**1. INTRODUCTION**

**1.1 Introduction to e-Medical Service Management**

What is e-Medical Service Management?

e-Medical Service Management is a web application aimed at revolutionizing the field of medicine. It is targeted to change the way the medical services are provided at Hospitals. So this application is planned to build a virtual home to provide the required services online to all the individuals irrespective of their location. For proper management of the provided medical services this system is built.

How e-Medical Service Management works?

Registration module will take care of maintaining the complete user profile and also the registration procedure. Provision for taking appointments for patients. Viewing their previous records, medical documents and various reports. Also they can get the details of expenses to be bared. Doctors will be able to view the history of the patients and provide appointments and e-prescriptions. Manager can go through the requested appointments and allocate them according to the requested doctor’s availability. In this system the manager acts like a MediAssist. System tracks all the medical errors and give the complete information to patient. In case any lapse from the doctor side the patient can escalate the issue. Issue tracker is provided by the system which is taken care by the manager. Issues will be received by the manager and the same will be passed to the doctor. Doctors can give clarifications through the system. The system provides a chat window and forum to discuss the issue. All the stakeholders will be able to access the chat whenever their appointment is scheduled. Online help also provided through this system. It has a provision to archive the complete data and generate various kinds of reports. A copy of the e-prescription generated by the doctor for a given patient will also be sent to the e-pharmacy that is in the system. The pharmacist can go through the medicines that are prescribed for the patient and can deliver them to the patient’s location accordingly. The prescription viewer allows the pharmacist to view the details of the patient and the doctor and the medicines and their recommended dosage. Pharmacist can be able to check the medicines availability through the system in order to deliver the medicines. There is a delivery report system which helps the pharmacist to check the delivery status of various medicine delivery requests and make the deliveries accordingly. Through this system the Lab Assistant will be able to login and upload the reports for a particular patient so that the patient and the doctor can view their medical reports. Doctors will be able to go through the reports online so that they will be able to diagnose the patient’s health condition while counseling the patient online and prescribe medicines accordingly.

**2. SYSTEM ANALYSIS**

**2.1 EXISTING SYSTEM:**

The existing system poses the drawback of unavailability of doctors at remote places. Limited time of services makes it unavailable at all times. Lack of advanced equipment for diagnosis and lack of attention in managing the patient data.

**2.2 PROPOSED SYSTEM:**

The proposed system will overcome all these drawbacks. The system will have five kinds of users – Patient, Doctor, Medi Assist, Pharmacist, Lab Assistant. The entry point of the system is that the patients and the doctors gets registered into the system. Existing users can directly login to the system and new users should get registered. Patients can take appointments for a particular doctor based upon their requirement. The specialization of the doctors will be provided. A chat window will be enabled to the users when their time of appointment is met. In this way the proposed system can be made available to the user at all times and also helps in providing continuous medical services to the users. There is a provision for purchasing medicines from the system and also uploading the medical lab reports. Pharmacist will be given a copy of the e-prescription generated for the patient and the pharmacist can make the delivery to the patient depending on their location. Lab assistant will be able to upload medical reports of the patients that have gone through diagnosis through the system. This helps the doctor to view the medical reports of the particular patient while counseling him/her. The Pharmacist will be able to view the delivery reports of the medicines and also check for availability of medicines in the e-Pharmacy. These e-pharmacy and e-Labs are the new features that are absent in the existing system.

**2.3 Feasibility study:**

There are aspects in the feasibility study portion of the preliminary investigation:

* Technology and system Feasibility
* Operation Feasibility
* Economic Feasibility

**Technology and System Feasibility:**

The assessment is based on an outline design of system requirements in terms of Input, Processes, Output, Fields, Programs, and Procedures. This can be quantified in terms of volumes of data, trends, frequency of updating, etc. in order to estimate whether the new system will perform adequately or not this means that feasibility is the study of the based in outline. The technical issue usually raised during the feasibility stage of the investigation includes the following:

Does the necessary technology exist to do what is suggested?

Do the proposed equipment have the technical capacity to hold the data required to use the new system?

Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?

Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security?

**Operational feasibility:**

It is a measure of how well a proposed system solves the problems, and takes advantages of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development Some of the important issues raised are to test the operational feasibility of a project includes the following

Is there sufficient support for the management from the users?

Will the system be used and work properly if it is being developed and implemented?

Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

**Economic feasibility:**

Economic analysis is the most frequently used method for evaluating the effectiveness of a new system. More commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. An entrepreneur must accurately weigh the cost versus benefits before taking an action.

**Time Based**:

Time management is most important when new application is develop and it effects cost of the developed system.

**Cost Based**:

No special investment is needed to manage the tool. No specific training is required for employees to use the tool. Investment requires only once at the time of installation. The software used in this project is freeware so the cost of developing the tool is minimal

**2.4 SOFTWARE REQUIREMENT SPECIFICATION**

**Functional Requirements:**

Registration module will take care of maintaining the complete user profile and also the registration procedure. Provision for taking online appointment for patients. Viewing their previous records, medical documents and various lab reports. Also they can get the details of expenses to be bared. Doctors should view the history of the patients and provide appointment and e-prescriptions. Manager can look at the appointments, perform day wise activity and calculate his commission. He is like a mediator(Ex: MediAssist) System should track all the medical errors and give the complete information to patient. In case any lapses from the doctor side the patient can escalate the issue. The same will be received by the admin and he will pass the same to doctor. Doctor will give the clarification through the system. Providing a chat window and forum to discuss the issues. All the stakeholders should access this window. Online help for patients. If required the local language support is to be given through the system. System should have a provision to archive the complete data and generate various kinds of reports. E-Pharmacy module will take care of checking the prescription details and delivering the required medicines to the patient’s location and also checking the delivery status and medicine availability. E-Labs module will be managed by a lab assistant where in he/she will be able to upload the medical reports of a patient so that the patient and the doctor can view the reports and will be helpful to prescribe medicines.

**Non-Functional Requirements:**

Security: Role based secure access of confidential data

Audit Trail: The system should log the activities of the users

Error logging: The system should log the errors

Multi language Support: NA

Performance: The system should be able to get a better performance at the peak time.

Scalability: Should be high and cater the needs 2000 users at a time

Availability / reliability: The system should be available 24 x 7

Data migration: NA

Data Retention: NA

Legal/ regulatory requirements: NA

Flexibility : Should be flexible and extendable.

**HARDWARE AND SOFTWARE REQUIREMENTS:**

**Hardware requirements**

Processor/RAM/HDD : Pentium/4 GB/75 GB

Web server : Apache, Tomcat 6.0

Database Server : Oracle 10g

**Software requirements**

OS for Web server : Windows

OS for Database Server : Linux

DBMS : Oracle

Third Party S/Ws : NIL

Technologies : J2EE, HTML 4.0, Struts 2.3, JPA, Toplink 1.2 Framework

**Development Environment Requirements**

IDE : Eclipse

Processor/RAM/HDD : Pentium/4 GB/75 GB

**3. SOFTWARE SPECIFICATION**

**3.1Introduction to Struts2.0 Framework**

A framework tries to automate the common tasks and provides a platform for the users to build applications quickly.

Struts 2 is based on the OpenSymphonyWeb Works Framework.

Struts 2 framework implements the Model-View-Controller (MVC) design pattern.

In Struts2 the model, view and controller are implemented by the action, result and FilterDispatcher respectively.

The controller's job is to map the user request to appropriate action.

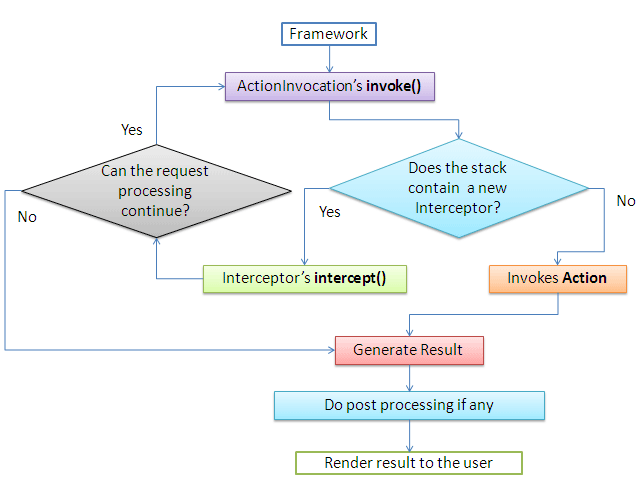
In Struts 2 FilterDispatcher does the job of Controller.

Model contains the data and the business logic.

In Struts 2 the model is implemented by the Action component.

View is the presentation component of the MVC Pattern.

In Struts 2 the view is commonly implemented using JSP, Velocity Template, Freemaker or some other presentation-layer technology.



The controller receives the user request and determine which Struts 2 action to invoke.

The framework creates an instance of this action and associate it with the newly created instance of the ActionInvocation.

In Struts 2 the invocation of action should pass through a series of interceptors as defined in the application's XML file.

The framework calls the ActionInvocation’s invoke() method to start the execution of the action.

Each time the invoke() method is called, ActionInvocation consults its state and executes whichever interceptor comes next.

ActionInvocation hands control over to the interceptor in the stack by calling the interceptors intercept() method.

The intercept() method of the interceptor in turn calls the invoke() method of the ActionInvocation till all the interceptors are invoked, in the end the action itself will be called and the corresponding result will be returned back to the user.

Some interceptor do work before the action is executed and some do work after the action is executed. It's not necessary that it should do something each time it is invoked.

These interceptors are invoke both before and after the action.

First all the interceptors are executed in the order they are defined in the stack.

Then the action is invoked and the result is generated.

Again all the interceptors present in the stack are invoked in the reverse order.

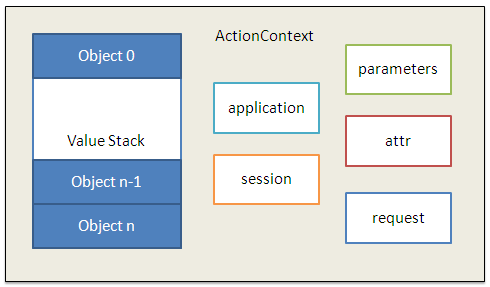
The other important features of Struts 2 are OGNL and ValueStack.

Object-Graph Navigation Language (OGNL) is a powerful expression language that is used to reference and manipulate data on the ValueStack.

OGNL help in data transfer and type conversion.

OGNL expression language provides simplified syntax to reference java objects.

OGNL is used to bind the java-side data properties to the string-based view layer.



In Struts 2 the action resides on the ValueStack which is a part of the ActionContext. ActionContext is a global storage area that holds all the data associated with the processing of a request.

When a request comes the params interceptor helps in moving the request data to the ValueStack.

Now the OGNL does the job of converting the string based form data to their corresponding java types. OGNL does this by using the set of available built-in type converters.

Again when the results are generated the OGNL converts the java types of the property on the ValueStack to the string-based HTML output.

ActionContext is thread local which means that the values stored in the ActionContext are unique per thread, this makes the Struts 2 actions thread safe.

**Why Struts 2.0?**

* **Simplified Design** - Programming the abstract classes instead of interfaces is one of design problem of struts1 framework that has been resolved in the struts 2 framework. Most of the Struts 2 classes are based on interfaces and most of its core interfaces are HTTP independent. Struts 2 Action classes are framework independent and are simplified to look as simple POJOs. Framework components are tried to keep loosely coupled.
* **Simplified Actions** - Actions are simple POJOs.Any java class with execute() method can be used as an Action class. Even we don't need to implement interfaces always. Inversion of Control is introduced while developing the action classes. This make the actions to be neutral to the underlying framework .
* **No more ActionForms** - ActionForms feature is no more known to the struts2 framework. Simple JavaBean flavoured actions are used to put properties directly. No need to use all String properties.
* **Simplified testability** - Struts 2 Actions are HTTP independent and framework neutral. This enables to test struts applications very easily without resorting to mock objects.
* **Intelligent Defaults** - Most configuration elements have a default value which can be set according to the need. Even there are xml-based default configuration files that can be overridden according to the need.
* **Improved  results**- Unlike ActionForwards, Struts 2 Results provide flexibility to create multiple type of outputs and in actual it helps to prepare the response.
* **Better Tag features** - Struts 2 tags enables to add style sheet-driven markup capabilities, so that we can create consistent pages with less code. Struts 2 tags are more capable and result oriented. Struts 2 tag markup can be altered by changing an underlying stylesheet. Individual tag markup can be changed by editing a FreeMarker template. Both JSP and FreeMarker tags are fully supported.
* **Annotations introduced** : Applications in struts 2 can use Java 5 annotations as an alternative to XML and Java properties configuration. Annotations minimize the use of xml.
* **Stateful Checkboxes** - Struts 2 checkboxes do not require special handling for false values.
* **QuickStart** - Many changes can be made on the fly without restarting a web container.
* **customizing controller** - Struts 1 lets to customize the request processor per module, Struts 2 lets to customize the request handling per action, if desired.
* **Easy Spring integration** - Struts 2 Actions are Spring-aware. Just need to add Spring beans!
* **Easy plugins** - Struts 2 extensions can be added by dropping in a JAR. No manual configuration is required!
* **AJAX support** - The AJAX theme gives interactive applications a significant boost.  
  The framework provides a set of tags to help you ajaxify your applications, even on Dojo. The AJAX features include:
  1. AJAX Client Side Validation
  2. Remote form submission support (works with the submit tag as well)
  3. An advanced div template that provides dynamic reloading of partial HTML
  4. An advanced template that provides the ability to load and evaluate JavaScript remotely
  5. An AJAX-only tabbed Panel implementation
  6. A rich pub-sub event model
  7. Interactive auto complete tag

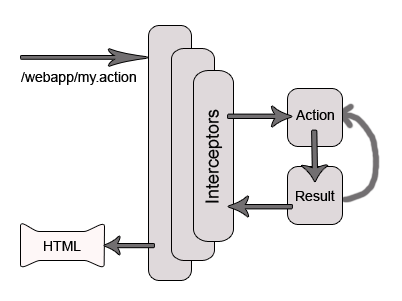
**Architecture of Struts 2.0:**

Struts and webwork has joined together to develop the Struts 2 Framework. Struts 2 Framework is very extensible and elegant for the development of enterprise web application of any size. In this section we are going to explain you the architecture of Struts 2 Framework.

**Request Lifecycle in Struts 2 applications**

1. **User Sends request:** User sends a request to the server for some resource.
2. **FilterDispatcher determines the appropriate action:** The FilterDispatcher looks at the request and then determines the appropriate Action.
3. **Interceptors are applied:** Interceptors configured for applying the common functionalities such as workflow, validation, file upload etc. are automatically applied to the request.
4. **Execution of Action:** Then the action method is executed to perform the database related operations like storing or retrieving data from the database.
5. **Output rendering:** Then the Result renders the output.
6. **Return of Request:** Then the request returns through the interceptors in the reverse order. The returning request allows us to perform the clean-up or additional processing.
7. **Display the result to user:** Finally the control is returned to the servlet container, which sends the output to the user browser.

**Struts 2 high level overview of request processing:**



**Struts 2 Architecture**

Struts 2 is a very elegant and flexible front controller framework based on many standard technologies like Java Filters, Java Beans, ResourceBundles, XML etc.

For the **Model**, the framework can use any data access technologies like JDBC, EJB, Hibernate etc and for the **View**, the framework can be integrated with JSP, JTL, JSF, Jakarta Velocity Engine, Templates, PDF, XSLT etc.

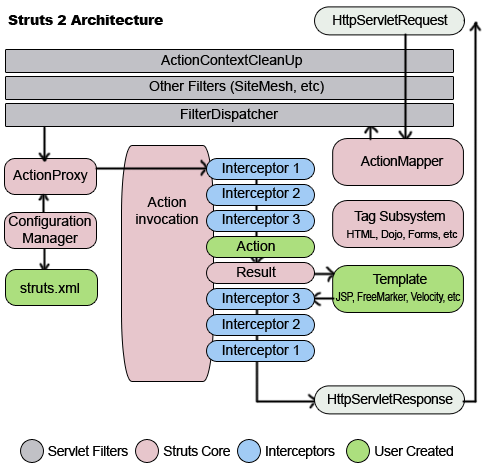
**Exception Handling:**

The Struts 2 Framework allows us to define exception handlers and inceptors.

* **Exception Handlers:**  
  Exception handlers allows us to define the exception handling procedure on global and local basis. Framework catches the exception and then displays the page of our choice with appropriate message and exception details.
* **Interceptors:**The Interceptors are used to specify the "request-processing lifecycle" for an action. Interceptors are configured to apply the common functionalities like workflow, validation etc., to the request.

**Struts 2 Architecture**

The following diagram depicts the architecture of Struts 2 Framework and also shows the initial request goes to the servlet container such as tomcat, which is then passed through standard filer chain.



The filter chain includes:

* **Action ContextCleanUp filter**:  
  The ActionContextCleanUp filter is optional and it is useful when integration has to be done with other technologies like SiteMash Plugin.
* **FilterDispatcher:**  
  Next the FilterDispatch is called, which in turn uses the ActionMapper to determine whether to invoke an Action or not. If the action is required to be invoked, the FilterDispatcher delegates the control to the ActionProxy.
* **ActionProxy:**  
  The ActionProxy takes help from Configuration Files manager, which is initialized from the struts.xml. Then the ActionProxy creates an ActionInvocation, which implements the command pattern. The ActionInvocation process invokes the Interceptors (if configured) and then invokes the action. Then the ActionInvocation looks for proper result. Then the result is executed, which involves the rendering of JSP or templates.   
    
  Then the Interceptors are executed again in reverse order. Finally the response returns through the filters configured in web.xml file. If the ActionContextCleanUp filter is configured, the FilterDispatcher does not clean the ThreadLocalActionContext. If the ActionContextCleanUp filter is not present then the FilterDispatcher will cleanup all the ThreadLocals present.

**Features of Struts 2.0 Framework**

The strut-2 framework is designed for the compilation of the entire development cycle including of building, developing and maintaining the whole application. It is very extensible as each class of the framework is based on an Interface and all the base classes are given an extra application and even you can add your own. The basic platform requirements are Servlet API 2.4, JSP API 2.0 and Java 5.   
  
Some of the general features of the current Apache Strut 2 framework are given below.   
  
**Architecture**? First the web browser request a resource for which the Filter Dispatcher decides the suitable action. Then the Interceptors use the required functions and after that the Action method executes all the functions like storing and retrieving data from a database. Then the result can be seen on the output of the browser in HTML, PDF, images or any other.   
  
**Tags** - Tags in Strut 2 allow creating dynamic web applications with less number of coding. Not only these tags contain output data but also provide style sheet driven markup that in turn helps in creating pages with less code. Here the tags also support validation and localization of coding that in turn offer more utilization. The less number of codes also makes it easy to read and maintain.   
  
**MVC**? The **Model View Controller** in Strut 2 framework acts as a coordinator between applications model and web view. Its Controller and View components can come together with other technology to develop the model. The framework has its library and markup tags to present the data dynamically.   
  
**Configuration** ?Provides a deployment descriptor to initialize resources in XML format. The initialization takes place simply by scanning all the classes using Java packages or you can use an application configuration file to control the entire configuration. Its general-purpose defaults allow using struts directly Out of the box.   
  
Configuration files are re-loadable that allows changes without restarting a web container.   
  
**Other Features:**

* All framework classes are based on interfaces and core interfaces are independent from HTTP.
* Check boxes do not require any kind of special application for false values.
* Any class can be used as an action class and one can input properties by using any JavaBean directly to the action class.
* Strut 2 actions are Spring friendly and so easy to Spring integration.
* AJAX theme enables to make the application more dynamic.
* Portal and servlet deployment are easy due to automatic portlet support without altering code.
* The request handling in every action makes it easy to customize, when required.

**Benefits of Struts 2.0 Framework**

Struts is a framework follows MVC design pattern. so it offers different advantages for the programmer.

It reduces the development maintenance time.

**Centralization of Configuration** :

Rather than hard coding information into java programs, many Struts values are represented in XML or property files. Struts\_config.xml file is the place from where you can get all information about your web application. It is organized, your Action class , Form bean and JSP page information is in Struts\_config.xml so don't need to search .

**Form Beans** :

Don't need to set the form vales to your value object . When you want to capture data from a form ( In the servlet you do request.getParameter()). In the struts you don't need to do explicitly request.getParameter(). Struts request processor will do for you. All the input data will be set to form bean.

**Bean Tags** :

Struts provides a set of custom JSP tags (bean:write,in particular) that let you easily output the properties of JavaBeans components. Basically, these are concise and powerful variations of the standard jsp:useBean and jsp:getProperty tags.

**HTML tags** :

Struts provides a set of custom JSP tags to create HTML forms that are associated with JavaBeans components.

**Form Field Validation** :

Apache Struts has built-in capabilities for checking that form values are in the required format or not .if values are not in proper format then the form can be automatically display with error message and with the previously entered values. This validation can be performed on the server (in Java),or both on the server and on the client (in JavaScript).

**3.2 Introduction to J2EE**

J2EE is Java, optimized for enterprise computing. Officially J2EE stands for Java 2 Platform, Enterprise Edition. Unlike the traditional Java, which is often used to build client enhancements, J2EE is designed to build server applications. Officially, the J2EE platform is "a set of coordinated specifications and practices that together enable solutions for developing, deploying, and managing" such server applications.

As an enterprise platform, the J2EE environment extends basic Java with tools that "provide a complete, stable, secure, and fast Java platform to the enterprise level."

One goal often stated by developers is that by using J2EE, they're reducing the cost and complexity of creating large-scale solutions. Because Java is a strongly typed language (meaning that it requires data types to be used as specified), use of the language is often inherently more secure in Web applications than Web applications built with less strong typing (such as C or even PHP and Perl).

**Architecture of J2EE**

The Java™ 2 Platform, Enterprise Edition (J2EE) provides a standard for developing multitier, enterprise services.

The economy and technology of today have intensified the need for faster, more efficient, and larger-scale information management solutions. The J2EE specification satisfies these challenges by providing a programming model that improves development productivity, standardizes the platform for hosting enterprise applications, and ensures portability of developed applications with an extensive test suite.

J2EE architecture supports component-based development of multi-tier enterprise applications. A J2EE application system typically includes the following tiers:

**Client tier**: In the client tier, Web components, such as Servlets and Java Server Pages (JSPs), or standalone Java applications provide a dynamic interface to the middle tier.

**Middle tier**: In the server tier, or middle tier, enterprise beans and Web Services encapsulate reusable, distributable business logic for the application. These server-tier components are contained on a J2EE Application Server, which provides the platform for these components to perform actions and store data.

**Enterprise data tier**: In the data tier, the enterprise's data is stored and persisted, typically in a relational database.

J2EE applications are comprised of components, containers, and services. Components are application-level components. Web components, such as Servlets and JSPs, provide dynamic responses to requests from a Web page. EJB components contain server-side business logic for enterprise applications. Web and EJB component containers host services that support Web and EJB modules.

**3.3 Introduction to  Java Persistence API**

**What is JPA?**

JPA is just an specification from Sun, which is released under JEE 5 specification. JPA standardized the ORM persistence technology for Java developers. JPA is not a product and can't be used as it is for persistence. It needs an ORM implementation to work and persist the Java Objects. ORM frameworks that can be used with JPA are Hibernate, Toplink, Open JPA etc.

These days most of the persistence vendors are releasing the JPA implementation of their persistence frameworks. So, developers can choose the best ORM implementation according to the application requirement. For example, production can be started from the free versions of ORM implementation and when the needs arise it can be switched to the commercial version of the ORM framework. You can switch the persistence provides without changing the code. So, ORM framework independence is another big benefit of JPA.

**Here are the benefits of JPA**

* Simplified Persistence technology
* ORM frameworks independence: Any ORM framework can be used
* Data can be saved in ORM way
* Supported by industry leaders

**ORM frameworks**

Here are the list of ORM frameworks that can be used with JPA specification.

* Hibernate
* Toplink
* iBatis
* Open JPA

**Why JPA?**

* JPA is standardized specification and part of EJB3 specification
* Many free ORM frameworks are available with can be used to develop applications of any size
* Application developed in JPA is portable across many servers and persistence products (ORM frameworks).
* Can be used with both JEE and JSE applications
* JSE 5 features such as annotations can be used
* Both annotations and xml based configuration support

The Java Persistence API provides an object/relational mapping facility to Java developers for managing relational data in Java applications. Java Persistence consists of three areas:

* The Java Persistence API
* The query language
* Object/relational mapping metadata

## About Persistence Entities

Persistent Data normally refers to permanent data in an application. The state of these data is made permanent by storing them in a persistent medium like database, files or a disk tape. In JPA terms, these persistent data are referred as entities. An entity refers to a logical collection of data that can be stored or retrieved as a whole.

Since entities form the heart of the JPA, they have some unique characteristics like persistability, identity and transactionability. The property of persistability deals with the storing and retrieving of entity from and to a persistent medium like database. Identity property is usually used to identity one unique entity among multiple entities (or multiple entity instances) in a database. All the CRUD operations (Create, Update and Delete) for entity objects will occur within a transactional context and it is one of the major characteristic for an entity object as the real state of an entity depends whether a transaction completes (commits/fails) or not.

## EntityManager

This class follows the standard Manager Design pattern for managing entities. Managing an entity or a set of entities refers to the act of bring a set of Java objects under the control of EntityManager. Unless entities don’t have any explicit association with EntityManager they are just ordinary java objects (though their corresponding classes have been marked with @Entity annotation).

This EntityManager API provides services for persisting an entity, removing an entity, querying and deleting entities.

In a J2SE application, a reference to an entity manager (EntityManager) can be obtained using the entity manager factory (EntityManagerFactory) and the Persistence class. The persistence class is a helper class (or a bootstrap) used to create EntityManagerFactory objects. With EntityManagerFactory objects, references to EntityManager objects can be obtained. The following code illustrates the same,

|  |  |
| --- | --- |
|  | EntityManagerFactory entityManagerFactory=Persistence.createEntityManagerFactory("PersistentUnitName");  EntityManagereManager = entityManagerFactory.createEntityManager( ); |

**Persistence Context**

To be very precise, a persistent context manages a set of entities which in turn is managed by the EntityManager. A persistent context keeps track of the state (or the changes) that an entity object may undergo. And the EntityManager takes the support of this persistence context to commit or to undo the changes. As soon as an EntityManager object is created, it is implicitly associated with a persistence context for managing a set of entities.

**3.4 Introduction to TopLink Framework**

In [computing](http://en.wikipedia.org/wiki/Computing), TopLink is an [object-relational mapping](http://en.wikipedia.org/wiki/Object-relational_mapping) (ORM) package for [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) developers. It provides a framework for storing Java objects in a relational database or for converting Java objects to XML documents.

TopLink Essentials is the [reference implementation](http://en.wikipedia.org/wiki/Reference_implementation) of the [EJB](http://en.wikipedia.org/wiki/Enterprise_JavaBeans) 3.0 [Java Persistence API (JPA)](http://en.wikipedia.org/wiki/Java_Persistence_API) and the open-source community edition of Oracle's TopLink product. TopLink Essentials is a limited version of the proprietary product. For example, TopLink Essentials doesn't provide cache synchronization between clustered applications, some cache invalidation policy, and query Cache.

**Features of Top Link Framework**

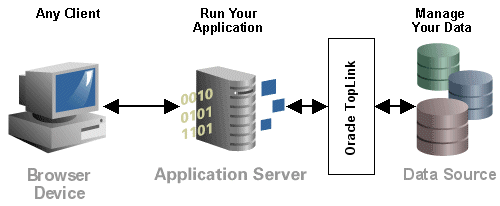
TopLink provides an extensive and thorough set of features. You can use these features to rapidly build high-performance enterprise applications that are both scalable and maintainable.

Some of the primary features of TopLink are the following:

* Nonintrusive, flexible, metadata-based Architectural flexibility: Plain Old Java Objects (POJO), Container-Managed Persistence (CMP), as well as Java Persistence API (JPA), Java API for XML Binding (JAXB), Service Data Objects (SDO), and Web services provided by EclipseLink.
* Advanced mapping support and flexibility: relational, object-relational data type, Enterprise Information Systems (EIS), and XML.
* Optimized for highly scalable performance and concurrency with extensive performance tuning options.
* Comprehensive object caching support including cluster integration for some application servers (such as, for example, Oracle Application Server).
* Extensive query capability including: TopLink Expressions framework, Java Persistence Query Language (JPQL), Enterprise JavaBeans Query Language (EJB QL), and native SQL.
* Just-in-time reading
* Object-level transaction support and integration with popular application servers and databases.
* Optimistic and pessimistic locking options and locking policies.
* Comprehensive visual design tools: Oracle JDeveloperTopLink Editor, Eclipse Dali, and Oracle TopLink Workbench.

**Basic TopLink Architecture**

You can use TopLink in a variety of application architectures, including three- and two-tier architectures, with or without Java EE, to access a variety of data types on both relational and nonrelational data sources.



This section introduces the TopLinkArchitecture that has been used in this project:

**Three-Tier**

The three-tier (or Java EE Web) application is one of the most common TopLink architectures. This architecture is characterized by a server-hosted environment in which the business logic, persistent entities, and the Oracle TopLink Foundation Library all exist in a single Java Virtual Machine (JVM)

The most common example of this architecture is a simple three-tier application in which the client browser accesses the application through servlets, JavaServer Pages (JSP) and HTML. The presentation layer communicates with TopLink through other Java classes in the same JVM, to provide the necessary persistence logic. This architecture supports multiple servers in a clustered environment, but there is no separation across JVMs from the presentation layer and the code that invokes the persistence logic against the persistent entities using TopLink.

**3.5 Conversion of Image to text in Java**

In Java, images can be converted to Base64 String format and Base64 String to Image. This procedure is used in this system to convert Medical reports to files that can be edited and again saved back to system data base to be used later.

**Image to Base64 String**

//Convert Image to byte[ ]

image.Save(ms, format);

byte[ ] imageBytes =ms.ToArray( );

//Convert byte[ ] to Base64 String

string base64String= Convert.ToBase64String(imageBytes);

return base64String;

**Base64 String to Image**

//Convert Base64 String to byte[ ]

byte[ ] imageBytes= Convert.FromBase64String(base64String);

MemoryStream ms=new MemoryStream(imageBytes,0,imageBytes.length);

//Convert byte[ ] to Image

ms.Write(imageBytes,0,imageBytes.Length);

Image image=Image.FromStream(ms,true);

return image;

This conversion has been used in the system to facilitate the editing of the medical reports by the doctor when necessary and the same will be saved back to the data base. The medical report which will be in the format of an image will be converted to text from the above procedure and after the text editing has been made it will be again converted back to the image and will be saved.

**3.6 DBMS MODELS**

A database is similar to data file in that it a storage space for data. Like a data file, a database does not present information directly to the user; the user runs an application that accesses data from the database and presents it to the user in an understandable format.

A database typically has two components; the file holding the physical database and the database management system (DBMS) software that applications use to access the data. The DBMS is responsible for enforcing the database structure, including:

* Maintaining the relationships between data in the database.
* Ensuring that data is stored correctly and that the rule defining data relationships are not violated.

The models popular are

* Hierarchical Model
* Network Model
* Relational Model

**HIERARCHICAL MODEL**

In this model the detailed data records of a master are linked through forward and backward pointers to its next and previous records respectively. All the master records are again linked to one another in the same fashion, in hierarchical model each of the detailed record is linked to only one master record.

**NETWORK MODEL**

This model is extension to hierarchical model where the master and detailed records are connected cross-across way. This makes each detail record to have connected to one or more master records. In network model, pointers are maintained from the transactions record to both the master records indicating that the detailed record is pertaining to both these masters. In network models a detail record is associated with one more master record.

**RELATIONAL MODEL**

In relational model all the data is assumed to be organized in the form of two-dimensional tables. The rows and columns present in the table represent the records and fields respectively. Data management under this model is performed using relational algebra. Each table represents a relational model the pointers to link records are made through the data elements within the same table.

**ABOUT RDBMS**

**Relational Database:**

Sometimes all the information of interest to a business operation can be stored in one table. SQL Server makes it very easy to link the data in multiple tables. Matching an employee to the department in which they work is one example. This is what makes SQL Server a relational database management system, or RDBMS. It stores data in two or more tables and enables you to define relationships between the tables and enables you to define relationships between the tables.

**Foreign Key:**

When a field is one table matches the primary key of another field is referred to as a foreign key. A foreign key is a field or a group of fields in one table whose values match those of the primary key of another table.

**Referential Integrity:**

Not only does SQL Server allow you to link multiple tables, it also maintains consistency between them. Ensuring that the data among related tables is correctly matched is referred to as maintaining referential integrity.

**Advantages of RDBMS**

* Redundancy can be avoided
* Inconsistency can be eliminated
* Data can be Shared
* Standards can be enforced
* Security restrictions can be applied
* Integrity can be maintained

**Disadvantages of DBMS**

A significant disadvantage of the DBMS system is cost. In addition to the cost of purchasing of developing the software, the hardware has to be upgraded to allow for the extensive programs and the workspace required for their execution and storage. While centralization reduces duplication, the lack of duplication requires that the database be adequately backed up so that in case of failure the data can be recovered.

**Introduction to Oracle**:

An Oracle **database** is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a [server](http://docs.oracle.com/cd/B19306_01/server.102/b14220/glossary.htm#i432724) reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.

Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost effective way to manage information and applications. Enterprise grid computing creates large pools of industry-standard, modular storage and servers. With this architecture, each new system can be rapidly provisioned from the pool of components. There is no need for peak workloads, because capacity can be easily added or reallocated from the resource pools as needed.

The database has **logical structures** and **physical structures**. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures.

**3.7 Module Details:**

**Login Form:**

This is the entry step to the system where the user logs in with the username and password and will be redirected to their appropriate home page depending of the type of the user. If the entered user name or password is incorrect then the user will be prompted to re-enter the login credentials. If it’s a new user then the user must register into the system.

**Lab Assistant Module:**

**Report Upload**

Here in this module the Lab Assistant uploads the medical reports of a particular patient. He/she needs to provide the credentials like the name of the patient, type of the report that they are uploading and report file itself. The system provides an e-Lab that can be used by the patients to check their medical reports and also then reports can be helpful for the doctors to assess the patient’s medical condition.

**Lab Report Viewer**

After uploading the lab report, the lab assistant can view the report by clicking on the ‘View Report’ link below the upload page. The uploaded report can also be viewed by the patient in their respective medical profile. Medical reports for a particular patient can also be viewed by the doctor while he/she is counseling the patient to study their health status and can prescribe medicines accordingly.

**Doctor Module:**

**Medical Report Editing**

The medical reports being uploaded by the Lab Assistant will be in the form of an image. Doctors will be able to view the medical reports of the patients that they are counseling along with their medical history. The system offers the provision of editing the medical reports by the doctors in case any change is needed. The edited report will be saved to the data base and the same will be referred from there on.

**Pharmacist Module:**

**View Prescription**

Pharmacist can view the prescription details of a particular patient by entering the patient name in the form. The prescription details can be used to check the medicines that are prescribed to the patient and if necessary then same should be delivered to the patient’s location on their request.

**Medicines Availability**

The details of the medicines that are currently available in the e-Pharmacy can be tracked through the system. Pharmacist can check the medicines availability so that he/she can arrange for medicines delivery upon the patient’s request.

**Medicine Delivery**

Pharmacist can check the delivery details and status of the medicine deliveries requested by the patients. The undelivered orders can be viewed through this system so that delivery can be made the patient’s location.

**Chat Application Module:**

The system supports for a one-one chat application wherein the patient and the doctor consultation can be made through this application. The patient logs in to the system with his/her user name and password and takes an appointment with the doctor of a particular specialization. The doctor will be able to check the appointments for the day. The chat server will be triggered by timing specified in the appointment. The chat server starts at the appointment start time on the doctor’s side. Only after the start of the chat server the chat client starts by an indication that the chat button on the patient’s page gets active only when the appointment time is approached. After the end of the consultation with the doctor the chat server is stopped which leads to auto termination of the chat client. This procedure is followed because the consultation time could be increased by a few minutes by the doctor so that the chat application does not get shut down right after the end of the appointment duration avoiding the abrupt end to the conversation.

**Functional Requirements**

The functional requirements describe the interactions between the system and its environment independent of its implementation.

**User Module**

The system should accept the user Name & Password as input from Administration of organization for successful login. The Database collects the details of the user and appropriate home page will be displayed according to the type of the user.

If the user is already exists then he will verify the user name and password with the data base. The system should accept the Username & Password as input from user of organization for successful login.

**User Registrations**: this takes the following as the input like User name, password, DOB, address, city, state, country, zip code.

**Non Functional Requirements**

Nonfunctional requirements describe user-visible aspects of the system that are not directly related to functionality of the system.

**Performance Constraints**

* Requests should be processed within no time.
* Users should be authenticated for accessing the requested data

**Error Handling and Extreme Conditions** In case of User Error, the System should display a meaningful error message to the user, such that the user can correct his Error.

The high level components in proposed system should handle exceptions that occur while connecting to database server, IOExceptions etc.

**Quality Issues**

Quality issues refer to how reliable, available and robust should the system be? While developing the proposed system the developer must be able to guarantee the reliability transactions so that they will be processed completely and accurately.

The ability of system to detect failures and recovery from those failures refers to the availability of system. Robustness of system refers to the capability of system providing information when concurrent users requesting for information.

**Security**

Security and confidentiality are the top most concerns of the client. The proposed system should provide the following.

* Each customer should also be provided with ID and password for controlled access.
* Access to database should also be restricted to the managing department only.

**3.8 Model used for Project:**

**Introduction:**

The waterfall model is a sequential software development model (a process for the creation of software) in which development is seen as flowing steadily downwards (like a waterfall) through the phases of requirements analysis, design, implementation, testing (validation), integration, and maintenance.

The model: In **waterfall model**, the following phases are followed in an order

* Design,
* Construction,
* Integration,
* Testing and debugging,
* Installation,
* Maintenance.

To follow the waterfall model, one proceeds from one phase to the next in a purely sequential manner. For example, one first completes requirements specifications, which are set in stone. When the requirements are fully completed, one proceeds to design. The software in question is designed and a blueprint is drawn for implementers (coders) to follow — this design should be a plan for implementing the requirements given. When the design is fully completed, an implementation of that design is made by coders. Towards the later stages of this implementation phase, disparate software components produced by different teams are integrated. After the implementation and integration phases are complete, the software product is tested and debugged; any faults introduced in earlier phases are removed here. Then the software product is installed, and later maintained to introduce new functionality and remove bugs.

The advantage of waterfall development is that it allows for departmentalization and managerial control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process.

**4. DESIGN**

**4.1 SOFTWARE DESIGN**

It is a process of planning the new or modified system. Analysis specifies what a new or modified system does. Design specifies how to accomplish the same. Design is essentially a bridge between requirement specification and the final solution satisfying the requirements. Design of a system is essentially a blue print or a solution for the system.

The design process for a software system has two levels. At first level the focus is on depending in which modules are needed for the system, the specification of these modules and how the modules should be interconnected. This is what is called system designing of top level design. In the second level, the internal design of the modules, or how the specification of the module can be satisfied is described upon. This design level is often called detailed design or logic design.

The first level produces system design, which defines the components needed for the system, and how the components interact with each other. It focus is on depending on in which that modules are needed for the system, the specification of these modules and how the module should be interconnected.

**4.2 Introduction to UML**

Unified Modeling Language is the one of the most exciting tools in the world of system development today. Because UML enables system builders to create blue prints that capture their visions in a standard, easy to understand way and communicate them to others. The UML is brain child of Grady Brooch, James Rum Baugh and Ivory Jacobson.

The UML is a language for

* Visualizing
* Specifying
* Constructing
* Documenting

These are the artifacts of a software-intensive system. The abbreviation for UML is Unified Modeling Language and is being brought of a designed to make sure that the existing ER Diagrams which do not serve the purpose will be replaced by this UML Diagrams where in these language as its own set of Diagrams.

Some of the Diagrams that help for the Diagrammatic Approach for the Object Oriented Software Engineering are

* Class Diagrams
* Use Case Diagrams
* Sequence Diagrams
* State Chart Diagrams
* Activity Diagrams

Using the above mentioned diagrams we can show the entire system regarding the working of the system or the flow of control and sequence of flow the state of the system and the activities involved in the system.

**Components of the UML:**

The UML consists of a number of graphical elements that combine to form diagrams. Because it’s a language, the UML has rules for combining these elements. The purpose of the diagrams to present multiple views of the system, and this set of multiple views is called a Model. A UML Model of a system is something like a scale model of a building. UML model describes what a system is supposed to do. It doesn’t tell how to implement the system. The following are the main nine component Diagrams of UM L:

**Class Diagram:**

A Class is a category or group of things that has similar attributes and common behavior. A Rectangle is the icon that represents the class it is divided into three areas. The upper most area contains the name, the middle area contains the attributes and the lowest areas show the operations. Class diagrams provides the representation that developers work from. Class diagrams help on the analysis side, too.

**Object Diagram:**

An object is an instance of a class- A specific thing that has specific values of the attributes and behavior. A Rectangle is the icon that represents the object diagram but the name is underlined. The name of the specific instance is on the left side of the colon, and the name of the class is on the right side of the colon.

**Use-Case Diagram:**

A Use-Case is a description of a systems behavior from a user’s stand point. For system developer this is a valuable tool: it’s a tried-and-true technique for gathering system requirements from a user’s point of view. That is important if the goal is to build a system that real people can use. A little stick figure is used to identify an actor the ellipse represents use-case.

**State Chart Diagram:**

At any given time, an object is in particular state. One way to characterize change in a system is to say that its objects change the state in response to events and to time. The UML State Diagram captures these kinds of changes it presents the states an object can be in along with the transitions between the states, and shows the starting point and end point of a sequence of state changes. A Rounded Rectangle represents a state, along with the solid line and arrow head that represents a transition. The arrow head points to the state being transition into. The solid circle symbolizes starting point and the bulls eye that symbolizes the end point.

**Sequence Diagrams:**

In a functioning system objects interacts with one another and these interactions occur over time. The UML Sequence Diagrams shows the time based dynamics of the interaction. The sequence diagrams consist of objects represented in the usual way-as named rectangles (If the name underlined), messages represented as solid line arrows and time represented as a vertical progression.

**Activity Diagrams:**

The state diagram shows the states of an object and represents activities as arrows connecting the states. The Activity Diagram highlights the activities. Each activity is represented by a rounded rectangle-narrower and more oval-shaped than the state icon. An arrow represents the transition from the one activity to the next.

**Collaboration Diagram:**

An object diagram shows the objects and their relationships with one another. A collaboration Diagram is an extension of the object diagram. In addition to the associations among objects, the collaboration diagram shows the messages the objects and each other.

**Use cases:**

**Use case Model:** It is the functional model, represented in UML. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. A use case describes a function provided by the system that yields a visible result for an actor. An actor can be human or an external system. In this proposed system actors are Administrator, Manager, Technical person and Customer.

**Kinds of classifiers**

A specification of the behavior of an entity in its interaction with outside agents

Use case

A computational resource.

Node

A named set of operations that characterize behavior.

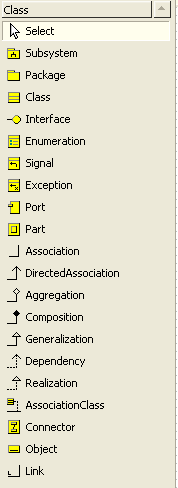
Interface

A concept from modeled system.

Class

An outside user of the system.

Actor

****

**4.3 USE CASE DIAGRAMS:**

UML provides the use case diagram to facilitate the process of requirements gathering. The Use case diagram models the interactions between the system’s external clients and the use cases of the system. Each use case represents a different capability that the system provides the client. Stick figure represents an actor.

Lab Assistant

Pharmacist

Doctor

**CLASS DIAGRAM:**

The object model, represented in UML with Class diagrams, Describes the structure of system in terms of objects, attributes, associations. The UML enables us to model the classes in a system and their relationships via class diagrams. In a class diagram, each class is modeled as a rectangle. This rectangle can be divided into three parts. The top part contains the name of the class. The middle part contains the class’s attributes. The bottom contains the class’s operations.

|  |
| --- |
| Login |
| Userid  Password  UserType |
| getUserId():String  setUserId(String):Void  setPassword(String):void  getPassword():String  setUserType(String):void  getUserType():String |

|  |
| --- |
| Report |
| labId  labName  testid  patientId  reportId  testName |
| getLabId():Integer  setLabId(Integer):Void  getLabName():String  setLabName(String):Void  getTestId():Integer  setTestId(Integer):Void  getPatientId():Integer  setPatientId(Integer):Void  getReportId():Integer  setReportId(Integer):Void  getTestName():String  setTestName(String):Void |

|  |
| --- |
| Pharmacy |
| prescriptionId  doctorId  patientId  medicineId  medicineName  price  deliveryId  deliveryStatus  billAmount  location |
| getPrescriptionId():Integer  setPrescriptionId(Integer):Void  getDoctorid():Integer  setDoctorId(Integer):Void  getPatientId():Integer  setPatientId(Integer):Void  getMedicineId():Integer  setMedicineId(Integer):Void  getPrice():Double  setPrice(Double):Void  getDeliveryId():Integer  setDeliveryId(Integer):Void  getDeliveryStatus():Char  setDeliveryStatus(Char):Void  getLocation():String  setLocation(String):Void |

**SEQUENCE DIAGRAM**

Sequence diagrams describe behavior as a sequence of messages exchanges among setoff objects. The sequence diagrams consists of objects represented in the usual way as named rectangles with underlined messages represented as solid-line arrows and time represented as a vertical progress.

Login

View Report

Logout

Lab Assistant Menu

Upload Report

1.Login

2.Upload Report

3.View Report

4.Logout

Login

Logout

Pharmacist Menu

1.Login

2.Logout

View/Edit Medical Report

Check Appointments

Counsel the patient

Login as Doctor

Save Report

1.Login

2.Check Appointment

3.Counsel patient

4.View/Edit report

5.Save Report

Receive e-Prescription

Wait for Appointment time

Get Counseling

Start the chat

Take Appointment

1.Take Appointment

2.Wait for Appointment

3.Chat

4.Get prescription

**STATE CHART DIAGRAM:**

State chart diagrams describe behavior in terms of states of individual objects and the possible transition between states. One way to characterize the change in a system is to say that its objects change their state in response to events and time. This also shows the starting and end point of a sequence of state changes.

Upload Medical Reports

Login as Lab Assistant

View Report

Logout

Logout

Perform an Action

Login as Pharmacist

Counsel Patient

Check Appointments

Login as Doctor

View/Edit Report

Save Report

Take Appointment

Wait for Appointment

Chat with Doctor

Get e-Prescription

Receive Counseling

**Activity Diagram**

An activity diagram describes a system in terms of activities. Activities are states that represent the execution of set of operations. These can be used to represent control flow i.e. the order in which operations occur and dataflow. This is useful for showing what happens in a business process or an operation.

Login as Lab Assistant

Upload Medical Reports

View Reports

Logout

Login as Pharmacist

Perform Action

Logout

Login as Doctor

Check Appointments

Counsel Patient

View/Edit Report

Save Report

Take Appointment

Wait for Appointment

Start Chat

Get Counseling

Get e-Prescription

**4.4 ORACLE CONCEPTS: DBMS**

DBMS is an organized collection of interrelated **DATA FLOW DIAGRAMS:**

A collection of components that works together to realize some objectives is called a system. An information system is nothing but a system that provides information to the peoples in an organization. A new system may be built afresh or by changing the existing system. System analysis is an important activity that takes places when new information systems are being built or existing once is changed. A set of steps that define how things are done is called process. A process is followed to determine what the system does and what is required of it is system analysis. This is central to the whole of the system development. It includes gathering the necessary data and developing plans for new system.

The data flow diagram (DFD) is one of the most important modeling tools used by the system analysis. DFD uses a number of symbols to represent systems. There are four kinds of symbols and are used to represent four kinds of systems components namely process, data stores, data flow and external entities. DFD’s are used to illustrate how data flows in a system.

**DATABASE DESCRIPTION:**

The data pertaining to proposed system is voluminous hence a careful design of the database must proceed before storing the data in the database.

A database management system provides flexibility in the storage and retrieval of data. The DBMS is a bridge between the application program, which determines what data are needed and how they are processed, and the operating system of the computer, which is responsible for placing data on the magnetic storage devices. A schema defines the database and a subschema defines the portion of the database that a specific program will use.

**Characteristics:**

* Represents complex relationships between data.
* Keeps control on data redundancy.
* Keeps a centralized data dictionary for the storage of information retain the data.
* Enforces data access authorization.
* Has automatic intelligent backup and recovery procedure for data.

**Database Administrator:**

A database administrator is a block of code loaded into memory, which organizes all information (database) between the users.

The DBA takes care of the following things:

* Updating database.
* Retrieving the data
* Accepting the queries
* Enforces security
* Optimizing queries

**Normalization:**

**First Normal Form**

A relation R is in first normal form if and only if all underlying domains contain atomic values only.

**Second Normal Form**

      A relation R is in second normal form if and only if it is in first normal form and every non-key attribute is fully dependent on primary key.

**Third Normal Form**

      A relation R is in third normal form if and only if t is in second normal form and every non-key attribute is non-transitively dependent on primary key.

**Boyce-Codd Normal Form**

      A relation R is in Boyce-Codd form if and only if every determinate is a candidate key. Normalization reduces redundancy. Redundancy is the unnecessary reputation of data. It can cause problems with storage and retrieval of data. Full-normalized record consists of a primary key, which identifies the entity and empty set of attribute.

**DATA DICTIONARY:**

A data dictionary is a file that contains metadata i.e. data about data. This file is consulates before actual data are read or modified in the database system. It is also known as system catalog. Among the types of information the system must store all the below items.

* Name of the relations
* Names of the attributes of each relation
* Domains and lengths of attributes
* Names of view defined on the database, and definitions of those views
* Integrity constraints.

**Architecture of DBMS:**

**Schema**

A schema is an overall design of the database. At the lowest level is the physical schema, at the intermediate level is the logical schema and at the highest level is the external schema.

The DBMS Architecture can be divided into 3 levels as described below.

**Internal/Physical Level**

This is at the lowest level of the Database abstraction and closest to physical storage method used, is defined by means of internal schema.

**Conceptual/Logical level**

This is at the lowest level of the Database and describes what data are stored in the database and what relationships exist among those data, is defined by means of conceptual schema.

**External/View level**

This is at the highest level of database abstraction where only portions of the database of concern to a user or application program are included, is defined by means of external schema.

**Relational Model**

In the Relational model all the data is assumed to be organized in the form of two dimensional tables and each table has multiple columns and each column can be identified by unique name.

**RDBMS**

RDBMS is a DBMS bases on relational model. It is a set of programs that manipulate and data by storing data in the form of tables. It can be defined as a DBMS where all data visible to the user organized strictly as tables of data and where all the database operations work on these tables. For any RDBMS to be accepted as a fully-fledged RDBMS. It has to follow the relational theory in mathematics and the 12 CODD’s Rules.

**Advantages of RDBMS:**

* Redundancy can be reduced.
* Inconsistency can be reduced.
* Data can be shared.
* Standards can be enforced.
* Security restrictions can be applied.
* Integrity can be maintained.

Oracle is the name of the database management system developed by Oracle Corporation. Oracle is a RDBMS as it stores and manages data using relational model. Oracle server manages data in the database. Users access server using SQL commands so oracle server receives SQL commands from the users and execute them on the database. Users of oracle database use logical view of the data.

**ORACLE’S ROLE IN CLIENT/SERVER COMPUTING.**

Client/server computing is a method in which

* Database is stored on the server in the network
* A dedicated program called back-end runs on the server to manage database, which is also stored on the server.
* User access the database by running application also called as front end running on the server.
* Applications running on the clients interact with the user.
* Back-end takes care of the total database Management
* Client application and back-end on different machines which may be of different types.

**FEATURES OF ORACLE:**

The following are some of the important features of oracle server.

**Largest database support:**

Oracle supports largest database, potentially hundreds of gigabytes.

**Data concurrence:**

Oracle supports concurrent access to database by multiple users. It automatically locks and unlocks rows to maintain integrity of data.

**Partitioning:**

It has improved performance via partitioning. A table can be divided into a number of pieces called as partition based on the value of one or more columns.

**Portability:**

It has new and improved data type’s portability. Oracle software can be ported to different operating systems and it is the same on systems. Application developed in the oracle can be ported to any operating system with little or no modification.

**Enforced integrity**:

Oracle allows users to define business rules and enforce them. These Rules need not be included at the application level.

**Data security:**

Oracle provides security in different levels-system level and object level. It also makes implementation of security easier through roles.

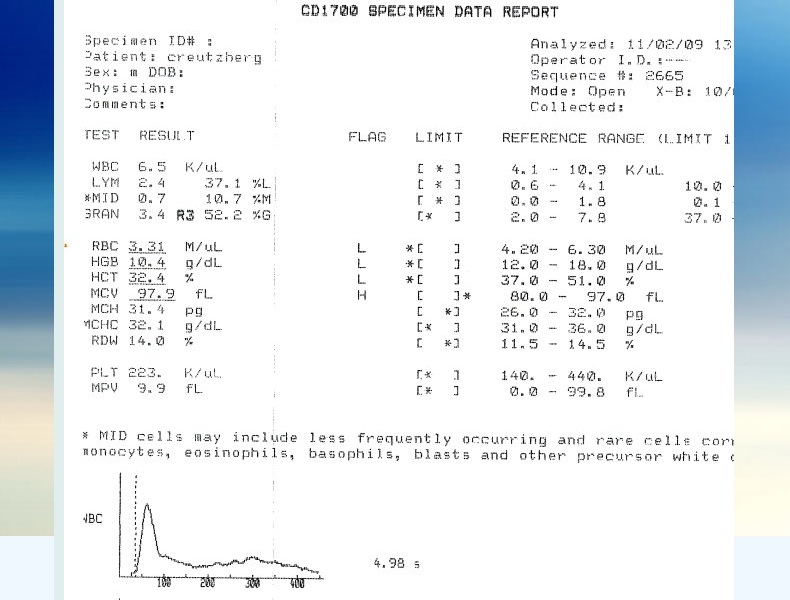
**Enhanced recovery features:**

Oracle has made tremendous improvements in the area of backup and recovery. Most of these features revolve around the recovery manager. Another recovery feature is the image copy backup, which can improve recovery time in the event of failure.

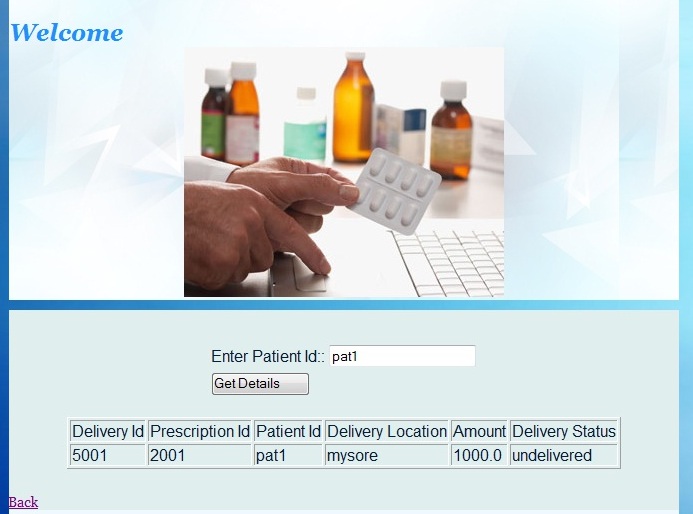
**5. SCREEN SHOTS**

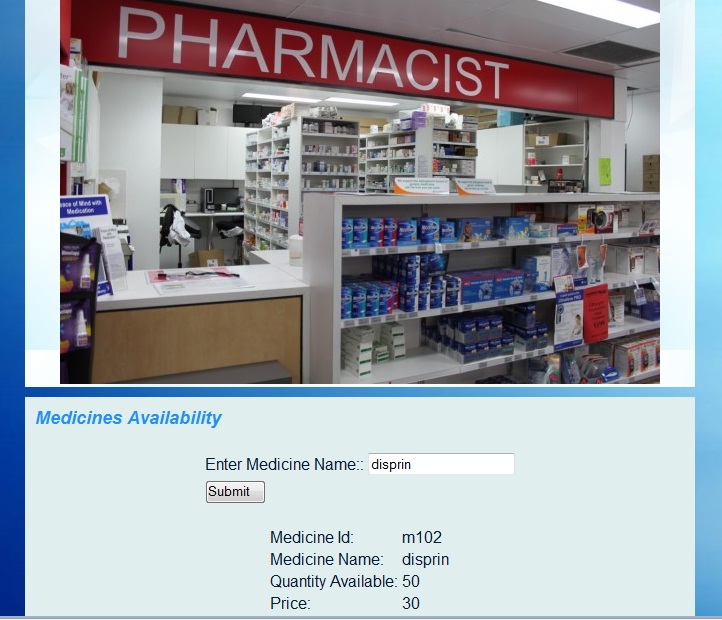
****











**6. TESTING**

**INTRODUCTION:**

**TESTING:**

Testing is the major quantity control measure employed during software development. Its basic function is to detect errors in the software. During requirement analysis and design, the output is a document that is usually textual and non-executable. This implies that testing not only has to uncover errors introduced during coding, but also uncovers requirement, design or coding errors in the programs. Consequently, different levels of testing are employed.

      The basic levels are unit testing, integration testing, system testing and acceptance testing. These different levels of testing attempt to detect different types of faults.

**Unit Testing:**

      The first level of testing is “unit testing “. In this different modules are tested against the specifications produced during design for the modules. Unit testing is essentially for verification of the code produced during the coding phase, and hence the goal is to test the internal logic of the modules.

**Integration Testing**:

      In this, many tested modules are combined into sub-systems, which are then tested. The goal is to see if the modules can be integrated properly, the emphasis being on testing interfaces between modules. This testing activity can be considered as testing the design, and hence the emphasis on testing module interactions.

**System Testing:**

      Here the entire software system is tested. The reference document for this process is the requirements document, and the goal is to see if the software meets its requirements.

**Acceptance Testing:**

      Acceptance testing is sometimes performed with realistic data of the client to demonstrate that the software is working satisfactorily. Testing here focuses on the external behavior of the system, the internal logic of the system is not emphasized.

**TYPES OF TESTING:**

  To accomplish task software is done under two categories of test case design techniques

* Structural/ White box Testing
* Functional/ Black box Testing

**Structural Testing (WHITE BOX)**

Structural testing is concerned with testing the implementation of the program. The intention of the structural testing is not to execute all the different input and output conditions, but to exercise the different programming and data structures used in the program. This testing is otherwise called as the “white box testing”.

**Functional Testing (BLACK BOX)**

      In functional testing the structure of the program is not considered. Test cases are decided solely on the basis of the requirements or specifications of the program or module and the internals of the module or the program are not considered for selection of test cases.

       Here the complete functionality of the system is tested. There are several test coverage criteria under the functional testing. This code is tested to meet the functional requirements of the system.

**System Testing:**

Testing is the process of finding differences between the expected behaviors specified by system Models and the observed behavior of the system.

**Testing Concepts:**

* A **component** is a part of the system that can be isolated for testing. A component can be an object, a group of objects, or one or more subsystems.
* A **fault**, also called bug or defect, is a design or coding mistake that may cause abnormal component behavior.
* An **error** is a manifestation of a fault during the execution of the system.
* A **failure** is a deviation between the specification of a component and its behavior. A failure is triggered by one or more errors.
* A **test case** is a set of inputs and expected results that exercise a component with the purpose of causing failures and detecting faults.

**Testing Activities:**

**Inspecting components,** which finds faults in an individual component through the manual inspection of its source code.

**Unit testing,** which finds faults by isolating an individual component using test stubs and drivers and by exercising the components using a test case.

**Integration testing,** which finds faults by integrating several components together.

**System testing,** which focuses on the complete system, its functional and nonfunctional requirements and its target environment.

**TEST PLAN:**

The plans for testing the course scheduling software. All major testing activities are specified here.

**Test Units:**  In this project we will perform three levels of testing:

* Unit testing
* Integrity testing
* System testing.

The basic units to be tested as follows Modules for processing input details.

* Modules for generating the routing details.
* Progress Update module.
* Modules for generating reports.

**Features to be tested:**

All the functional features specified in the requirements documents will be tested. No testing is done for the performance requirements like response time.

**Approach for testing:**

For unit testing structural testing based on the branch coverage criteria will be used. The goal is to achieve branch coverage of more than 95%. System testing will be largely functional in nature.

**Test Deliverables:**

The following documents are required:

* Unit Test Report for each unit.
* Test case specification for system testing.
* Test report for system test

**Various test cases**

**1. Test Case: Login Testing**

|  |  |
| --- | --- |
| Test Case Name: | Login Tester |
| Test classification: | Black Box test |
| Test objective: | Appropriate response and validation of webpage |
| Means of Control: | Test the Users’ authentication |
| Data: | The User name and Password of the user |
| Test Procedure: | Verifies the Password of the particular User |

**2. Test Case: Patient Name Testing**

|  |  |
| --- | --- |
| Test Case Name: | Patient Name Tester |
| Test classification: | Equivalence test |
| Test objective: | Test to check if the patient name entered is in the data base |
| Means of Control: | Testing for valid patient name being entered |
| Data: | The information given by the Pharmacist |
| Test Procedure: | Verifies if the entered patient name is valid and in the data base |

**3. Test Case: Medicine Name Testing**

|  |  |
| --- | --- |
| Test Case Name: | Medicine Name Tester |
| Test classification: | Equivalence test |
| Test objective: | Test to check if the medicine name entered is in the data base |
| Means of Control: | Testing for valid medicine name being entered |
| Data: | The information given by the Pharmacist |
| Test Procedure: | Verifies if the entered patient name is valid and in the data base |

**4. Test Case: Database Testing**

|  |  |
| --- | --- |
| Test Case Name: | Database Tester |
| Test classification: | Black Box test |
| Test objective: | Checking the Database constraints |
| Means of Control: | Site maintainer: Test driver for uploading the data in appropriate tables. |
| Data: | At the time of uploading the files in local disk, we have to upload the data related to file attributes in database. |
| Test Procedure: | By opening the database tables and checks it with uploaded data. |

**5. Test Case: File Upload Testing**

|  |  |
| --- | --- |
| Test Case Name: | File Upload Tester |
| Test classification: | Black Box test |
| Test objective: | Test to check if the file being uploaded is in proper format that can be saved to data base |
| Means of Control: | Testing for valid format of the file being uploaded |
| Data: | File being uploaded by the Lab Assistant |
| Test Procedure: | Verifies if the file being uploaded is in proper format to be saved in the data base |

**7. CONCLUSION**

**CONCLUSION:**

This system integrates all the basic medical services that are being provided at hospitals in the present world. The system is controlled by the Medi Assist internally where he/she will be maintaining and managing the users accessing the system.

The design of the system is made by taking the users basic computing and language skills into consideration. Effort has been made to reduce redundancy to the optimum level.

It also includes issue tracking system and online forum where in the user can communicate with other users of the system and can raise and issue if required which will be handled by the admin. Furthermore, considerations regarding non-functional requirements are addressed.

This document is intended for designers, testers and implementation unit.

**8.BIBLIOGRAPHY**

BOOKS REFERRED

* Struts 2 Black Book

Author: Kogent Solutions INC

* Java 2 Complete Reference, Third Edition

Author: Patrick Naughton, Herbert Schildt

* OBJECT-ORIENTED SOFTWARE ENGINEERING

Author: Bernd Bruegge& Allen H.Dutoit, Pearson Education Asia

* SQL Server:

Author: GunderloyJorden, BPB Publisher

WEBSITES REFERRED

* [www.roseIndia.com/](http://www.roseIndia.com/)
* [www.stackoverflow.com](http://www.stackoverflow.com)
* [www.wikipedia.com](http://www.wikipedia.com)
* [www.google.com](http://www.google.com)