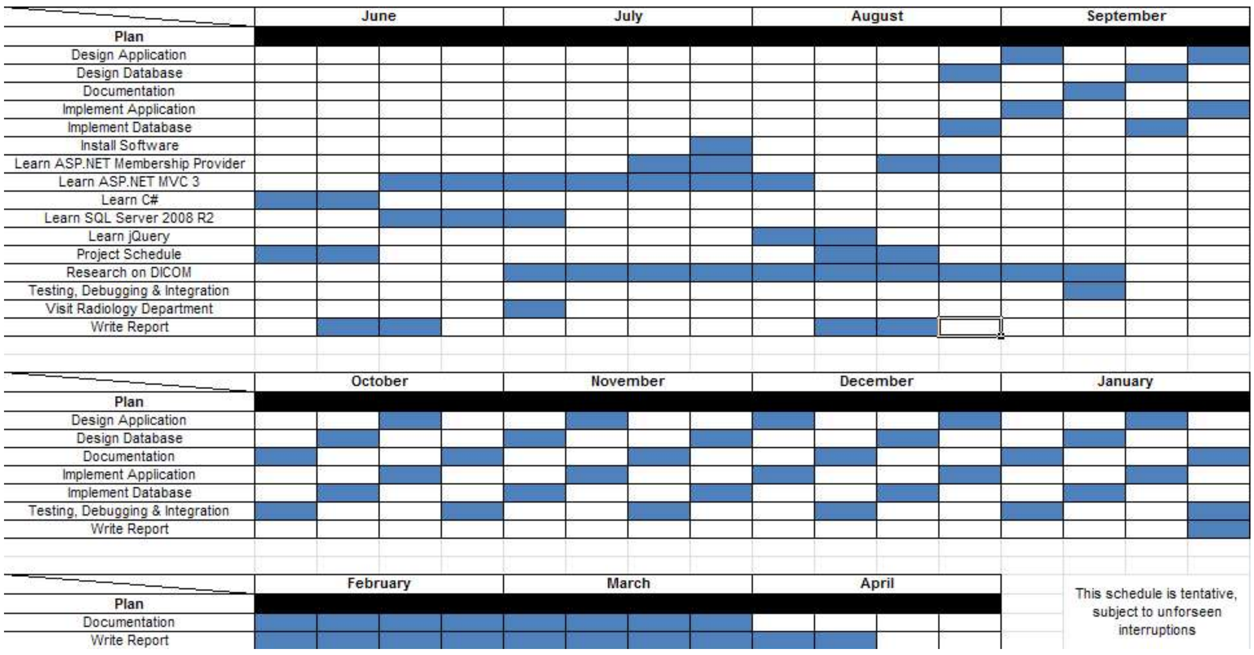
The scope of this project is as follows:

- Visit a radiology department of any medical institution to gather as much information as possible on the workflow process
- Understand the commonly used technical jargons in radiology
- Analyze the requirements of the RIS based upon information gathered
- Learn and understand DICOM, the standard used in medical imaging
- Learn ASP.NET, the web technology upon which the RIS will be hosted on
- Learn C#, the primary programming language that will be used in this project
- Learn SQL Server 2008 R2, the database that will be used in this project
- Design and develop the backend database iteratively
- Design and develop the RIS web application iteratively
- Learn about technologies that might potentially shorten the development time of the RIS and utilize them, if possible
- Explore about the possibility of optimizing the RIS for mobile platforms, if time permits

The fully implemented RIS shall fulfill the objectives of this project, as stated in section 1.2.

1.4 Project Planning and Schedule

The initial proposed project schedule is indicated in Figure 1. As the project goes into the implementation stage, it is expected that difficulties will force a rescheduling of the development flow.



	June				July				August				September			
Plan																
Design Application													🟦			🟦
Design Database												🟦			🟦	
Documentation														🟦		
Implement Application													🟦			🟦
Implement Database												🟦			🟦	
Install Software								🟦								
Learn ASP.NET Membership Provider							🟦	🟦			🟦	🟦				
Learn ASP.NET MVC 3			🟦	🟦	🟦	🟦	🟦	🟦	🟦							
Learn C#	🟦	🟦														
Learn SQL Server 2008 R2			🟦	🟦	🟦											
Learn jQuery									🟦	🟦						
Project Schedule	🟦	🟦								🟦	🟦					
Research on DICOM					🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦		
Testing, Debugging & Integration														🟦		
Visit Radiology Department					🟦											
Write Report		🟦	🟦							🟦	🟦					

	October				November				December				January			
Plan																
Design Application			🟦			🟦			🟦			🟦			🟦	
Design Database		🟦			🟦			🟦			🟦			🟦		
Documentation	🟦			🟦			🟦			🟦			🟦			🟦
Implement Application			🟦			🟦			🟦			🟦			🟦	
Implement Database		🟦			🟦			🟦			🟦			🟦		
Testing, Debugging & Integration	🟦			🟦			🟦			🟦			🟦			🟦
Write Report																🟦

	February				March				April			
Plan												
Documentation	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦				
Write Report	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦		

This schedule is tentative, subject to unforseen interruptions

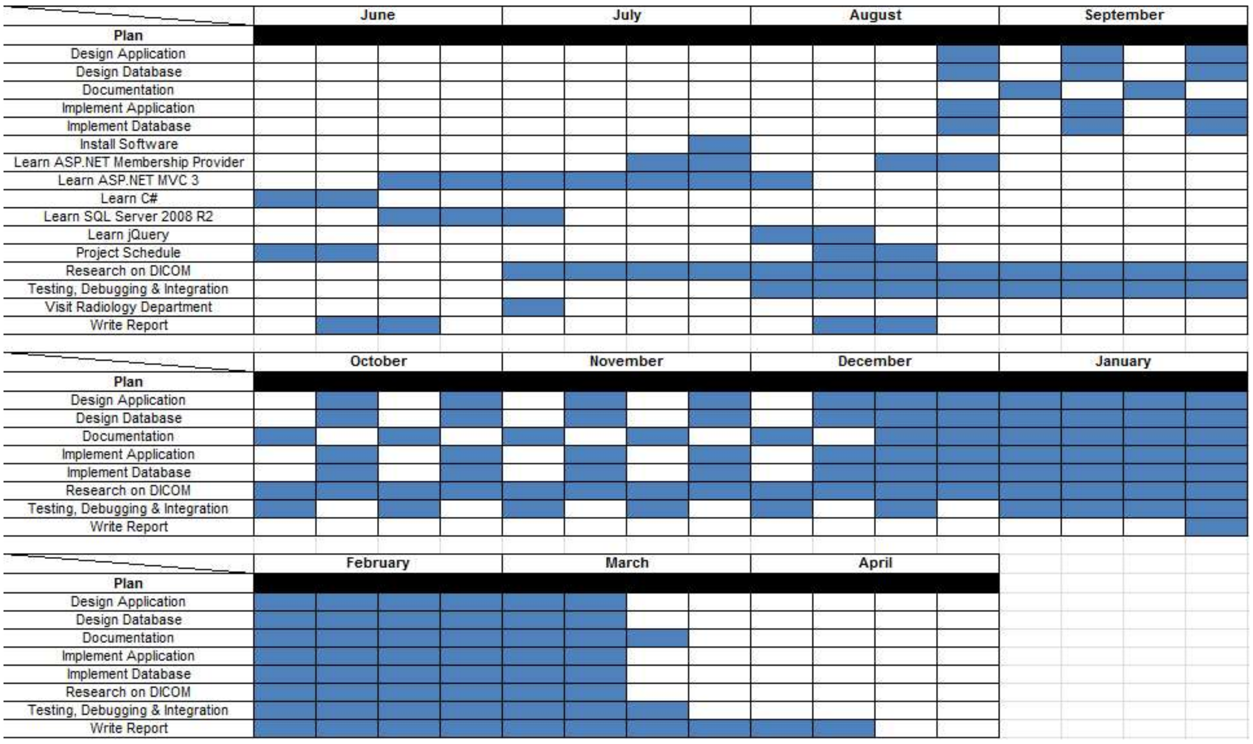
Figure 1 – Original Schedule

Figure 2 shows the actual development that took place, due to the expected difficulties that were predicted during initial planning.

Notice that development activity became unusually heavy since the second week of December. This was due to discovering that NHibernate, a critical software framework for abstracting interactions with the underlying Microsoft SQL Server 2008 R2, was deemed to be incompatible with a new feature of the database very late during development.

As such, all work that was done up to that point had to be abandoned, and the project essentially restarted from scratch. Moreover, the DICOM standard specifications are still posing a challenge at the time of writing, thereby necessitating more research activity on it until the last iteration of development.

Thankfully, despite the restarting of development work, the database schema was able to be reused due to a conscious effort from the beginning to separate the data persistence and application logic layers.



	June				July				August				September			
Plan	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Design Application												🟦		🟦		🟦
Design Database												🟦		🟦		🟦
Documentation													🟦		🟦	
Implement Application												🟦		🟦		🟦
Implement Database												🟦		🟦		🟦
Install Software								🟦								
Learn ASP.NET Membership Provider							🟦	🟦			🟦	🟦				
Learn ASP.NET MVC 3			🟦	🟦	🟦	🟦	🟦	🟦	🟦							
Learn C#	🟦	🟦														
Learn SQL Server 2008 R2			🟦	🟦	🟦											
Learn jQuery									🟦	🟦						
Project Schedule	🟦	🟦								🟦	🟦					
Research on DICOM					🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
Testing, Debugging & Integration									🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
Visit Radiology Department					🟦											
Write Report		🟦	🟦							🟦	🟦					

	October				November				December				January			
Plan	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Design Application		🟦		🟦		🟦		🟦		🟦	🟦	🟦	🟦	🟦	🟦	🟦
Design Database		🟦		🟦		🟦		🟦		🟦	🟦	🟦	🟦	🟦	🟦	🟦
Documentation	🟦		🟦		🟦		🟦		🟦		🟦	🟦	🟦	🟦	🟦	🟦
Implement Application		🟦		🟦		🟦		🟦		🟦	🟦	🟦	🟦	🟦	🟦	🟦
Implement Database		🟦		🟦		🟦		🟦		🟦	🟦	🟦	🟦	🟦	🟦	🟦
Research on DICOM	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
Testing, Debugging & Integration	🟦		🟦		🟦		🟦		🟦		🟦		🟦	🟦	🟦	🟦
Write Report																🟦

	February				March				April			
Plan	■	■	■	■	■	■	■	■	■	■	■	■
Design Application	🟦	🟦	🟦	🟦	🟦	🟦						
Design Database	🟦	🟦	🟦	🟦	🟦	🟦						
Documentation	🟦	🟦	🟦	🟦	🟦	🟦	🟦					
Implement Application	🟦	🟦	🟦	🟦	🟦	🟦						
Implement Database	🟦	🟦	🟦	🟦	🟦	🟦						
Research on DICOM	🟦	🟦	🟦	🟦	🟦	🟦						
Testing, Debugging & Integration	🟦	🟦	🟦	🟦	🟦	🟦	🟦					
Write Report	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦		

Figure 2 – Actual Schedule

1.5 Resources

Two specific software suites were utilized in the design and development of RIS:

- Microsoft SQL Server 2008 R2
- Microsoft Visual Studio 2010 Ultimate

In addition to design and development, this project also utilized other software as follows:

- AJAX Control Toolkit. An AJAX UI framework built on top of the default ASP.NET AJAX controls
- ASP.NET. A web application framework from Microsoft to build dynamic web applications
- dotPeek. A .NET decompiler
- Git. A revision control managing the changes to program files, database files, documentation etc.
- Microsoft Office. A productivity suite for formally documenting the development process in the form of a report
- NuGet. A package manager for the .NET Framework
- ReSharper. A productivity extension to Visual Studio that helps to maintain and improve C# code (among many others) in Visual Studio projects.

1.6 Report Organization

This report is organized in the following manner:

Chapter 1 presents an introduction the reader regarding the project. It includes the background information to help readers understand the objectives of the project.

Chapter 2 presents an explanation of the business and working model of a sample radiology department. It gives a sample radiology workflow and layman explanations of jargons. A literature review of DICOM, C#, Microsoft SQL Server 2008 R2 and aspects of ASP.NET that were utilized is also given.

Chapter 3 presents the database system design that is used in the web application. It provides the specifics of the schema of the database and provides the rationale for the choices made, where applicable.

Chapter 4 gets into the technical implementation of the web application. The server and client side designs are elaborated upon in this section. It provides the rationale and advantages of critical design decisions.

Chapter 5 presents the experimental system and results. Details will be given on the rationale behind implementation. Where appropriate, screenshots will be provided.

Chapter 6 concludes the project and includes discussions on possible future development and enhancements.

2 Literature Review

2.1 Radiology Business Model

To better understand the workflow processes and jargons used within a radiology department, a paid, introductory tour was arranged with Raffles Medical Group. The tour occurred during the first week of July 2011.

According to the medical center, a typical process for outpatients would be as shown in Figure 3, whereas a typical process for inpatients would be as shown in Figure 4.

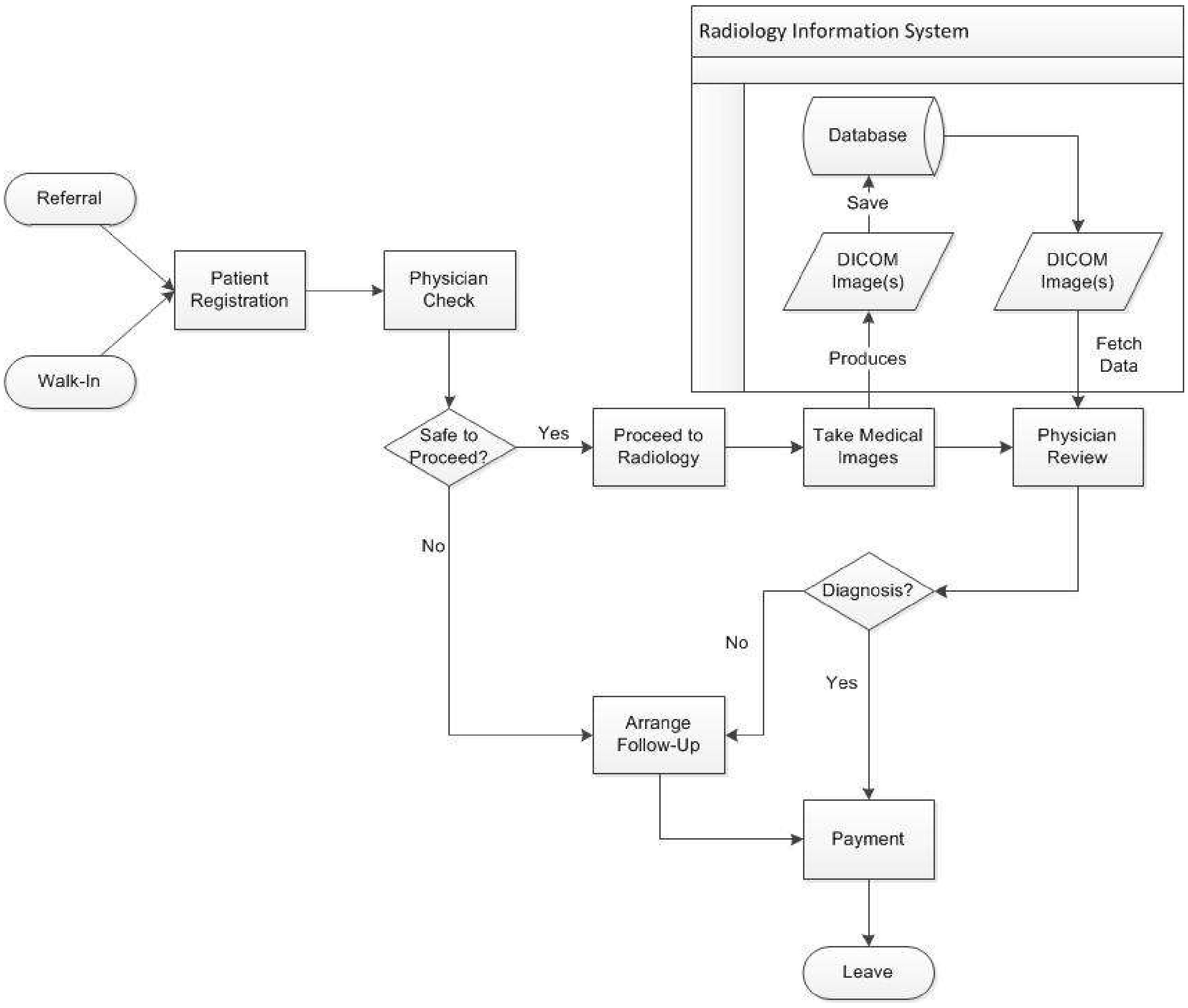


Figure 3 – Outpatient Process

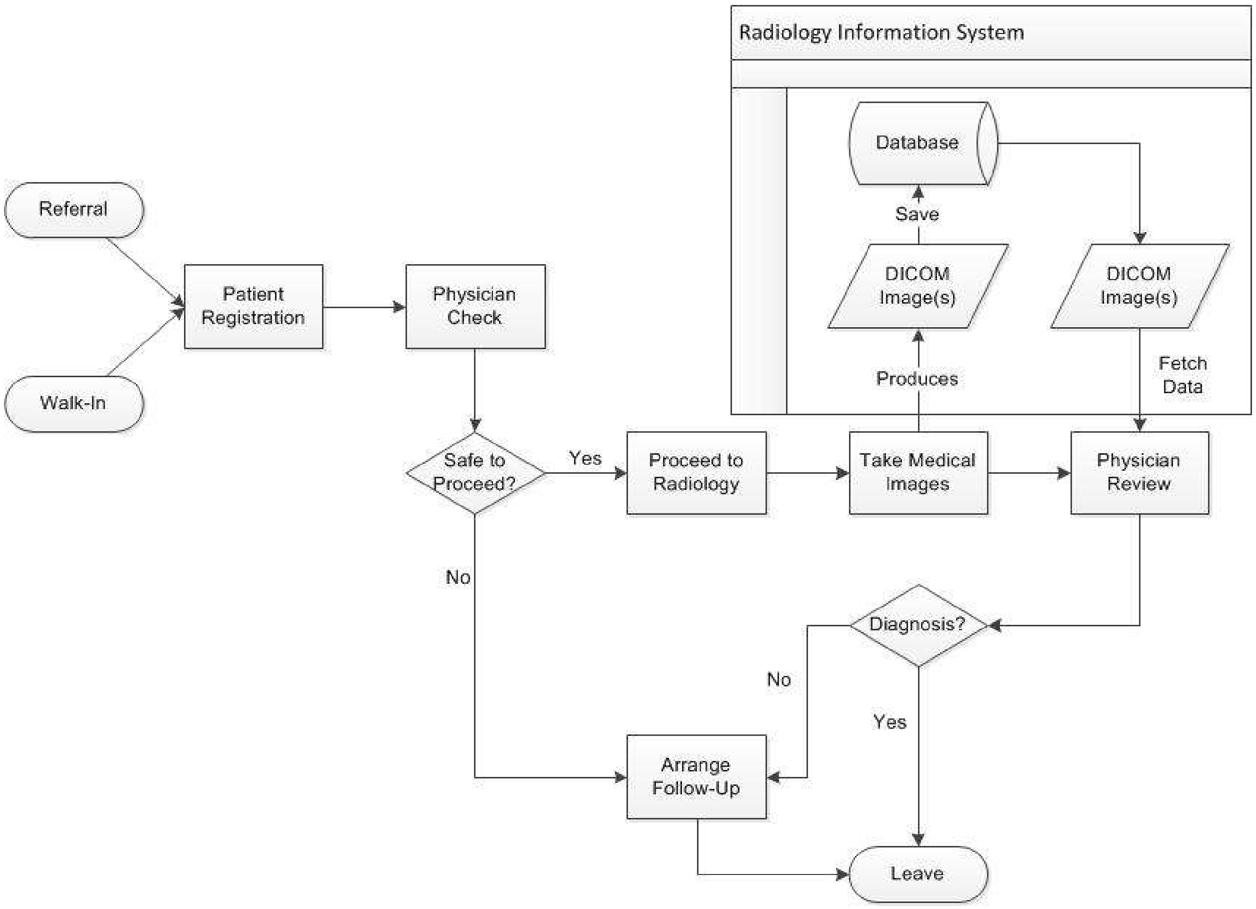


Figure 4 – Inpatient Process

For both processes, we see that the RIS is only used during the saving and retrieving of medical images.

When a patient is referred to the radiology department, s/he has started a study. In a study, different types of imaging equipment (series) might be used. Each series represents a different modality, and can have several images. This information is also encapsulated in DICOM (explained later), as seen in the image below.

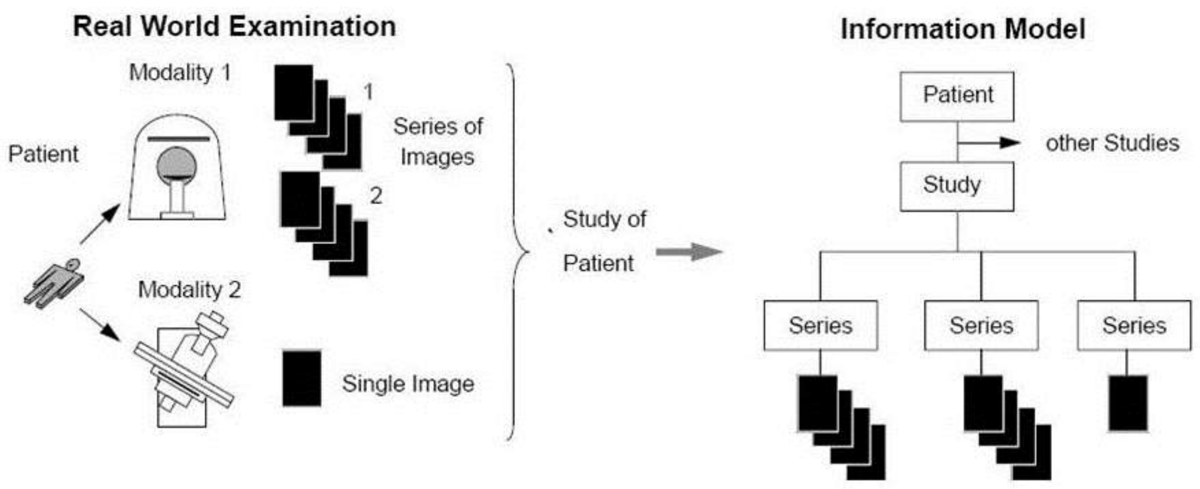


Figure 5 – Radiology to DICOM

2.2 ASP.NET

ASP.NET is a web application framework by Microsoft. It is built on top the Common Language Runtime, allowing .NET supported languages to be used. The main building block of ASP.NET is its web pages, officially known as Web Forms.

ASP.NET web forms contain static HTML markup, as well as markup defining server-side web controls and user controls for both static and dynamic content on the page. Static content is placed in the client facing .aspx file, while dynamic content server are placed within a <% %> block in a separate .aspx.cs (otherwise known as a code-behind file) linked that the .aspx file.

An ASP.NET project has three other file types that aid in reducing development time – the web.config, master and sitemap files. Web.config is an XML document containing the main settings and configuration file for an ASP.NET application. Specifically, the web.config file contains information that controls module loading, security configuration, session state, application language, compilation settings and database connection strings. The master file is the file from which all other .aspx files in the application derives their layout from, and the sitemap file simplifies site management by consolidating all web pages into one centralized location.

Recent versions of ASP.NET added in a Membership provider feature, and AJAX framework. Membership handles common security tasks such as logins, registration, authentication and role management, while AJAX provides asynchronous controls for the client. The RIS has AJAX Control Toolkit, a AJAX framework by Microsoft, installed for future improvements.

2.2 C#

C# is a programming language designed for the Common Language Runtime. The most recent version, 4.0, was released on April 2010. C# is most commonly used for application and business logic in the code-behind files for the ASP web pages.

2.3 Microsoft SQL Server 2008 R2

Microsoft SQL Server 2008 R2 is an updated release of Microsoft SQL Server 2008. The key feature of this release is a feature named FileStream, which integrates the database engine with NTFS by storing varbinary(max) binary large object (BLOB) data as files on the file system itself. There are three key reasons why FileStream is used for RIS:

1) DICOM files are on average larger than 1MB in size
2) Fast read access is considered an important requirement
3) RIS uses a middle tier for application and business logic

The major disadvantage of FileStream is its incompatibility with NHibernate, an open source object-relational mapper. As such, a significant portion of relational data persistence related programming tasks that could have been abstracted now needs to be done manually.

Like other major commercial database management systems, SQL Server has its own management software called SQL Server Management Studio. It is through this software that most of the initial interactions with SQL Server take place, before it is accessed programmatically by RIS.

2.4 DICOM

DICOM is a standard for the creation, transmission, and storage of digital medical image and report data. It defines a data dictionary, data structures, file formats, client and server services, workflow, and compression, among other things. The most important aspects of DICOM are that it contains a file header portion, a File Meta Information portion, and a single SOP (Service-Object Pair) instance [1].

The header is made up of a 128 byte preamble, followed by the characters 'D', 'I', 'C', 'M'. The preamble is usually not used, and thus contains all zeroes, although applications may use it for proprietary data [2].

After the header is the File Meta Information. This portion follows a tagged file format, and contains information about the file, the series and study it belongs to, and the patient that it belongs to [1]. This information is frequently parsed and used in the RIS itself.

The actual DICOM standard itself is very difficult to understand and use as it includes support for ACR and NEMA, two non-compatible predecessor standards. This problem is compounded by the fact that medical equipment manufacturers need not be fully compliant with the standard itself, thereby resulting in a myriad of possible data combinations.

A visual reference of a fully compliant DICOM file is provided in the following page.

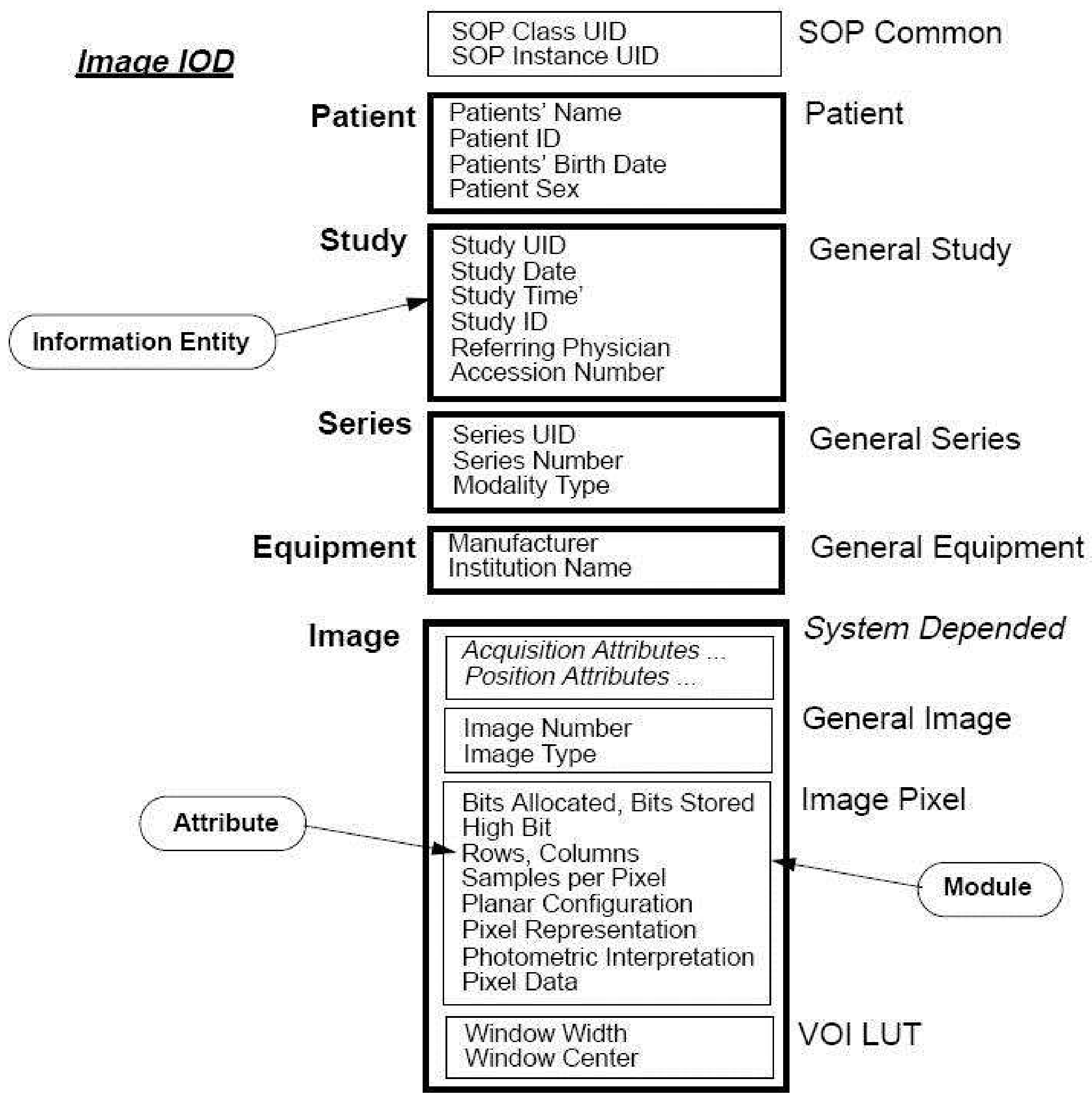


Figure 6 – Sample DICOM File

2.5 DICOM Frameworks

Owing to its complexity, the author has spent a considerable amount of time searching for and testing out DICOM frameworks compatible with the .NET platform. The most notable frameworks currently are openDICOM and dicomSharp, both developed natively on the .NET platform, but both of which has long seen development abandoned.

However, at the time of writing the final draft of this report, no single framework has yet been found to be suitable for usage in RIS. As such, the application itself does not use any DICOM frameworks to simplify interactions with DICOM files. Instead a simple workaround, described in later chapters, is used.

The author recommends that future versions of RIS be migrated to the Java platform as development of DICOM frameworks there are much more active and maintained.

3 Database System Design

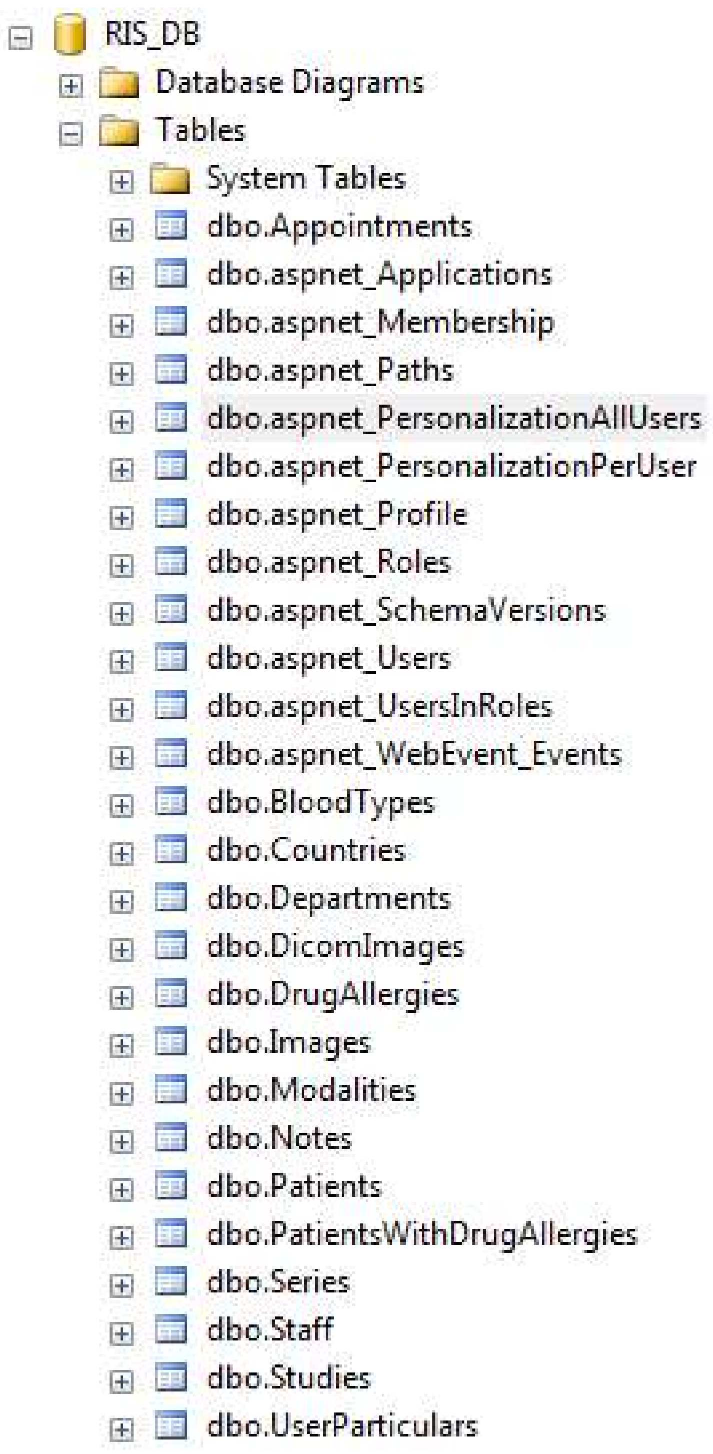
3.1 Overview

The RIS web application database is made up of two logically distinct collections of SQL tables. The first collection of tables is used to model domain entities in an object relational environment, and the other collection of tables is used to handle security issues common to web applications.

The latter is offloaded to ASP.NET's Membership framework, which is automatically installed together with the .NET framework. It handles logins, registrations, authentication and role management; thereby reducing development time, and thus freeing up more time to solve problems in the business domain.

Both collections of tables are linked together with several referential integrities, and located within the same database system, so as not to make the system overly complicated. A point to take note of is that although security has been offloaded to ASP.NET, code still needs to be written to call the Membership framework functions.

Figure 5 shows an overview of all the SQL tables used in the RIS web application. For an overall view of all the tables and their relationships, please refer to the appendix.

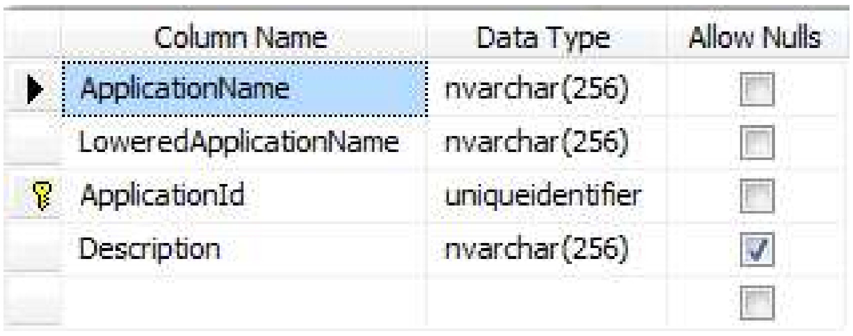


RIS_DB
Database Diagrams
Tables
System Tables
dbo.Appointments
dbo.aspnet_Applications
dbo.aspnet_Membership
dbo.aspnet_Paths
dbo.aspnet_PersonalizationAllUsers
dbo.aspnet_PersonalizationPerUser
dbo.aspnet_Profile
dbo.aspnet_Roles
dbo.aspnet_SchemaVersions
dbo.aspnet_Users
dbo.aspnet_UsersInRoles
dbo.aspnet_WebEvent_Events
dbo.BloodTypes
dbo.Countries
dbo.Departments
dbo.DicomImages
dbo.DrugAllergies
dbo.Images
dbo.Modalities
dbo.Notes
dbo.Patients
dbo.PatientsWithDrugAllergies
dbo.Series
dbo.Staff
dbo.Studies
dbo.UserParticulars

Figure 7 – All Database Tables

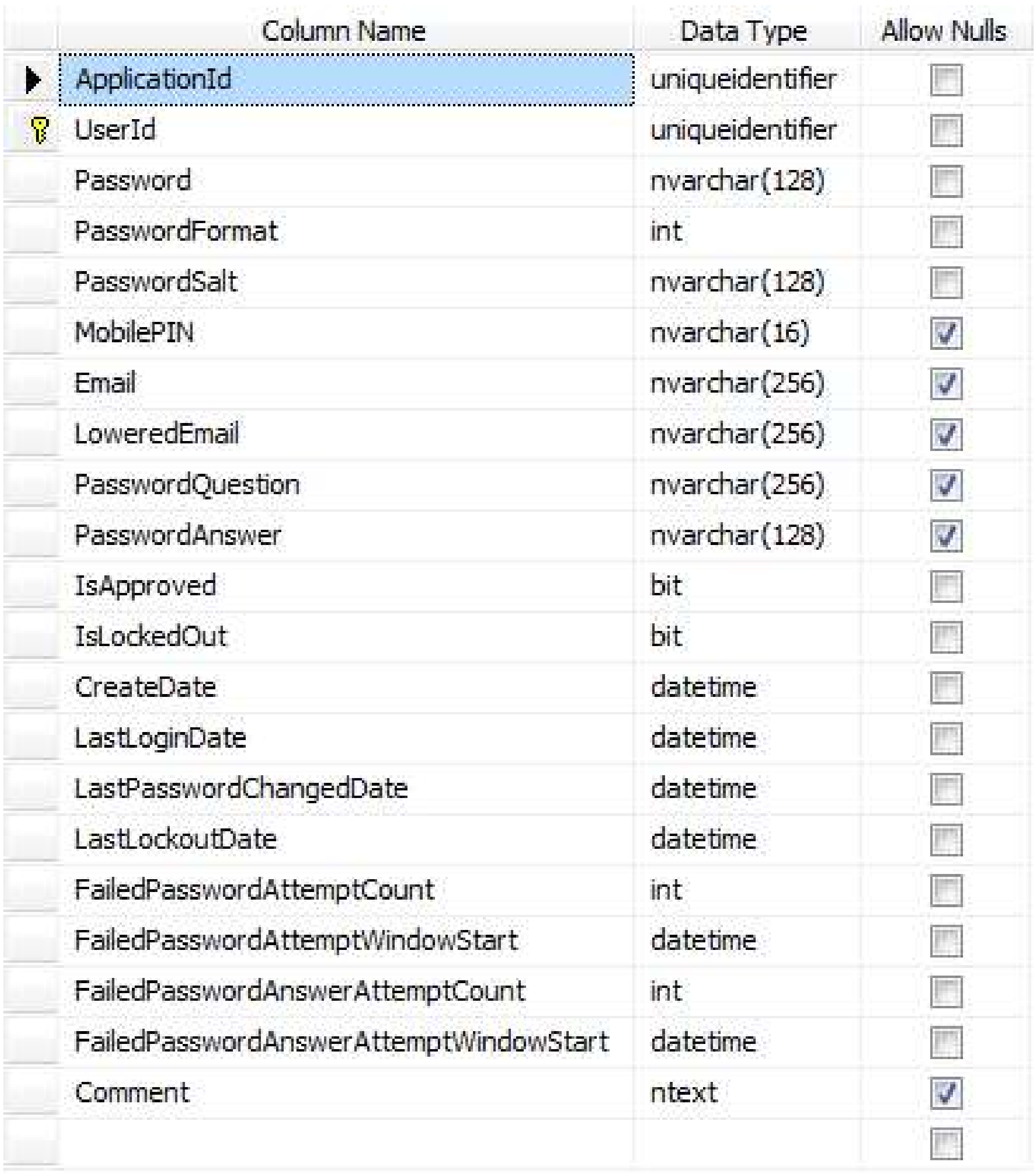
3.2 ASP.NET Tables

As described in the previous section, the ASP.NET Membership framework handles user registration, login, authentication and role management features. Detailed schemas for the ASP.NET Membership tables are included below.



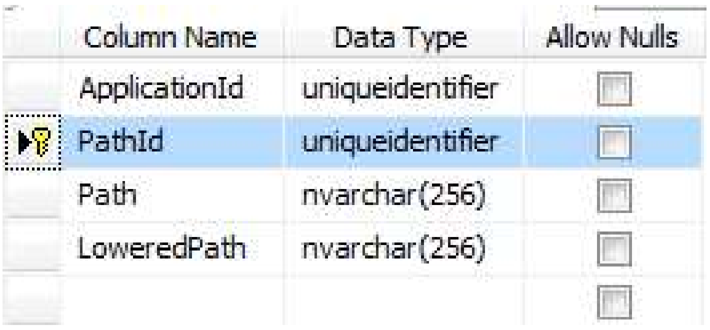
Column Name	Data Type	Allow Nulls
ApplicationName	nvarchar(256)	☐
LoweredApplicationName	nvarchar(256)	☐
ApplicationId	uniqueidentifier	☐
Description	nvarchar(256)	✓
		☐

Figure 8 – aspnet_Applications



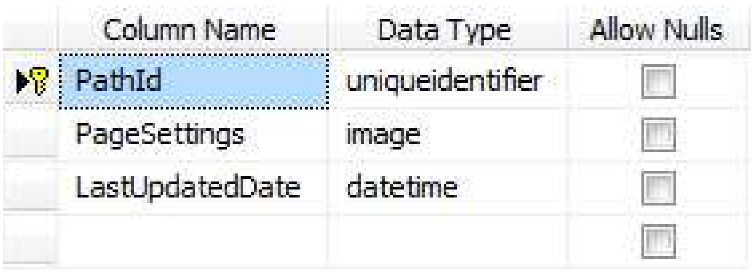
Column Name	Data Type	Allow Nulls
ApplicationId	uniqueidentifier	☐
UserId	uniqueidentifier	☐
Password	nvarchar(128)	☐
PasswordFormat	int	☐
PasswordSalt	nvarchar(128)	☐
MobilePIN	nvarchar(16)	✓
Email	nvarchar(256)	✓
LoweredEmail	nvarchar(256)	✓
PasswordQuestion	nvarchar(256)	✓
PasswordAnswer	nvarchar(128)	✓
IsApproved	bit	☐
IsLockedOut	bit	☐
CreateDate	datetime	☐
LastLoginDate	datetime	☐
LastPasswordChangedDate	datetime	☐
LastLockoutDate	datetime	☐
FailedPasswordAttemptCount	int	☐
FailedPasswordAttemptWindowStart	datetime	☐
FailedPasswordAnswerAttemptCount	int	☐
FailedPasswordAnswerAttemptWindowStart	datetime	☐
Comment	ntext	✓
		☐

Figure 9 – aspnet_Membership



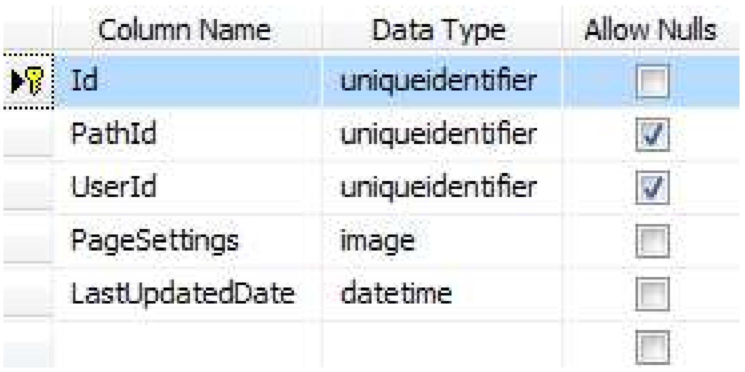
Column Name	Data Type	Allow Nulls
ApplicationId	uniqueidentifier	☐
PathId	uniqueidentifier	☐
Path	nvarchar(256)	☐
LoweredPath	nvarchar(256)	☐
		☐

Figure 10 – aspnet_Paths



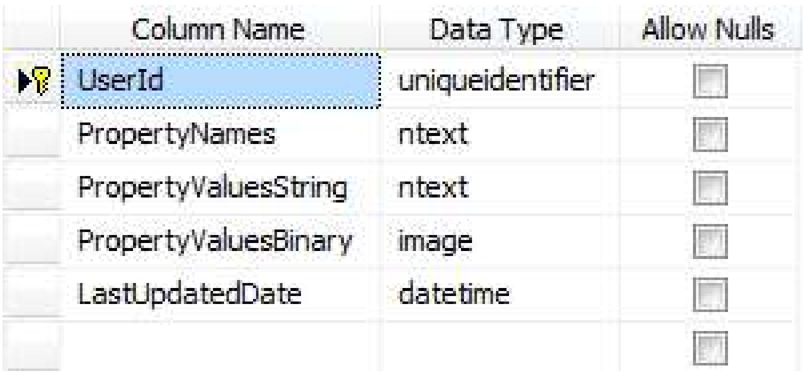
Column Name	Data Type	Allow Nulls
PathId	uniqueidentifier	☐
PageSettings	image	☐
LastUpdatedDate	datetime	☐
		☐

Figure 11 – aspnet_PersonalizationAllUsers



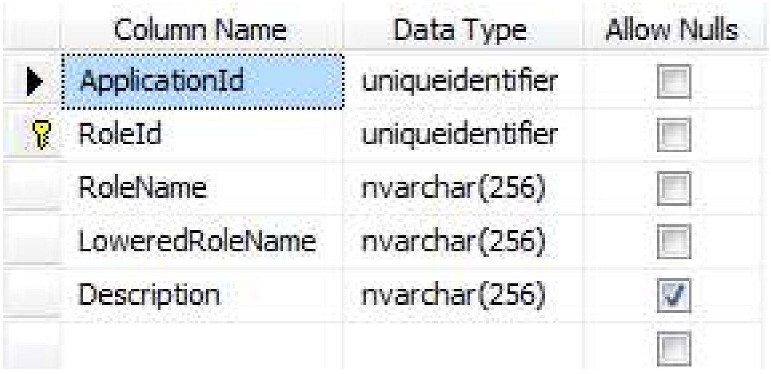
Column Name	Data Type	Allow Nulls
Id	uniqueidentifier	☐
PathId	uniqueidentifier	✓
UserId	uniqueidentifier	✓
PageSettings	image	☐
LastUpdatedDate	datetime	☐
		☐

Figure 12 – aspnet_PersonalizationPerUser



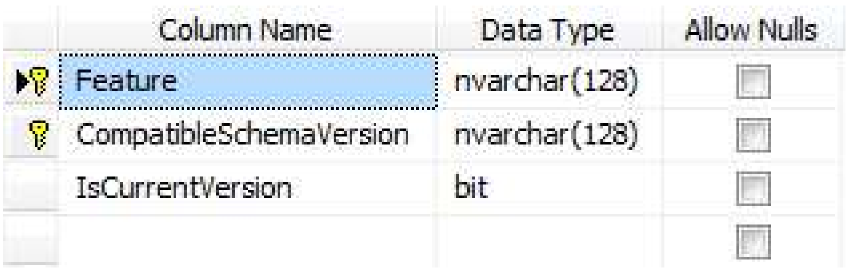
Column Name	Data Type	Allow Nulls
UserId	uniqueidentifier	☐
PropertyNames	ntext	☐
PropertyValuesString	ntext	☐
PropertyValuesBinary	image	☐
LastUpdatedDate	datetime	☐
		☐

Figure 13 – aspnet_Profile



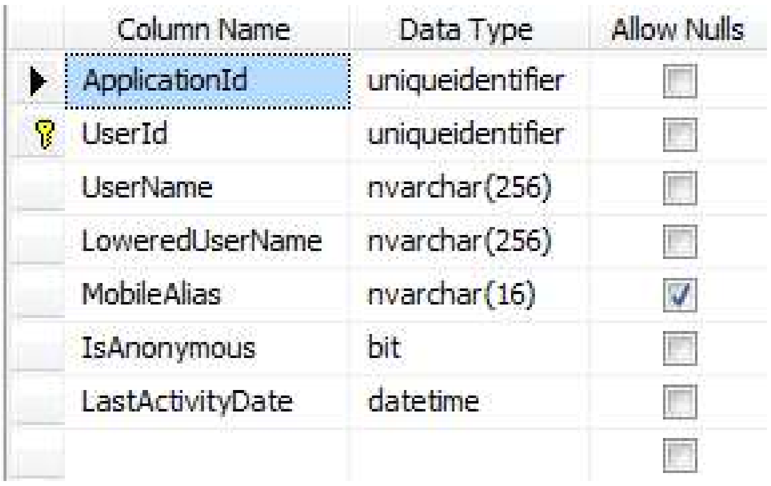
Column Name	Data Type	Allow Nulls
ApplicationId	uniqueidentifier	☐
RoleId	uniqueidentifier	☐
RoleName	nvarchar(256)	☐
LoweredRoleName	nvarchar(256)	☐
Description	nvarchar(256)	✓
		☐

Figure 14 – aspnet_Roles



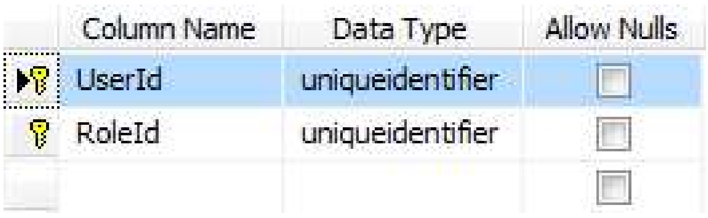
Column Name	Data Type	Allow Nulls
Feature	nvarchar(128)	☐
CompatibleSchemaVersion	nvarchar(128)	☐
IsCurrentVersion	bit	☐
		☐

Figure 15 – aspnet_SchemaVersions



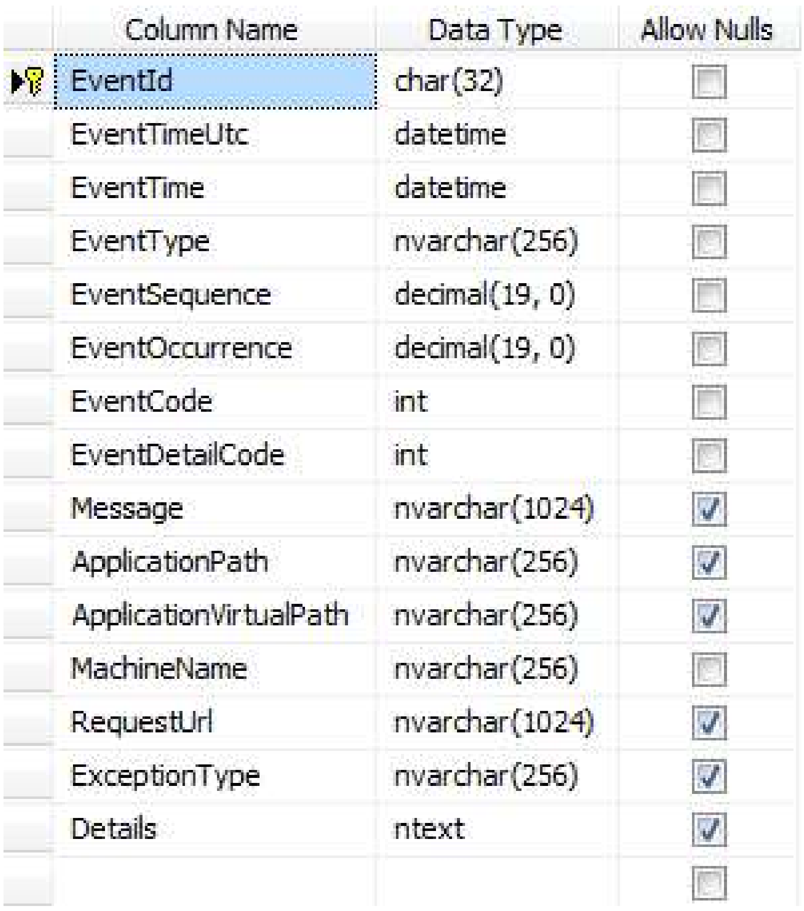
Column Name	Data Type	Allow Nulls
ApplicationId	uniqueidentifier	☐
UserId	uniqueidentifier	☐
UserName	nvarchar(256)	☐
LoweredUserName	nvarchar(256)	☐
MobileAlias	nvarchar(16)	✓
IsAnonymous	bit	☐
LastActivityDate	datetime	☐
		☐

Figure 16 – aspnet_Users



Column Name	Data Type	Allow Nulls
UserId	uniqueidentifier	☐
RoleId	uniqueidentifier	☐
		☐

Figure 17 – aspnet_UsersInRoles



Column Name	Data Type	Allow Nulls
EventId	char(32)	☐
EventTimeUtc	datetime	☐
EventTime	datetime	☐
EventType	nvarchar(256)	☐
EventSequence	decimal(19, 0)	☐
EventOccurrence	decimal(19, 0)	☐
EventCode	int	☐
EventDetailCode	int	☐
Message	nvarchar(1024)	✓
ApplicationPath	nvarchar(256)	✓
ApplicationVirtualPath	nvarchar(256)	✓
MachineName	nvarchar(256)	☐
RequestUrl	nvarchar(1024)	✓
ExceptionType	nvarchar(256)	✓
Details	ntext	✓
		☐

Figure 18 – aspnet_WebEvent_Events

To create the above tables in the database, we navigate to C:\%windir%\Microsoft.NET\Framework\version and run the aspnet_regsql executable tool.

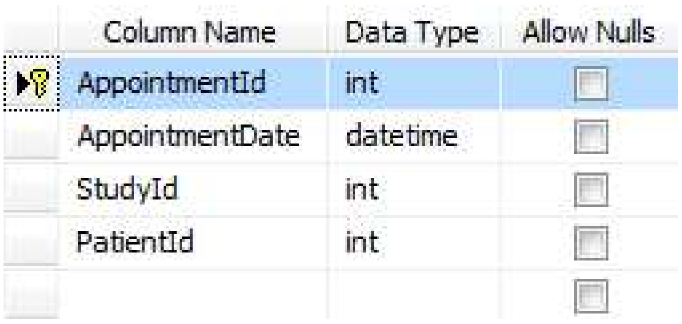
Thereafter, we have to enable the Membership feature in the global web.config file by adding the following lines of code:

```xml
<configuration>
  <connectionStrings>
    <add name="ApplicationServices" connectionString="Data Source=.;Initial Catalog=
RIS_DB;User Id=sa;Password=93324577;" providerName="System.Data.SqlClient" />
  </connectionStrings>
  <system.web>
    <authentication mode="Forms">
      <forms loginUrl="~/Account/Login.aspx" protection="All" slidingExpiration="tru
e" ticketCompatibilityMode="Framework40" />
      <!--Comment the line above, and uncomment below when deploying to production--
>
      <!--
<forms loginUrl="~/Account/Login.aspx" protection="All" slidingExpiration="true" req
uireSSL="true" ticketCompatibilityMode="Framework40" />-->
    </authentication>
    <membership>
      <providers>
        <clear />
        <add name="AspNetSqlMembershipProvider" type="System.Web.Security.SqlMembers
hipProvider" connectionStringName="ApplicationServices" enablePasswordRetrieval="fal
se" enablePasswordReset="true" requiresQuestionAndAnswer="true" requiresUniqueEmail=
"false" maxInvalidPasswordAttempts="5" minRequiredPasswordLength="6" minRequiredNona
lphanumericCharacters="1" passwordAttemptWindow="10" applicationName="/" />
      </providers>
    </membership>
    <profile>
      <providers>
        <clear />
        <add name="AspNetSqlProfileProvider" type="System.Web.Profile.SqlProfileProv
ider" connectionStringName="ApplicationServices" applicationName="/" />
      </providers>
    </profile>
    <roleManager enabled="true">
      <providers>
        <clear />
        <add connectionStringName="ApplicationServices" applicationName="/" name="As
pNetSqlRoleProvider" type="System.Web.Security.SqlRoleProvider" />
        <add applicationName="/" name="AspNetWindowsTokenRoleProvider" type="System.
Web.Security.WindowsTokenRoleProvider" />
      </providers>
    </roleManager>
</configuration>
```

Figure 19 – Membership Configuration

3.3 Radiology Information System (RIS) Tables

In addition to the ASP.NET default tables, the RIS web application also has its own tables that capture the domain entities of a radiology department in object relational tables as shown below. All primary key columns with integer data type are configured to be auto-incrementing identity values.

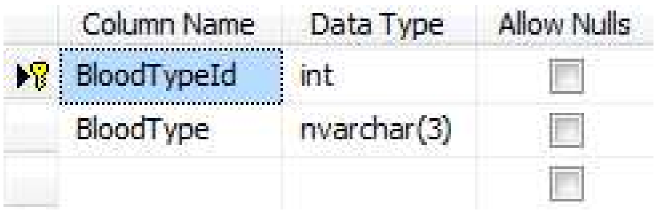


Column Name	Data Type	Allow Nulls
AppointmentId	int	☐
AppointmentDate	datetime	☐
StudyId	int	☐
PatientId	int	☐
		☐

Figure 20 – Appointments

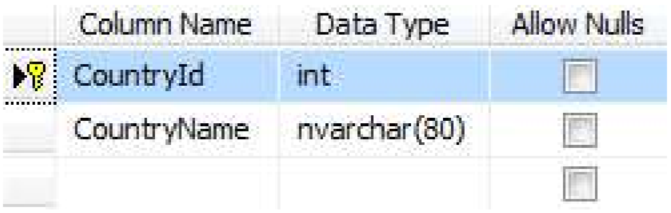
The Appointments table has:

- One-to-many relationship with Studies table
- Many-to-one relationship with Patients table



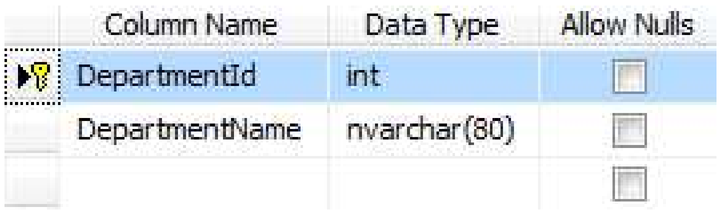
Column Name	Data Type	Allow Nulls
BloodTypeId	int	☐
BloodType	nvarchar(3)	☐
		☐

Figure 21 - BloodTypes



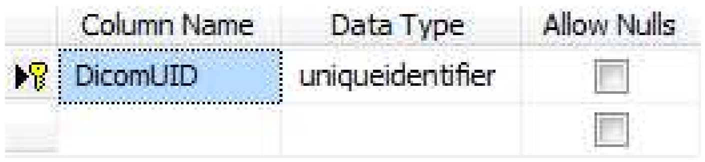
Column Name	Data Type	Allow Nulls
CountryId	int	☐
CountryName	nvarchar(80)	☐
		☐

Figure 22 - Countries



Column Name	Data Type	Allow Nulls
DepartmentId	int	☐
DepartmentName	nvarchar(80)	☐
		☐

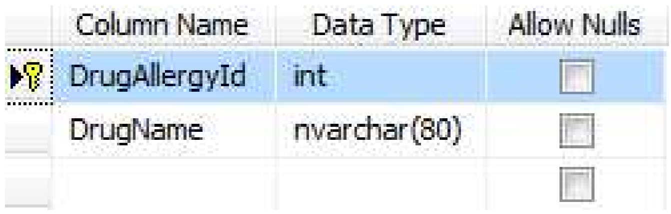
Figure 23 - Departments



Column Name	Data Type	Allow Nulls
DicomUID	uniqueidentifier	☐
		☐

Figure 24 - DicomImages

The DicomImages table uses a Globally Unique Identifier (GUID) as its data type for the primary key column. This number is automatically generated by SQL Server, and guaranteed to be a unique value based upon a time seed.

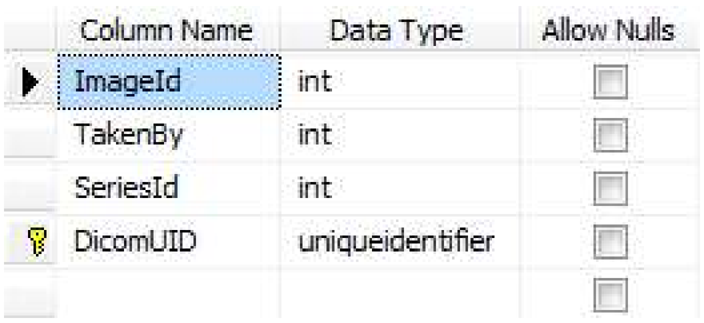


Column Name	Data Type	Allow Nulls
DrugAllergyId	int	☐
DrugName	nvarchar(80)	☐
		☐

Figure 25 - DrugAllergies

The DrugAllergies table has:

- Many-to-many relationship with Patients table

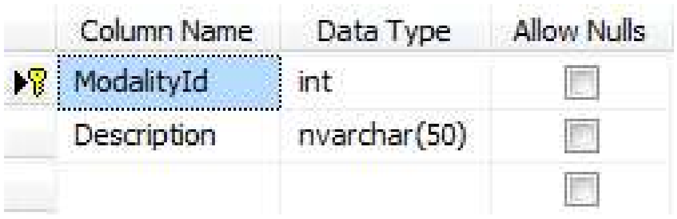


	Column Name	Data Type	Allow Nulls
▶	ImageId	int	☐
	TakenBy	int	☐
	SeriesId	int	☐
🔑	DicomUID	uniqueidentifier	☐
			☐

Figure 26 - Images

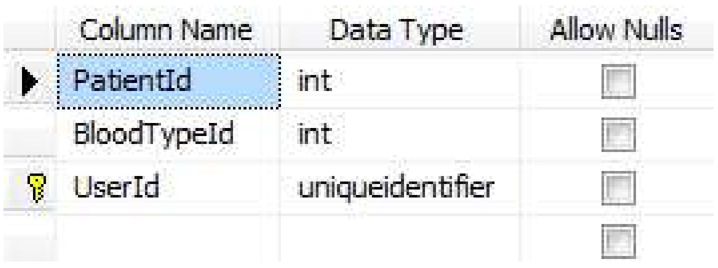
The Images table has:

- Many-to-one relationship with Staff table
- Many-to-one relationship with Series table
- One-to-one relationship with DicomImages table



	Column Name	Data Type	Allow Nulls
▶🔑	ModalityId	int	☐
	Description	nvarchar(50)	☐
			☐

Figure 27 - Modalities

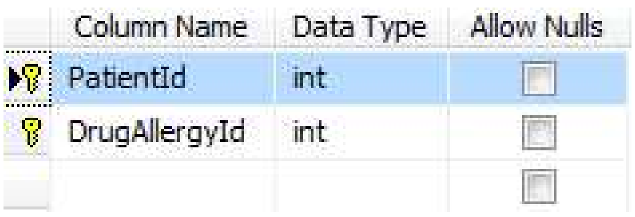


Column Name	Data Type	Allow Nulls
PatientId	int	☐
BloodTypeId	int	☐
UserId	uniqueidentifier	☐
		☐

Figure 28 - Patients

The Patients table has:

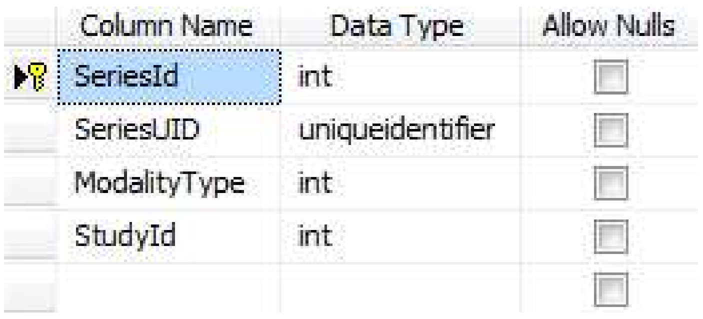
- Many-to-one relationship with BloodType table
- One-to-one relationship with aspnet_Users table
- Many-to-many relationship with DrugAllergies table



Column Name	Data Type	Allow Nulls
PatientId	int	☐
DrugAllergyId	int	☐
		☐

Figure 29 – PatientsWithDrugAllergies

The PatientsWithDrugAllergies is a join table between Patients and DrugAllergies.

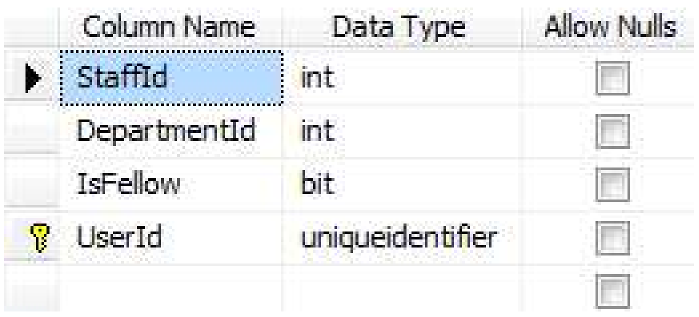


Column Name	Data Type	Allow Nulls
SeriesId	int	☐
SeriesUID	uniqueidentifier	☐
ModalityType	int	☐
StudyId	int	☐
		☐

Figure 30 – Series

The Series table has:

- Many-to-one relationship with Modalities table
- Many-to-one relationship with Studies table

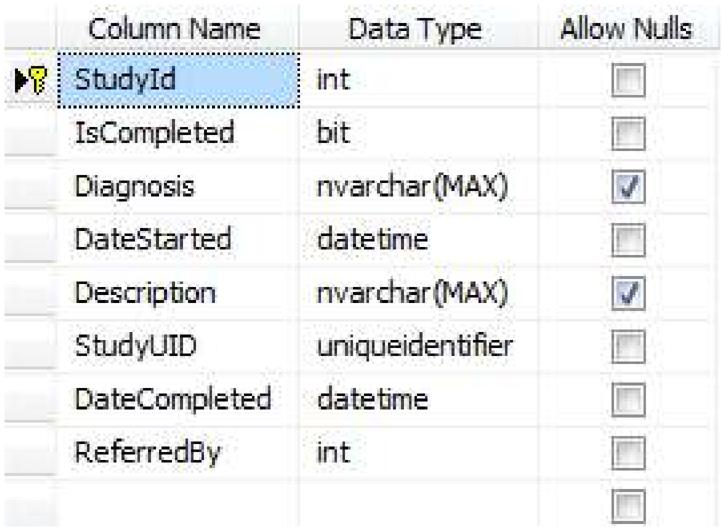


Column Name	Data Type	Allow Nulls
StaffId	int	☐
DepartmentId	int	☐
IsFellow	bit	☐
UserId	uniqueidentifier	☐
		☐

Figure 31 – Staff

The Staff table has:

- Many-to-one relationship with Department table
- One-to-one relationship with aspnet_Users table

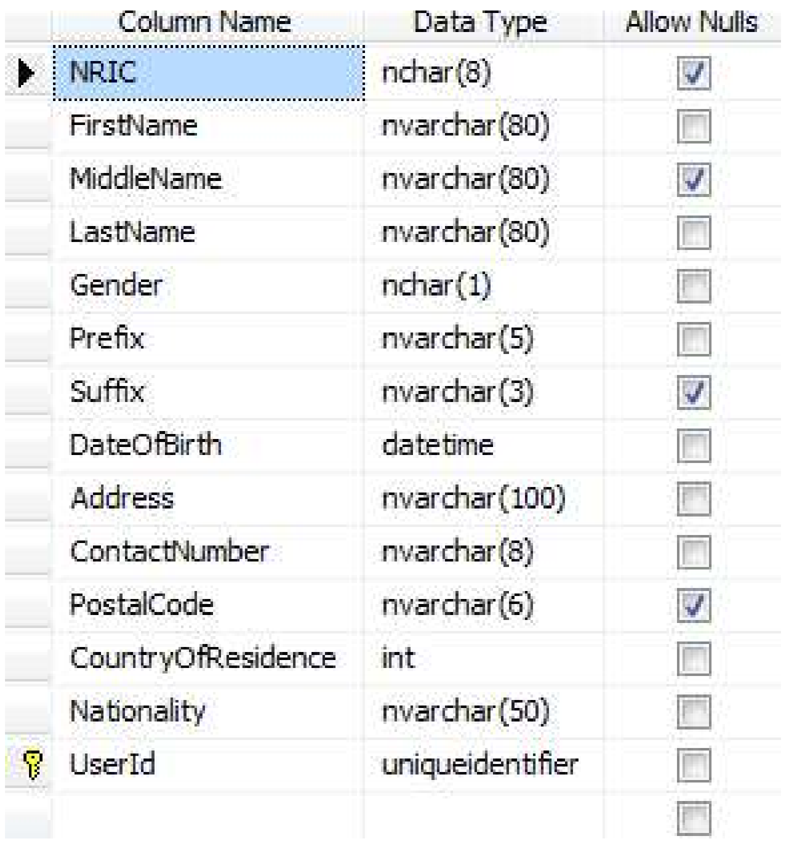


Column Name	Data Type	Allow Nulls
StudyId	int	☐
IsCompleted	bit	☐
Diagnosis	nvarchar(MAX)	✓
DateStarted	datetime	☐
Description	nvarchar(MAX)	✓
StudyUID	uniqueidentifier	☐
DateCompleted	datetime	☐
ReferredBy	int	☐
		☐

Figure 32 - Studies

The Studies table has

- Many-to-one relationship with Staff table



Column Name	Data Type	Allow Nulls
NRIC	nchar(8)	✓
FirstName	nvarchar(80)	
MiddleName	nvarchar(80)	✓
LastName	nvarchar(80)	
Gender	nchar(1)	
Prefix	nvarchar(5)	
Suffix	nvarchar(3)	✓
DateOfBirth	datetime	
Address	nvarchar(100)	
ContactNumber	nvarchar(8)	
PostalCode	nvarchar(6)	✓
CountryOfResidence	int	
Nationality	nvarchar(50)	
UserId	uniqueidentifier	

Figure 33 - UserParticulars

The UserParticulars table has:

- One-to-one relationship with aspnet_Users table

4 Web Application Modules

4.1 Overall Application Design

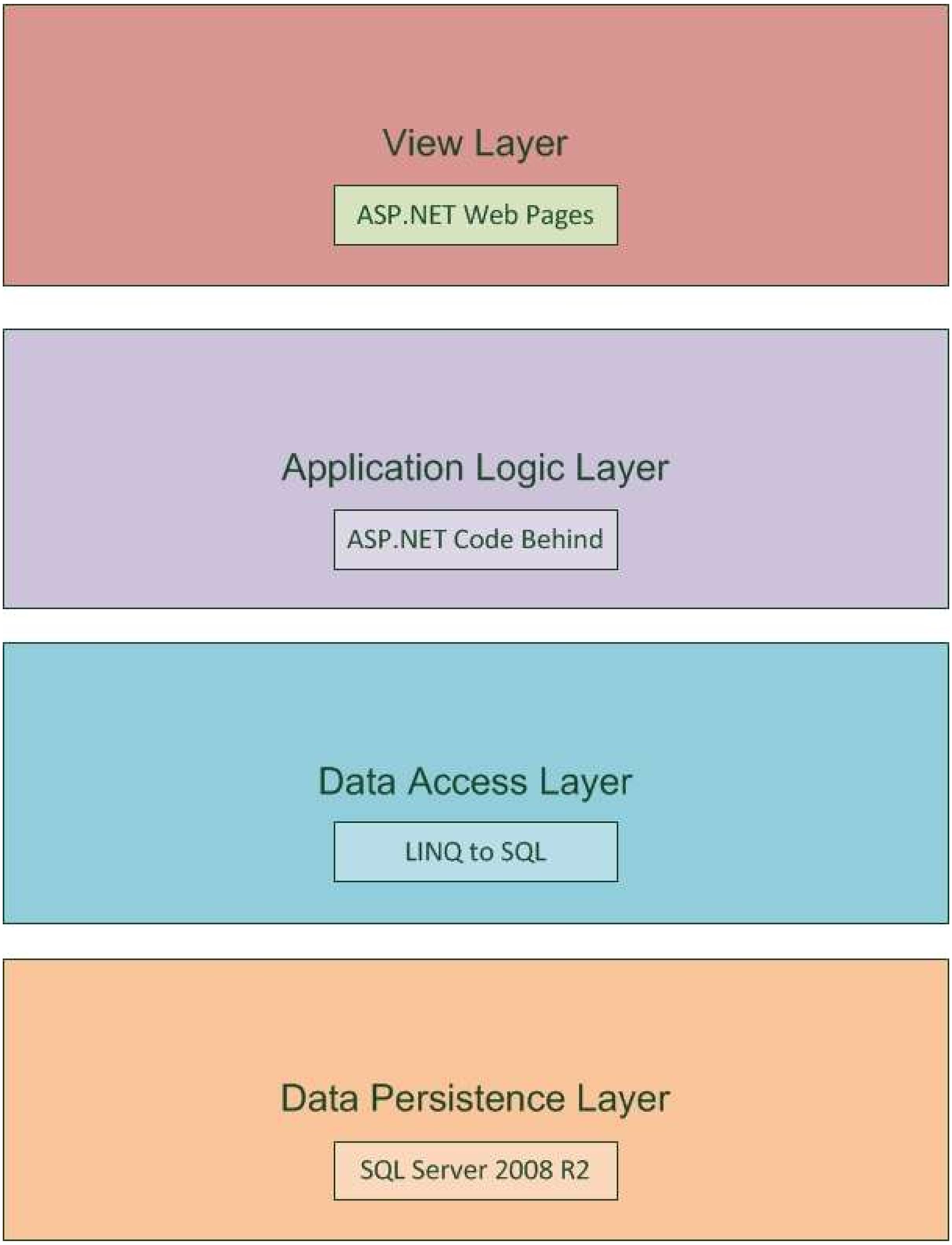


Figure 34 – Overall Design

As seen in the diagram above, the overall design of RIS is that of several distinct layers. Within each layer is the major component driving it. By adopting a layered architecture, changes made to an arbitrary layer will not adversely affect the services it offers to layers above it. In the long term, it is expected that RIS will be highly scalable due to this logical segregation. This chapter focuses on all but the data persistence layer, as it was already covered in the previous chapter.

4.2 Data Access Layer

The data access layer is responsible for handling queries to the database. The main technology used in this layer is LINQ. Language Integrated Query (LINQ, pronounced "link") is a Microsoft .NET Framework component that adds native data querying capabilities to .NET languages. LINQ can be used with any data source and can express query behavior using a programming language, transform data query results into suitable formats, and then manipulate the results.

LINQ to SQL is an O/RM (object relational mapping) implementation that ships in the .NET Framework, and which allows a relational database to be modeled using .NET classes. It can then be queried using LINQ, as well as update/insert/delete data from it. LINQ to SQL fully supports transactions, views, and stored procedures.

To model a database using .NET classes, the tables in the database either have to be imported into Visual Studio or created from scratch. In either scenario, a DataContext class will be automatically generated. This class is the main conduit by which we query entities from the database as well as apply changes. The DataContext class created will have properties that represent each Table modeled within the database, as well as methods for each stored procedures.

RIS itself uses LINQ to SQL exclusively for data access. The figure below shows a small section of the RIS database as modeled in LINQ to SQL.

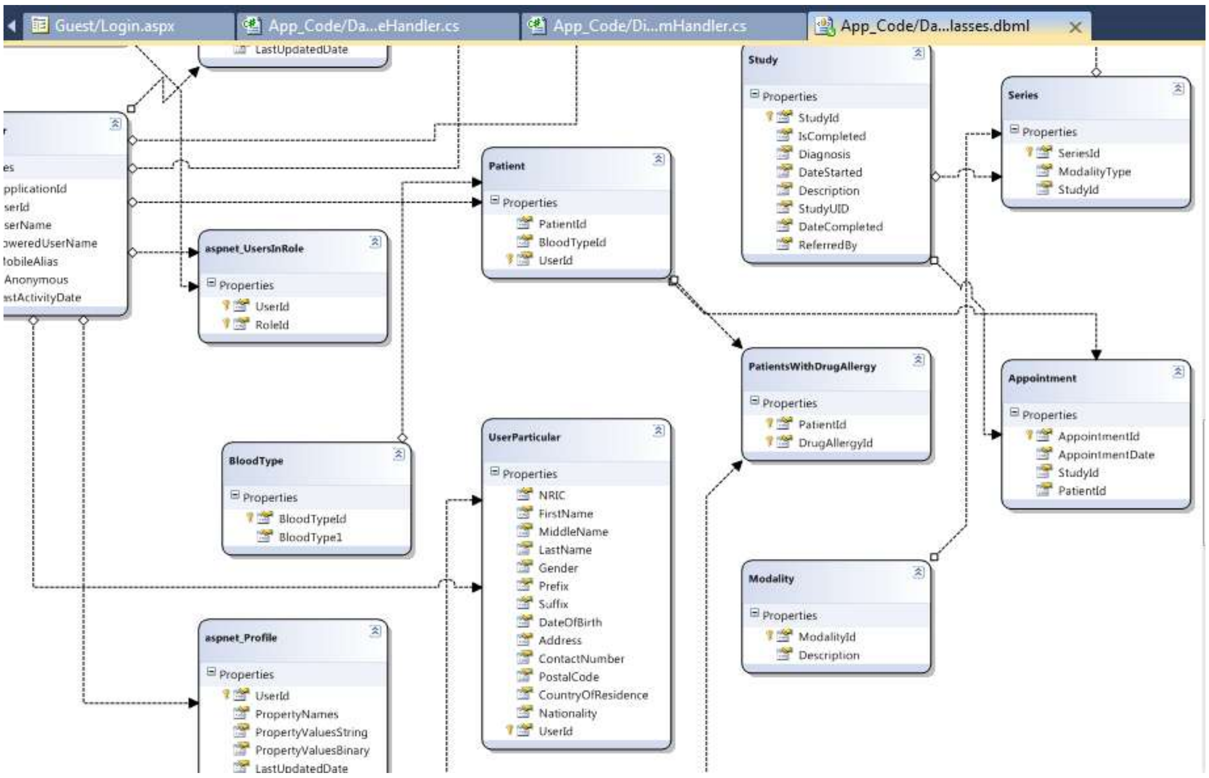


Figure 35 – Section of LINQ Classes

Notice how the modeled tables have arrows pointing to one another. The arrows represent actual relationships between the actual tables in the underlying database, namely one-to-one, one-to-many, and many-to-many.

4.3 Application Logic Layer

The application logic layer is responsible for most of the ‘heavy lifting’ in the application. Its scope of work includes:

- Desensitizing and validating user input
- Desensitizing application output
- Appending and translating query strings
- Maintaining session state
- Ensuring user role segregation
- Providing appropriate role based services to users
- Interacting with the data access layer to query data
- Interacting with the view layer to display data

Although large in scope, we utilize as much as possible features available in ASP.NET to provide some of the low level coding needed to implement the layer’s responsibilities. For example, point 1 uses the HttpUtility class and the various ASP.NET validators in page markup to achieve the desired effect.

The main component of this layer comes from a feature of ASP.NET called Code Behind. As its name implies, Code Behind promotes the separation of code and content. That is, the user interface and the user interface programming logic need not necessarily be written in a single page. There are two ways to separate code and content:

1) By using Code-behind files that are pre-compiled modules written in any of the Microsoft .NET runtime compliant languages.
2) By developing the frequently used page logic in to separate files called Pagelets (also known as page controls) and by using them several times

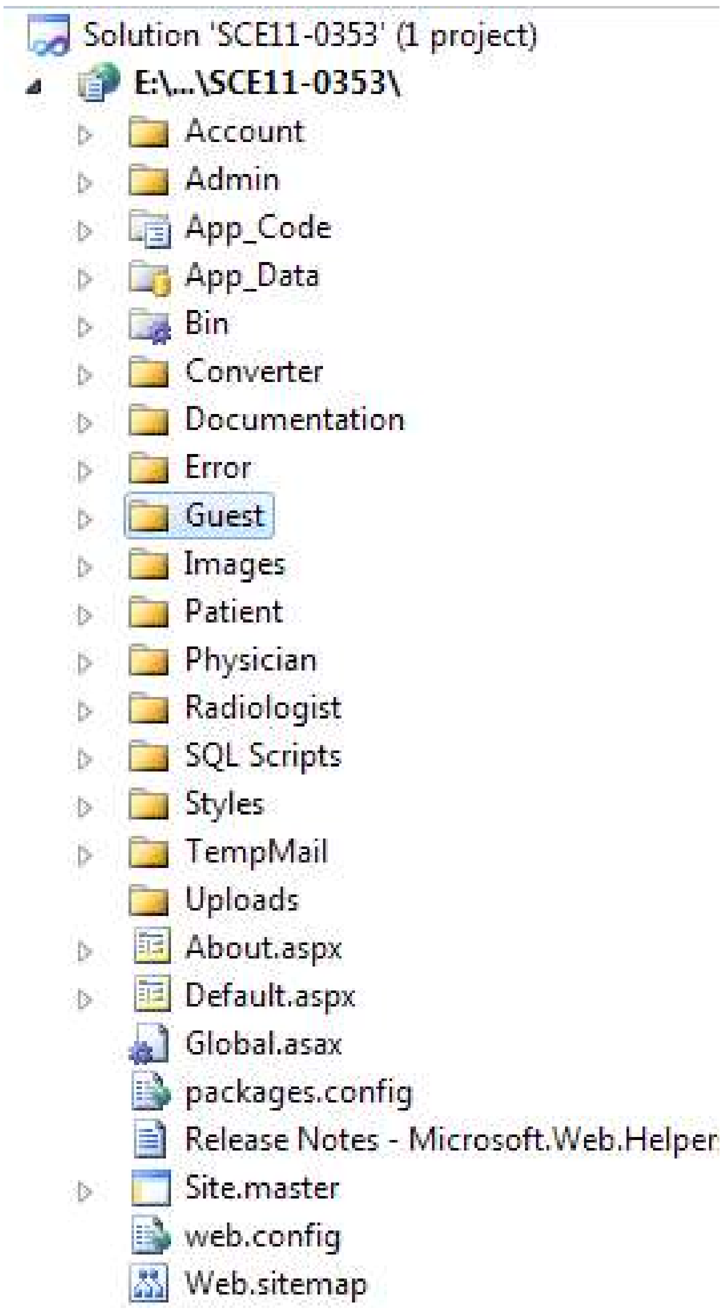
The application uses the first method exclusively.

4.4 View Layer

The view layer renders the model into a human readable form suitable for interaction. An ASP.NET page interacts with the application logic layer in order to generate an appropriate user interface, though ultimately getting its data from the data persistence layer. This layer is similar to its counterpart in the MVC design pattern, only that the overall application does not use this pattern. At present, this layer does not have full AJAX functionality.

4.5 Physical Organization

Although this section chapter is primarily devoted to the logical design of RIS, it is only appropriate that the actual arrangement of the application files is discussed as well. The diagram below shows the file structure of RIS and goes over each important folder briefly.



```
Solution 'SCE11-0353' (1 project)
  E:\...\SCE11-0353\
    Account
    Admin
    App_Code
    App_Data
    Bin
    Converter
    Documentation
    Error
    Guest
    Images
    Patient
    Physician
    Radiologist
    SQL Scripts
    Styles
    TempMail
    Uploads
    About.aspx
    Default.aspx
    Global.asax
    packages.config
    Release Notes - Microsoft.Web.Helper
    Site.master
    web.config
    Web.sitemap
```

Figure 36 – Folder Layout

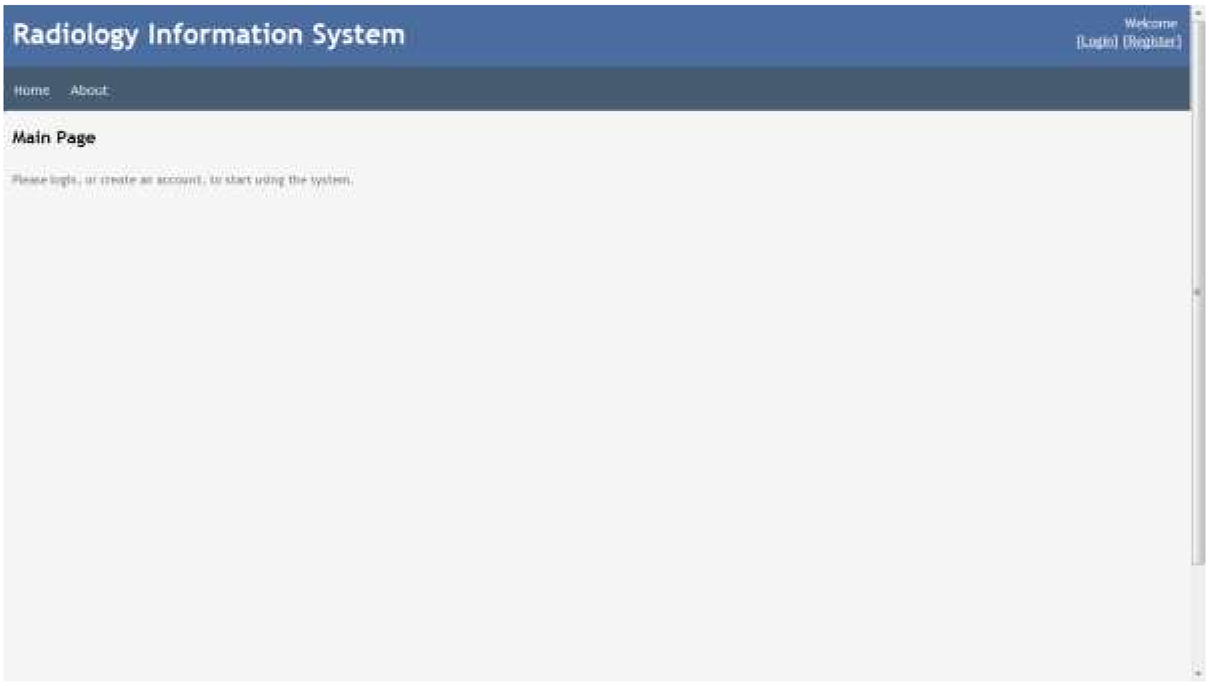
- Admin
    - Contains the ‘actions’ that can be performed by administrators.
- App_Code
    - Contains special executable class files needed for RIS.
- App_Data
    - Contains the actual SQL Server 2008 R2 database instance.
- Bin
    - Contains references to all external libraries and frameworks.
- Converter
    - Contains the command line utility that converts DICOM files into a binary data array for native display in web browsers. Access blocked to all users.
- Documentation
    - Contains the documentation and anything else related. Access blocked to all users.
- Error
    - Contains the custom error page that is to be displayed to remote users.
- Guest
    - Contains the ‘actions’ that anonymous users can perform.
- Images
    - Contains images that are used to make the pages more attractive.
- Patient
    - Contains the ‘actions’ that patients can perform.

- Physician
    - Contains the ‘actions’ that physicians can perform.
- Radiologist
    - Contains the ‘actions’ that radiologists can perform.
- SQL Scripts
    - Contains some executable SQL scripts for populating the database. Access blocked to all users.
- Styles
    - Contains the CSS style sheets for RIS.
- TempMail
    - Meant as a temporary email drop-off folder during development. Access blocked to all users.
- Uploads
    - Meant as a temporary drop-off folder for uploaded DICOM files

The reason for organizing the files in the application is twofold. The first is for easier site management as the numbers of pages grow. The second is for easier migration to ASP.NET MVC in future.

5 Experimental Systems and Results

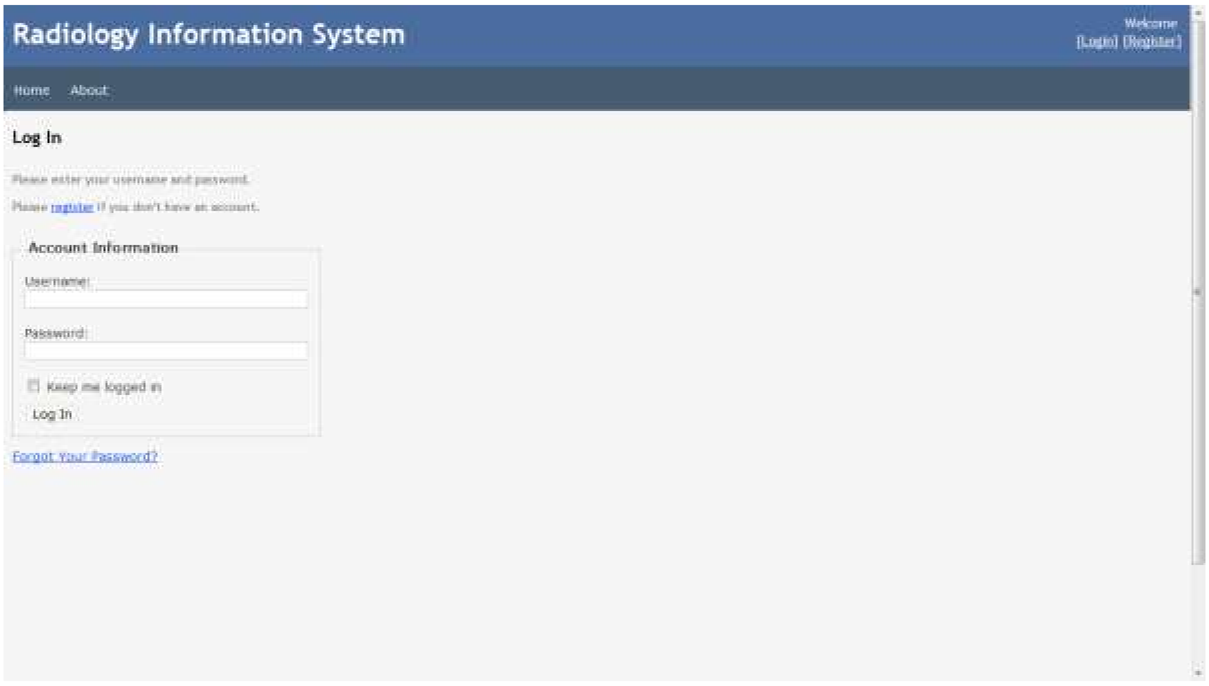
5.1 All Users



Main Page

Please login, or create an account, to start using the system.

Figure 37 – Main Page



Log In

Please enter your username and password.

Please register if you don't have an account.

Account Information

Username:

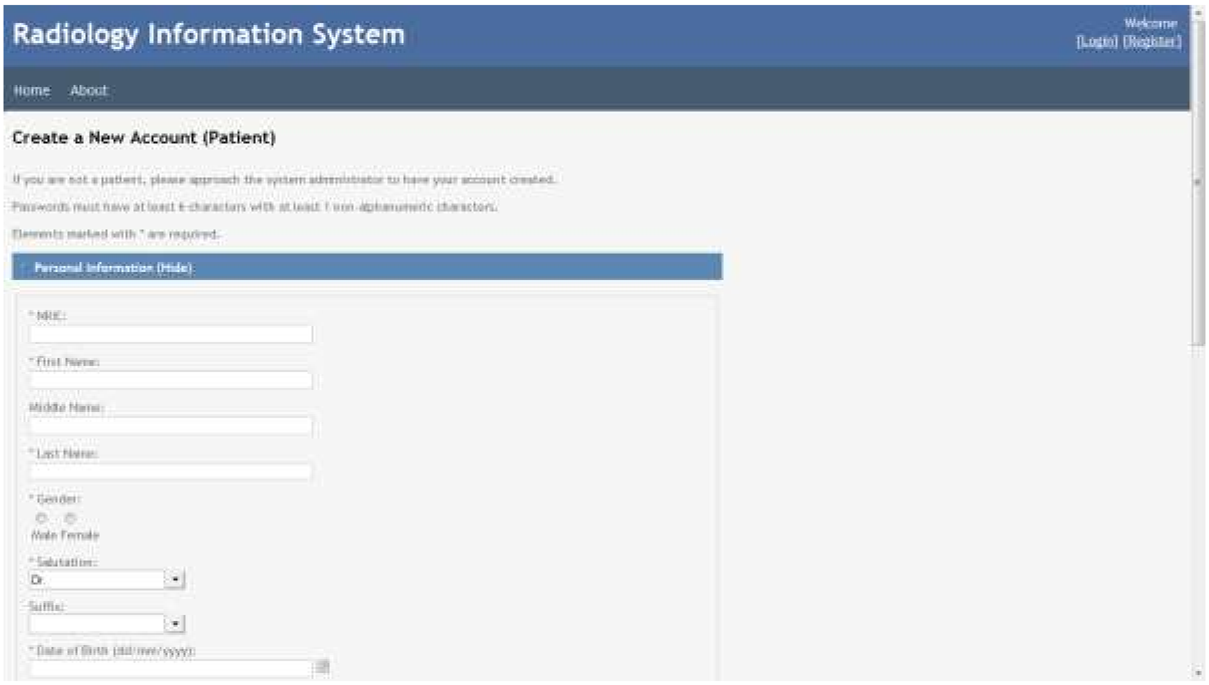
Password:

☐ Keep me logged in

Log In

Forgot Your Password?

Figure 38 – Log In


Radiology Information System

Welcome
[Login] [Register]

Create a New Account (Patient)

If you are not a patient, please approach the system administrator to have your account created.

Passwords must have at least 6 characters with at least 1 non-alphanumeric character.

Elements marked with * are required.

Personal Information (Hide):

* NRIC:

* First Name:

Middle Name:

* Last Name:

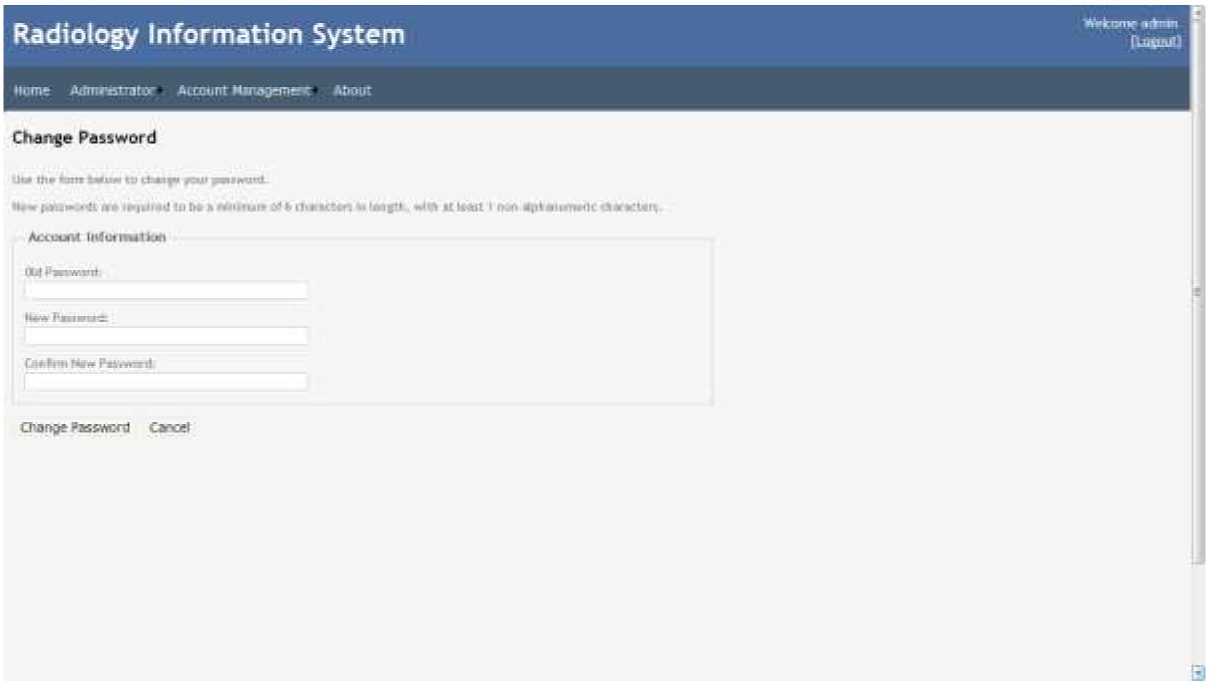
* Gender:
Male Female

* Salutation:
Dr.

Suffix:

* Date of Birth (dd/mm/yyyy):

Figure 39 – Create New Account (Patient)


Radiology Information System

Welcome admin
[Logout]

Change Password

Use the form below to change your password.

New passwords are required to be a minimum of 6 characters in length, with at least 1 non-alphanumeric character.

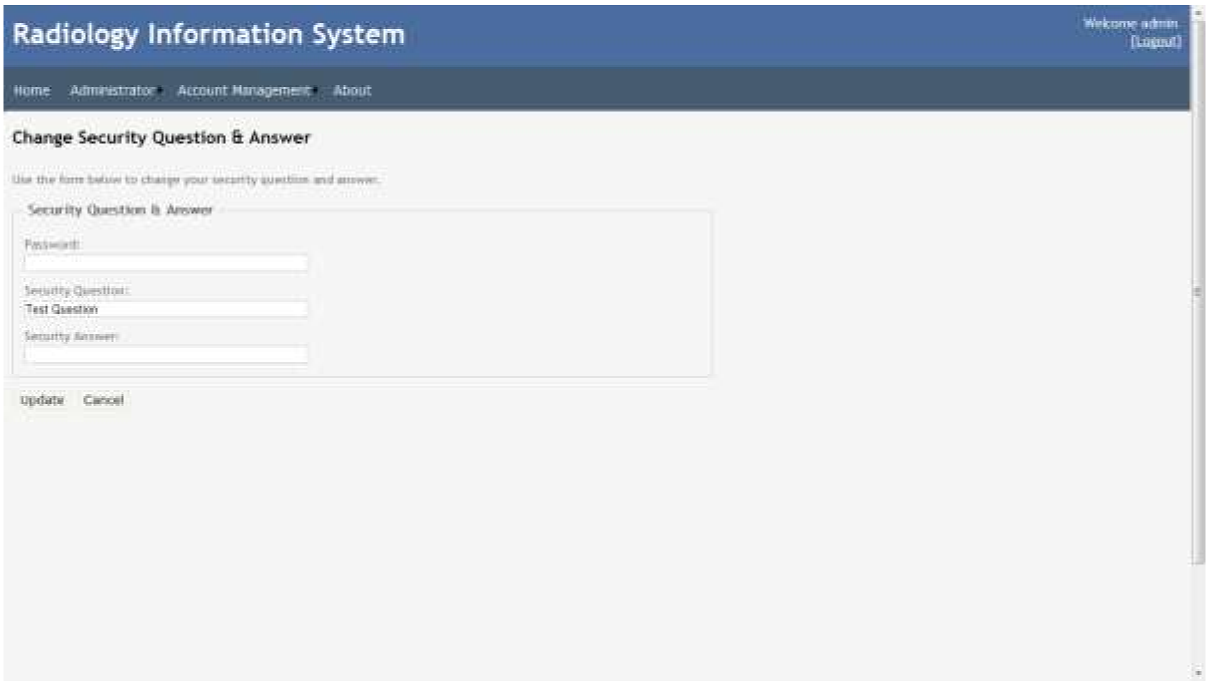
Account Information

Old Password:

New Password:

Confirm New Password:

Figure 40 – Change Password



Change Security Question & Answer

Use the form below to change your security question and answer.

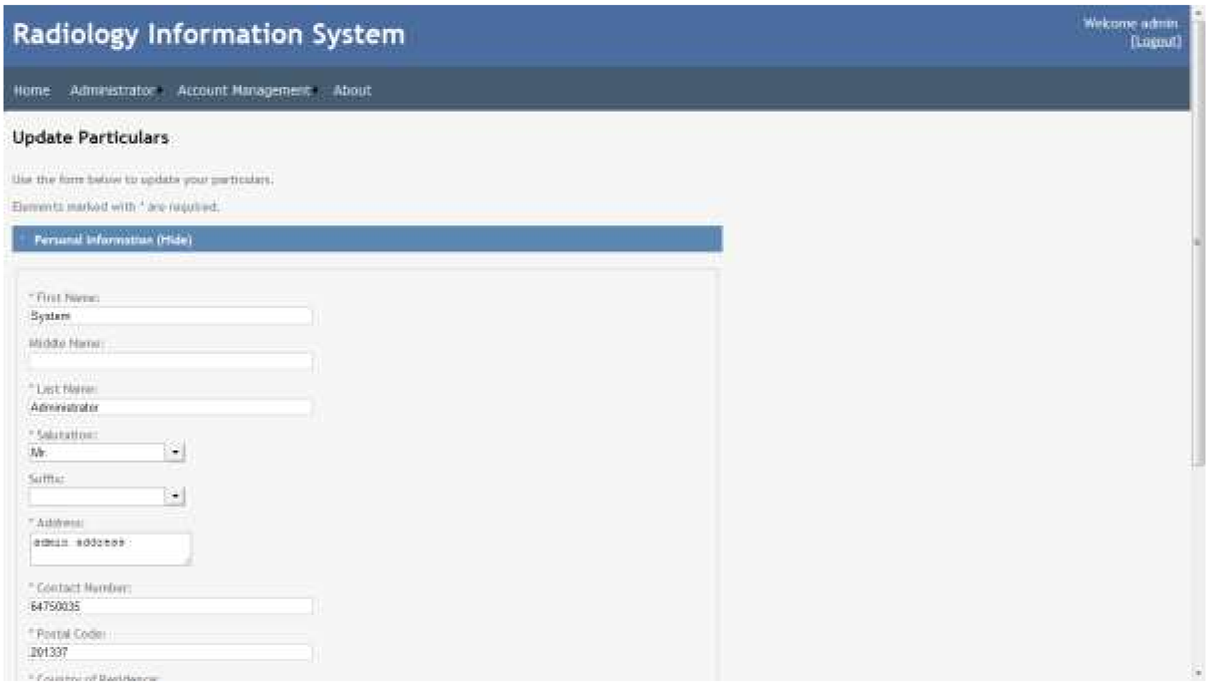
Security Question & Answer

Password:

Security Question: Test Question

Security Answer:

Figure 41 – Change Reset Question



Update Particulars

Use the form below to update your particulars.

Elements marked with * are required.

Personal Information (Hide)

* First Name: System

Middle Name:

* Last Name: Administrator

* Salutation: Mr

Suffix:

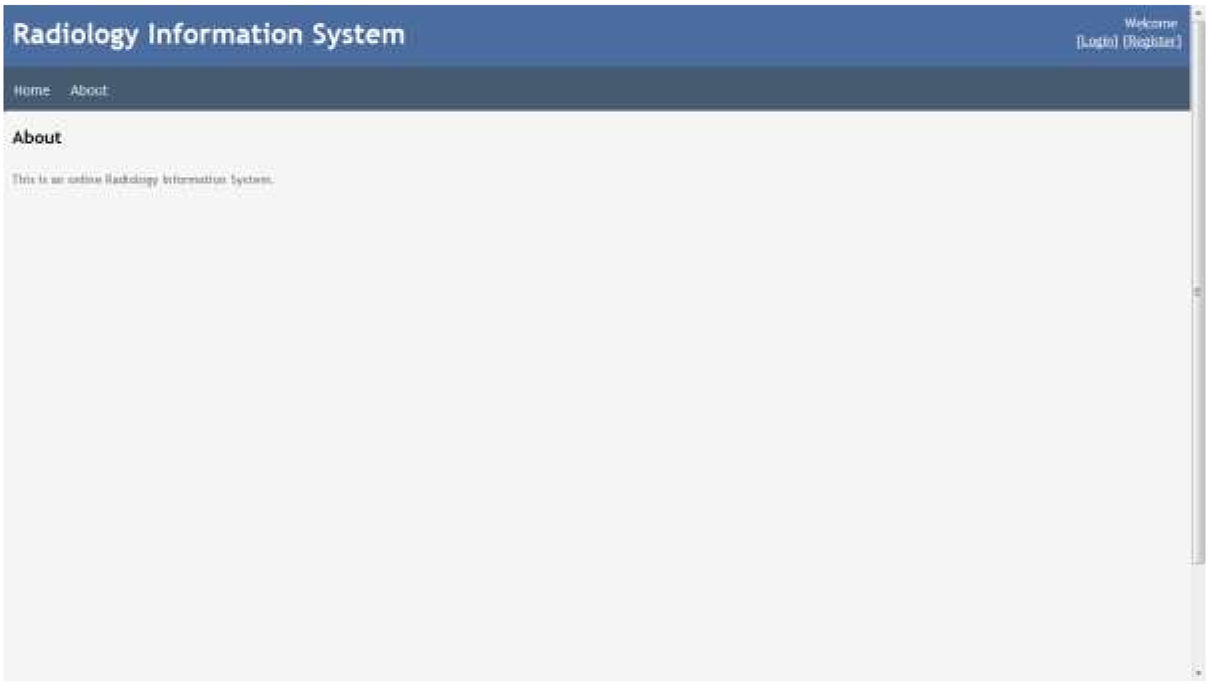
* Address: admin address

* Contact Number: 64750035

* Postal Code: 201337

* Country of Residence:

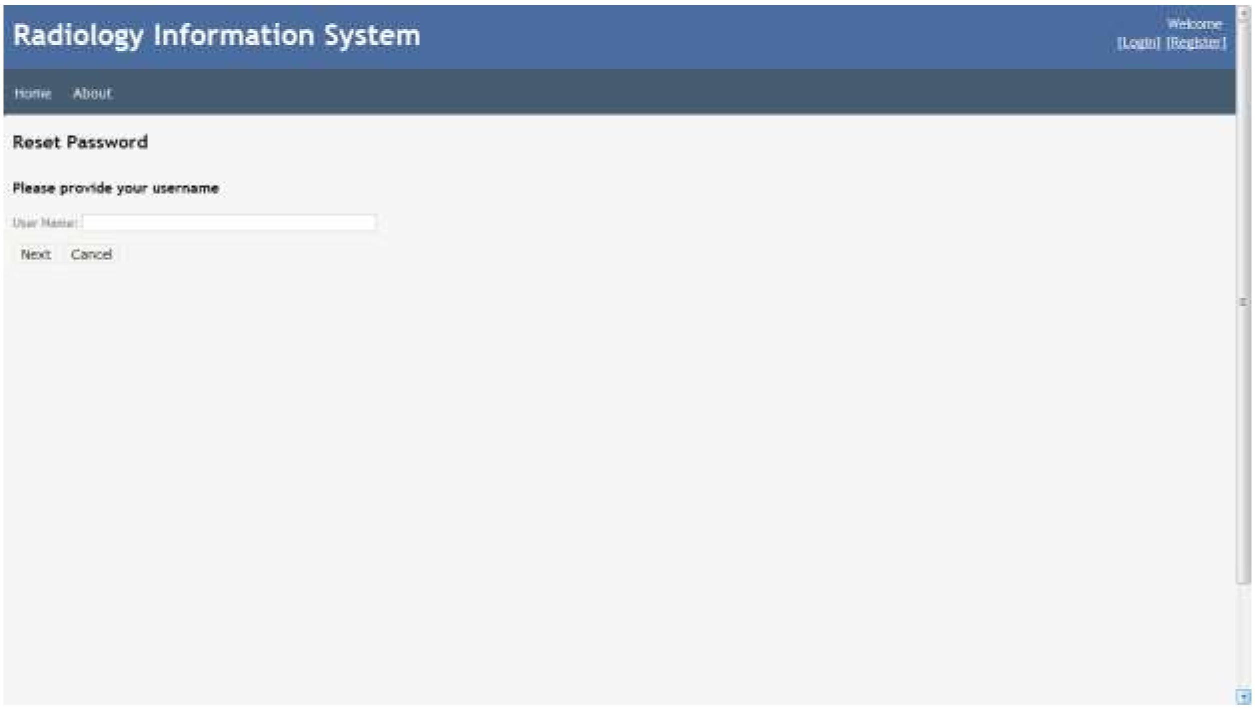
Figure 42 – Update Particulars



About

This is an online Radiology Information System.

Figure 43 – About

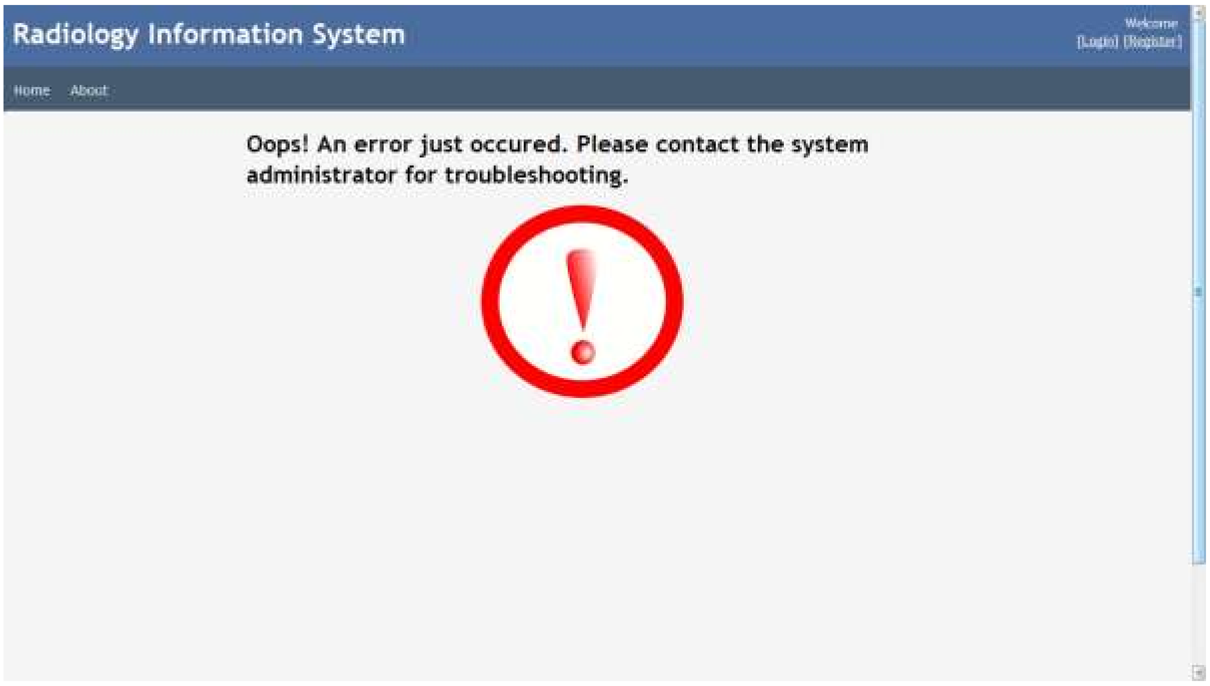


Reset Password

Please provide your username

User Name:

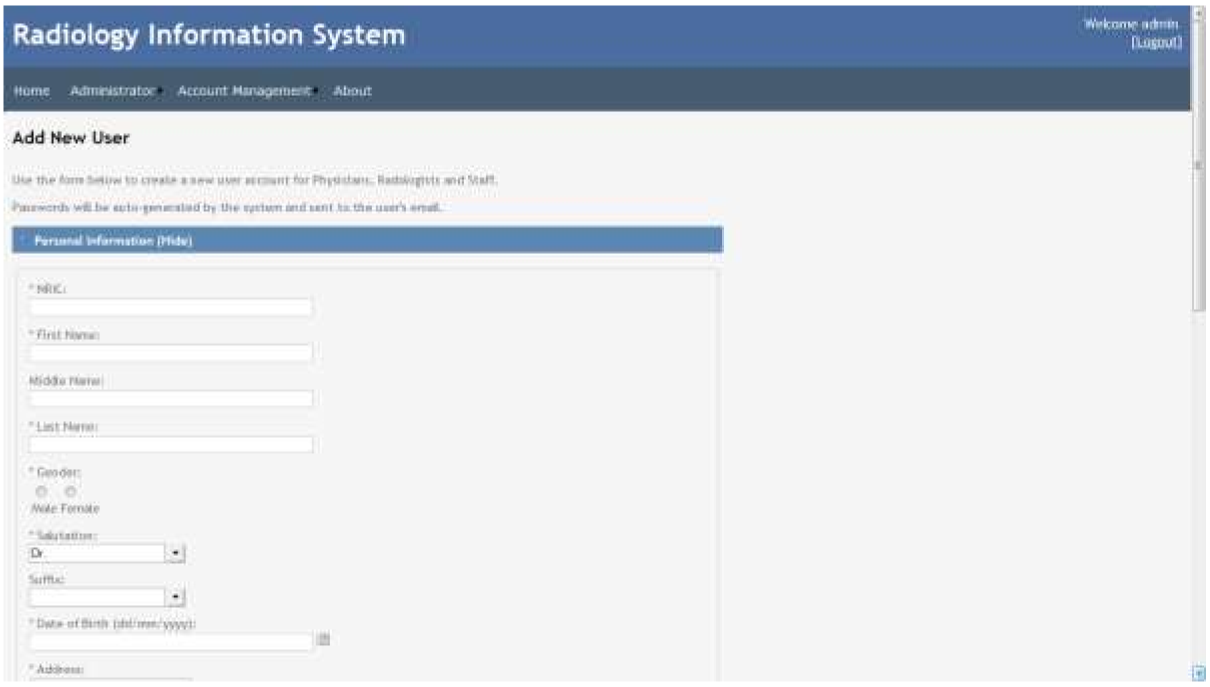
Figure 44 – Reset Password



Oops! An error just occured. Please contact the system administrator for troubleshooting.

Figure 45 – Error

5.2 Admin



Welcome admin
[Logout]

Home Administrator Account Management About

Add New User

Use the form below to create a new user account for Physicians, Radiologists and Staff.

Passwords will be auto-generated by the system and sent to the user's email.

Personal Information (Hide)

* NRIC:

* First Name:

Middle Name:

* Last Name:

* Gender:
Male Female

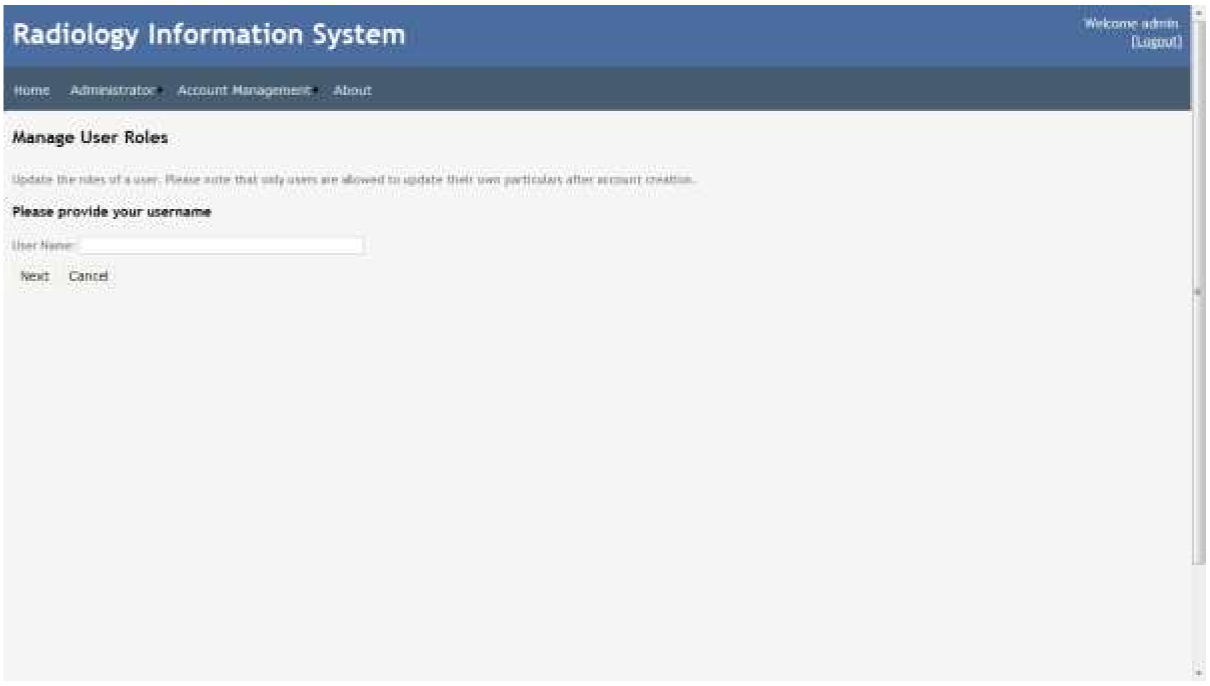
* Salutation:
Dr

Suffix:

* Date of Birth (dd/mm/yyyy):

* Address:

Figure 46 – Add User



Welcome admin
[Logout]

Home Administrator Account Management About

Manage User Roles

Update the roles of a user. Please note that only users are allowed to update their own particulars after account creation.

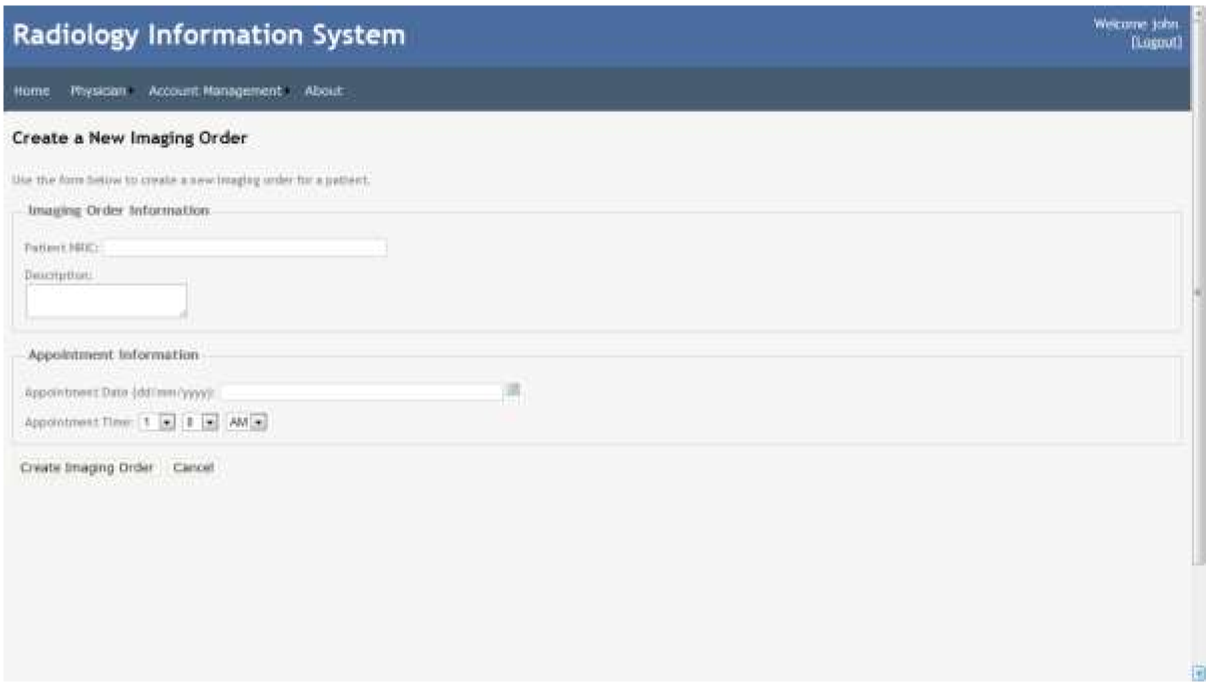
Please provide your username

User Name:

Next Cancel

Figure 47 – Manage User Roles

5.3 Physician



Create a New Imaging Order

Use the form below to create a new imaging order for a patient.

Imaging Order Information

Patient NRIC:

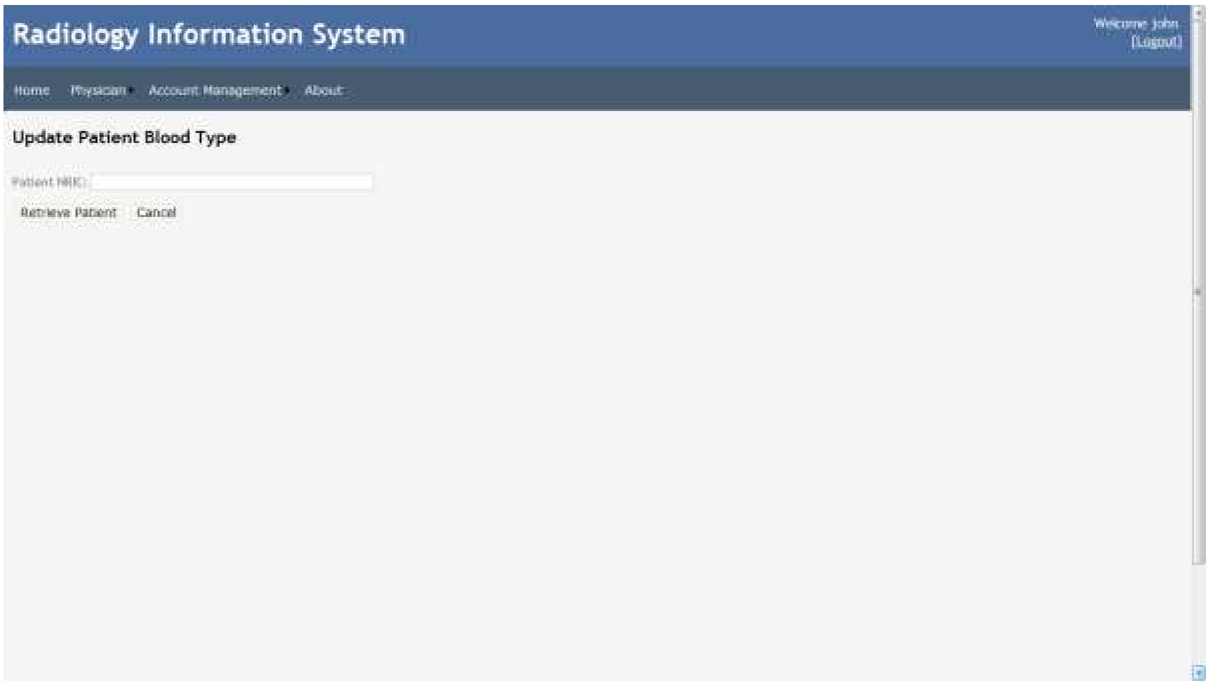
Description:

Appointment Information

Appointment Date (dd/mm/yyyy):

Appointment Time: 1 8 AM

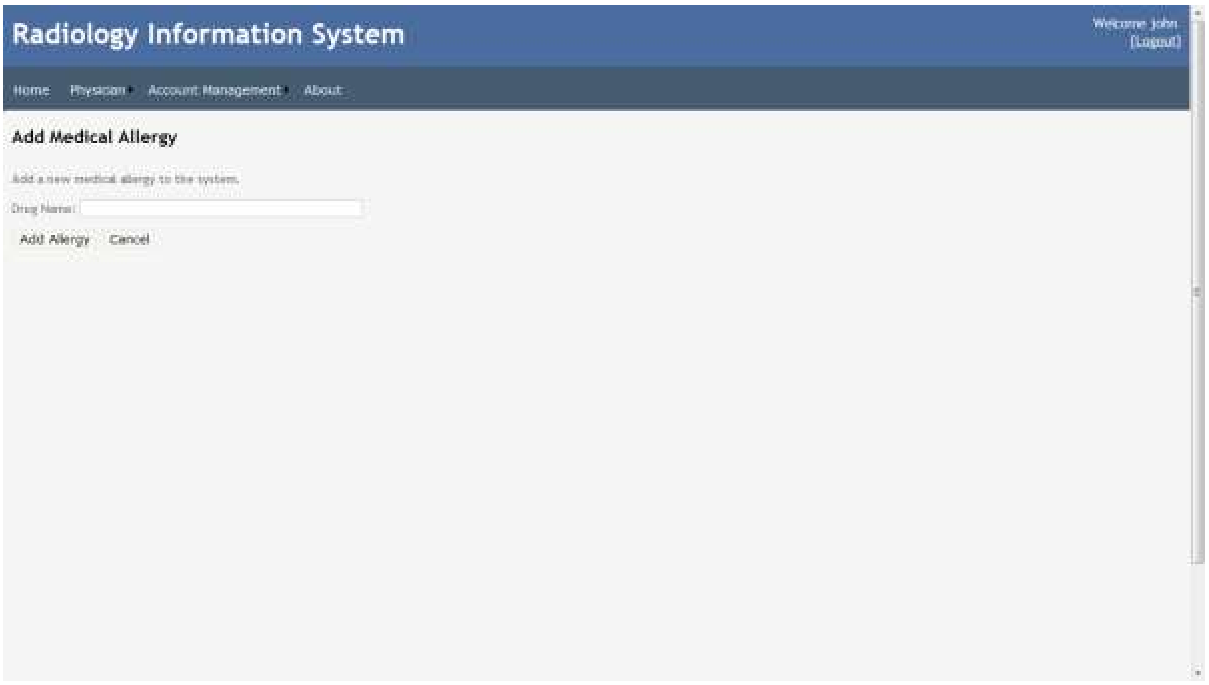
Figure 48 – New Imaging Order



Update Patient Blood Type

Patient NRIC:

Figure 49 – Update Patient Blood Type



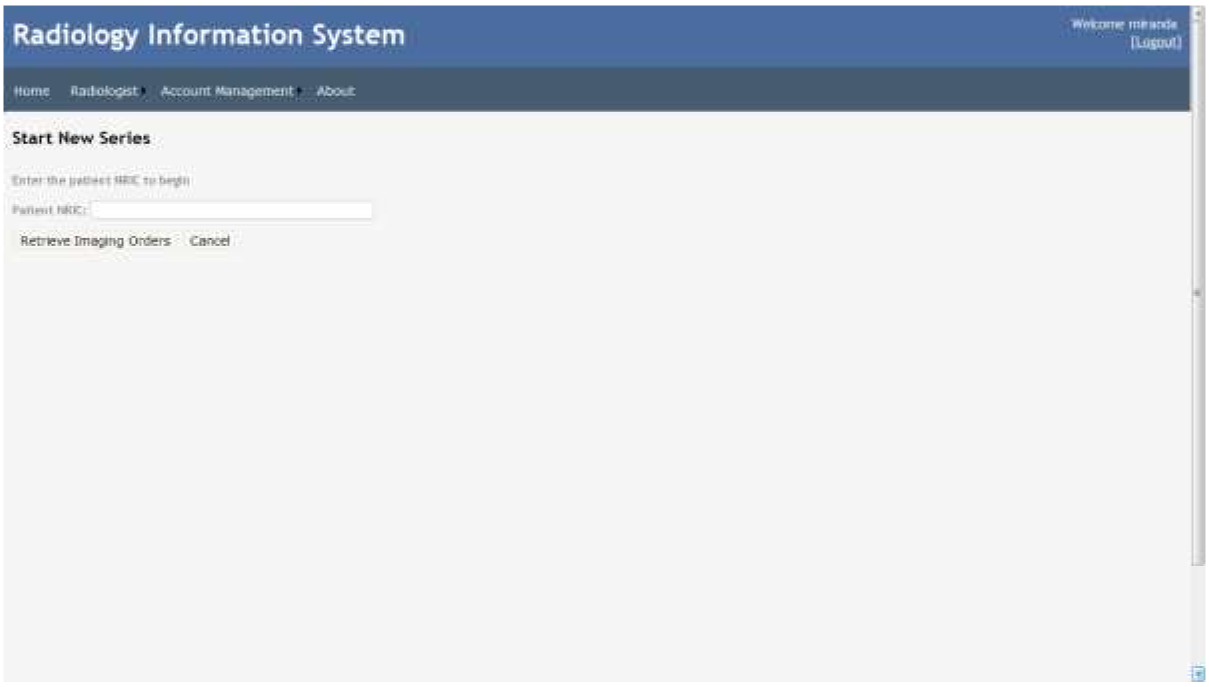
Add Medical Allergy

Add a new medical allergy to the system.

Drug Name:

Figure 50 – Add Medical Allergy

5.4 Radiologist

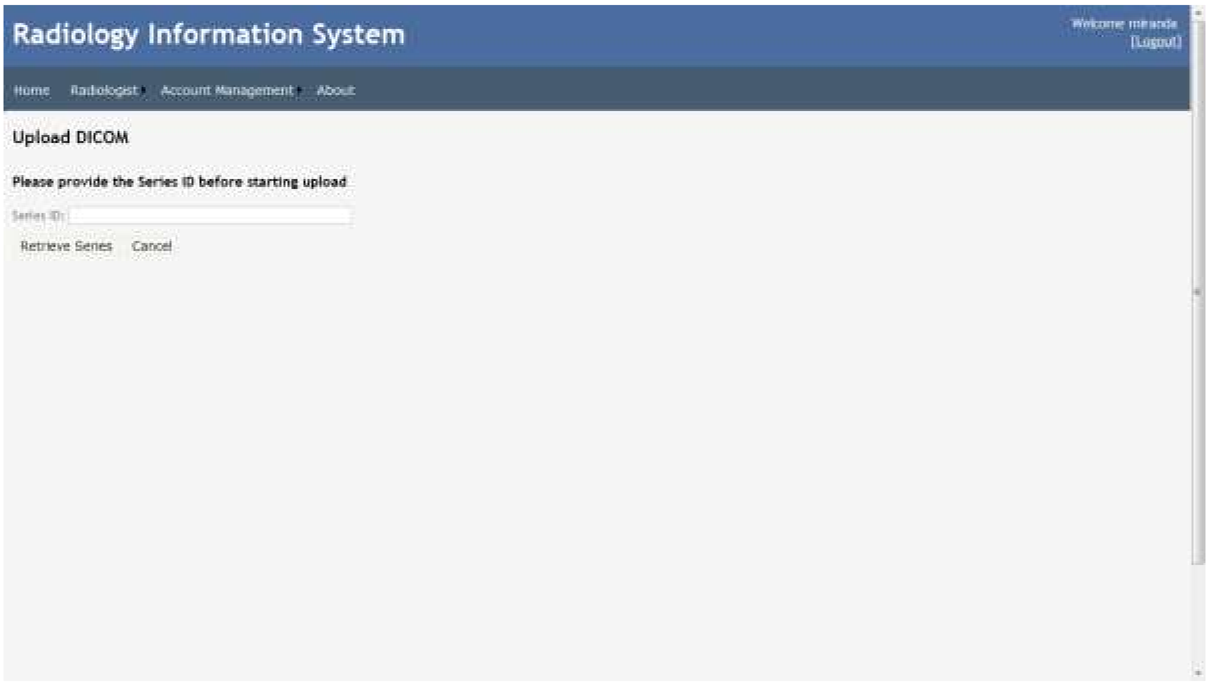


Start New Series

Enter the patient NRIC to begin

Patient NRIC:

Figure 51 – Start New Series



Upload DICOM

Please provide the Series ID before starting upload

Series ID:

Figure 52 – Upload DICOM

6 Discussion and Conclusion

6.1 Summary

This report covered several topics at length. First, it provided background information on telemedicine and the inspiration for developing a web based Radiology Information System (RIS). Second, the author presented his research findings on the software tools, standards and potential frameworks that will be utilized by the application. The architectural design of the application was then presented before drilling into the specifics.

As the application was being developed, the complexity of the DICOM standard persisted as a problem, eventually forcing the author to convert DICOM files uploaded to the system in order to circumvent it.

6.2 Discussion

RIS, like most web applications, does not require any complex "roll out" procedure to deploy; a compatible web browser is all that is needed. The low barrier to entry will not doubt help to drive adoption rates, further increasing usage of telemedicine in Singapore as a whole.

6.3 Future Work

Like every software endeavor, there are several ways in which RIS can be improved on.

The first way is to incorporate AJAX functionality into more pages. AJAX is a development technique used on the client-side to offer a highly interactive user interface whose responsiveness rivals that of desktop applications. This also helps to cut down on the amount of data that needs to be exchanged between the client and server as only a minimum amount of data is used.

The second way is to migrate it to an ASP.NET MVC 3 Web Application project. Based on ASP.NET, the migration will allow us to redesign RIS as a composition of three roles: Model, View and Controller. The other big reason for doing so is that a new HTML helper, WebImage, would be available for use. This greatly simplifies the need to parse byte data arrays into the Response object in order to send imaging data to the client.

The final way is to utilize caching. Instead of having to query the database everytime a page is requested or posted back, the server can cache commonly accessed data to reduce processing times. This will be especially useful if image data is cached, as it is not expected to change once created.

References

[1] D. Evans, “A Very Basic DICOM Introduction,” dcm4che.org, 21 August 2006. [Online]. Available: http://www.dcm4che.org/confluence/display/d2/A+Very+Basic+DICOM+Introduction. [Accessed 16 October 2011].

[2] C. Rorden, "The DICOM Standard," [Online]. Available: http://www.cabiatl.com/mricro/dicom/index.html. [Accessed 10 December 2011].

Appendix

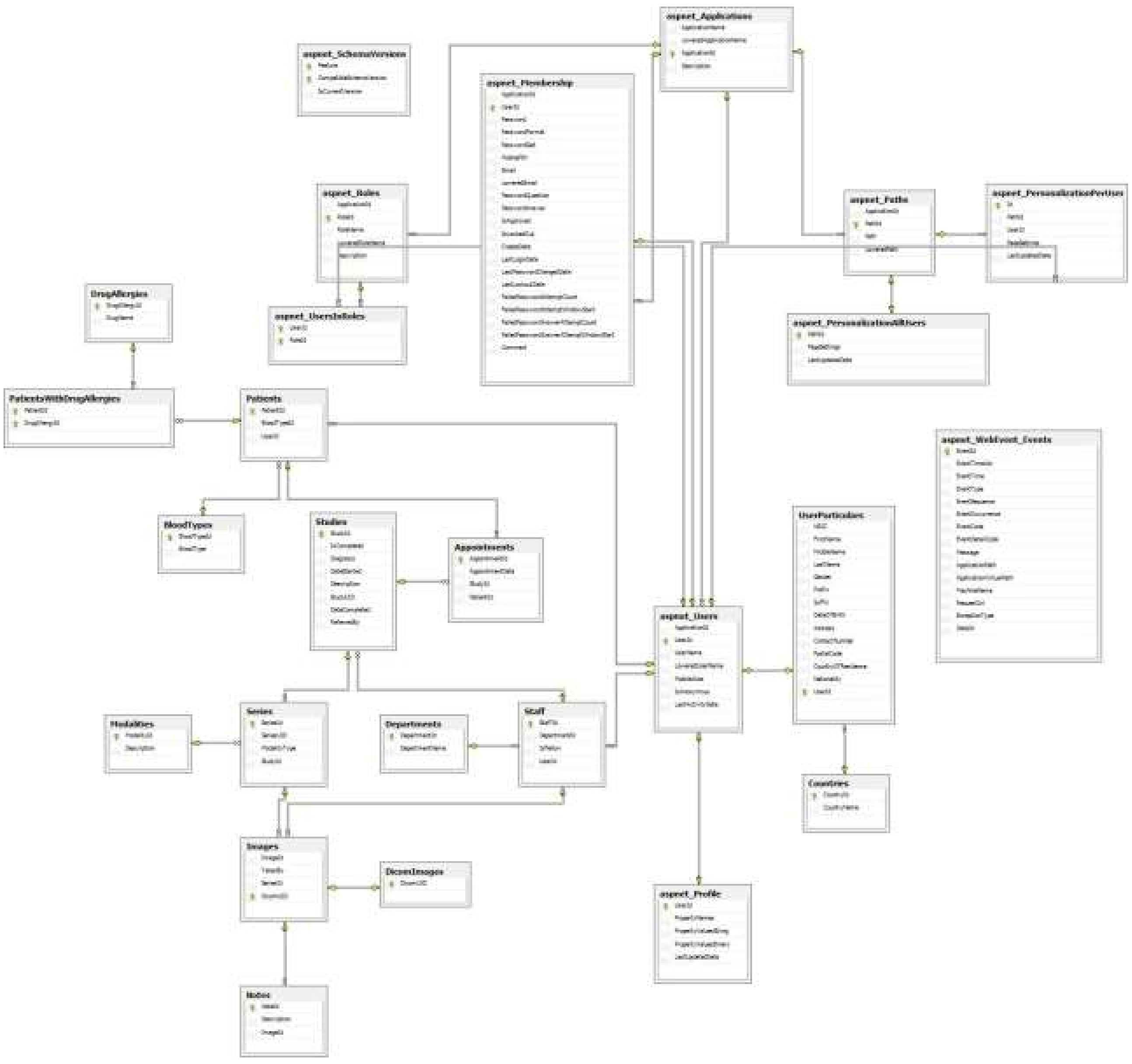


Figure 53 - Database Schema