FYP DOCUMENT

SEMANTIC ENABLED CLOUD BASED EMR

ADVISOR:

Dr. Khalid Latif

CO-ADVISORS:

Dr. Hafiz Farooq Ahmed, Mr. Maqbool Hussain

TEAM MEMBERS:

Hassan Ahmed Usmani	**2007-NUST-BIT-113**
Mubashar Hussain	**2007-NUST-BIT-123**
Rashid Ali	**2007-NUST-BIT-138**
Waqas Ashraf	**2007-NUST-BIT-142**

DATED:

20-Jun-2011

DEDICATION

To Allah the Almighty & To my Parents and Faculty

CERTIFICATE OF APPROVAL

It is certified that the contents and form of thesis entitled "Semantic enabled Cloud based EMR" submitted by Hassan Ahmed Usmani (2007-NUST-BIT-113), Mubashar Hussain (2007-NUST-BIT-123), Rashid Ali (2007-NUST-BIT-138) and Waqas Ashraf (2007-NUST-BIT-142) have been found satisfactory for the requirement of the degree.

Advisor:	Dr. Khalid Latif	**Signature:** _____
Co-Advisor:	Dr. Hafiz Farooq	**Signature:** _____
Co-Advisor:	Mr. Maqbool Hussain	**Signature:** _____

ABSTRACT

The core focus area of a hospital is the healthcare services that they provide to a patient. Managing a complex hospital management system by the hospital is a daunting task and this makes their focus diverted from healthcare services and into managing a server, a network and software. If the hospital information system is based on a cloud computing then all the management efforts are taken care of by the vendor and the hospital can again focus in their core focus area.

EMR is usually a computerized legal medical record created in an organization that delivers care, such as a hospital and doctor's surgery and it allows storage, retrieval and manipulation of records. It is not a paper record made electronic. Growing trend of cloud computing reaches into EMR and allows all users to use the same software that is dynamically scalable.

The project semantic enabled Cloud based EMR: Ambulatory module is a part of a bigger project being developed in HLH project. Health level Horizon (HLH) is a project of Semantic Research Lab, SEECS-NUST in which research is being carried out on interoperability of Healthcare information systems. In this project HL7 middleware software is also being developed with the funding of ICTR&D fund. Primary focus of this project is on HL7 version 3.0 Standard.

Our proposed solution is cloud based and semantically healthy. It must be noted that each module in this system is a service and semantically sound. The prominent data exchange module should be given extra concentration as data on cloud should be compatible and accessible to all other EMR systems. We built an ambulatory module that will produce and maintain an EMR of patients whose medical condition requires hospitalization in emergency situation.

In the developed system, following benefits are observed:

- EMR web services that will reside on cloud.
- There is no need of full fledge team for EMR software maintained because the solution will reside on cloud and it is accessible via web services.
- Due to our cloud infrastructure based solution, cost is greatly reduced and capital expenditure is converted to operational expenditure.
- Device and location independence enable users to access systems using a web browser regardless of their location or what device they are using.
- As we follow the functional criteria of HL7 model so we will gain the interoperability feature so that we can share data among different EMR systems.

ACKNOWLEDGMENTS

All praises to Almighty Allah who gave us the power of thinking and reasoning. His blessings and mercy are the real source of all human accomplishments.

We are highly indebted to (Dr. Khalid Latif our advisor and Dr. Hafiz Farooq & Mr. Maqbool Hussain our co-advisors) for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express our gratitude towards our families for their kind co-operation, support and encouragement which helped us in completion of this project.

Our thanks and appreciations also go to our colleagues who have willingly helped and supported us with their abilities in weak moments.

Table of Contents

Table of Figures

Abbreviations

Acronym	Definition
AJAX	Asynchronous JavaScript & XML
API	Application Programming Interface
CCHIT	Certification Commission for Health Information Technology
DEO	Data Entry Operator
DTO	Data Transmission Object
EHR	Electronic Health Record
EMR	Electronic Medical Record
ERD	Entity Relationship Diagram
GWT	Google Web Toolkit
HL7	Health Level Seven International
HLH	Health level Horizon
NS	Notification Service
NU	Nurse
OWL	Web Ontology Language
PH	Physician
RDF	Resource Description Framework
RDQL	RDF Data Query Language
RE	Receptionist
REST	Representational State Transfer
SA	System Administrator
SOAP	Simple Object Access Protocol
SPARQL	SPARQL Protocol and RDF Query Language
UDDI	Universal Description, Discovery and Integration
WSDL	Web Services Description Language

1. INTRODUCTION *Chapter 01*

1.1. Overview

The project Cloud based EMR: Ambulatory module is a part of a bigger project being developed in HLH project. Health level Horizon (HLH) is a project of Semantic Research Lab, SEECS-NUST in which research is being carried out on interoperability of Healthcare information systems. In this project HL7 middleware software is also being developed funded by ICT R&D. Primary focus of this project is on HL7 version 3.0 Standard.

Health Level Seven (HL7) is a non-profit organization involved in development of international healthcare standards. There standards are mostly related to clinical and administration aspect of a healthcare information system. This standard allows interoperability among HITs of different organization or different vendors.

An EMR is not a paper record made electronic. An electronic medical record (EMR) is a computerized medical record created in an organization that delivers care, such as a hospital where as ambulatory care is the care of patients whose condition requires hospitalization in emergency situation. [8]

We will build an ambulatory module that will produce and maintain an EMR of patients admitted in a hospital. In the project Cloud based EMR: Ambulatory module, cloud computing technology is being used. Cloud computing is Internet-based computing, whereby shared resources, software, and information are provided to computers and other devices on demand, like the electricity grid.

Our ambulatory module follows the Ambulatory EHR 2011 Certification criteria set by CCHIT. The Certification Commission for Health Information Technology (CCHIT) is a private not-for-profit organization that will apply to be an ONC-Authorized Testing and Certification Body (ONCATCB) of electronic health records (EHR). In this project Cloud based EMR: Ambulatory module, not all of the

certification criteria set by CCCHIT will be implemented but initially we will focus on some of the core system functionalities.

1.2. Problem Background

Traditional paper-based patient records require a significant amount of storage space compared to digital records. In US, most states require physical records be held for a minimum of seven years. The costs of storage media, such as paper and film, per unit of information differ dramatically from that of electronic storage media. Patients very rarely see one doctor anymore, but instead opt for a variety of specialists about a particular problem. This poses a problem when doctors have to transfer paper files among themselves.

In order to automate workflows of healthcare system for patient care, many initiatives are launched for developing open source solutions. A full fledge IT-Staff is required all the time to configure, deploy and maintain the system. As compared to existing traditional healthcare systems our solution is highly cost deficient. There is no need of paper, huge IT-staff and heavy servers to maintain. The existing solutions do not follow any standard of interoperability so there is no data sharing facility among different EMR systems. But in our system, physicians can easily share patient data among themselves through electronic media. Patients do not need to be in the hospital for making appointments with doctors because our system facilitates them to make appointments from anywhere by using their mobiles. Patients will be reminded for their appointments accordingly. Patients do not need to keep their prescription with them when they go to some medical store; they will just provide their Medical Record Number (MRN), generated by our system for every patient, and chemist can view all the required data to give dose of medicines.

1.3. Objective of the Project

Our motive is to provide an internet-based, HL7 & CCHIT compliant and open source solution designed over multi-tiered architecture which will be interoperable, reliable and scalable.

1.4. Benefits

This project will help healthcare systems to automate their workflows in order to deliver patient care. As we are following international standards due to which communication between healthcare institutions would be very easy and reliable. It also accompanies modern technologies which can help to grow this product exponentially.

2. LITERATURE REVIEW *Chapter 02*

Currently, only 25% of US physicians are utilizing systems that facilitate electronic health records, according to an October report by the Robert Wood Johnson Foundation. The report, which reviewed data from three dozen surveys conducted in the past decade, found that not enough physicians are opting to use electronic medical records, despite their benefits. In fact, fewer than 10 percent of the nation's physicians are using fully operational electronic records that collect patient information, display test results, allow providers to enter medical orders and prescriptions and aid in treatment.

Health leaders are calling for increased use of electronic records because they are more accurate cut down on medical errors and can save money. All existing EMRs are variants in terms of Prices ($1000 to $50,000+/physician) and Features.

Few popular existing EMRs are discussed below:

2.1. Open EMR

OpenEMR is a free medical practice management, electronic medical records, prescription writing, and medical billing application. These programs are also referred to as electronic health records. OpenEMR is licensed under the General Gnu Public License (General GPL). It is a free open source replacement for medical applications such as Medical Manager, Health Pro, and Misys. [2]

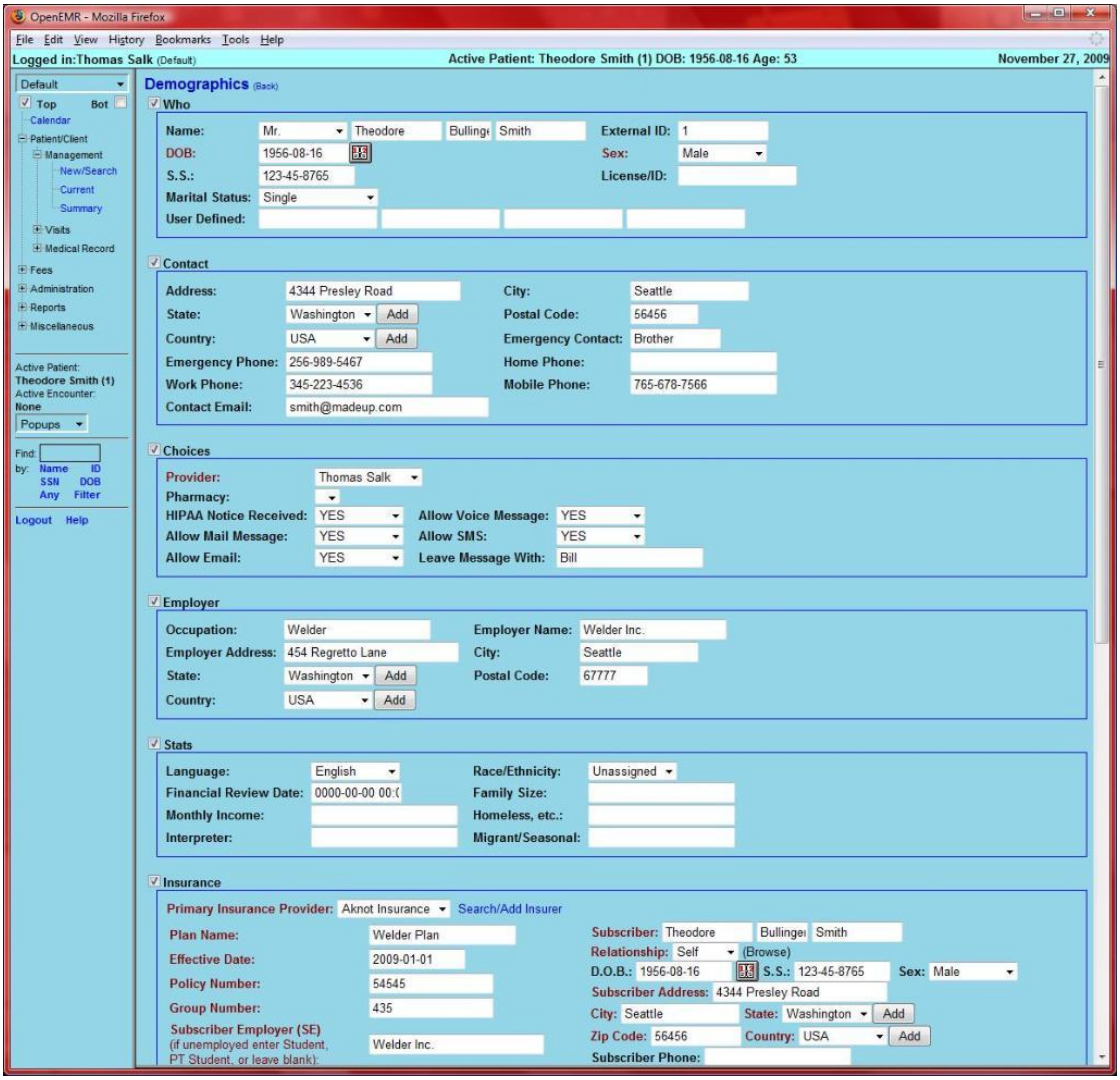
OpenEMR is one of the most popular free electronic medical records in use today. SourceForge has recorded over 3,400 downloads per month. This is the equivalent in fair market value of $100,000,000 in donated software a month.

- OpenEMR online demo can be visited at [16]

- OpenEMR Virtual Appliance can be visited at [15]

Below is screenshot from OpenEMR for patient demographics:

Figure 2-1: Patient Demographics [18]



Pros:

- Free
- Open Source
- Multi-language Support
- Free Upgrades
- Free online support

- Electronic Billing
- Document management
- Integrated practice management
- E-Prescription
- Insurance tracking (3 insurances)
- Easy to customize
- Easy Installation
- Voice recognition ready (MS Windows Operating Systems)
- Web based (Secure access with SSL certificates)
- Integration with external general accounting program SQL-Ledger
- Built in Scheduler
- Multi-facility capable
- Prescriptions by printed script, fax or email
- Any language can be added and translated in a collaborative Google Docs spreadsheet maintained by the OpenEMR community

Cons:

- Highly coupled
    - MVC Model not followed
    - Not extensible
- Not user-friendly

2.2. Child Health Improvements through Computer Automation (CHICATM) System

It is a computer-based decision support system that combines these elements:

I. Pediatric preventive care guidelines encoded in Arden Syntax
II. A dynamic, scannable paper user interface
III. HL7-compliant interface to existing electronic medical record systems

The result is a system that both delivers "just in time" patient-relevant guidelines to physicians during the clinical encounter and accurately captures structured data from all who interacts with the system. The system performs these tasks while remaining sensitive to the workflow constraints of a busy outpatient pediatric practice. [20]

We have developed this system as an extension of the Regenstrief Medical Record System (RMRS), an inpatient and outpatient information system which contains 30 years of data and more than 300 million numeric or coded patient observations [14]. In this setting we are able to rapidly deploy the system for evaluation, and expect that our use of industry standards will allow this system to be easily adapted to other settings as well.

CHICA has client-server architecture. The client displays a "grease board" with a list of the patients to be seen, their medical record numbers, and their appointment status during the visit. The client also drives the printer and scanner. The server is designed to service multiple clients and it consists of a data store, a rule processor for Arden Syntax MLMs and a HL7 listener and parser.

This is a guideline-based decision support system that could seamlessly integrate into the delicate workflow of a high volume pediatric clinic. There are six essential criteria to be considered:

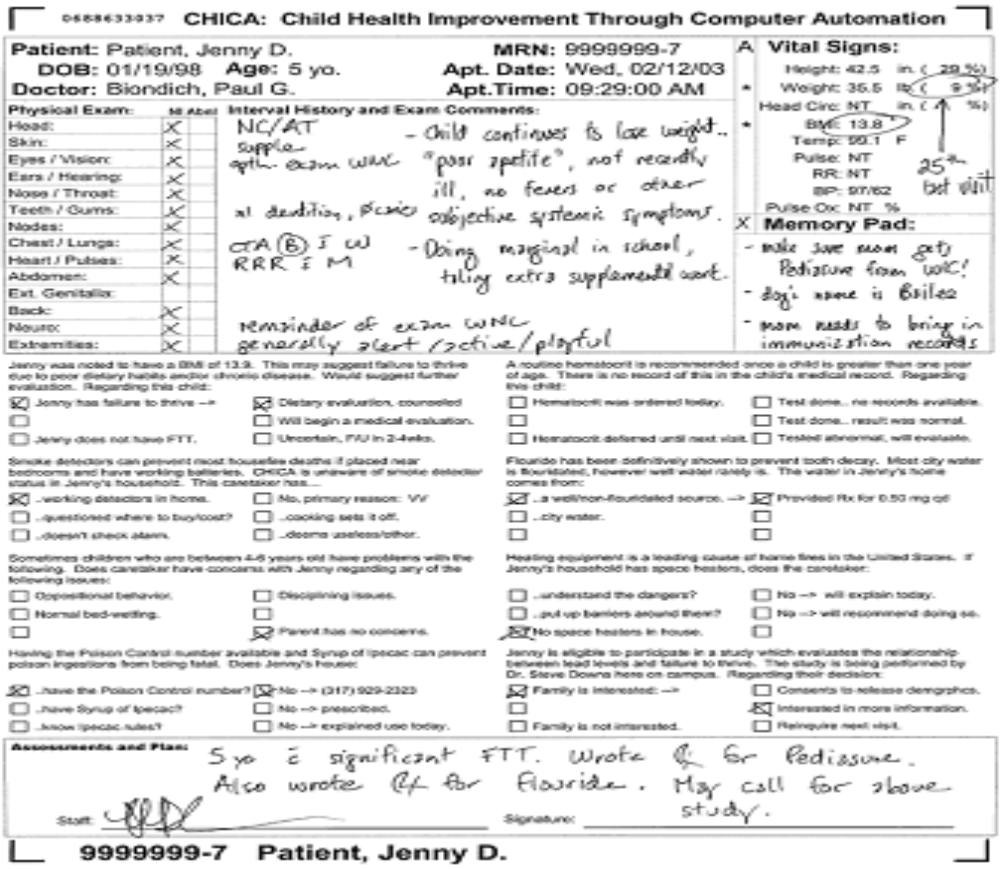
I. Collecting data directly from patients or their parents
II. Providing reminders to nurses about age- appropriate screening data
III. Prioritizing needed preventive services
IV. Providing tailored prompts and reminders to physicians unobtrusively during the encounters with patients
V. Capturing data directly from physicians
VI. Requiring little or no training of staff

Preliminary work by one of its investigators has demonstrated the feasibility of using tailored scannable paper forms to provide patient specific reminders to physicians and capture data through optical scanning. They expanded this model, using advances in

Optical Character Recognition (OCR) technology and international standards for knowledge representation (Arden Syntax) and data communication (HL7).

The software interprets the scanned data and writes all newly-recorded observations into the EMR. Below is The Physician Worksheet (PWS)

Figure 2-2: The Physician Worksheet (PWS) [20]



Pros:

- HL7 compliant
- Decision Support System

Cons:

- Paper is used to interact with an electronic data repository.
- Standardized paper forms are printed and then "scanned".

- Character-recognition

2.3. OpenMRS

OpenMRS is a community-developed, open source, enterprise electronic medical record system platform. OpenMRS is a flexible electronic medical record platform that can be customized to the needs of clinical and research organizations. It's robust enough for a nationwide health care system, and nimble enough for a field-based clinic. And best of all, it is open source, which means you can build on what we've done to realize new ideas and create the ideal software for your needs.

OpenMRS is based on the principle that information should be stored in a way which makes it easy to summarize and analyze, i.e., minimal use of free text and maximum use of coded information

OpenMRS is a software platform and a reference application which enables design of a customized medical records system with no programming knowledge (although medical and systems analysis knowledge is required). It is a common platform upon which medical informatics efforts in developing countries can be built. The system is based on a conceptual database structure which is not dependent on the actual types of medical information required to be collected or on particular data collection forms and so can be customized for different uses.

Pros:

- **Central concept dictionary:** Definitions of all data (both questions and answers) are defined in a centralized dictionary, allowing for robust, coded data
- **Security:** User authentication
- **Privilege-based access:** User roles and permission system
- **Patient repository:** Creation and maintenance of patient data, including demographics, clinical observations, encounter data, orders, etc.

- **Multiple identifiers per patient:** A single patient may have multiple medical record numbers
- **Data entry:** With the FormEntry module, clients with InfoPath (included in Microsoft Office 2003 and later) can design and enter data using flexible, electronic forms. With the HTML FormEntry module, forms can be created with customized HTML and run directly within the web application.
- **Data export:** Data can be exported into a spreadsheet format for use in other tools (Excel, Access, etc.)
- **Standards support:** HL7 engine for data import
- **Modular architecture:** An OpenMRS Module can extend and add any type of functionality to the existing API and web application.
- **Patient workflows:** An embedded patient workflow service allows patient to be put into programs (studies, treatment programs, etc.) and tracked through various states.
- **Cohort management:** The cohort builder allows you to create groups of patients for data exports, reporting, etc.
- **Relationships:** Relationships between any two people (patients, relatives, caretakers, etc.)
- **Patient merging:** Merging duplicate patients
- **Localization / internationalization:** Multiple language support and the possibility to extend to other languages with full UTF-8 support.
- **Support for complex data:** Radiology images, sound files, etc. can be stored as “complex” observations
- **Reporting tools:** Flexible reporting tools
- **Person attributes:** The attributes of a person can be extended to meet local needs

Cons:

- No support of billing and appointment scheduling.
- Data model not suitable for large number of clients but it can analyze the data of limited number of clients very well.

3. BACKGROUND *Chapter 03*

This chapter covers the background of the methods and concepts used in the project. The background will cover what sort of related work has been done regarding the field. We will be looking on three main things that are:

I. Semantic Web
II. GWT Framework
III. Restful web services

3.1. Semantic Web

Semantics is the systematic study of meaning. The term "Semantic Web" is often used more specifically to refer to the formats and technologies that enable it. These technologies include the Resource Description Framework (RDF), a variety of data interchange formats (e.g. RDF/XML, N3, Turtle, N-Triples), and notations such as RDF Schema (RDFS) and the Web Ontology Language (OWL), all of which are intended to provide a formal description of concepts, terms, and relationships within a given knowledge domain.

The new recommendations for the Resource Description Framework (RDF) and the Web Ontology Language (OWL) have just been published. They provide a simple triple-based representation of knowledge, with formal semantics allowing for automated inference. RDFS and OWL also provide some useful vocabulary, particularly for building schema and ontologies.

In different EMR systems, they are facing challenges that there can be many meaning for the term ‘Doctor’. Doctor can be called as a Physician, a Healthcare provider or a Doctor. Semantics help our system to perceive the same meaning (doctor) for all different names. This keeps our system clean and enables us to share data among different EMR system easily without any modification of data.

.

3.1.1. What is Jena?

Jena is a leading Semantic Web toolkit for Java programmers. The heart of the Semantic Web recommendations is the RDF Graph as a universal data structure. An RDF graph is simply a set of triples (S, P, O), where P names a binary predicate over (S, O). Jena similarly has the Graph as its core interface around which the other components are built. OWL Full is a semantic extension of RDF; Jena's ontology support is targeted at OWL Full. Future Semantic Web standardization is likely to include work on query languages, and possibly Web APIs for the Semantic Web.

The main contribution of Jena is the rich Model API for manipulating RDF graphs. Around this API, Jena1 provided various tools, including I/O modules for: RDF/XML, N3, and N-triple; and the query language RDQL. Using the API the user can choose to store RDF graphs in memory or in persistent stores. Jena provided an additional API for manipulating DAML+OIL.

3.1.2. Jena Architecture Overview

The heart of the Jena architecture is the RDF graph, a set of triples of nodes. This layer, following the RDF abstract syntax, is minimal by design: wherever possible functionality is done in other layers. This permits a range of implementations of this layer such as in memory or persistence triple stores. The Graph layer is the extension point on which to build APIs: within Jena the functionality offered by the Graph layer is used to implement the Jena Model API and the new Ontology functionality for OWL and RDFS, upgrading the Jena DAML API. I/O is done in the Model layer, essentially for historical reasons. The Jena architecture supports fast path query that goes all the way through the layers from RDQL at the top right through to an SQL database at the bottom, allowing user queries to be optimized by the SQL query optimizer. We give some more detail on the three layers below.

3.1.3. The Graph Layer: Triples as the Universal Data Structure

The Graph layer is based on the RDF Abstract Syntax. It is straightforward to implement any of:

- Triple stores, both in memory and backed by persistent storage.
- Read-only views of non-triple data as triples, such as data read from a computer file system hierarchy, or scraped from a web page.
- Virtual triples corresponding to the results of inference processes over some further set of triples as premises.

Implementations of the Graph layer provided with Jena give a variety of concrete (materialized) triple stores, and built-in inference for RDFS and a subset of OWL.

3.1.4. The Model Layer: Views for Application Programmers

Jena maintains the Model API from Jena1 as the primary abstraction of the RDF graph used by the application programmer. This gives a much richer set of methods for operating on both the graph itself (the Model interface) and the nodes within the graph (the Resource interface and its subclasses). Further, the DAML API is updated and enhanced in Jena to form an Ontology API that can be realized as a DAML API or an OWL API.

3.1.5. The Graph Layer: Multiple Simultaneous Views

Both the Model and the Ontology layers lie on top of the Graph layer via an intermediate layer: the Graph layer. This provides an extension point for providing views of graphs, and views of nodes within a graph. This generalizes the needs of both the Model and the Ontology APIs, and, significantly, makes the design decision that such presentation layers must be stateless: all significant state is within the graph. (Caching of state is permitted by the presentation layers). The Graph layer is mutability – this is reflected through the use of Java exceptions and testable through a Capabilities interface. The most significant part of the core Graph interface is the find operation. The primitive find(Node S, Node P, Node O) delivers an iterator over all the triples of the Graph which "match" the triple (S, P, O). To "match" means to be equal to or for the S/P/O node to be ANY. This allows the Graph to be queried for e.g. all the properties of some particular subject, all the predicates with some particular

object, or indeed all the triples in the Graph. This is the extensibility point that the inference engines and Graph combiner use for generating virtualized triples.

3.1.6. Fast Path Query

One of the goals of the Graph layer is to allow queries to be expressed which can exploit underlying efficient query engines, and which can return different kinds of results-variable bindings or sub-graphs, for example. Rather than add many operations to Graph it, each Graph has an associated query handler which manages the more complex queries. A standard simple query handler is provided which implements the complex queries in terms of the find primitive for Graphs not offering more efficient possibilities.

A Query consists of a collection of triple patterns to be matched against some Graphs. A triple pattern is a Triple that may contain the extended ANY and Variable nodes mentioned above. So a Query might contain

(?x P ?y) (?y Q ?z)

To request all the bindings for ?x, ?y, and ?z for which matching triples can be found in the Graph. The query is executed so as to find all possible bindings of the variables; from this matched sub-graphs can be computed. Jena's memory-based Graph model simply implements the triple pattern matches by iterating over the Graph using find. The RDB-based Graphs instead compile some queries into SQL to be submitted to the database query engine. The query handling operates over all the triples expressed by the Graph; however they are generated - as base assertions or as inferred consequences. RDQL uses this interface to do the non-constraint parts of its query handling.

3.1.7. RDQL – RDF QUERY

RDQL (RDF Data Query Language) was pioneered in Jena1. The Jena implementation is the de facto reference implementation. An RDQL query consists of a graph pattern, expressed as a list of triple patterns. Each triple pattern is comprised

of named variables and RDF values (URIs and literals). An RDQL query can additionally have a set of constraints on the values of those variables, and a list of the variables required in the answer set.

SELECT ?x WHERE (

?x, <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>,<http://example.com/someType>)

This triple pattern matches all statements in the graph that have predicate http://www.w3.org/1999/02/22-rdf-syntaxns#type and object http://example.com /someType. The variable "?x" will be bound to the label of the subject resource. All such "x" is returned. An RDQL query treats an RDF graph purely as data. If the implementation of that graph provides differencing to appear as "virtual triples" (i.e. triples that appear in the graph but are not in the ground facts) then an RDQL will include those triples as possible matches in triple patterns. RDQL makes no distinction between inferred triples and ground triples. The next phase of the Semantic Web activity by the W3C is likely to address RDF query. We hope that this work will take our positive experiences with RDQL into account. Jena's RDQL implementation will evolve as a result of the new work at the W3C.

3.1.8. Persistent Storage

Jena supports persistent storage of RDF Models in a conventional database. Implemented at the Graph layer, it provides all the usual Graph operations (add, delete, find) and efficiently supports reification.

3.1.9. Query Processing

Queries are executed against graphs which may have multiple statement tables. For each statement table there is a handler to convert between the graph view of Jena and the triple view of SQL. To evaluate a triple pattern, the query processor passes the pattern, in turn, to each table handler for evaluation. A goal of Jena is support for fast path query processing for RDQL. This is evaluated in a nested-loops fashion in Java by using the results of one triple pattern to bind values to variables and then generating new triple patterns for evaluation. Jena's RDQL uses the Graph query interface to pass

all the triple patterns into the database graph; the goal of fast path query processing is to use the database engine to process the entire query, rather than single patterns. A full discussion of fast path query processing is beyond the scope of the paper. Here, we present two simple cases and mention the difficulties for the general case. For the first simple case, assume that all the triple patterns reference only the triple table. As mentioned above, a single triple pattern can be completely evaluated over a table by a single SQL query. To evaluate multiple patterns in the database engine, it is sufficient to combine the SQL statements for the individual patterns and add additional join conditions for the linking variables. The second simple case is when all patterns can be completely evaluated by a single property table. This is similar to the first case. However, here it may be possible to eliminate joins if the patterns reference properties stored together (since the property values for the same subject are stored in the same row). When the triple patterns for a query apply to multiple tables, it is more difficult to construct a single SQL query to satisfy all patterns.

3.1.10. The RDF Web API

To make RDF repositories available across the Internet, the RDF Web API requires each graph to have a URL for the purposes of naming and routing query traffic to the repository providing that graph. One host repository may have several RDF graphs available, so it is necessary to direct queries to the right one based on both network location and on name. URLs provide the mechanism for this. The protocol used for query is HTTP, specifically the GET verb. In order to provide compatibility with regular web use, a plain GET (no query string provided) is interpreted as fetching the whole RDF graph. A query string provides refinement of the GET to extract a sub graph of the target graph. The query string consists of identification of the query language and a query language specific string giving the query itself. The full details of this can be found in the member submission to the W3C.

3.2. GWT Framework

Google Web Toolkit (GWT) is a development toolkit for building and optimizing complex browser-based applications. Its goal is to enable productive development of high-performance web applications without the developer having to be an expert in browser quirks, XMLHttpRequest, and JavaScript. GWT is used by many products at Google, including Google Wave and the new version of AdWords. It's open source, completely free, and used by thousands of developers around the world.

The GWT SDK provides a set of core Java APIs and Widgets. These allow you to write AJAX applications in Java and then compile the source to highly optimized JavaScript that runs across all browsers, including mobile browsers for Android and the iPhone. Constructing AJAX applications in this manner is more productive thanks to a higher level of abstraction on top of common concepts like DOM manipulation and XHR communication. You aren't limited to pre-canned widgets either. Anything you can do with the browser's DOM and JavaScript can be done in GWT, including interacting with hand-written JavaScript.

Performance bottlenecks aren't limited to JavaScript. Browser layout and CSS often behave in strange ways that are hard to diagnose. Speed Tracer is a new Chrome Extension in Google Web Toolkit that enables you to diagnose performance problems in the browser.

3.2.1. Why GWT?

Using GWT framework for building interface has many advantages; some of them are described below:

- Static type checking in the Java language boosts productivity while reducing errors.
- Common JavaScript errors (typos, type mismatches) are easily caught at compile time rather than by users at runtime.
- Code prompting/completion is widely available
- Automated Java refactoring is pretty snazzy these days.

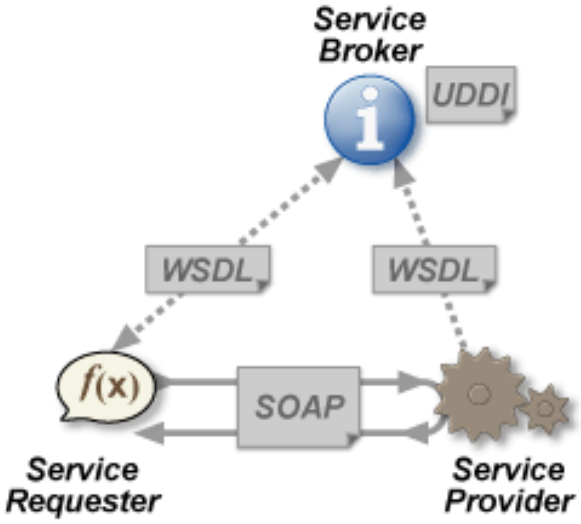
- Java-based OO designs are easier to communicate and understand, thus making your AJAX code base more comprehensible with less documentation.
- No need to learn/use JavaScript language
    - Leverage Java programming knowledge you already have
- No need to handle browser incompatibilities and quirks
    - GWT handles them for you
    - Browser history
    - Forward/backward buttons
- No need to learn/use DOM APIs
    - Use Java APIs
- No need to build commonly used Widgets
    - Most of them come with GWT
- Leverage various tools of Java programming language for writing/debugging/testing
    - NetBeans
    - Eclipse
- JUnit integration
    - GWT's direct integration with JUnit lets you unit test both in a debugger and in a browser and you can even unit test asynchronous RPCs
- Internationalization
    - GWT internationalization support provides a variety of techniques to internationalize strings, typed values, and classes

3.3. Restful Web services

3.3.1. What are web services?

A Web service is a method of communication between two electronic devices over a network. The term Web services describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards

Figure 3-1: Web services architecture [21]



over an Internet protocol backbone. XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing what services are available. Used primarily as a means for businesses to communicate with each other and with clients, Web services allow organizations to communicate data without intimate knowledge of each other's IT systems behind the firewall.

3.3.2. Why we used web services?

We have used web services because it has following advantages:

a) Exposing the existing function on to network:
   - A Web service is a unit of managed code that can be remotely invoked using HTTP, that is, it can be activated using HTTP requests. So, Web Services allows you to expose the functionality of your existing code over the network. Once it is exposed on the network, other application can use the functionality of your program.

b) Connecting Different Applications i.e. Interoperability:
   - Web Services allows different applications to talk to each other and share data and services among themselves. Other applications can also use the services of the web services. For example VB or .NET application can talk to java web services and vice versa. So, Web services are used to make the application platform and technology independent.

c) Standardized Protocol:

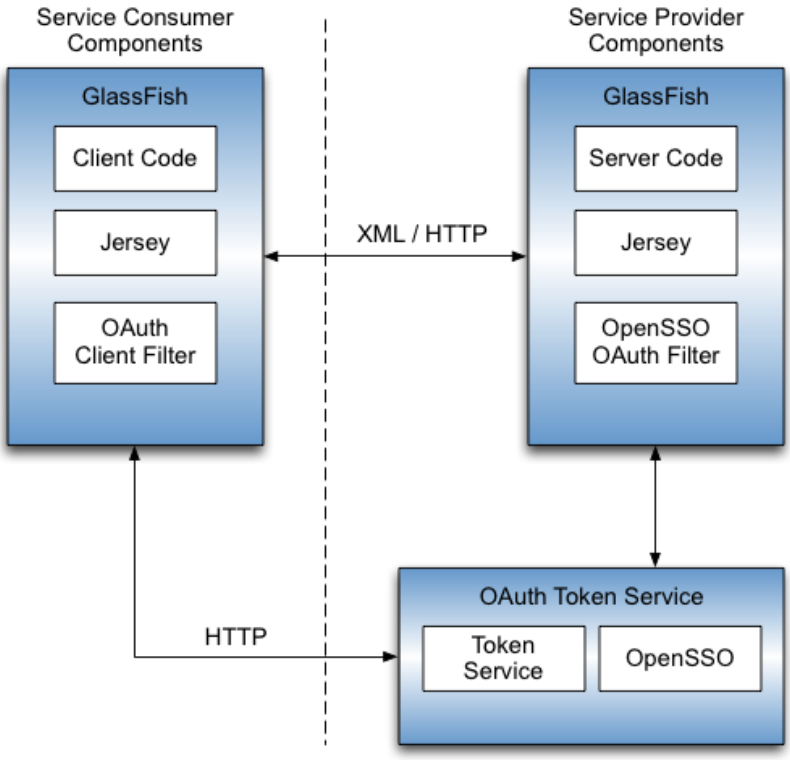
- Web Services uses standardized industry standard protocol for the communication. All the four layers (Service Transport, XML Messaging, Service Description and Service Discovery layers) use the well-defined protocol in the Web Services protocol stack. This standardization of protocol stack gives the business many advantages like wide range of choices, reduction in the cost due to competition and increase in the quality.

d) Low Cost of communication:
   - Web Services uses SOAP over HTTP protocol for the communication, so you can use your existing low cost internet for implementing Web Services. This solution is much less costly compared to proprietary solutions like EDI/B2B. Beside SOAP over HTTP, Web Services can also be implemented on other reliable transport mechanisms like FTP etc.

3.3.3. RESTful web services:

RESTful web services are built to work best on the Web. Representational State Transfer (REST) is an architectural style that specifies constraints, such as the uniform interface, that if applied to a web service induces desirable properties, such as performance, scalability, and modifiability that enable services to work best on the Web. In the REST architectural style, data and functionality are considered resources and are accessed using Uniform Resource Identifiers (URIs), typically links on the Web. The resources are acted upon by using a set of simple, well-defined operations. The REST architectural style constrains architecture to client/server architecture and is designed to use a stateless communication protocol, typically HTTP. In the REST architecture style, clients and servers exchange representations of resources by using a standardized interface and protocol.

Figure 3-2: REST-Based Web Service Deployment Architecture [19]



3.3.4. Principles of RESTful web services

The following principles encourage RESTful applications to be simple, lightweight, and fast:

a) Resource identification through URI:
   - A RESTful web service exposes a set of resources that identify the targets of the interaction with its clients. Resources are identified by URIs, which provide a global addressing space for resource and service discovery.

b) Uniform interface:
   - Resources are manipulated using a fixed set of four create, read, update, delete operations: PUT, GET, POST, and DELETE. PUT creates a new resource, which can be then deleted by using DELETE. GET retrieves the current state of a resource in some representation. POST transfers a new

state onto a resource. See Responding to HTTP Resources for more information.

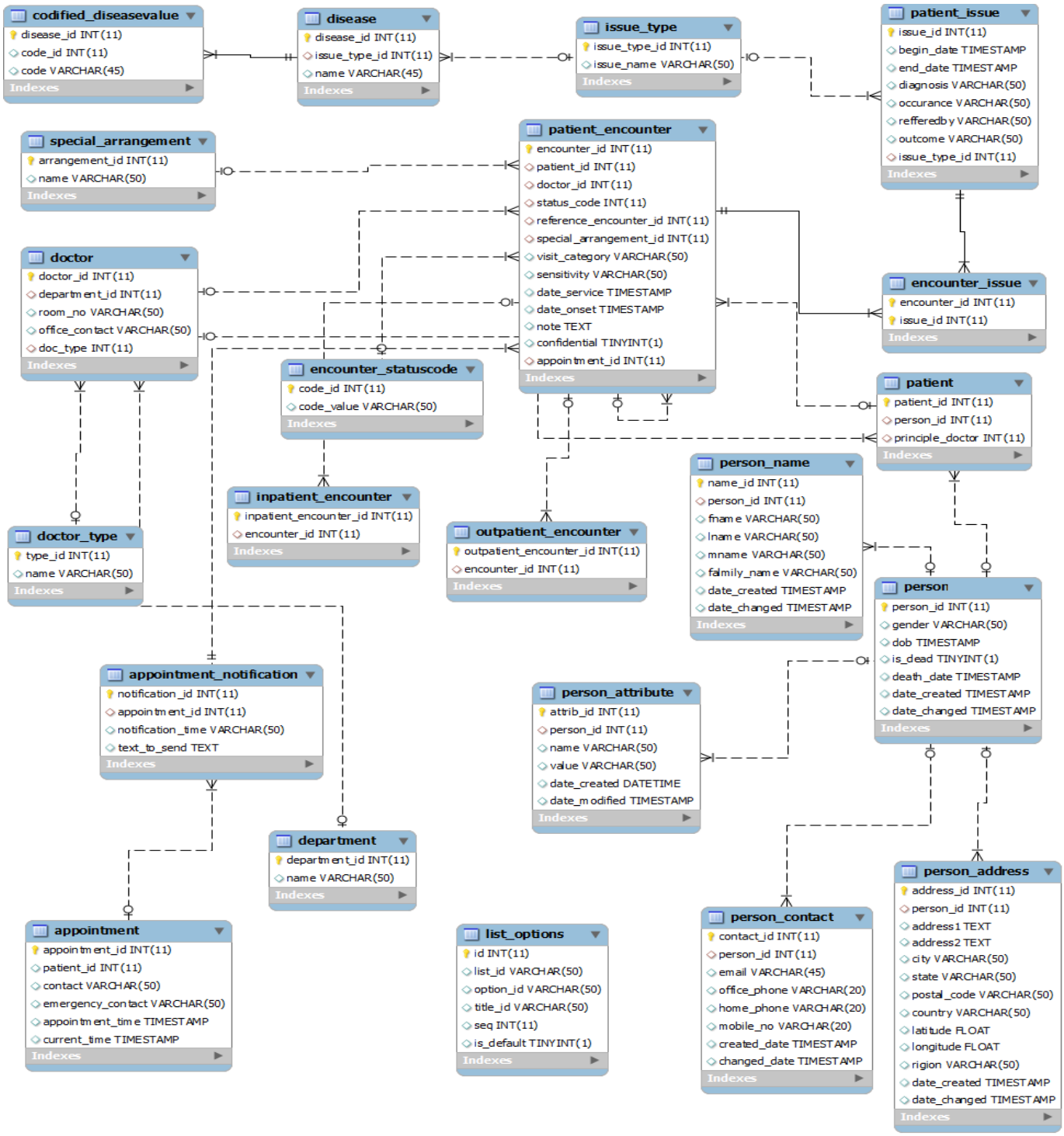
c) Self-descriptive messages:
   - Resources are decoupled from their representation so that their content can be accessed in a variety of formats, such as HTML, XML, plain text, PDF, JPEG, JSON, and others. Metadata about the resource is available and used, for example, to control caching, detect transmission errors, negotiate the appropriate representation format, and perform authentication or access control. See Responding to HTTP Resources and Using Entity Providers to Map HTTP Response and Request Entity Bodies for more information.

d) Stateful interactions through hyperlinks:
   - Every interaction with a resource is stateless; that is, request messages are self-contained. Stateful interactions are based on the concept of explicit state transfer. Several techniques exist to exchange state, such as URI rewriting, cookies, and hidden form fields. State can be embedded in response messages to point to valid future states of the interaction.

4. DESIGN *Chapter 04*

4.1. ERD

Figure 4-1: Entity Relationship Diagram



4.2. Brief description of tables

Person: In this table, some generic information is stored about patients, doctors and any person related to our system. It contains the following attributes:

1. Person_id
2. Gender
3. DOB
4. Is_dead
5. Death_date

Person_Name: This table contains the all the information about the person name having following attributes:

1. Name_id
2. Fname
3. Mname
4. Lname
5. familyName

Person_contact: It contains all the contact information about a person having following attributes:

1. Contact_id
2. Email
3. Home_phone
4. Office_phone
5. Mobile_number

Person_address: This table contains all the information about person's address having following attributes:

1. Address_id
2. Street_address_1
3. Street_address_2

4. Postal_code
5. City
6. state
7. Country
8. Region
9. Latitude
10. longitude

Patient: This table is to store the necessary information about a patient having following attributes:

1. Patient_id
2. Person_id
3. Principle_doctor

Patient_encounter: This table is to store the information of a patient required at the time of encounter when patient comes in the hospital, having following attributes:

1. Encounter_id
2. Patient_id
3. Doctor_id
4. Status
5. Confidentiality
6. Sensitivity
7. Visit_category
8. Reference_encounter_id
9. Appointment_id
10. Vital_id
11. Note
12. Date_service
13. Date_onset

Patient_issue: This table is to store the information of a patient's disease and his issues when his encounter is being processed, having following attributes:

1. Issue_id
2. Patient_id
3. Reffered_by
4. Diagnosis
5. Occurrence
6. outcome
7. begin_date
8. end_date

Issue_type: Every issue has a type and is assigned and this information is stored in table having following information:

1. issue_type_id
2. issue_name

Disease: This table is to store the information of disease. Each disease is assigned a code to uniquely identify it and its diagnosis:

1. disease_code
2. disease_name
3. issue_type_id

Vitals: This table is to store the vital information of patients at the time of encounter. It has following attributes.

1. weightKg
2. weightLb
3. heightInch
4. heightCm
5. BPSystolic
6. BPDiastolic
7. pulse
8. respiration
9. temperatureF
10. temperatureC

11. oxygenSaturation
12. headinCircumference
13. headcmCircumference
14. waistinCircumference
15. waistcmCircumference
16. BMI
17. BMIStatus
18. otherNotes
19. tempLocation

Appointment: This table is to store the information patient required at the time of making an appointment with a doctor:

1. appointment_id
2. doctor_id
3. patient_id
4. contact_no
5. emergency_contact
6. appointment_time

Doctor: This table is to store the information of doctor having attributes as follows:

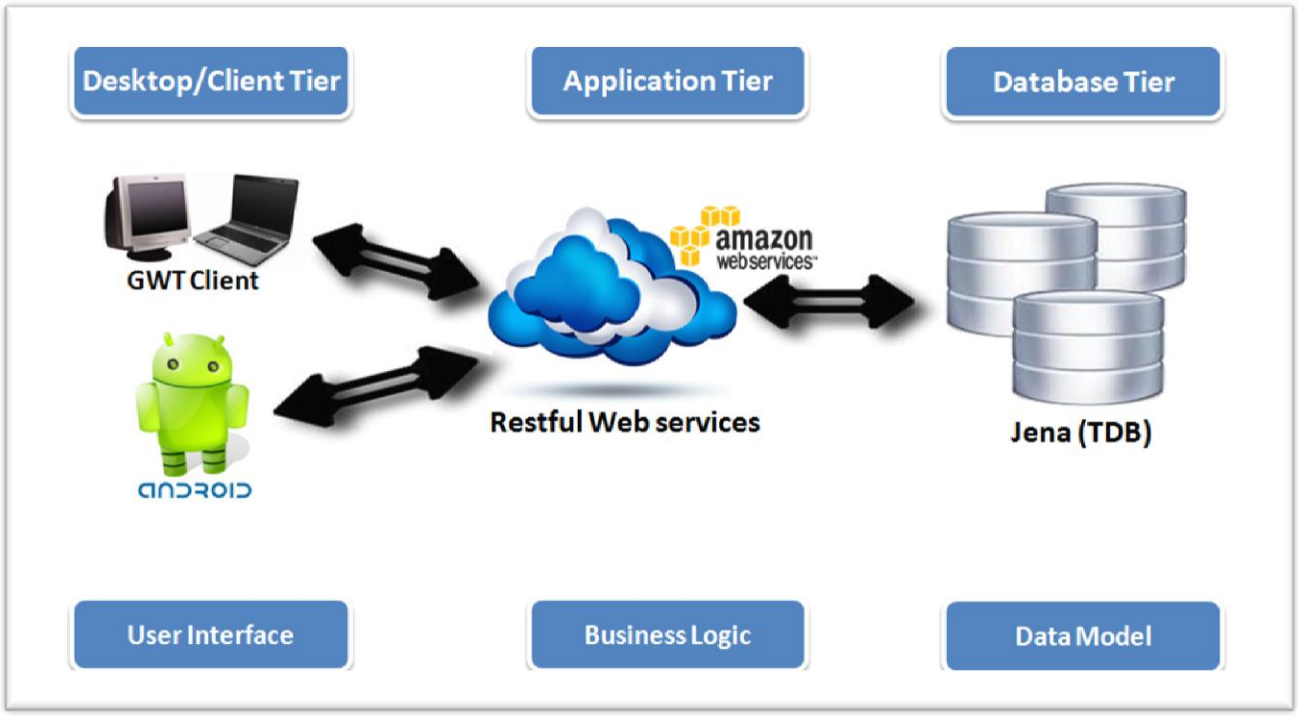
1. person_id
2. doctor_id
3. room_no
4. department_id
5. office_hours

4.3.Architecture

Our system has 3-tier architectural design.

1. Client Tier
2. Application Tier

Figure 4-2: Architecture Diagram



3. Database Tier

4.3.1. Client Tier

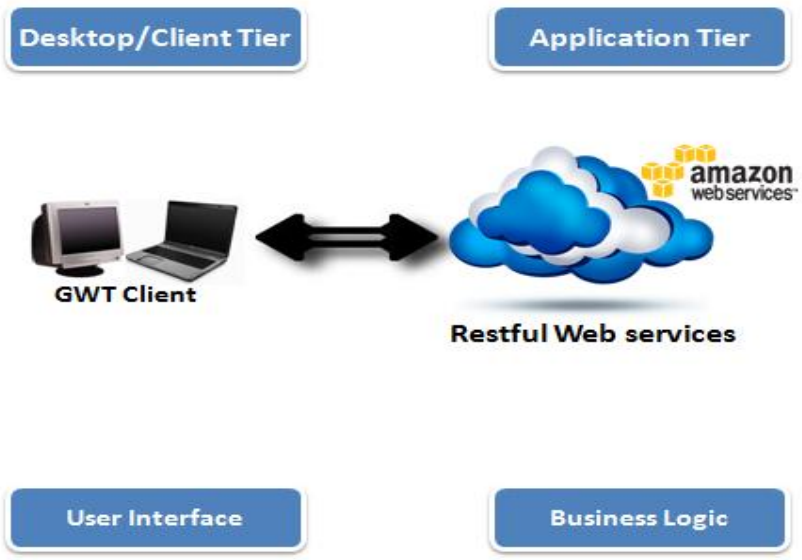
In client tier, we have two types of clients. One is web based interface and other is Android phone.

1. Web based Client/Browser
2. Android Phone

Web based Client/Browser

We have provided a web interface for interacting with our EMR system. Our client tier is totally independent from our application tier. Our client sends request over HTTP to our Application tier and also gets response from application tier over HTTP.

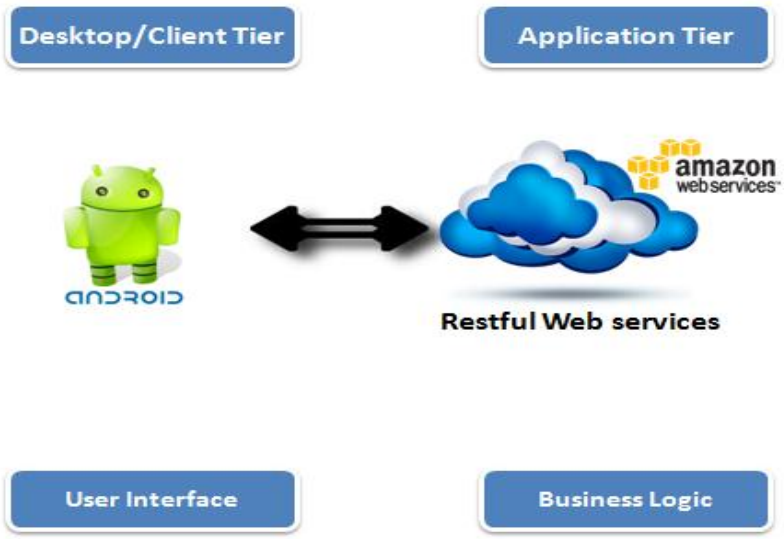
Figure 4-3: Web Client Interaction



Android Phone

We have provided an android interface for interacting with our EMR system. Our android application sends request over HTTP to our Application tier and also gets response from application tier over HTTP.

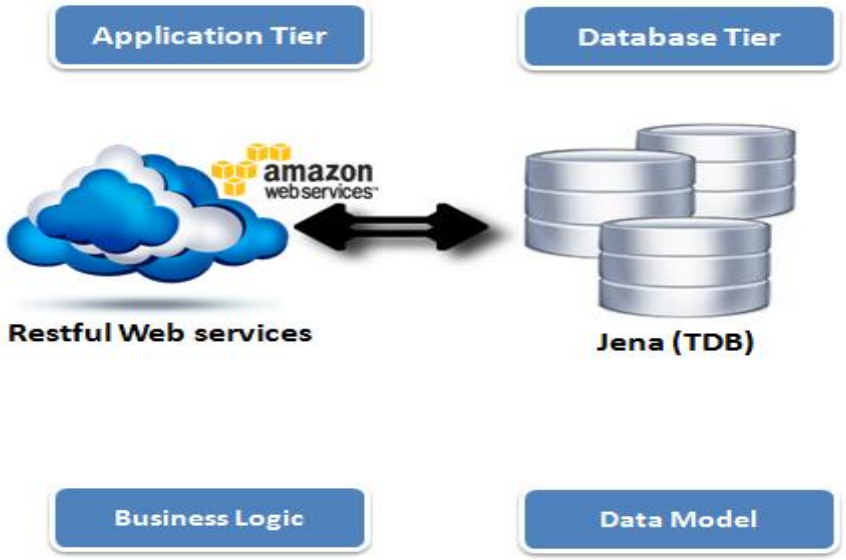
Figure 4-4: Android Interaction



4.3.2. Application Tier

Our application tier is developed using restful web services. Restful web services receive the HTTP request and do business logic. After applying business logics the restful web services send data to database tier for storage/retrieval.

Figure 4-5: Application Tier



4.3.3. Database Tier

Our database tier receive request from application tier and respond accordingly. Application may want to store data or retrieve data so our database tier responds accordingly.

4.4.System features

4.4.1. Patient registration

Introduction/purpose of feature

This feature provides the ability for registration, identification and maintenance of patient record.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	1
Use Case	**Record patient registration**
Actor	Data Entry Operator (DEO)/Receptionist (RE)
Purpose	Purpose of the use case is to create identity of a patient that has come to the hospital for health care.
Cross Reference	View Patient Demographic Information
Functional requirements	1. The system shall create a single patient record for each patient. 2. The system shall associate (store and link) key identifier information (e.g., system ID, medical record number) with each patient record. 3. The system shall provide the ability to store more than one

	identifier for each patient record. 4. The system shall provide a field which will identify patients as being exempt from reporting functions. 5. The system shall provide the ability to merge patient information from two patient records into a single patient record.
Pre-conditions	➢ Patient must not have an existing identity in the system ➢ DEO/RE should be authorized and has access privileges.
Course of event	1. DEO or RE press Load Patient Registration menu item. 2. System loads the corresponding form and displayed. 3. DEO or RE enters demographic information like; a) Patient Name b) Patient Gender c) Patient Age And press Save button to save information. 4. System validates the information and save the information successfully. After successful saving it generates unique patient medical record number.
Alternate course of event	Invalid demographic information. 1. System displays errors message by flagging fields of wrong data. 2. DEO updates the information and perform step 3 above.

Use Case id	2
Use Case	**Manage patient demographics**
Actor	Data Entry Operator (DEO)/Receptionist (RE)

Purpose	Purpose of the use case is to modify and maintain patient demographic information.
Cross Reference	View Patient Demographic Information
Functional requirements	1. The system shall provide the ability to include demographic information in reports. 2. The system shall provide the ability to maintain and make available historic information for demographic data including prior names, addresses, phone numbers and email addresses. 3. The system shall provide the ability to modify demographic information about the patient. 4. The system shall store demographic information in the patient medical record in separate discrete data fields, such that data extraction tools can retrieve these data. 5. The system shall provide the ability to access demographic information such as name, date of birth and gender needed for patient care functions. 6. The system shall capture and maintain demographic information as discrete data elements as part of the patient record. 7. The system shall provide the ability to query for a patient by more than one form of identification.
Pre-conditions	➢ Registration of the patient must exist. ➢ DEO/RE should be authorized and has access privileges.
Course of event	1. DEO or RE clicks edit demographic information form. 2. System displays patient search form.

	3. DEO or RE filters and searches of patient demographic information using selection criteria such as last name, medical record number, account number telephone number and clicks edit button. 4. System displays edit form. 5. DEO or RE modifies data and clicks save button. 6. System saves information.
Alternate course of event	Invalid demographic information. 1. System displays errors message by flagging fields of wrong data. 2. DEO/RE performs step 3.

4.4.2. Manage problem/diagnosis list

Introduction/purpose of feature

This feature provides the ability for capturing and maintaining all problem/diagnoses associated with a patient.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	3
Use Case	**Record problems/diagnosis**
Actor	Physician
Purpose	To provide the ability to record, capture all problems/diagnosis and updates.
Cross Reference	View Patient Demographic Information
Functional	1. The system shall provide the ability to capture, maintain

requirements	and display, as discrete data elements, all problems/diagnoses associated with a patient. 2. The system shall provide the ability to record the chronicity (chronic, acute/self-limiting, etc.) of a problem/diagnosis. 3. The system shall provide the ability to record the user ID and date of all updates to the problem/diagnosis list. 4. The system shall provide the ability to capture, maintain and display free text comments associated with the problem / diagnosis.
Pre-conditions	➢ Physician should be authorized and has access privileges. ➢ Patient must be registered
Course of event	1. Physician clicks record problem/diagnosis button. 2. System displays a problem/diagnosis form. 3. Physician enters appropriate data and clicks save button 4. System saves information.
Alternate course of event	Invalid or incomplete problem/diagnosis information. 1. System displays error message by flagging fields of wrong or incomplete data. 2. Physician performs step 3.

Use Case id	4
Use Case	**Maintain problems/diagnosis list**
Actor	Physician
Purpose	To provide the ability to maintain and update the resolution and medication of problem/diagnosis.

Cross Reference

Functional requirements	1. The system shall provide the ability to maintain the onset date of the problem/diagnosis. 2. The system shall provide the ability to maintain the resolution date of the problem/diagnosis. 3. The system shall provide the ability to associate orders, medications, and notes with one or more problems/diagnoses. 4. The system shall provide the ability to associate orders and medications with one or more codified problems/diagnoses. 5. The system shall provide the ability to maintain a coded list of problems/diagnoses. 6. The system shall provide the ability to display different views of the problem / diagnosis list based upon the status of the problem.
Pre-conditions	➢ Physician should be authorized and has access privileges. ➢ Patient must be registered
Course of event	1. Physician clicks update problem/diagnosis button. 2. System displays a problem/diagnosis form. 3. Physician enters appropriate data and clicks save button 4. System saves information.
Alternate course of event	Invalid or incomplete problem/diagnosis information. 1. System displays errors message by flagging fields of wrong or incomplete data. 2. Physician performs step 3.

4.4.3. Manage medication list

Introduction/purpose of feature

This feature provides the ability for recording and maintaining all medications associated with a patient.

Associated functional requirements and stimulus/response sequence – use case

Use Case id 5

Use Case	**View medication**
Actor	Physician, Data Entry Operator
Purpose	To provide the ability to view all medication records and history.
Cross Reference	Edit medication
Functional requirements	1. The system shall provide the ability to display medication history for the patient. 2. The system shall provide the ability to display a patient-specific medication list based on current medication orders or prescriptions. 3. The system shall provide the ability to display a view that includes only current medications. 4. The system shall provide the ability to exclude a medication from the current medication list (e.g. marked inactive, erroneous, completed, discontinued) and document reason for such action. 5. The system shall provide the ability to print a current medication list. 6. The system shall provide the ability to display that the patient takes no medications.
Pre-conditions	➢ Physician should be authorized and has access privileges.

	➢ Patient must be registered
Course of event	1. Physician clicks on search button and enter query for specific patient 2. System displays specified patient 3. Physician clicks view medication button. 4. System displays all medication history associated with specified patient.
Alternate course of event	Invalid or incomplete patient query in search box. 1. System displays error message of wrong query 2. Physician reenter correct query.

Use Case id	6
Use Case	**Add medication**
Actor	Physician, Data Entry Operator
Purpose	To provide the ability to add medication to particular patient.
Cross Reference	view medication
Functional requirements	1. The system shall provide the ability to record the prescribing of medications including the identity of the prescriber. 2. The system shall provide the ability to capture medications entered by authorized users other than the prescriber. 3. The system shall store medication information in discrete data fields. At a minimum, there must be one field for each of the following:- medication name, form and strength;- dispense quantity;- refills; and- sig.

	4. The system shall provide the ability to enter uncoded or free text medications when medications are not on the vendor-provided medication database or information is insufficient to completely identify the medication. 5. The system shall provide the ability to enter or further specify in a discrete field that the patient takes no medications. 6. The system shall provide the ability to record the date of changes made to a patient's medication list and the identity of the user who made the changes. 7. The system shall provide the ability to capture and maintain, as discrete data elements, all current medications including over-the-counter and complementary medications such as vitamins, herbs and supplements.
Pre-conditions	➢ Physician should be authorized and has access privileges. ➢ Patient must be registered
Course of event	1. Physician clicks on add medication 2. System displays a form to add medication 3. Physician enters appropriate date 4. Physician clicks on add medication button 5. System inserts that record into database and displays a message
Alternate course of event	Invalid or incomplete data in add medication form. 1. System displays error message stating wrong input 2. Physician performs step 3.

Use Case id	7

Use Case	**Update medication**
Actor	Physician
Purpose	To provide the ability to update particular medication record.
Cross Reference	view medication
Functional requirements	1. The system shall provide the ability to maintain medication ordering dates. 2. The system shall provide the ability to maintain other dates associated with medications including start, modify, renewal and end dates as applicable. 3. The system shall provide the ability to update a patient-specific medication list based on current medication orders or prescriptions. 4. The system shall provide the ability to capture and maintain, as discrete data elements, all current medications including over-the-counter and complementary medications such as vitamins, herbs and supplements.
Pre-conditions	➢ Physician should be authorized and has access privileges. ➢ Patient must be registered
Course of event	1. Physician clicks on search button and enter query for specific patient 2. System displays specified patient 3. Physician clicks update medication button 4. Physician updates appropriate data fields 5. System displays all medication history associated with specified patient
Alternate	Invalid or incomplete patient query in search box.

course of event	1. System displays errors message of wrong query 2. Physician reenter correct query.

4.4.4. Allergy information

Introduction/purpose of feature

This feature provided the ability to input view and manipulate the allergies of the admitted patient. This list will be available when ordering medication for the patient so that any conflicts can be determined.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	8
Use Case	**Record Patient Allergy Information**
Actor	Physician (PH)/ Nurse (NU)
Purpose	Purpose of the use case is to record all allergies related effects of a patient.
Cross Reference	
Functional requirements	1. The system shall provide the ability to modify or inactivate an item on the allergy and adverse reaction list. 2. The system shall provide the ability to specify the type of allergic or adverse reaction in a discrete data field. 3. The system shall provide the ability to capture and maintain, as discrete data, the identity of the user who added, modified, inactivated or removed items from the allergy and adverse reaction list, including attributes of the changed items. The user ID and date/time stamp shall be recorded. 4. The system shall provide the ability for a user to explicitly capture and maintain, as discrete data, that the allergy list

	was reviewed. The user ID and date/time stamp shall be recorded when the allergies reviewed option is selected. 5. The system shall provide the ability to explicitly indicate in a discrete field that a patient has no known drug allergies or adverse reactions. 6. The system shall provide the ability to capture, maintain and display, as discrete data, lists of medications and other agents to which the patient has had an allergic or other adverse reaction.
Pre-conditions	➢ PH/NU should be authorized and has access privileges.
Course of event	1. PH or NU clicks on record allergy information menu item. 2. System displays entry form. 3. PH or NU fills form and clicks save button. 4. System saves information.
Alternate course of event	Invalid Allergy information. 1. System displays errors message by flagging fields of wrong data. 2. PH/NU updates the information and perform step 3.

Use Case id	9
Use Case	**Update Allergy Information Notice**
Actor	Notification Service (NS)
Purpose	Purpose of the use case is to notify physician that patient allergy information needs to be updated.
Cross Reference	view medication
Functional	1. When allergies are “Unknown” or “Unable to Assess

requirements	Allergies," the system shall provide the ability to inform the clinician for the need of an update.
Pre-conditions	
Course of event	1. NS search into its log for active notification, and prepare proper allergy notification to intended user. 2. The system takes notification of NS and displays to the user on proper scheduled time.
Alternate course of event	

Use Case id	10
Use Case	**View Allergy Information**
Actor	Physician (PH)/ Nurse (NU)
Purpose	Purpose of the use case is to view allergy information list
Cross Reference	
Functional requirements	1. The system shall provide the ability to display the allergy list, including date of entry. 2. The system shall provide the ability to display information which has been inactivated or removed from the allergy and adverse reaction list. 3. When the display of the allergy list exceeds the current screen or printed page, the system shall indicate that the list continues.
Pre-conditions	➢ PH/NU should be authorized and has access privileges.
Course of event	1. PH or NU clicks in patient search form. 2. System displays patient search form.

	3. PH or NU applies filters to search patient according to selection criteria and clicks view allergy information button. 4. System displays allergy information list.
Alternate course of event	

4.4.5. Capture external clinical documents

Introduction/purpose of feature

This feature is used to help the system to communicate with external enjoinment. With the help of this feature we can share documents (documents include scanned documents, images, or digital documents etc.) with external environment. This covers all types of documents received by the practice that would typically be incorporated into a medical record, including but not limited to faxes, referral authorizations, consultant reports, and patient correspondence of a clinical nature.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	11
Use Case	**Capture external clinical documents**
Actor	Patient, physician, lab Technicians
Purpose	The purpose of this use case is to share documents between system and above mentioned Actors.
Cross Reference	order medication, Generate and record patient specific instructions
Functional requirements	1. The system shall provide the ability to save scanned documents as images. 2. The system shall provide the ability to receive, store in the patient's record, and display text-based outside reports.

3. The system shall provide the ability to index scanned documents and associate a date and document type to the document.
4. The system shall provide the ability to retrieve indexed scanned documents based on document type and date.
5. The system shall provide access to clinical images. They must be accessible from within the patient's chart and labeled and date-time stamped or included in a patient encounter document. These images may be stored within the system or be provided through direct linkage to external sources.
6. The system shall provide the ability to accept, store in the patient's record, and display clinical results received through an interface with an external source.
7. The system shall provide the ability to record that patient specific instructions or educational material were provided to the patient.

Pre-conditions	➢ User must be logged on. ➢ User must have enough access control to share the documents.
Course of event	1. User click on send document button. 2. User will select document. 3. User will give the name of receiver. 4. User will click on send button. 5. If document is sent from external environment to our

system then system will save the document.

6. System will save the whole transaction.

Alternate course of event	Connection failure. 1. System displays errors message by flagging fields of wrong data. 2. User will click on send button again.

4.4.6. Generate and record patient specific instructions

Introduction/purpose of feature

This feature is used to provide information and instruction to patient that will help the patient to decide.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	12
Use Case	**Generate and record patient specific instructions**
Actor	Patient, physician, health organizations
Purpose	The purpose of this use case is to provide useful instructions to patient.
Cross Reference	Edit existing instruction
Functional requirements	1. The system shall provide the ability to produce patient instructions and patient educational materials which may reside within the system or be provided through links to external source. 2. The system shall have the ability to provide access to patient-specific test and procedure instructions that can be modified by the physician or health organization; these instructions are to be given to the patient. These

	instructions may reside within the system or be provided through links to external sources. 3. The system shall provide the ability to record that patient specific instructions or educational material were provided to the patient.
Pre-conditions	➢ User must be logged on. ➢ User must have enough access control to give instructions.
Course of event	1. User goes to patient profile. 2. User clicks on give instruction button. 3. User types instruction in text box. 4. User clicks on send button. 5. System will save this instruction.
Alternate course of event	If system fails to send instruction, system will generate an error message and ask the user to send instruction again.

Use Case id	13
Use Case	**Edit existing instruction**
Actor	physician, health organizations
Purpose	The purpose of this use case is to edit the instructions.
Cross Reference	Generate and record patient specific instructions
Functional requirements	1. The system shall provide the ability to produce patient instructions and patient educational materials which may reside within the system or be provided through links to external source. 2. The system shall have the ability to provide access to

	patient-specific test and procedure instructions that can be modified by the physician or health organization; these instructions are to be given to the patient. These instructions may reside within the system or be provided through links to external sources. 3. The system shall provide the ability to record that patient specific instructions or educational material were provided to the patient.
Pre-conditions	➢ User must be logged on. ➢ User must have enough access control to give instructions.
Course of event	1. User visit to patient profile. 2. User selects an existing instruction. 3. User clicks on edit instruction button. 4. User edits the instruction. 5. User clicks on send button. 6. System will save this instruction.
Alternate course of event	If system fails to send instruction, system will generate an error message and ask the user to send instruction again.

4.4.7. Order medication

Introduction/purpose of feature

This feature will enable the system to provide the ability to create prescription or other medication orders with sufficient information for correct filling and dispensing by a pharmacy.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	14
Use Case	**Create prescription with sufficient information**

Actor	Physician
Purpose	The purpose of this use case is to create prescription and make it available to intended peoples.
Cross Reference	Edit existing instruction
Functional requirements	1. The system shall provide the ability to create prescription or other medication orders with sufficient information for correct filling and dispensing by a pharmacy. 2. The system shall provide the ability to record user and date stamp for prescription related events, such as initial creation, renewal, refills, discontinuation, and cancellation of a prescription.
Pre-conditions	➢ User must be logged on. ➢ User must have enough access control to give prescription.
Course of event	1. User selects a patient. 2. User clicks on new prescription button. 3. User fills the prescription form. 4. User adds prescription to the patient profile. 5. System will save this transaction.
Alternate course of event	

Use Case id	15
Use Case	**Check validity of prescription**
Actor	System
Purpose	The purpose of this use case is to check the date of prescription.

Cross Reference	
Functional requirements	1. The system shall provide the ability to identify medication samples dispensed, including lot number and expiration date.
Pre-conditions	
Course of event	1. System gets the validation date of prescription. 2. System gets the current date. 3. System compares current date with validation date of prescription. 4. System generates an alarm if prescription has expired. 5. System sends a message to patient.
Alternate course of event	

Use Case id	16
Use Case	**Create an event based on prescription**
Actor	System
Purpose	The purpose of this use case is to generate events based on prescription.
Cross Reference	
Functional requirements	The system shall provide the ability to record user and date stamp for prescription related events, such as initial creation, renewal, refills, discontinuation, and cancellation of a prescription.
Pre-conditions	

Course of event	System gets the date of event from prescription. System gets the current date. System compares current date with the date of event mentioned in prescription. System sends a message to patient if current date matches to event.
Alternate course of event	

Use Case id	17
Use Case	**Dose calculator**
Actor	System
Purpose	The intent is to allow input of dose-per-weight and patient weight and calculate the corresponding dose.
Cross Reference	
Functional requirements	1. The system shall provide the ability to display a dose calculator for patient-specific dosing based on weight.
Pre-conditions	
Course of event	1. System gets the weight of patient. 2. System calculates dose amount based on some formula.
Alternate course of event	

Use Case id	18

Use Case	**Reorder a prior prescription**
Actor	Patient
Purpose	Use existing prescription without reentering the prescription.
Cross Reference	
Functional requirements	1. The intent is to allow input of dose-per-weight and patient weight and calculate the corresponding dose.
Pre-conditions	➢ User must be logged on. ➢ User must have enough access control to share the documents.
Course of event	1. Patient selects prescription. 2. Patient clicks on reorder prescription. 3. System saves the transaction.
Alternate course of event	

4.4.8. Clinical Task assignment

Introduction/purpose of feature

In this feature the user can create a task and he can edit or reassign the task. User can also view the list of the entire task as well as completed task.

Associated functional requirements and stimulus/response sequence – use case

Use case id	19
Use Case	**Create and assign task**
Actor	Provider
Purpose	The purpose of this use case is to assign a task to user.
Cross Reference	
Functional	1. The system shall provide the ability to create and

requirements	assign tasks by user
Pre-conditions	2. The user must be login to create task. 3. User should be authorized and has access privileges.
Course of event	1. The user will click on creak task on the main form. 2. System shall take the user to task creation form. 3. On task creation page the user will enter the task description. 4. Select the person to assign the task. 5. Click on save button to create the task.
Alternate course of event	In case of task is not saved the system should display proper error message.

Use Case id	20
Use Case	**Edit the assigned task**
Actor	Provider
Purpose	The purpose of this use case is to edit/reassign a task to user.
Cross Reference	
Functional requirements	1. The system shall provide the ability to edit/reassign tasks.
Pre-conditions	➢ User should be authorized and has access privileges.
Course of event	1. The user will list the entire task. 2. User will select a task to edit. 3. System shall take the user to task edit form. 4. On task edit page the user will enter the task description if he want to change.

	5. Select the person to reassign the task. 6. Click on save button to edit the task.
Alternate course of event	In case of task is not saved the system should display proper error message.

Use Case id	21
Use Case	**Display complete list of task and completed task**
Actor	Provider
Purpose	The purpose of this use case view the complete list of task.
Cross Reference	
Functional requirements	1. The system shall provide the ability to present a list of tasks by user.
Pre-conditions	➢ User should be authorized and has access privileges.
Course of event	1. The user will list the entire task. 2. The user will click completed task check box to view completed task.
Alternate course of event	

4.4.9. Pharmacy communication

Introduction/purpose of feature

In this feature the user has the ability to communicate the pharmacies to place an order using some electronic means.

Associated functional requirements and stimulus/response sequence – use case

Use Case id 22

Use Case	**Communication between prescribers and pharmacies**
Actor	Prescribers, Pharmacies
Purpose	The purpose of use case is to Provide Electronic mean of communication between prescribers and pharmacies
Cross Reference	
Functional requirements	1. The system shall have the ability to provide electronic communication between prescribers and pharmacies or other intended recipients of the medication order.
Pre-conditions	➢ User should be authenticated. ➢ User must have the privileges to place the order.
Course of event	1. The user clicks the button to open the form for placing the order. 2. The user selects the medicine and enters the quantity. 3. User selects the pharmacies. 4. User click on place order button to place the order.
Alternate course of event	

4.4.10. Provider demographics

Introduction/purpose of feature

In this feature administration can make a directory of provider. Administration also has the option to view the provider currently login to the system.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	23
Use Case	**Maintain directory of providers**
Actor	Administration

Purpose	Purpose of the use case is to maintain and display the directory of providers which are currently accessing the system.
Cross Reference	
Functional requirements	1. The system shall provide the ability to maintain a directory of all clinical personnel who currently use or access the system. 2. The system shall provide the ability to maintain a directory which contains identifiers required for licensed clinicians to support the practice of medicine
Pre-conditions	➢ User should be authenticated. ➢ User must have the privileges to manage provider list.
Course of event	1. User clicks the manage provider button to view the providers which are currently login. 2. User will have the option to create a provider directory. 3. When user click on create button a form will displayed. 4. In the form he will suggest a name for the directory. 5. Add the list of provider to that directory and click save button to save the directory.
Alternate course of event	If system is unable to get list of currently login provider than system shall display the error message. If the provider directory is not saved than error will be shown that the directory is not created.

4.4.11. Scheduling

Introduction/purpose of feature

In this feature the system shall provide the ability to display the schedule of patient appointments.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	24
Use Case	**Display a schedule of patient appointments**
Actor	Administration, Receptionist (RE)
Purpose	Purpose of the use case is to display a schedule of patient appointments because it is very important to view the schedule appointments of a patient.
Cross Reference	
Functional requirements	1. The system shall provide the ability to display a schedule of patient appointments, populated either through data entry in the system itself or through an external application interoperating with the system.
Pre-conditions	➢ User should be authenticated. ➢ User must have the privileges to view patient appointments.
Course of event	1. First user searches the patient using it demographics info. 2. User will select the patient. 3. After selecting the patient user will have the option to view its schedule appointments.
Alternate course of event	If user is not found than system will report that user not exists. If user has no appointment than a message will be shown that user has no appointment.

4.4.12. Report generation

Introduction/purpose of feature

Using report generation feature user can make a report for a specific patient. User can also make summary reports and reports for multiple patients.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	25
Use Case	**Generate report**
Actor	Administration, Provider
Purpose	Purpose of the use case is to generate report in order analyze the patient history. The report generation is a mandatory part everywhere.
Cross Reference	
Functional requirements	1. The system shall provide the ability to generate reports of clinical and administrative data using either internal or external reporting tools. 2. The system shall provide the ability to generate reports consisting of all or part of an individual patient's medical record (e.g. patient summary). 3. The system shall provide the ability to generate reports regarding multiple patients (e.g. diabetes roster). 4. The system shall provide the ability to access reports outside the EHR application.
Pre-conditions	➢ User should be authenticated. ➢ User must have the privileges to generate reports.
Course of event	1. First user searches the patient using it demographics info. 2. User will select the patient. 3. After selecting the patient user will have the option to generate its report. 4. User has the option to generate report using a specific outsource e.g. File, print.
Alternate course	If user is not found than system will report that user not exists.

of event	If system is unable to generate report on specific outsource than I will display error e.g. cannot copy to file.

4.4.13. Health record output

Introduction/purpose of feature

Heath record output is used to enforce confidentiality. System will generate the report but the user confidentiality will not compromise.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	26
Use Case	**Generate report using specific format**
Actor	Administration, Provider
Purpose	Purpose of the use case is to generate report using specific format so that user confidentiality is not compromised e.g. Names, Postal address information, other than town or city, state and zip code, Telephone numbers, Fax numbers, Electronic mail addresses, Social security numbers, Medical record numbers of a user should not be printed on reports.
Cross Reference	
Functional requirements	1. The system shall provide the ability to define one or more reports as the formal health record for disclosure purposes. 2. The system shall provide the ability to generate hardcopy or electronic output of part or all of the individual patient's medical record. 3. The system shall provide the ability to generate hardcopy and electronic output by date and/or date range 4. The system shall provide the ability to export structured

	data which removes those identifiers listed in the HIPAA definition of a limited dataset e.g. Names, Postal address information, other than town or city, state and zip code, Telephone numbers, Fax numbers, Electronic mail addresses, Social security numbers, Medical record numbers.
Pre-conditions	➢ User should be authenticated. ➢ User must have the privileges to generate reports.
Course of event	1. First user searches the patient using it demographics info. 2. User will select the patient. 3. After selecting the patient user will have the option to generate its report.
Alternate course of event	

4.4.14. Enforcement of confidentiality

Introduction/purpose of feature

Enforcement of confidentiality feature will confirm that certain information as confidential and only make that accessible by appropriately authorized users.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	27
Use Case	**Ensure the user confidentiality**
Actor	System operator
Purpose	Purpose of the use case is to enforce the confidentiality.
Cross Reference	
Functional requirements	➢ The system shall provide a means to document a patient's dispute with information currently in their chart.

- The system shall provide the ability to identify certain information as confidential and only make that accessible by appropriately authorized users.
- The system shall provide the ability to prevent specified user(s) from accessing a designated patient's chart.

Pre-conditions	
Course of event	When a user want to access information than he must be authenticated.
Alternate course of event	If user is not authenticated than the system will display the error message.

4.4.15. Clinical Documentation

Introduction/purpose of feature

The purpose of this feature is to record and display the identity and credentials of all users who entered all or part of a note even if they did not finalize the note.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	28
Use Case	**Record and display the identity and credentials of all users who enter a note**
Actor	Physicians (PH) / Nurse (NU)
Purpose	To record and display the identity and credentials of all users who enter a note or involve in the creation of a note
Cross Reference	
Functional requirements	1. This criterion does not require that the system identify or display which portion or portions of a final note were entered by each user but rather that the system record and

	display which users were involved in any part of the creation of the note.
Pre-conditions	➢ PH/NU should be authorized to view this information
Course of event	1. PH or NU clicks in patient search form. 2. System displays patient search form. 3. PH or NU applies filters to search patient according to selection criteria and clicks view standard document button. 4. System displays formatted document
Alternate course of event	

4.4.16. Access Control

Introduction/purpose of feature

This feature provides the ability to control unauthorized access to patient data.

Associated functional requirements and stimulus/response sequence – use case

Use Case id	29
Use Case	**Record and display the identity and credentials of all users who enter a note**
Actor	D
Purpose	Purpose of the use case is to assign or edit information access control privileges to users registered with the system
Cross Reference	
Functional requirements	1. The system shall provide the ability for authorized administrators to assign restrictions or privileges to users. 2. The system must be able to associate permissions with a user using one or more of the following access controls:

	1) user-based (access rights assigned to each user); 2) role-based (users are grouped and access rights assigned to these groups); or 3) context-based (role-based with additional access rights assigned or restricted based on the context of the transaction such as time-of-day, workstation location, Emergency-mode, etc.)
Pre-conditions	➢ User must be register with the system. ➢ SA should be authorized and has access privileges
Course of event	1. SA clicks in user search form menu item. 2. System displays user search form. 3. SA applies filters to search user according to selection criteria and clicks view patient status. 4. System displays edit/assign user rights form. 5. SA gives privileges to user clicks save button. 6. System saves information.
Alternate course of event	

4.4.17. Authentication

Introduction/purpose of feature

This feature provides the ability to authenticate any access to the system using different

Associated functional requirements and stimulus/response sequence – use case

Use Case id	30
Use Case	**Authenticate a User**
Actor	System Operator
Purpose	Purpose of the use case is to authenticate any access to the system using different methods including password protected access.

Cross Reference

Functional requirements	1. When passwords are used, the system shall support password strength rules that allow for minimum number of characters, and inclusion of alpha-numeric complexity. 2. The system shall enforce a limit of (configurable) consecutive invalid access attempts by a user. 3. When passwords are used, the system shall use either standards-based encryption, e.g., 3DES, AES, or standards-based hashing, e.g., SHA1 to store or transport passwords. 4. When passwords are used, the system shall prevent the reuse of passwords previously used within a specific (configurable) timeframe
Pre-conditions	➢ The person who is authenticated must have an account in the system
Course of event	1. The user clicks the Sign In button 2. Enter the username and password 3. Click the button to get signed in.
Alternate course of event	Show error if username and password is not correct and ask user to retry and re-enter the username and password.

4.4.18. Documentation

Introduction/purpose of feature

The system shall include documentation to guide users to install and use the system services.

Associated functional requirements and stimulus/response sequence – use case

Use Case id 31

Use Case	**Provide Documentation**
Actor	System Operator
Purpose	Purpose of the use case is to provide documentation to guide users to install and use the system services.
Cross Reference	
Functional requirements	1. The system shall include documentation that describes the patch (hot-fix) handling process the vendor will use for EHR, operating system and underlying tools 2. The system shall include documented procedures for product installation, start-up and/or connection. 3. If the system includes hardware, the system shall include documentation that covers the expected physical environment necessary for proper secure and reliable operation of the system including: electrical, HVAC, sterilization, and work area. 4. The system shall include documentation that describes the steps needed to confirm that the system installation was properly completed and that the system is operational. 5. The system shall include documentation available to the customer that provides guidelines for configuration and use of the security controls necessary to support secure and reliable operation of the system, including but not limited to: creation, modification, and deactivation of user accounts, management of roles, reset of passwords, configuration of password constraints, and audit logs.

Pre-conditions	➢ The person who uses documentation should be registered as a member of the system ➢ He should be authenticated ➢ He should be authorized for viewing the documents
Course of event	1. The user clicks the index page for documentation 2. Then selects the domain from menu for which he want to get knowledge
Alternate course of event	

4.4.19. Backup/recovery

Introduction/purpose of feature

This feature is to generate a backup copy of the application data, security credentials, and log/audit files.

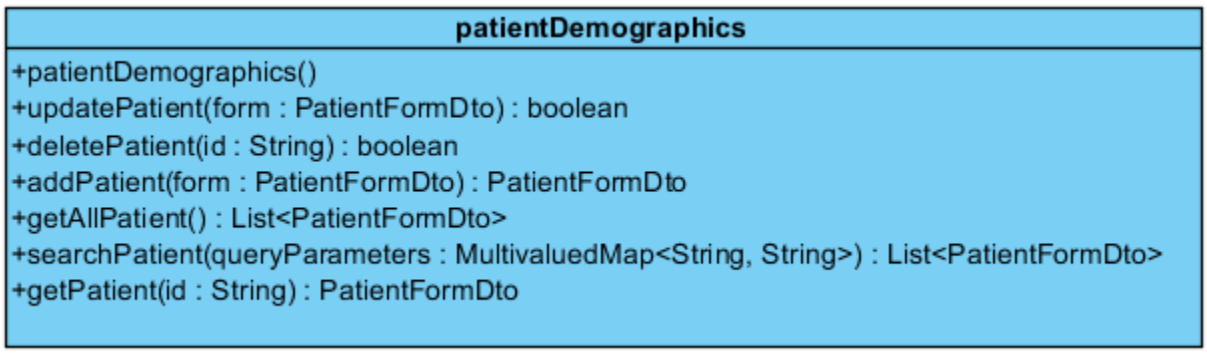
Associated functional requirements and stimulus/response sequence – use case

Use Case id	32
Use Case	**Provide Documentation**
Actor	System Operator
Purpose	Purpose of the use case is to provide documentation to guide users to install and use the system services.
Cross Reference	
Functional requirements	1. The system restore functionality shall result in a fully operational and secure state. This state shall include the restoration of the application data, security credentials, and log/audit files to their previous state.

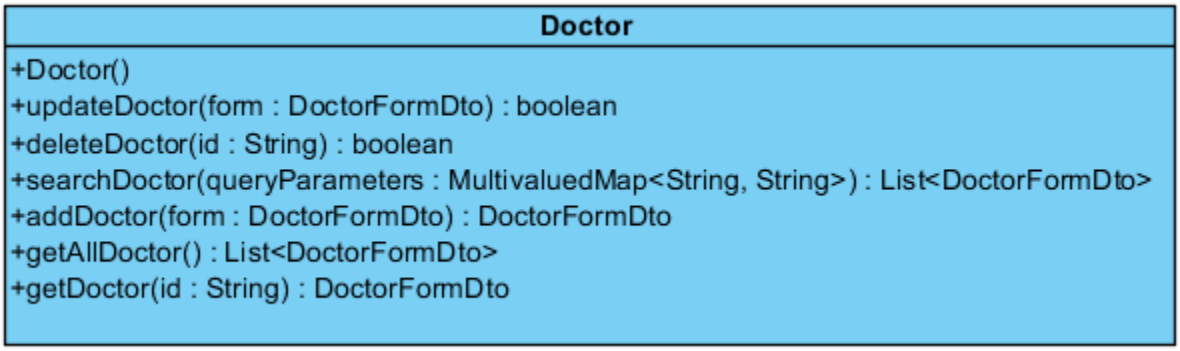
Non-Functional requirements	1. If the system claims to be available 24x7 then the system shall have ability to run a backup concurrently with the operation of the application.
Pre-conditions	➢ The person who uses recovery functionality should be registered as a member of the system ➢ He should be authenticated ➢ He should be authorized for making the recovery
Course of event	1. User clicks the option of recovery from the menu 2. Then selects the area of system of which recovery is to be made. 3. Then start recovery and a status bar will be shown to show the progress of recovery
Alternate course of event	In case of failure, an error dialogue box will be shown and ask the user to retry and start the recovery process again

4.5.Class Diagrams

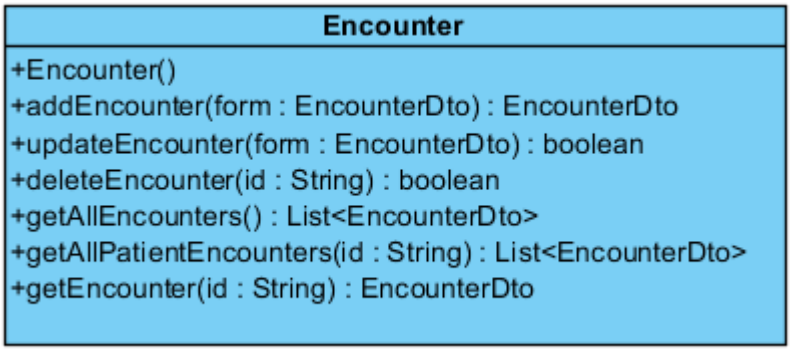
Patient



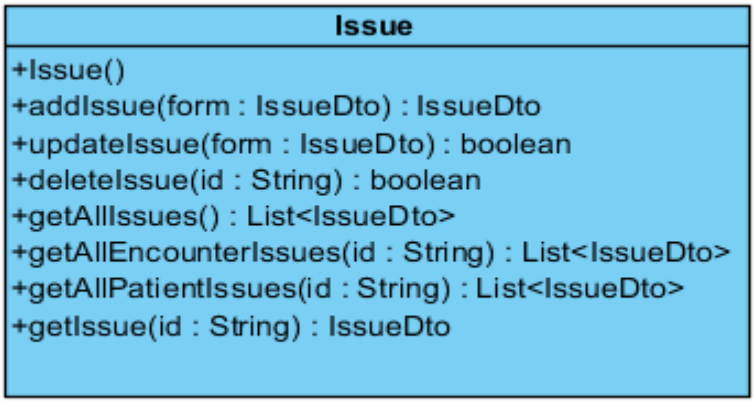
Doctor



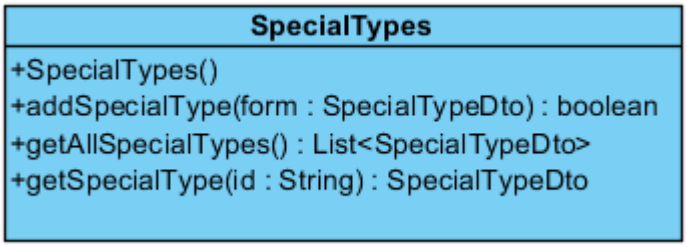
Encounter



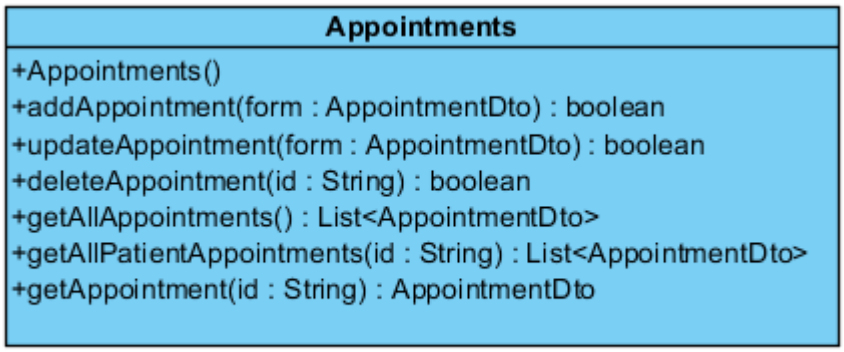
Issue



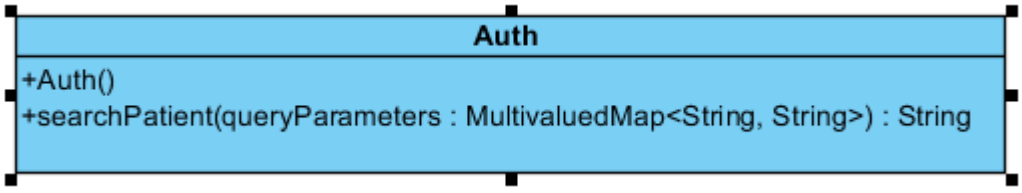
Special Type



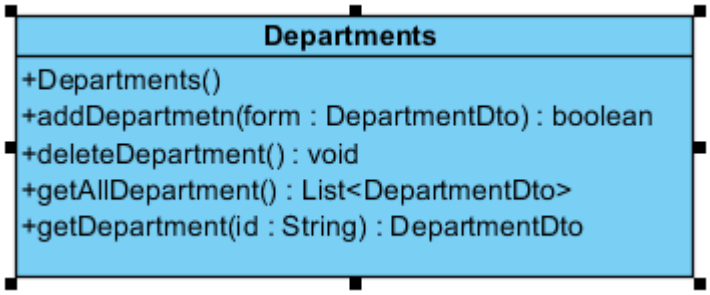
Appointment



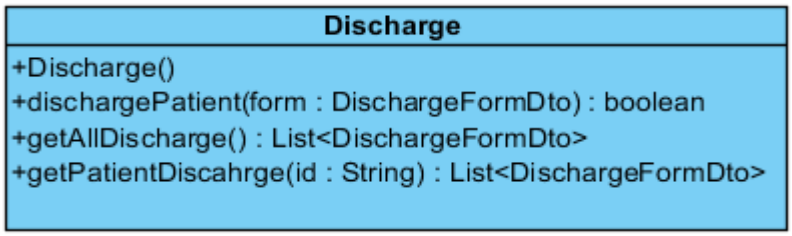
Auth



Department


Departments
+Departments()
+addDepartmetn(form : DepartmentDto) : boolean
+deleteDepartment() : void
+getAllDepartment() : List<DepartmentDto>
+getDepartment(id : String) : DepartmentDto

Discharge


Discharge
+Discharge()
+dischargePatient(form : DischargeFormDto) : boolean
+getAllDischarge() : List<DischargeFormDto>
+getPatientDiscahrge(id : String) : List<DischargeFormDto>

5. IMPLEMENTATION *Chapter 05*

5.1. System Components:

The three main components of EMR are:

- Client tier
  - Android client
  - Web Based client
- Application Tier
  - Restul web services
- Database Tier
  - Triple DB (JENA)

We will shortly discuss the implementation of three components with respect to available resources and guidance.

5.2. Client Tier

5.2.1. Web based Client

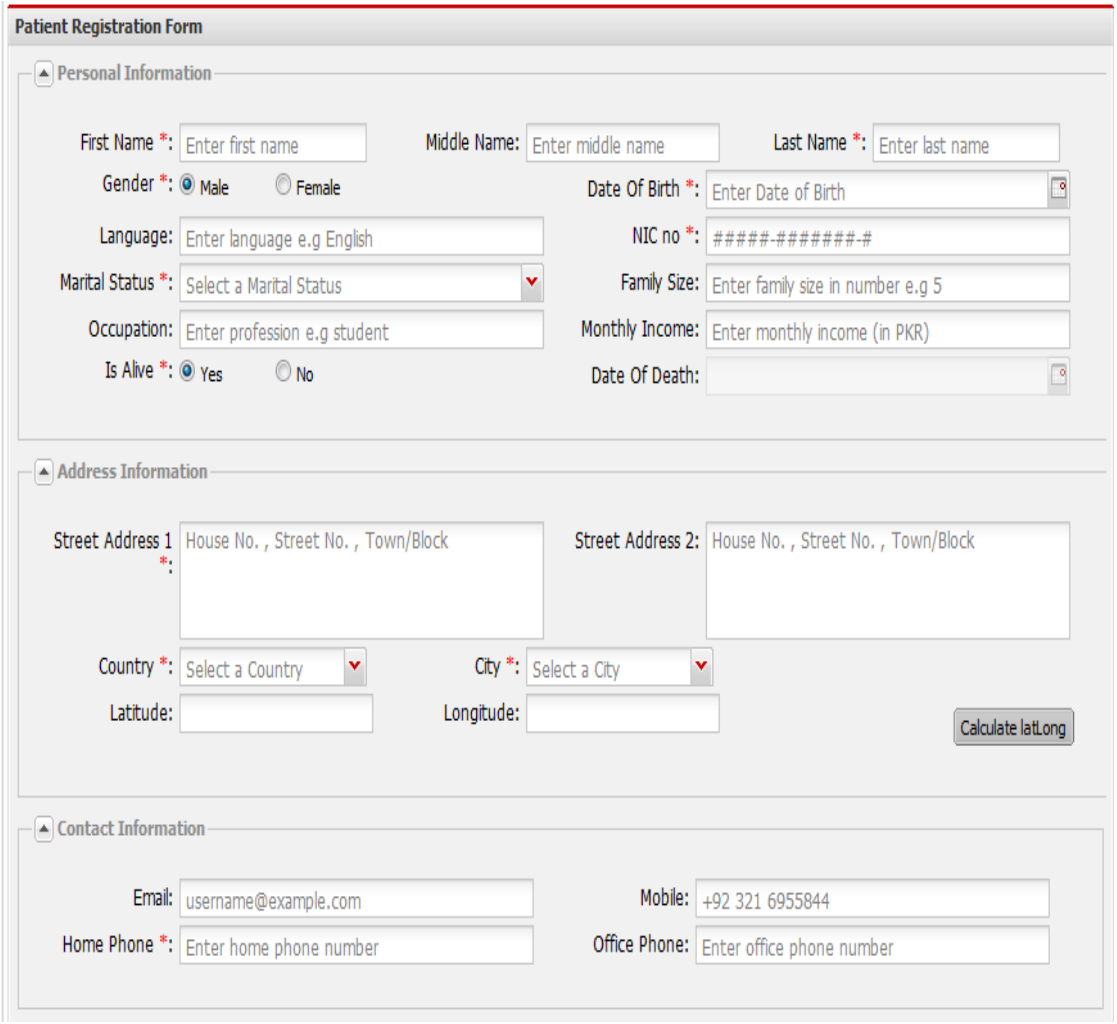
For our web based client we used GWT. Google Web Toolkit (GWT) is a development toolkit for building and optimizing complex browser-based applications. Its goal is to enable productive development of high-performance web applications without the developer having to be an expert in browser quirks, XMLHttpRequest, and JavaScript. GWT is used by many products at Google, including Google Wave and the new version of AdWords. It's open source, completely free, and used by thousands of developers around the world.

We provide different type of functionality: we will discuss each of them one by one. First we will discuss about patient demographics.

Add Patient

First we make form for patient registration in GWT. The snapshot of our patient registration form is given below:

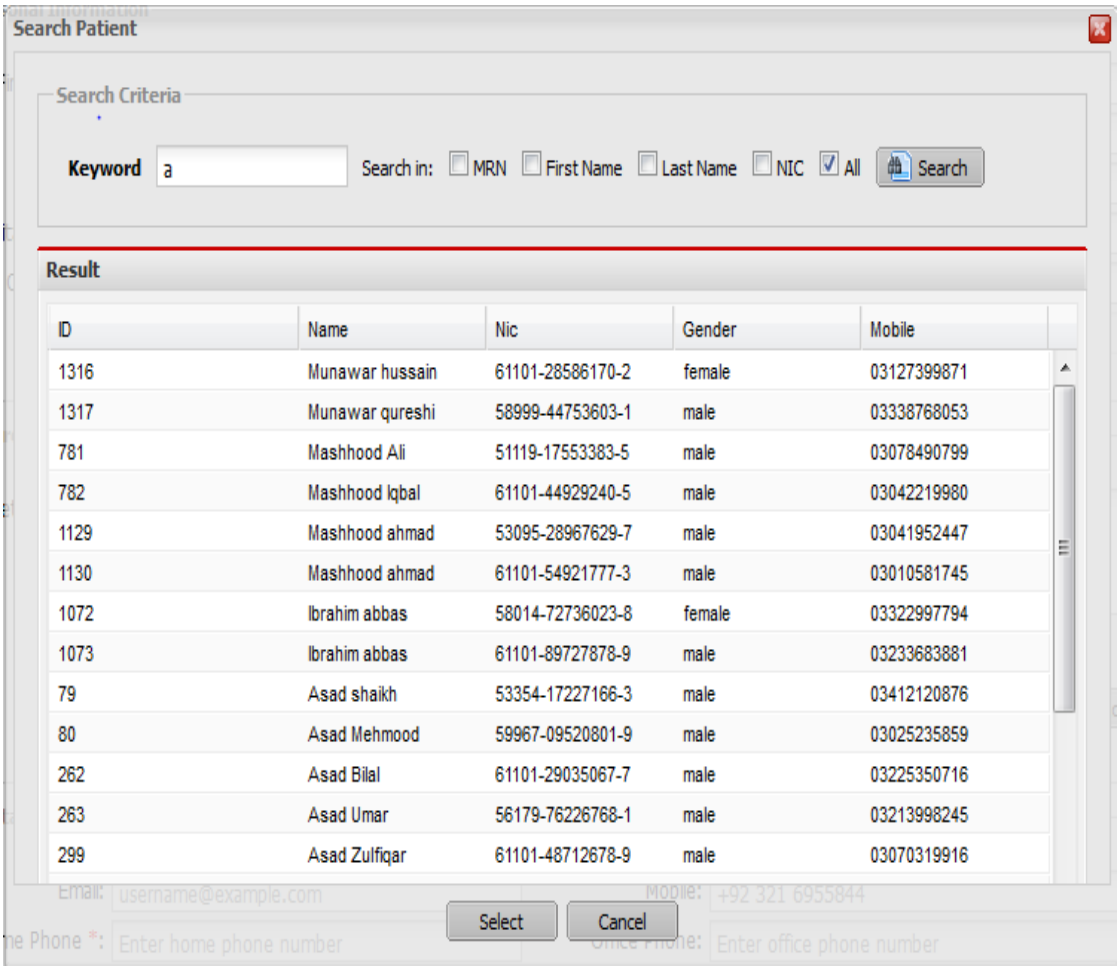
Figure 5-1: Patient Demographics Form



We make a DTO (Data transmitting object) for every form. When user fills this form and submits this form, every field of this form is set into the DTO by calling the DTO setter function. After populating the DTO object. This DTO object to pass to Google GSON API. The Google GSON API converts that java object into serialized json object. And after converting java object into json, this json is passed to our application tier using a HTTP POST request. Our application tier get the json object from the request and convert by into java object using Google GSON API. After converting into java object, data is sent to database layer i.e. JENA. The Jena saves the data into database.

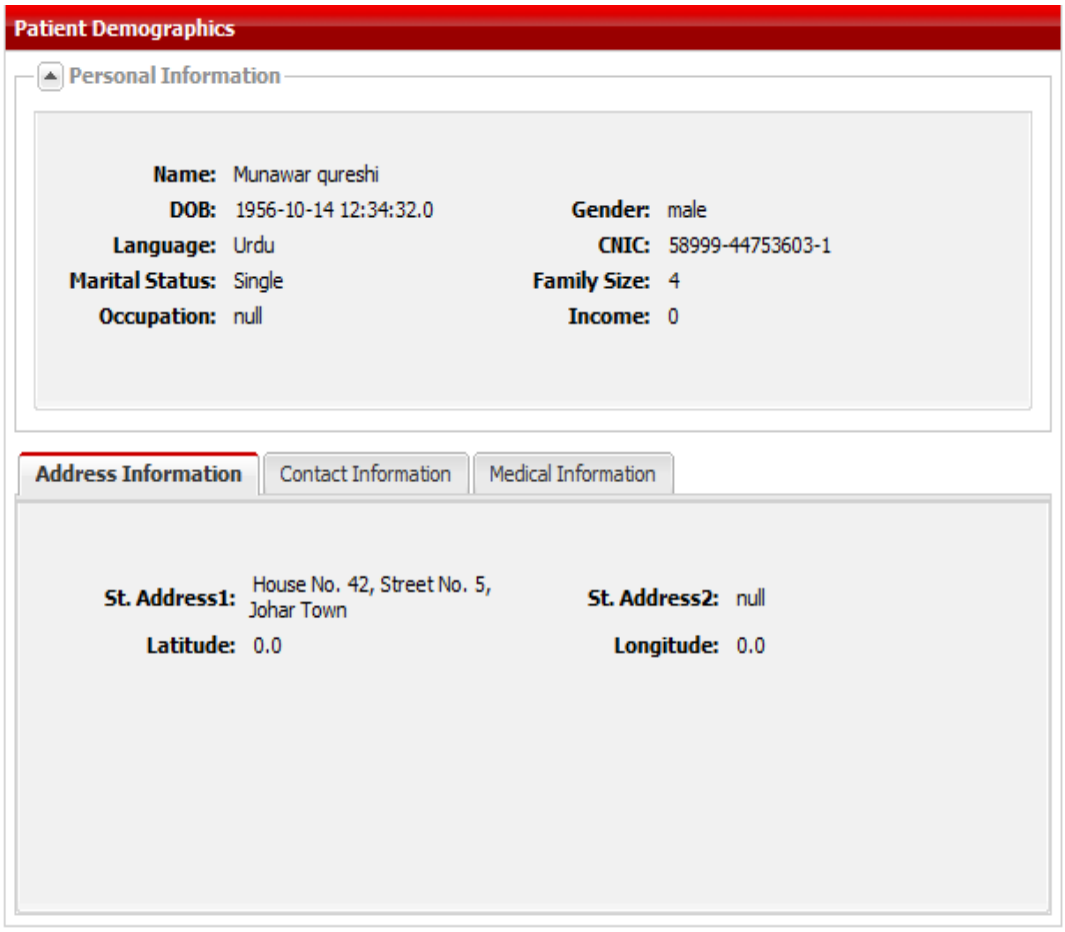
Search Patient

Figure 5-2: Patient Search Form



For search patient user enter some keyword and define his search criteria like search from MRN, firstname, lastname or NIC. After entering his search and selecting his criteria the user press search button. Now the search query is passed to application tier as a HTTP Request and application tier get result from database layer and passed back to our client. The client shows the result in a grid. After selecting a user form search the user information is shown in a table like the fig. below:

Figure 5-3: Patient Demographics Info

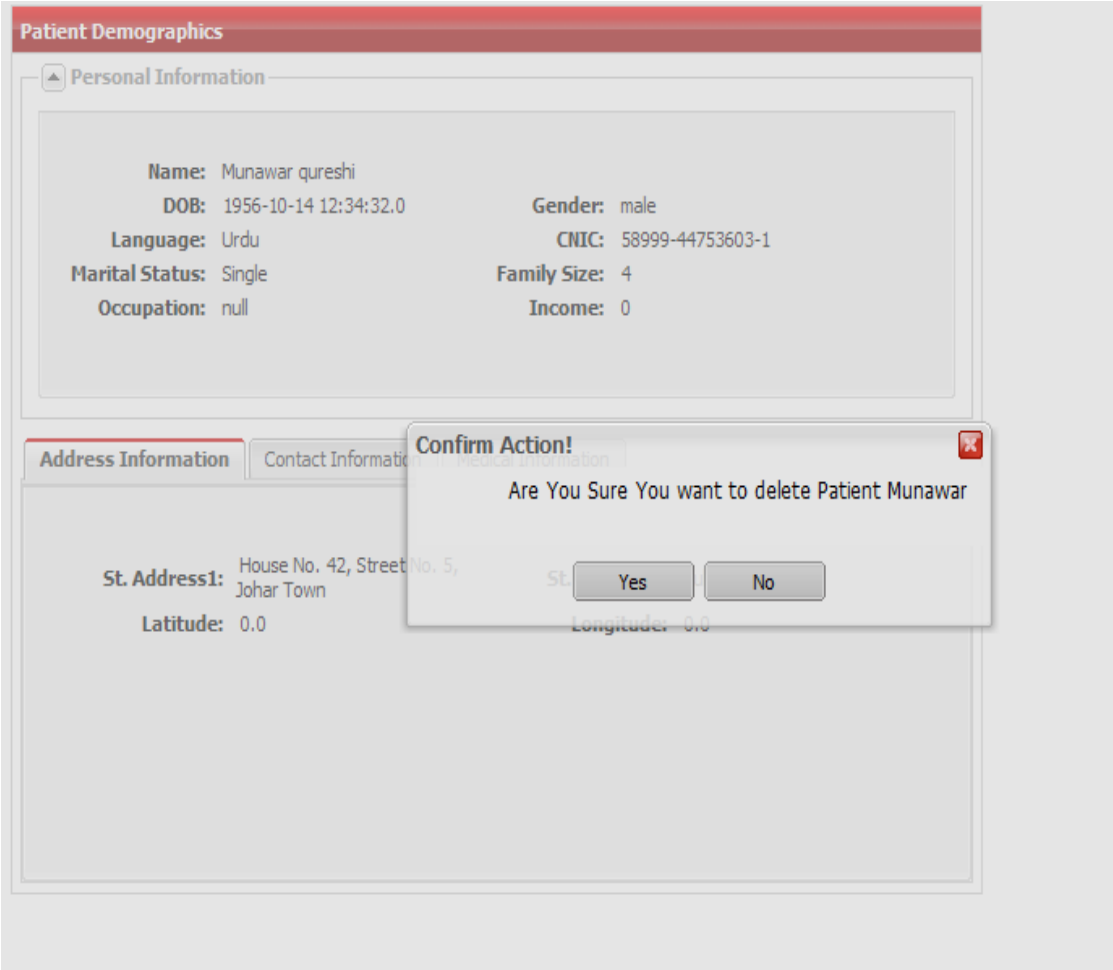


When the user selects a patient from the list the patient which is selected is become active. Now user can edit or delete the active patient.

Delete Patient

For deleting a patient user select delete menu and application confirm from user before deleting the active patient as shown in fig.

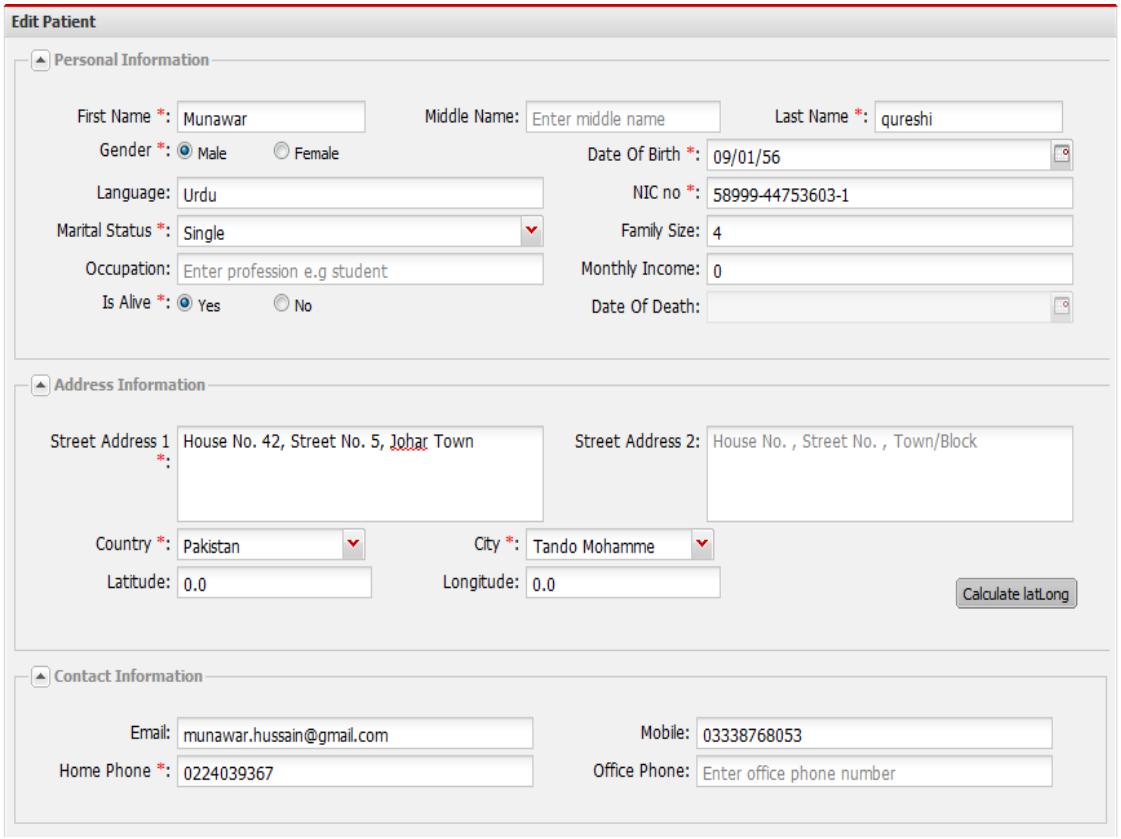
Figure 5-4: Delete Patient



Edit Patient

When user clicks on edit the active patient the form is populated with the previous value. Now user can edit the patient info and click update button to reflect the changes.

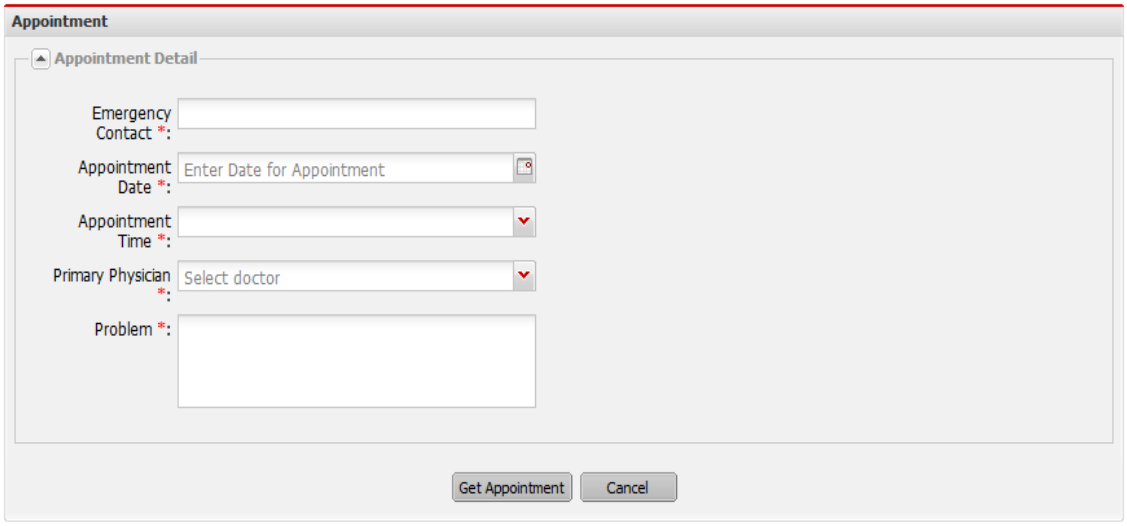
Figure 5-5: Edit Patient



Appointment

Using our EMR system user can get appointment from the doctor.

Figure 5-6: New Appointment

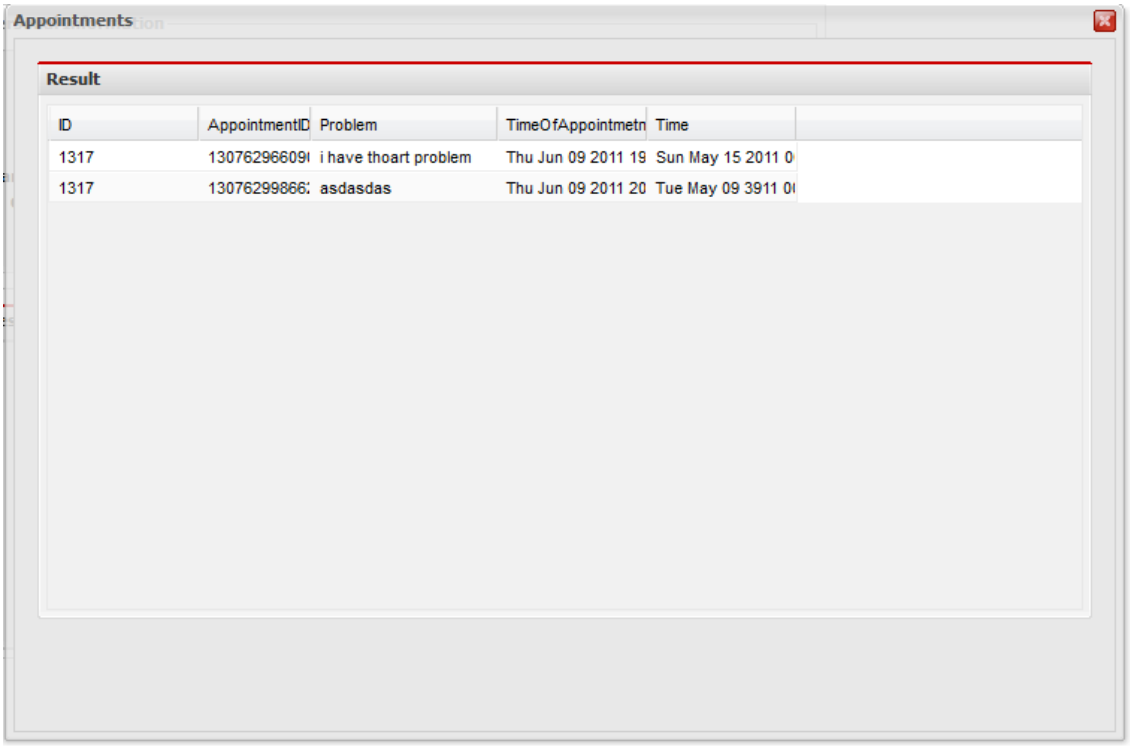


For getting appointment user fill the appointment form. After filling the form user click on get appointment .And the user appointment is made to that doctor.

Edit Appointment

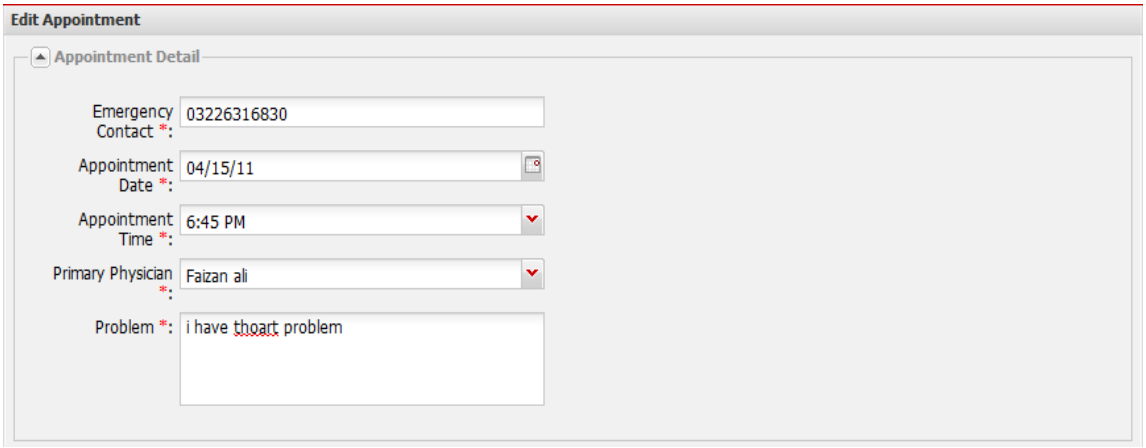
User can edit the appointments. When user clicks on edit appointment menu, list of all his appointment is shown to him in a grid.

Figure 5-7: Appointments



After selecting the appointment the edit appointment form is shown. Now user can update appointment data.

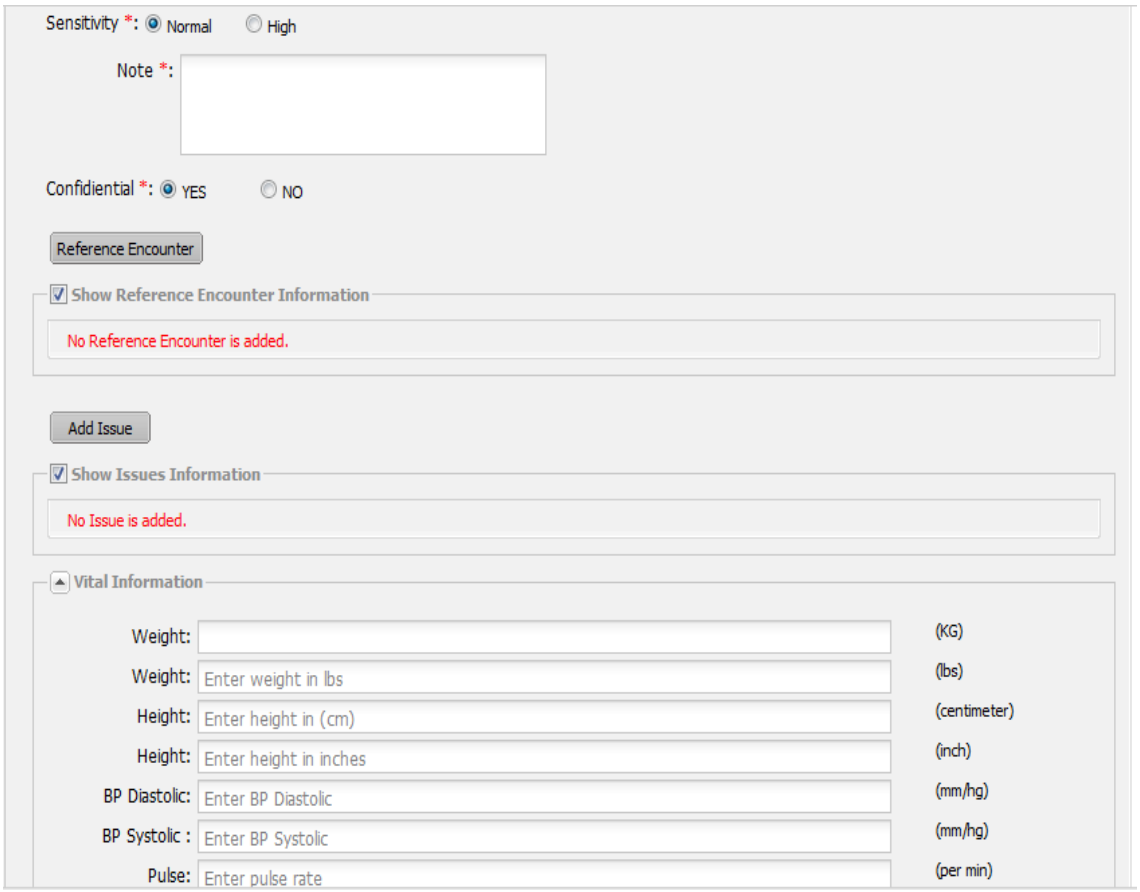
Figure 5-8: Edit Appointment



Add Encounter

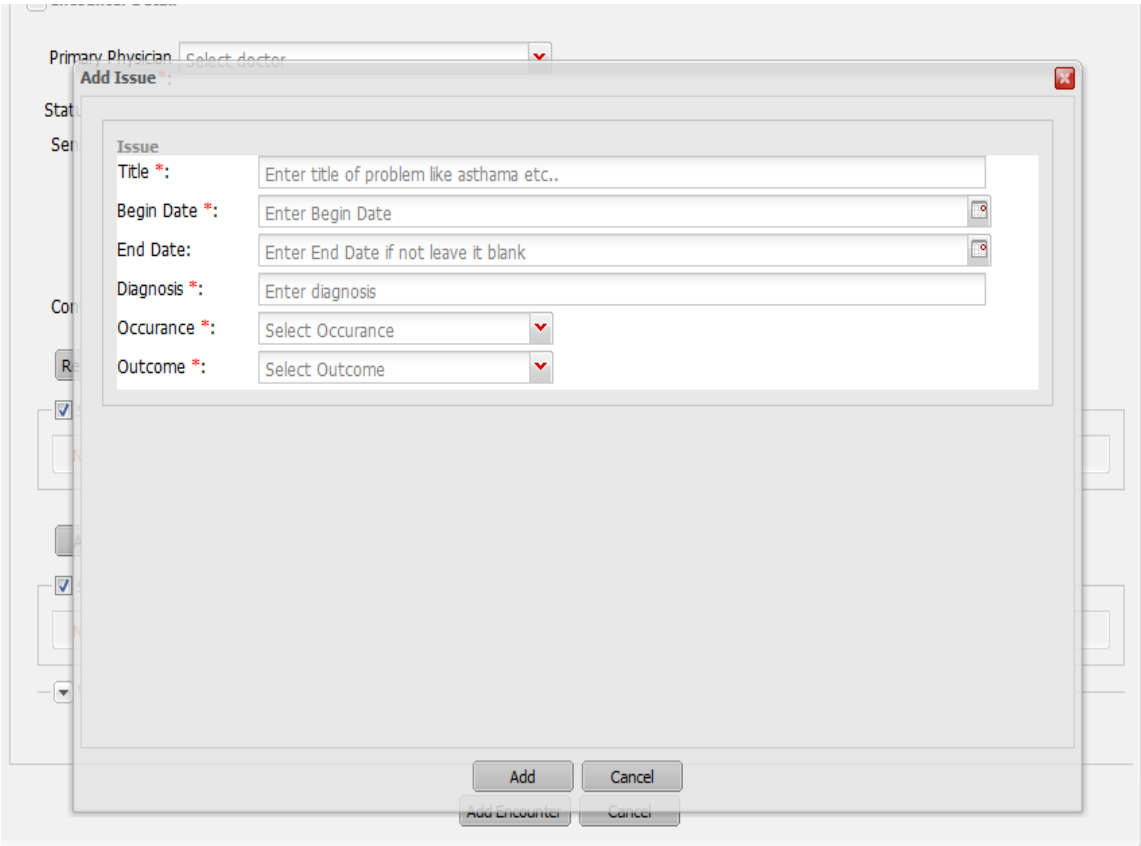
Data entry operator can add patient encounter. For this purpose he opens the encounter form and fills the necessary data.

Figure 5-9: Add Encounter



An encounter can have multiple issues so data entry operator can add multiple issues for an encounter like the fig shown below.

Figure 5-10: Add Issue



An encounter can be a reference encounter of a previous encounter so user can add reference encounter. After providing the information data entry operator click on add encounter button and the same procedure is happen. The data is converted into DTO. The DTO is converted into JSON and sent to application tier as HTTP Request.

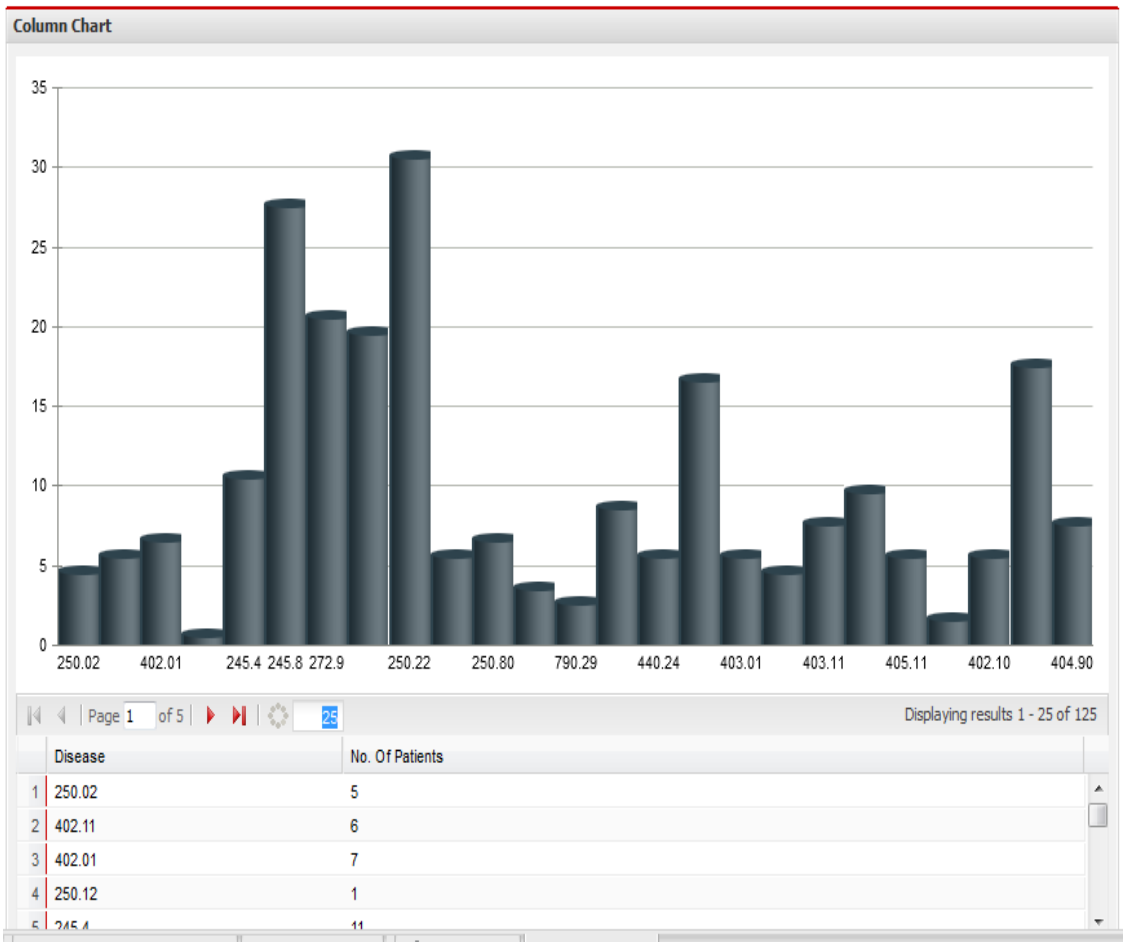
Analysis

User can do different type of analysis on the data. Some of them are:

- Most Popular Disease
- Department vs. Encounter
- Doctor vs. Encounter
- Disease vs. Patient

The chart of Disease vs. Patient is shown in fig. below:

Figure 5-11: Disease vs. Patient Chart



5.3. Application Tier

For application tier we use restful Web service. A Web service is a method of communication between two electronic devices over a network. The term Web services describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet protocol backbone. XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing what services are available. Used primarily as a means for businesses to communicate with each other

and with clients, Web services allow organizations to communicate data without intimate knowledge of each other's IT systems behind the firewall.

List of implemented Web services in our system are discussed below:

- Patient
- Doctor
- Department
- Issue
- Encounter
- Appointment
- Analysis

1. Patient Service

GET	: Get All Patient Data
POST	: Add a Patient to DB
PUT	: Edit Patient Info
DELETE /{ID}	: Delete a specific patient from DB
GET/{ID}	: Get the info of a patient using the given ID

2. Doctor Service

GET	: Get All Doctor Data
POST	: Add a Doctor to DB
PUT	: Edit Doctor Info
DELETE /{ID}	: Delete a specific Doctor from DB
GET/{ID}	: Get the info of a Doctor using the given ID

3. Department Service

GET	: Get All Department Data
POST	: Add a Department to DB
DELETE /{ID}	: Delete a specific Department from DB

GET/{ID} : Get the info of a Department using the given ID

4. Issue Service

GET : Get all issue Data
POST : Add an Issue to DB
DELETE /{ID} : Delete a specific Issue from DB
GET/{ID} : Get the info of an Issue using the given ID

5. Encounter Service

GET : Get all encounter Data
POST : Add an encounter to DB
DELETE /{ID} : Delete a specific encounter from DB
GET/{ID} : Get the info of an encounter using the given ID

6. Appointment Service

GET : Get All Appointment Data
POST : Add an Appointment to DB
DELETE /{ID} : Delete a specific Appointment from DB
GET/ {ID} : Get the info of an Appointment using the given ID

5.4. Database Tier

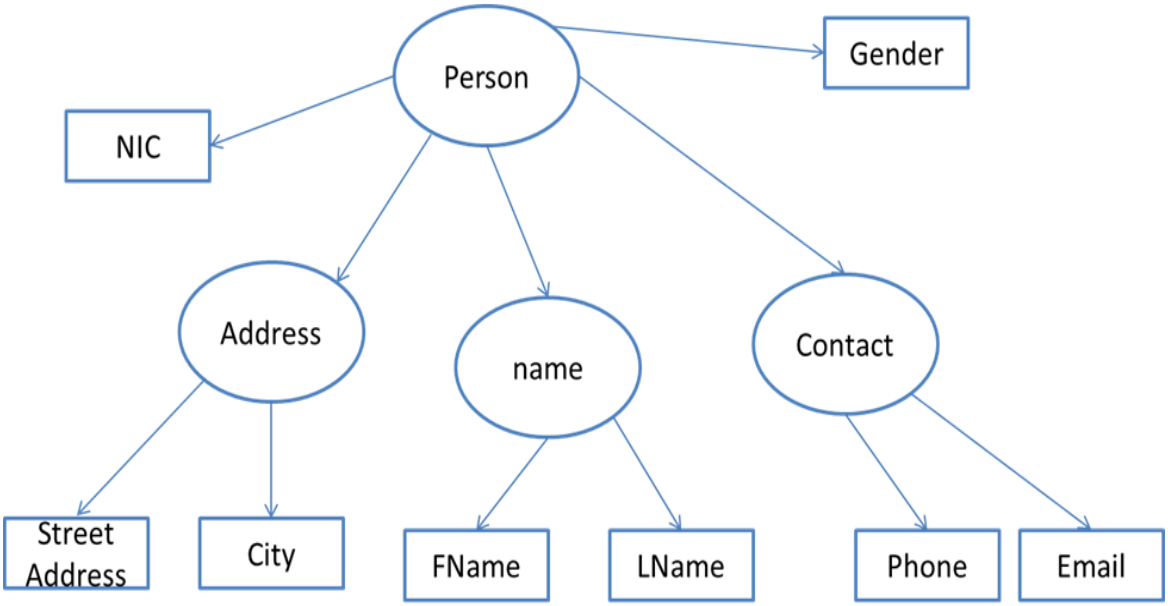
For database we use Jena API. Jena is a leading Semantic Web toolkit for Java programmers. The heart of the Semantic Web recommendations is the RDF Graph as a universal data structure. An RDF graph is simply a set of triples (S, P, O), where P names a binary predicate over (S, O). Jena similarly has the Graph as its core interface around which the other components are built. OWL Full is a semantic extension of RDF; Jena's ontology support is targeted at OWL Full. Future Semantic Web standardization is likely to include work on query languages, and possibly Web APIs for the Semantic Web.

The main contribution of Jena is the rich Model API for manipulating RDF graphs. Around this API, Jena1 provided various tools, including I/O modules for: RDF/XML, N3, and N-triple; and the query language RDQL. Using the API the user can choose to store RDF graphs in memory or in persistent stores. Jena provided an additional API for manipulating DAML+OIL.

Patient RDF graph is shown below:

Patient

Figure 5-12: Patient RDF graph



6. TESTING *Chapter 06*

6.1. Unit testing

Unit testing is used to test a single unit which is a smallest testable part of an application. Generally we do unit testing before their integration in an application. In an application mostly modules are dependent on the output of other modules. If one module is not working properly or giving wrong output, then this bug can propagate into whole system. That's why it is good approach to test each unit before its integration.

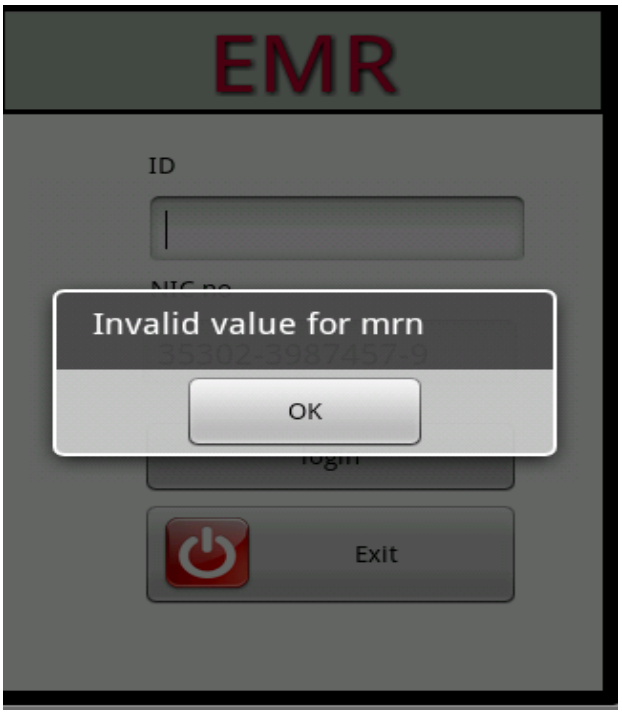
As we have deployed our services on cloud and access them from web and android based clients. To test that our services are working properly , we had tested them from web and android.

Here are few screen shots of web testing and android testing.

6.1.1. Test Case 1 (Login with invalid Id):

Let the user enter an invalid id and try to login

Figure 6-1: Android Login Page (Invalid MRN)

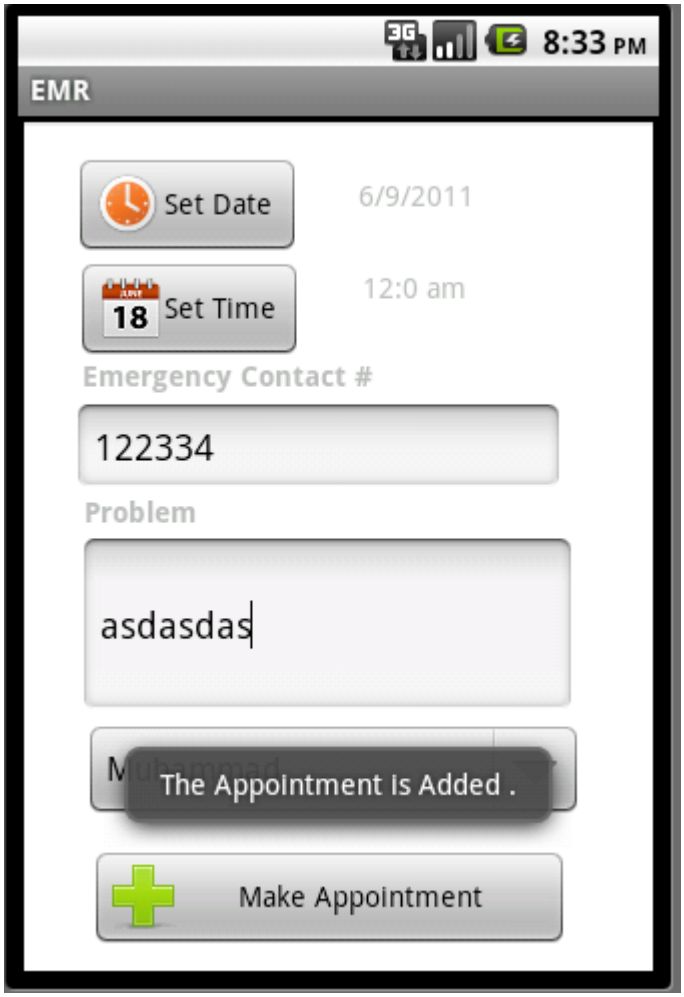


6.1.2. Test Case 2 (Make an appointment):

Let a user enter all required information for an appointment and press make appointment button

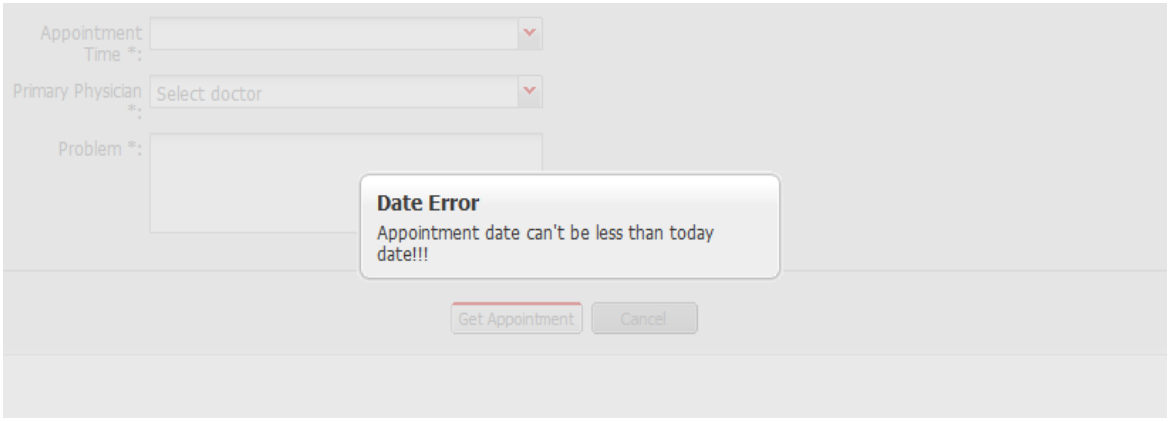
.

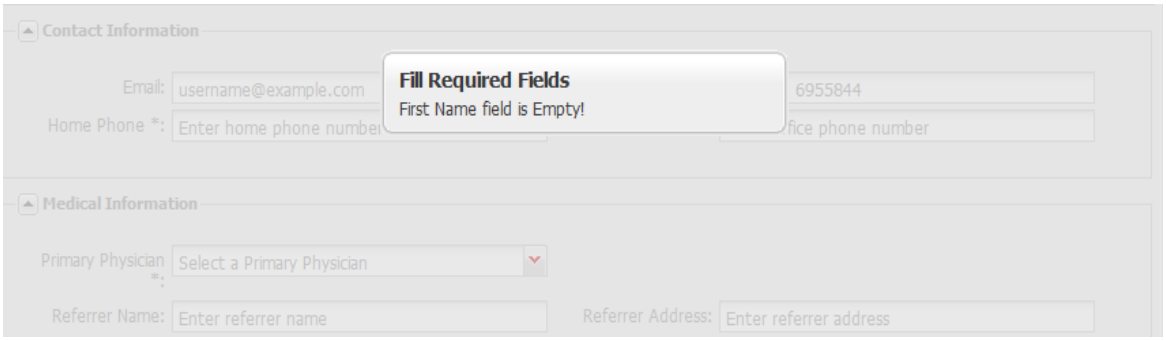
Figure 6-2: Android Make Appointment



6.1.3. Test Case3 (Add a patient using web client):

Let we add a patient and some required fields are empty.





6.2. Integrated Testing:

This testing is used to check that whole application is working properly after integrating all the units. After unit testing of all the units and their integration in our application, integrated testing was done. We had tested all parts and their functionality on device. We had found some bugs and we had fixed them. For example Gson (a Google library) was not working properly on android device and we were getting some NULL values. We resolved this issue. We also tested our web services using our web client.

7. BIBLIOGRAPHY

[1] Introduction to Adoption of Electronic Health Records. (2008, 01 22).

[2] OpenEMR Home Page. (2009). Retrieved 08 11, 2010, from http://www.oemr.org/

[3] CCHIT Certified 2011 Ambulatory EHR. (2010, 04 07).

[4] EMR in Ambulances. (2011, 05 05). Retrieved 06 01, 2011, from http://emergencymedicalparamedic.com/emr-for-paramedics

[5] A forum for discussion of ehr, emr implementation. (n.d.). Retrieved 11 25, 2010, from http://www.emrandehr.com/

[6] CCHIT Certified Ambulatory EHR 7. (n.d.). Retrieved 10 2, 2010, from http://www.cchit.org/certify/2011/cchit-certified-2011-ambulatory-ehrc

[7] Diseases and Injuries. (n.d.). Retrieved 05 16, 2011, from Wikipedia: http://icd9cm.chrisendres.com/

[8] Electronic medical record. (n.d.). Retrieved 10 25, 2010, from Wikipedia: http://en.wikipedia.org/wiki/Electronic_medical_record

[9] Electronic Medical Records, Electronic Health Records. (n.d.). Retrieved 06 2011, 01, from http://www.openclinical.org/emr.html

[10] EMR and EHR System Implementation Best Practices. (n.d.). Retrieved 11 20, 2010, from http://ezinearticles.com/?EMR-and-EHR-System-Implementation-Best-Practices---Part-4---Literature-Review&id=3591342

[11] Garets, D., & Davis, M. (2005, 10). Electronic Patient Records: EMRs AND EHRs. Retrieved 12 23, 2010, from http://www.providersedge.com/ehdocs/ehr_articles/Electronic_Patient_Records-EMRs_and_EHRs.pdf

[12] Health Information Technology. (n.d.). Retrieved 11 25, 2010, from www.allhcealth.org/Publications/Health_information_technology/health_information_technology_toolkit.asp#Keyfacts

[13] List of open source healthcare software. (n.d.). Retrieved 11 05, 2010, from Wikipedia: http://en.wikipedia.org/wiki/List_of_open_source_healthcare_software

[14] McDonald, C. e. (1992). The Regenstrief Medical Record System: 20 years of experience in hospitals, clinics, and neighborhood health centers. In MD Computing (pp. 206-17).

[15] Miller, B. (n.d.). OpenEMR 3.2.0 Virtual Appliance. Retrieved 11 2, 2010, from http://www.openmedsoftware.org/wiki/OpenEMR_3.2.0_Virtual_Appliance

[16] Miller, B. (n.d.). OpenEMR Version 3.2.0 Demo. Retrieved 11 03, 2010, from http://www.openmedsoftware.org/wiki/OpenEMR_Version_3.2.0_Demo

[17] ONC-ATCB Certified 2011/2012. (n.d.). Retrieved 11 2010, 26, from http://www.cchit.org/get_certified/ONC-ATCB-Certified-2011-2012

[18] OPENEMR Appliance pro Electronic Medical Record EMR. (n.d.). Retrieved 02 24, 2011, from http://cgi.ebay.com/OPENEMR-APPLIANCE-PRO-ELECTRONIC-MEDICAL-RECORD-EMR-/160512406031

[19] Simhachalam, M. (n.d.). Securing REST Web Services With OAuth. Retrieved 11 23, 2010, from http://developers.sun.com/identity/reference/techart/restwebservices.html

[20] Vibha, A., Paul, B., Gilbert, L., Marc, R., & Stephen, D. (n.d.). Child Health Improvement through Computer Automation: The CHICA System.

[21] Web service. (n.d.). Retrieved 12 23, 2010, from Wikipedia: http://en.wikipedia.org/wiki/Web_service