# Introduction

## Purpose

This web application is for pharma market. Pharma market would like to analyze which medicine are available in market and which medicine mostly doctor used for particular disease. And what are the ingredients available in the medicine. So that if pharma company will invent new medicine then that pharma company know the scope of medicine. For this my company planning to develop this kind of web application.

## Product Scope

This application keeps the data in a **centralized way** which is available pharma company.

It is very easy to manage historical data in database.

## Overview

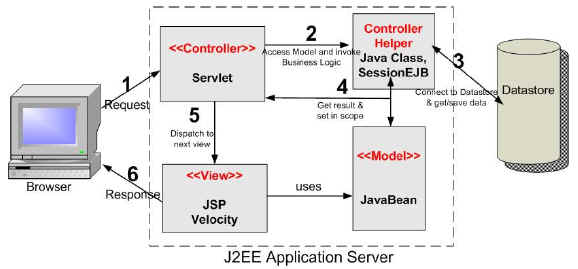
* Now a days, for any pharma company there is not more options rather than paper work(file) to save the details of medicine available in market and need of market. It’s very time consuming process to collect the details from market, make report and analyze all reports.
* Also for doctors, they have to manage all patients’ reports, medicine reports, diagnosis reports etc. i.e. time consuming and lots of paper work.
* So to overcome from all above the problems and typical process we create a web application that is very helpful to doctors as well as any pharma company.

1. **Technology and Literature Review**

**2.1 About Tools and Technology**

We use the java for this project using netbeans.and for database we use ms sql server.

In java we use the jsp and servlet.how this work we see in fig.

****

**Fig 2 (a)**

## 3 System Requirements Study

## 3.1 Hardware Interfaces

1. PC should be sufficiently fast with adequate memory space. At least 64 MB RAM and 2 GB hard-disk space is required to run this application.
2. Screen resolution of atleast 800\*600 is required to properly view the screens.

## 

## 3.2 Software Interfaces

1. Any windows operating system.
2. For the database handling MYSQL must be installed. These products are open source products.

**3.3 Design and Implementation Constraints**

* **Database:**  
  The system shall use the MySQL Database, which is open source and free.
* **Operating System**  
  The Development environment shall be Windows based.
* **Web-Based**  
  The system shall be a Web-based application.

**3.4 Assumptions and Dependencies**

* It is assumed that one hundred IBM compatible computers will be available before the system is installed and tested.
* It is assumed that the pharma company will have enough trained staff to take care of the system

**4 System Analysis**

**4.1 Study of Current System**

* Now a days, for any pharma company there is not more options rather than paper work(file) to save the details of medicine available in market and need of market. It’s very time consuming process to collect the details from market, make report and analyze all reports.
* Also for doctors, they have to manage all patients’ reports, medicine reports, diagnosis reports etc. i.e. time consuming and lots of paper work.

**4.2 Advantage of new system**

* Manually system changes into computerized system.
* Time saving.
* Save paper work.
* Give facility of different type of inquiry.
* Data’s are easily approachable.
* Doctor can view patient’s history
* Pharmaceutical company get all the data related to the medicine
* Pharmaceutical company get efficient information for invention of new product.
* Doctor make record of diagnosis of patient in efficient manner.

**4.3 Features:**

Pharma Market Analyzer has lots of features as following:

* **Doctor :**
* Login
* Easily save patients details
* View patient history
* Write/Upload and analyze prescription
* Search cases
* Search patient disease-wise.
* Analyze monthly/yearly patient report
* **Pharma Company:**
* Easily analyze diseases and medicine reports so get idea about medicine requirements in market
* View Graph
* Add new doctors
* Update and Delete Doctors detail
* View doctors’ reports
* View Medicine history

**4.4 Process In New System**

* Pharma market analyzer is a software application which avoids the effort to making files of records for patients. This application keeps the data in a **centralized way** which is available pharma company. It is very easy to manage historical data in database.
* It is very easy to record the information of patient’s like patient’s name’s, patient’s disease and all the patient’s history ,their reports , medicine list in the databases.

4.5 Class Diagram

* The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application.
* The class diagram describes the attributes and operations of a class and also the constraints imposed on the system.
* The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages.
* The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a *structural diagram*



**Fig 4.5(a)**

4.6 Use case diagram

* To introduce use case diagrams two things to remember is:

1) Actors

2) Use cases

Dependency relationships between use cases:

* An *actor* is anyone or anything that must interact with the system.
* Actors are not part of the system
* In the UML, an actor is represented as a stickman.
* A use case must deliver something of value to an actor.
* The collection of use cases for a system constitutes all the defined ways the system may be used.
* In the UML, a use case is represented as an oval.
* A use case diagram is a diagram that shows a set of use cases and actors and their relationships.
* Actors may be connected to use cases only by association.
* An association between an actor and a use case indicates that the actor and the use case communicate with one another, each one possibly sending and receiving messages.

**Use case diagrams are used to …**

* Model the context of a system.
* Model the requirements of a system

**Dependency Relationships between Use Cases:**

* *Extend*: Specifies that the target use case extends the behavior of the source use case.
* *Include*: Specifies that the source use case explicitly incorporates the behavior of another use case at a location specifies by the source.

**Symbols for Use case diagram**

|  |  |
| --- | --- |
| **Symbol name** | **Symbol** |
| System Boundary |  |
| Use-Case |  |
| Uses |  |
| Actor |  |

High level Usecase



**Fig4.6(a)**

**Fig4.6(b)**

**Fig4.6(c)**





**Fig4.6(d)**

4.7 Sequence Diagram

It illustrates how objects interact with each other. It emphasizes time ordering of

messages it can model simple sequential flows, branching, iteration, recursion and

concurrency.

**Table: 4.4 Symbols for Sequence diagram**

|  |  |
| --- | --- |
| **Symbol Name** | **Symbol** |
| Object | Object |
| Life Line |  |
| Active |  |
| Message |  |

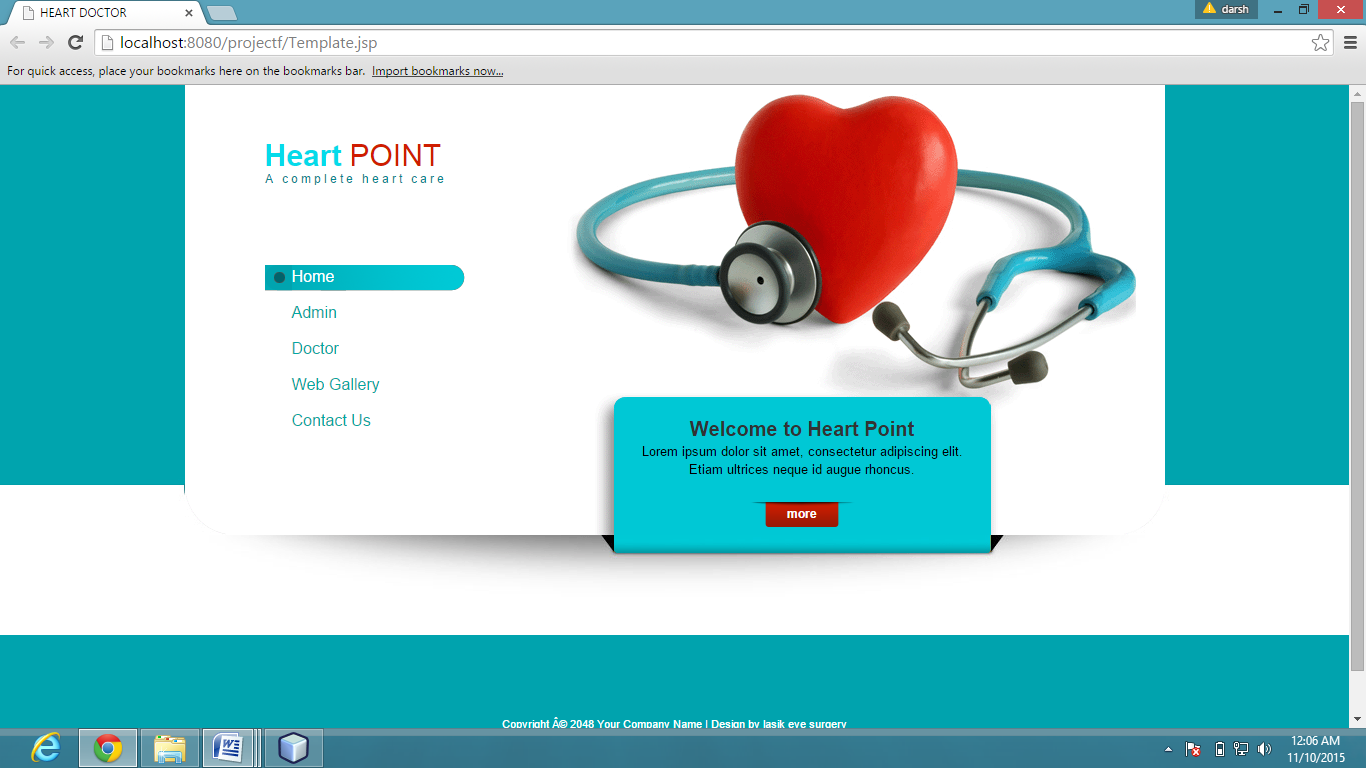


**Fig 4.7 (a) admin**

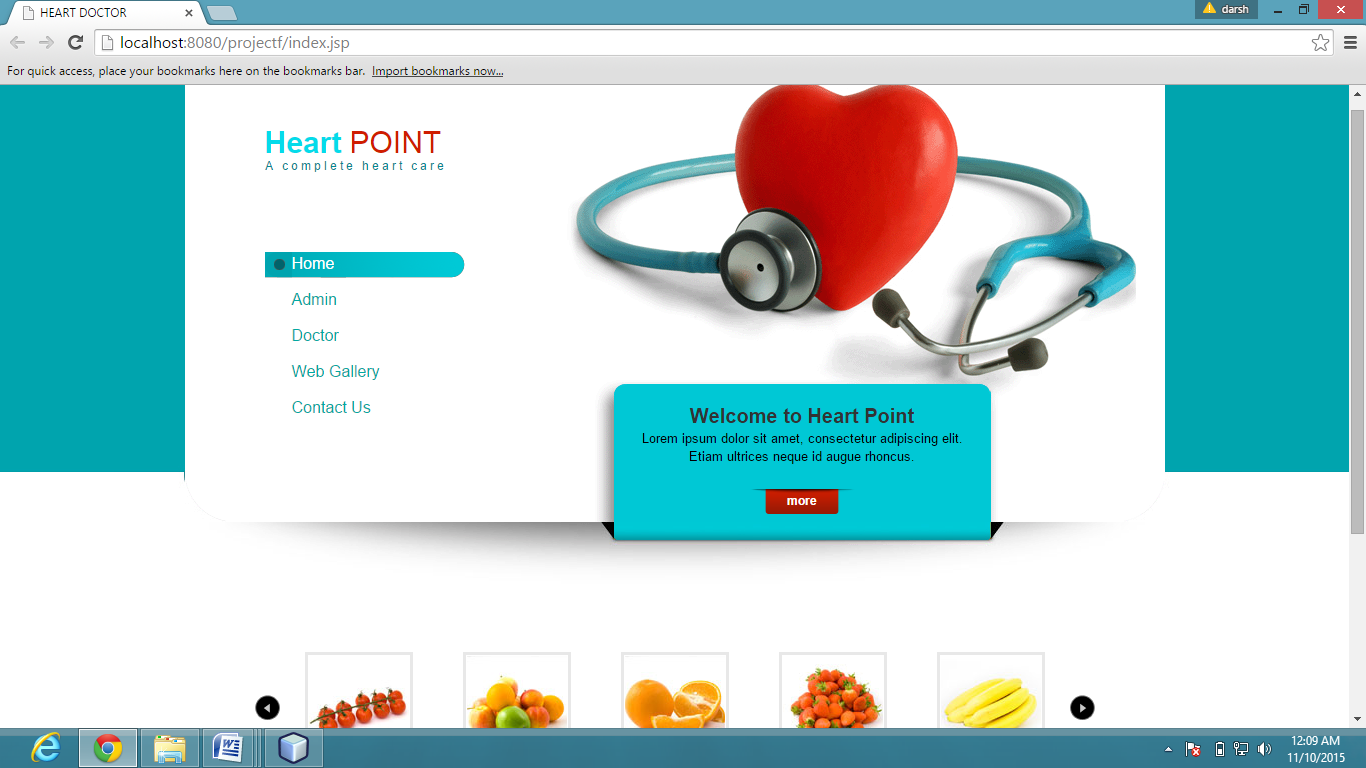


1. **System Design**

**5.1 System Application Design**



When click on the home page



**5.2 Database Design**

Data Dictionary

Table no-1 Doctor Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data type** | **Size** | **Constrain** | **Reference** |
| Doid | Integer | 10 | Primary key |  |
| uname | Varchar | 20 | Not null |  |
| password | Varchar | 10 | Not null |  |
| doname | Varchar | 10 | Not null |  |
| contactno | Varchar | 20 | Unique key |  |
| qulification | Varchar | 10 | Not null |  |
| emailaddress1 | Varchar | 30 |  |  |
| hname | Varchar | 20 | Not null |  |
| Hcity | Varchar | 10 | Not null |  |
| specialization | Varchar | 10 |  |  |

Table no-2 Patient Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data type** | **Size** | **Constrain** | **Reference** |
| pid | Integer | 15 | Primary key |  |
| doid | Integer | 40 | Foreign Key | Reference from doctor Table |
| puid | Varchar | 10 | Not null |  |
| name | Varchar | 14 | Not null |  |
| emailaddress | Varchar | 14 |  |  |
| weight | Integer | 20 | Super Key |  |
| age | Integer | 18 | Not null |  |
| gender | Varchar | 10 | Not null |  |
| contactno | Varchar | 40 | Not null |  |
| condition | Varchar | 12 |  |  |
| Disease ID | Integer | 10 | Foreign Key | Reference from diseaseTable |
| Medicine ID | Integer | 10 | Foreign Key | Reference from medicine Table |
| dosage | Varchar | 12 | Not null |  |
| city | Varchar | 12 | Not null |  |

Table no-3 Disease Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data type** | **Size** | **Constrain** | **Reference** |
| Disease ID | Integer | 10 | Primary key |  |
| Disease Name | Varchar | 20 | Not null |  |

Table no-4 Medicine Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data type** | **Size** | **Constrain** | **Reference** |
| Disease Id | Integer | 10 | Foreign key | Reference from disease table |
| Medicine ID | Integer | 10 | Not null |  |
| Medicine Name | Varchar | 15 | Not null |  |

## 5.3 Input/Output and Interface Design

**5.3.1 Activity diadgram**

Activity diagrams are normally used for business process modeling, for modeling the logic captured by a single use case or usage scenario, or for modeling the detailed logic of business rule.

**Table: 5.1 Symbols for Activity diagram**

|  |  |
| --- | --- |
| **Symbol Name** | **Symbol** |
| Decision |  |
| Control Flow |  |
| Final State |  |
| Initial State |  |
| State |  |
| Action State |  |
| Fork Line |  |
| Join Line |  |



Fig5.3(a) Admin activity diagram

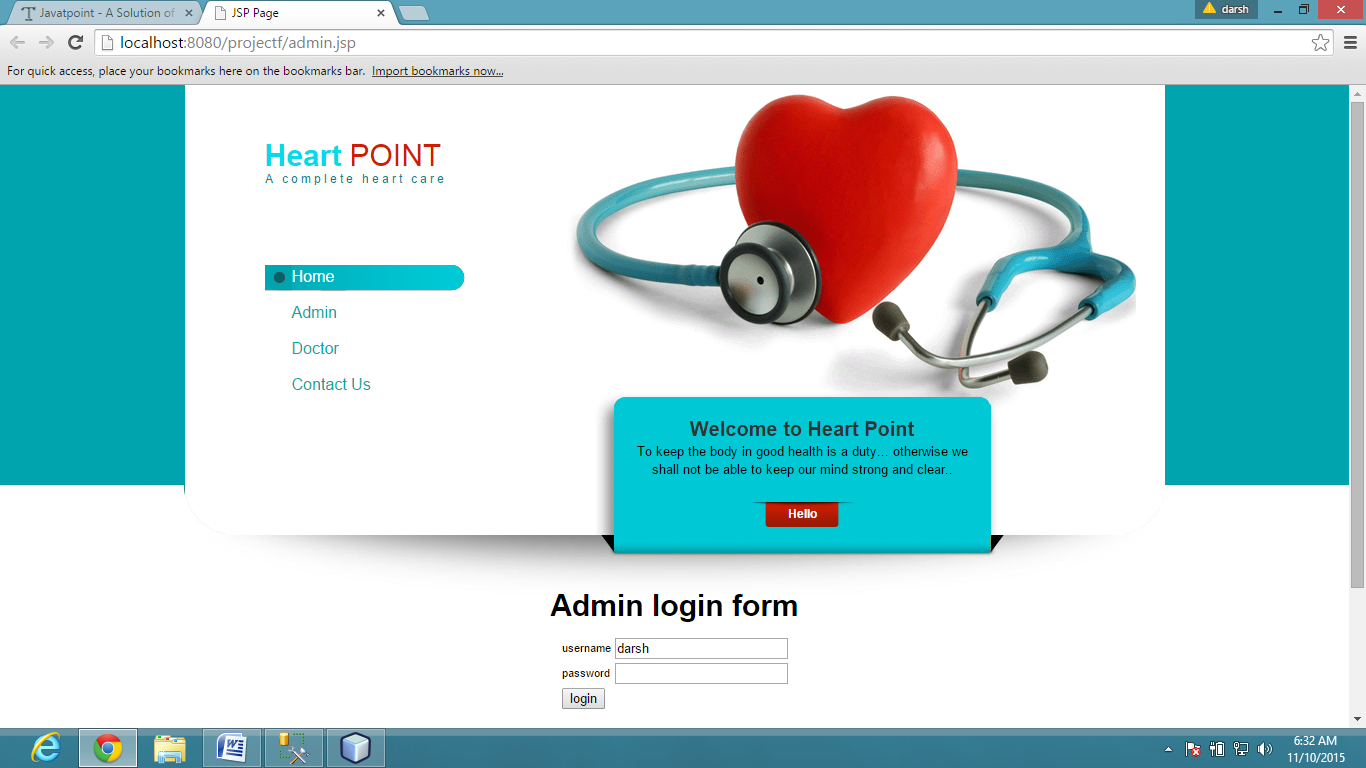


Fig 5.3(b)Doctor activity diagram

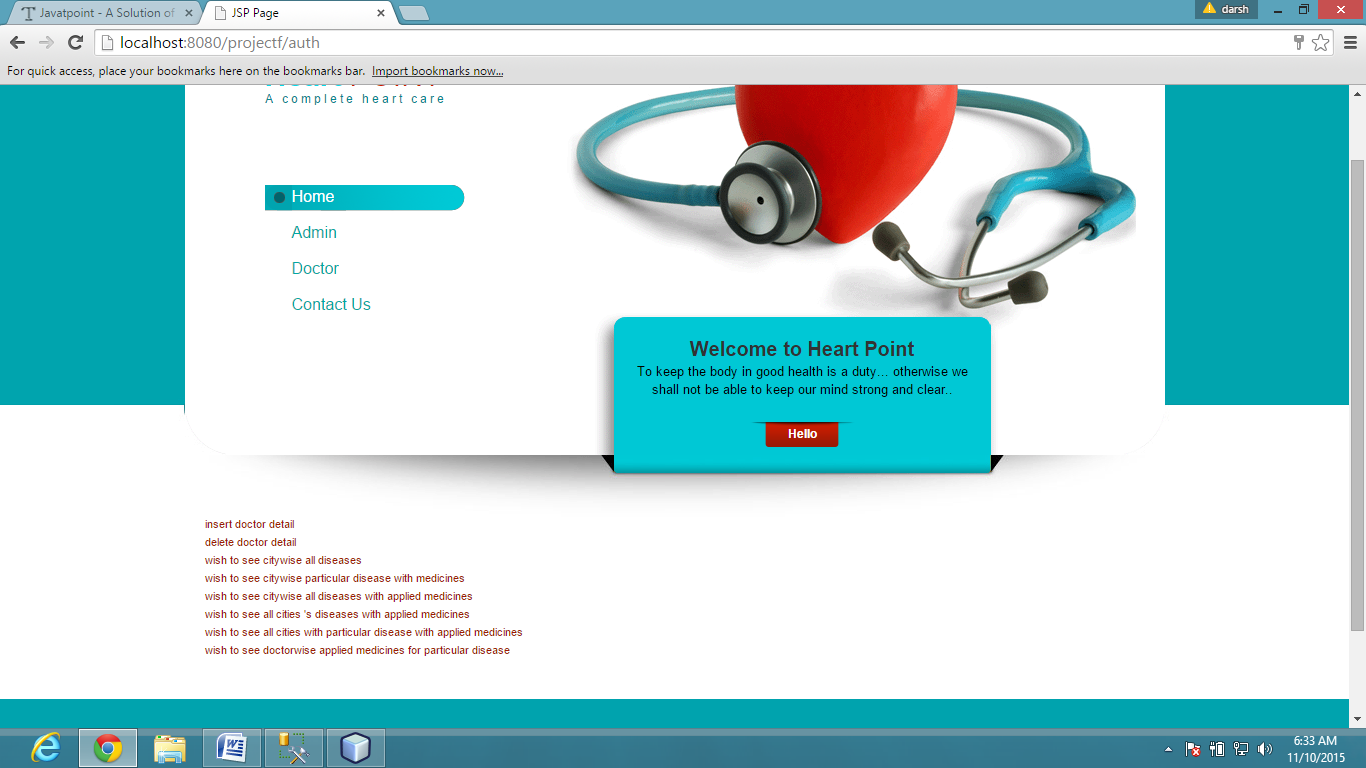
* + 1. **Samples Of Forms, Reports and Interface**

**Admin**

1. **Admin login form**

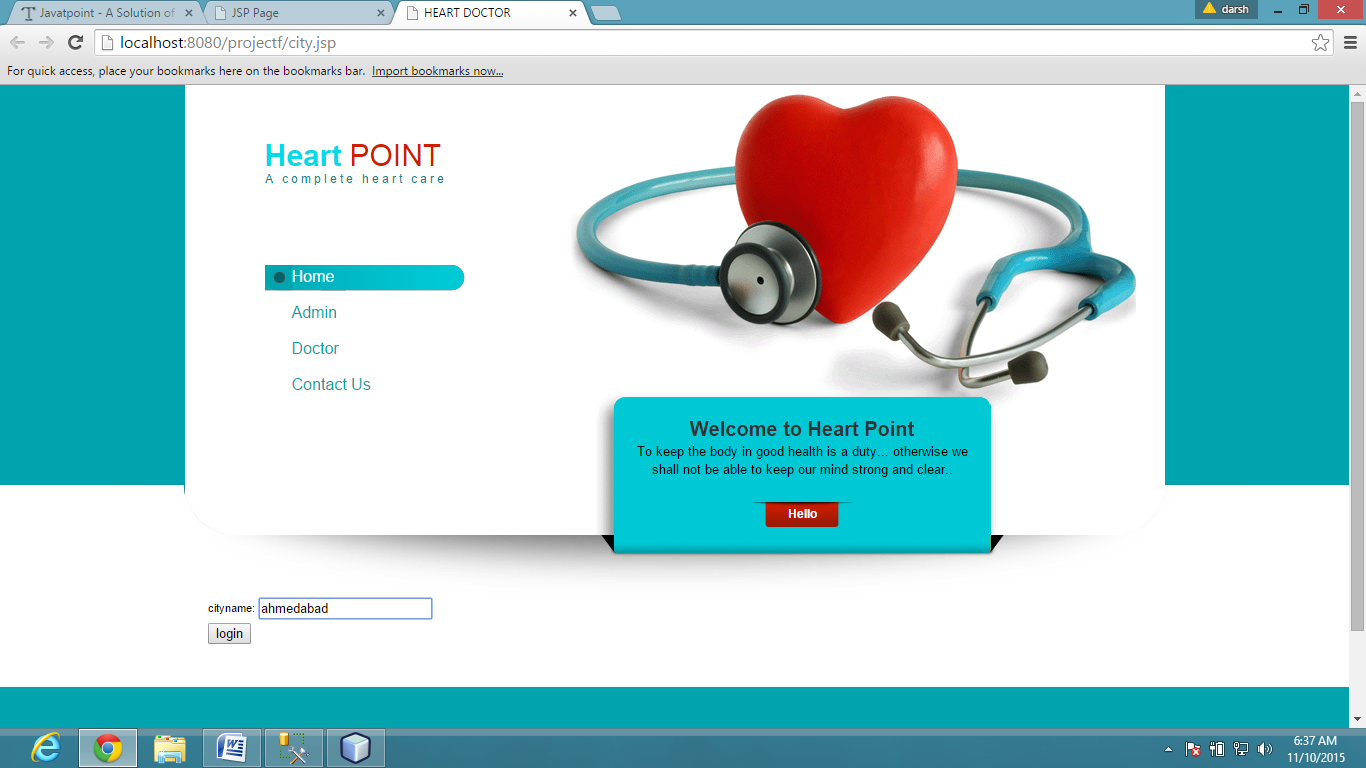
****

**If password and username is correct then see below page**

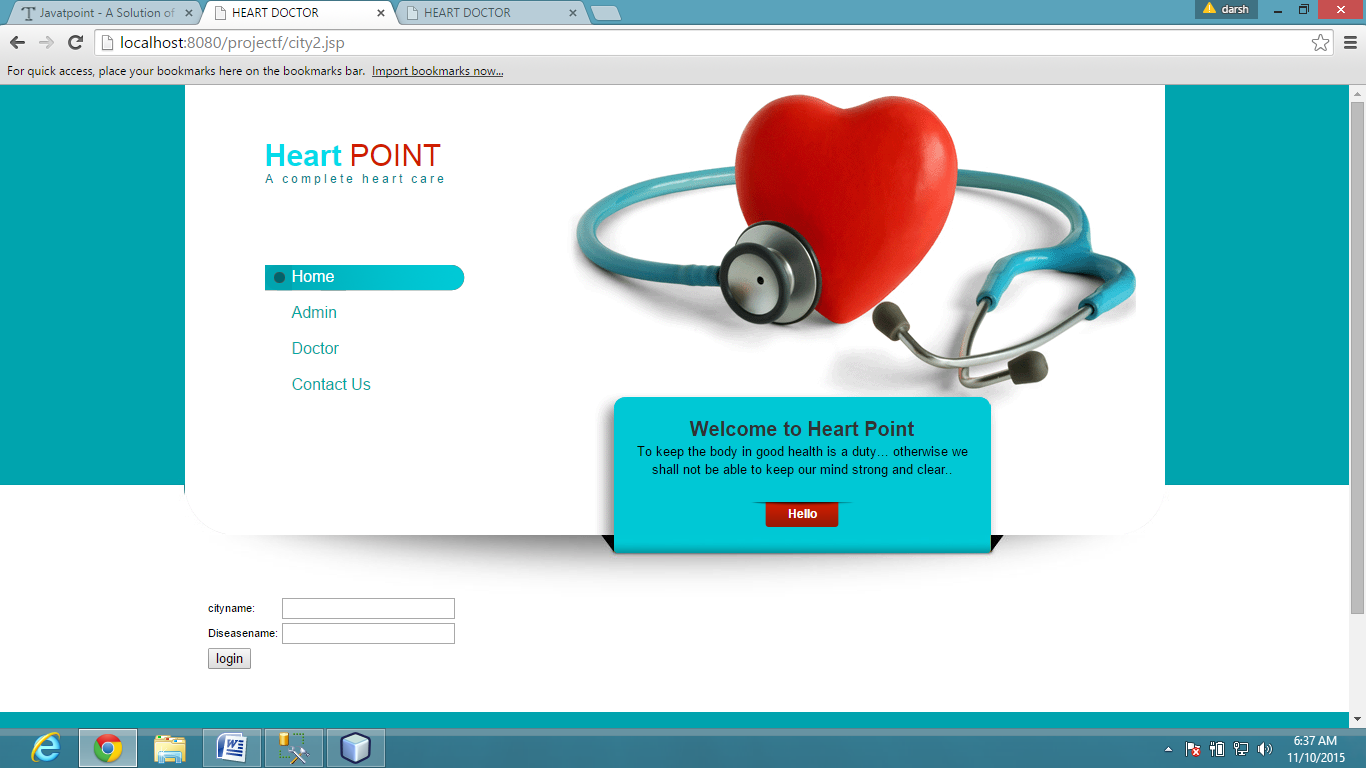
****

Different functionality have different form and page

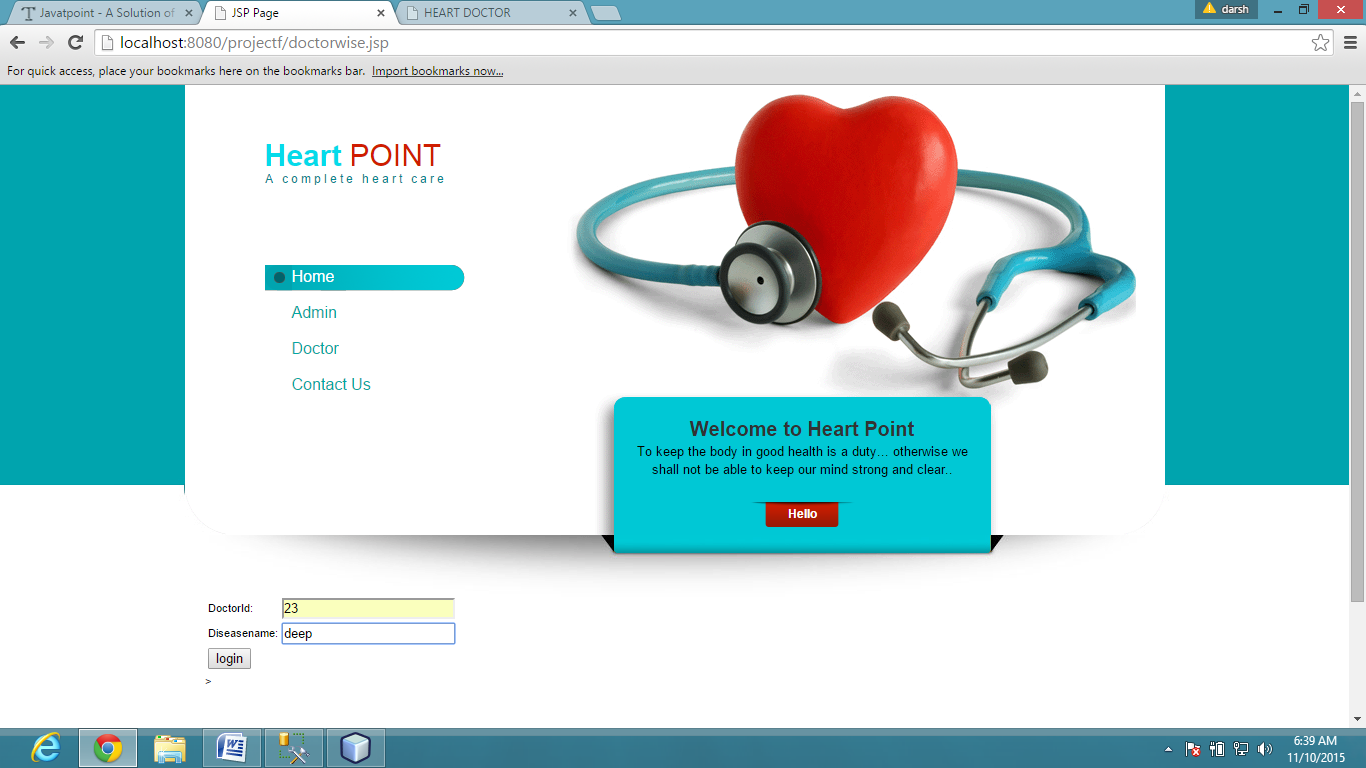
1. To see citywise all diseases



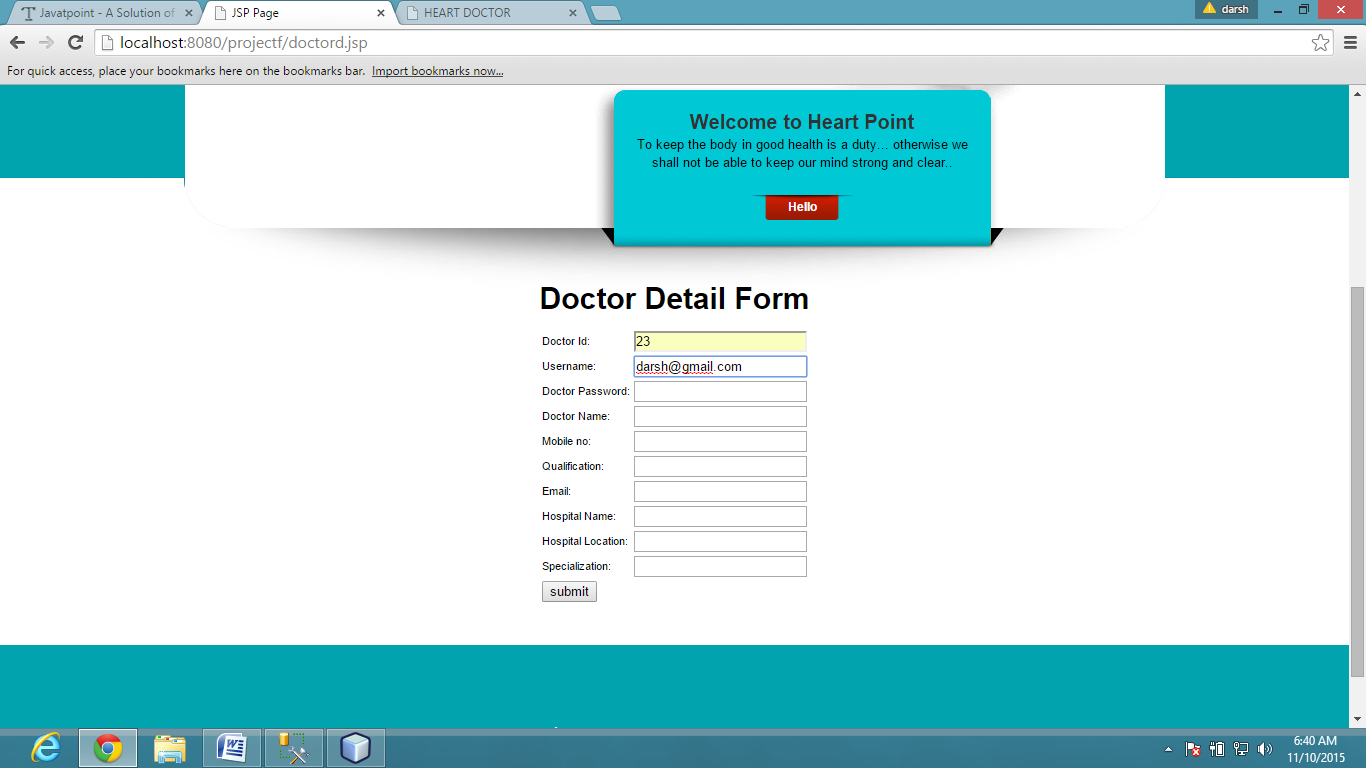
1. to see citywise particular disease with medicines



4) To see doctorwise applied medicines for particular disease.

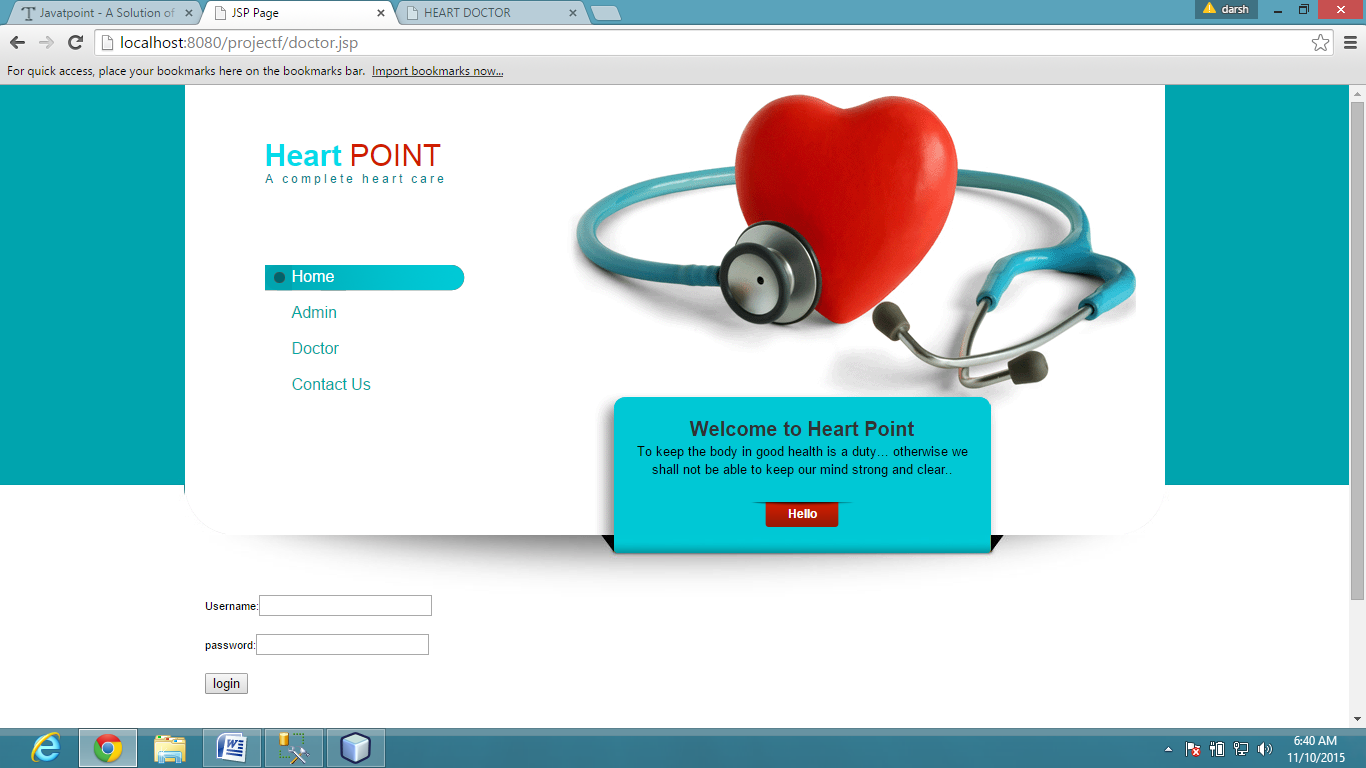


5) Insert doctor detail

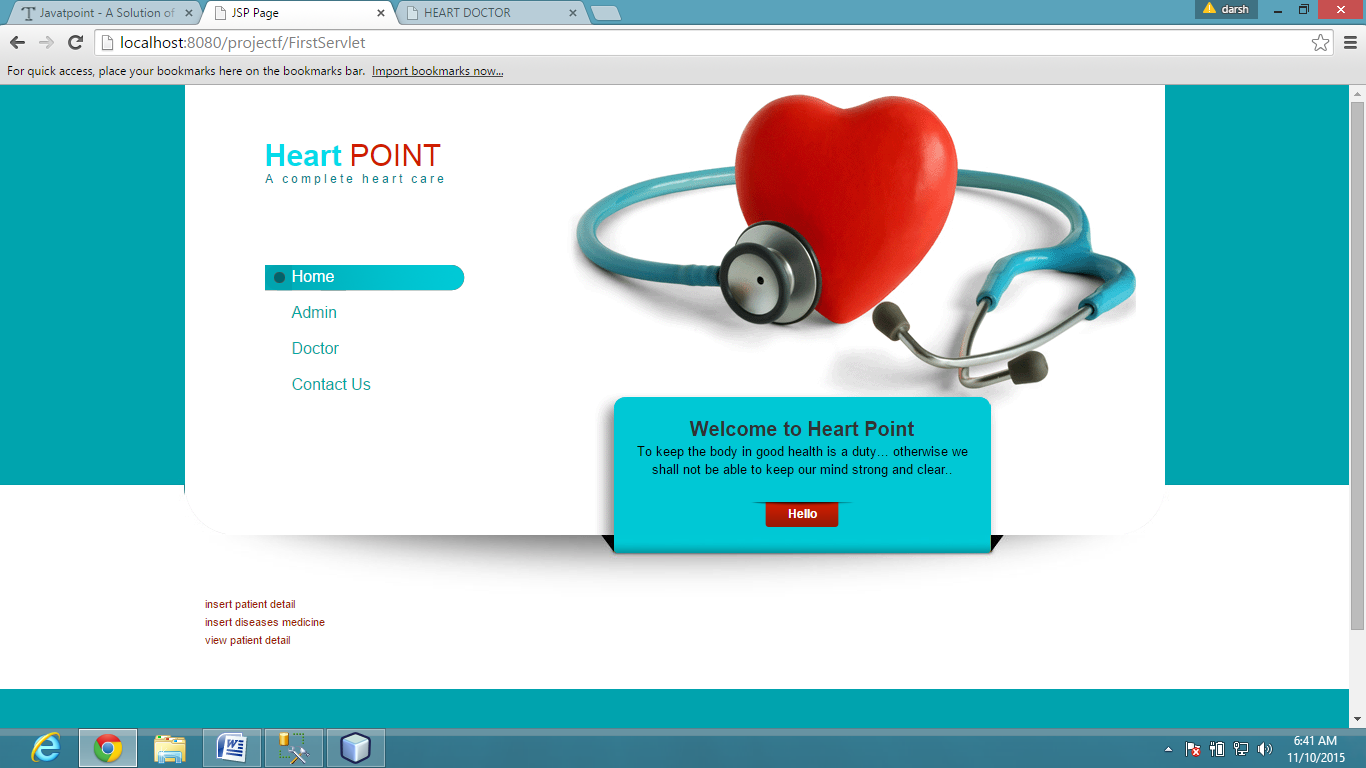


Doctor

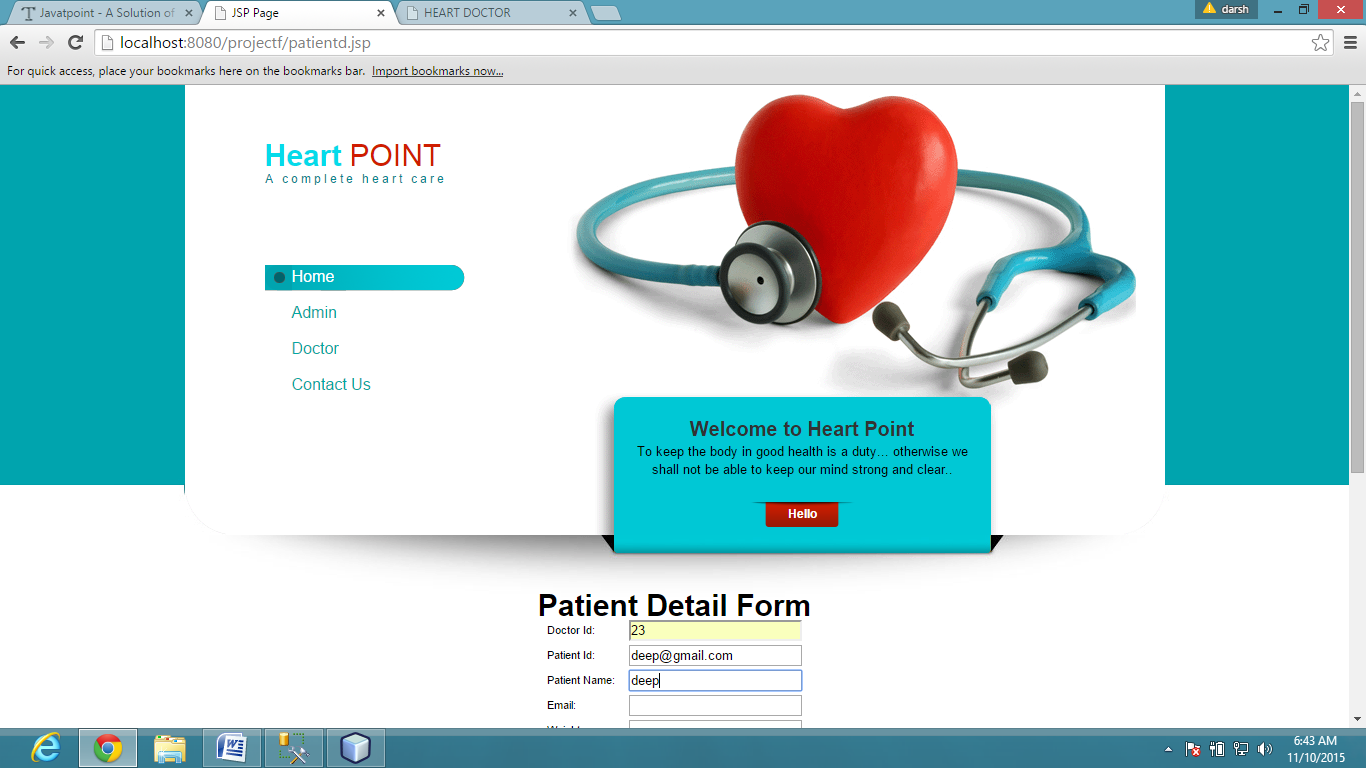
1. doctor login form



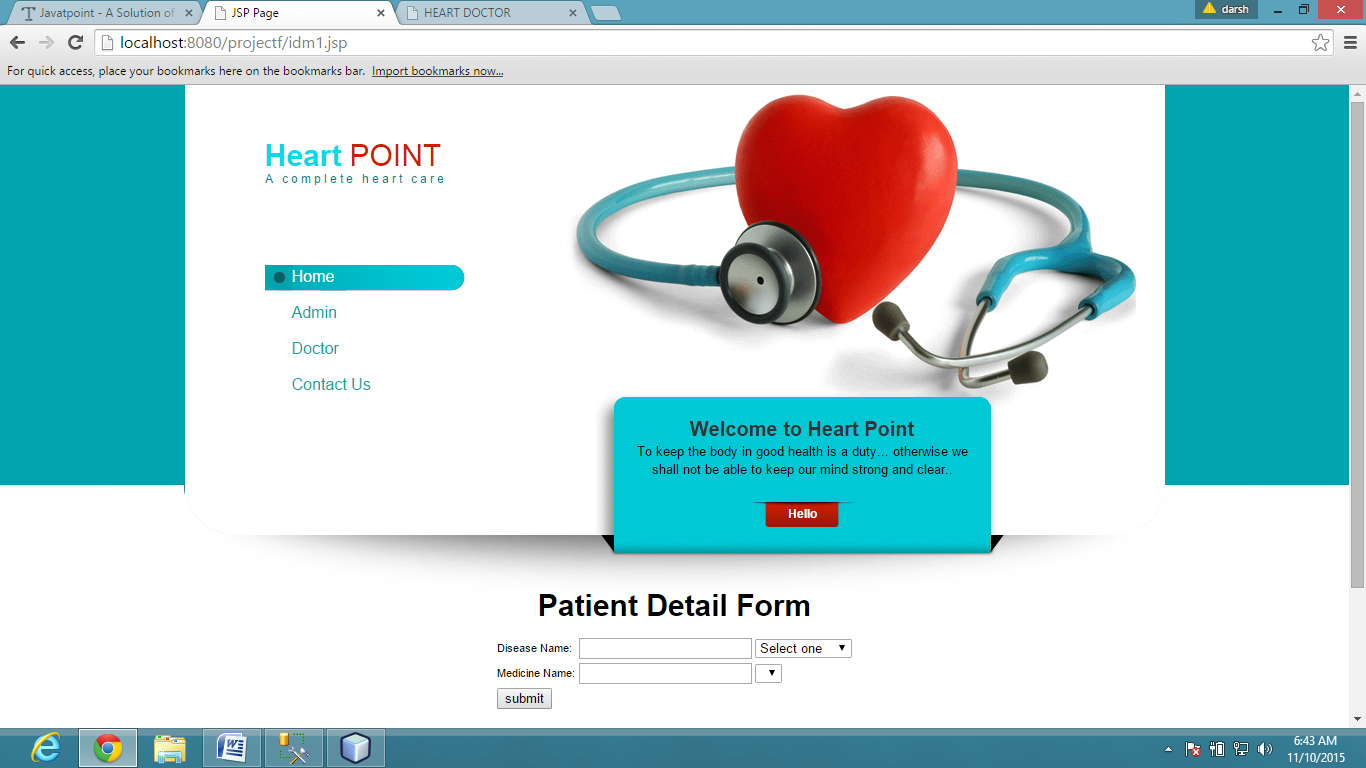
1. if username and password correct see this page



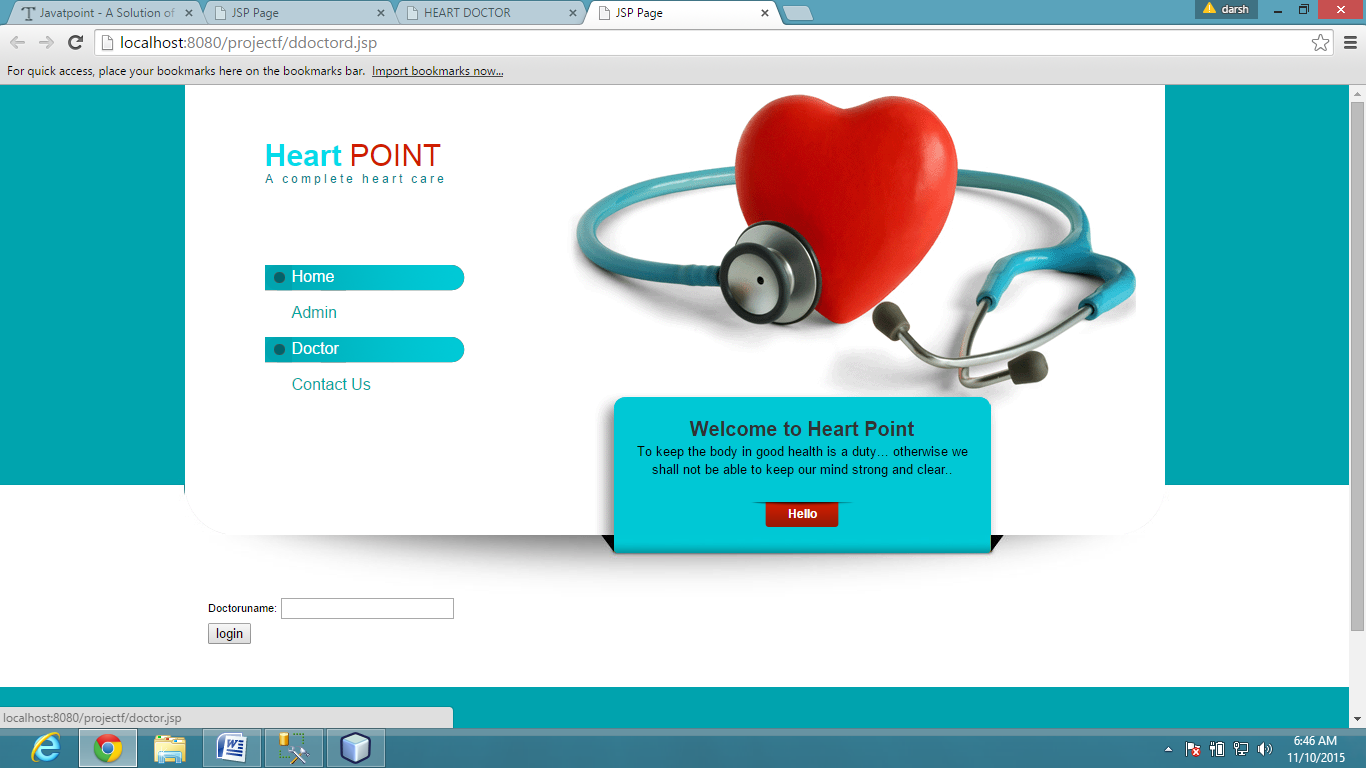
3) insert patient detail



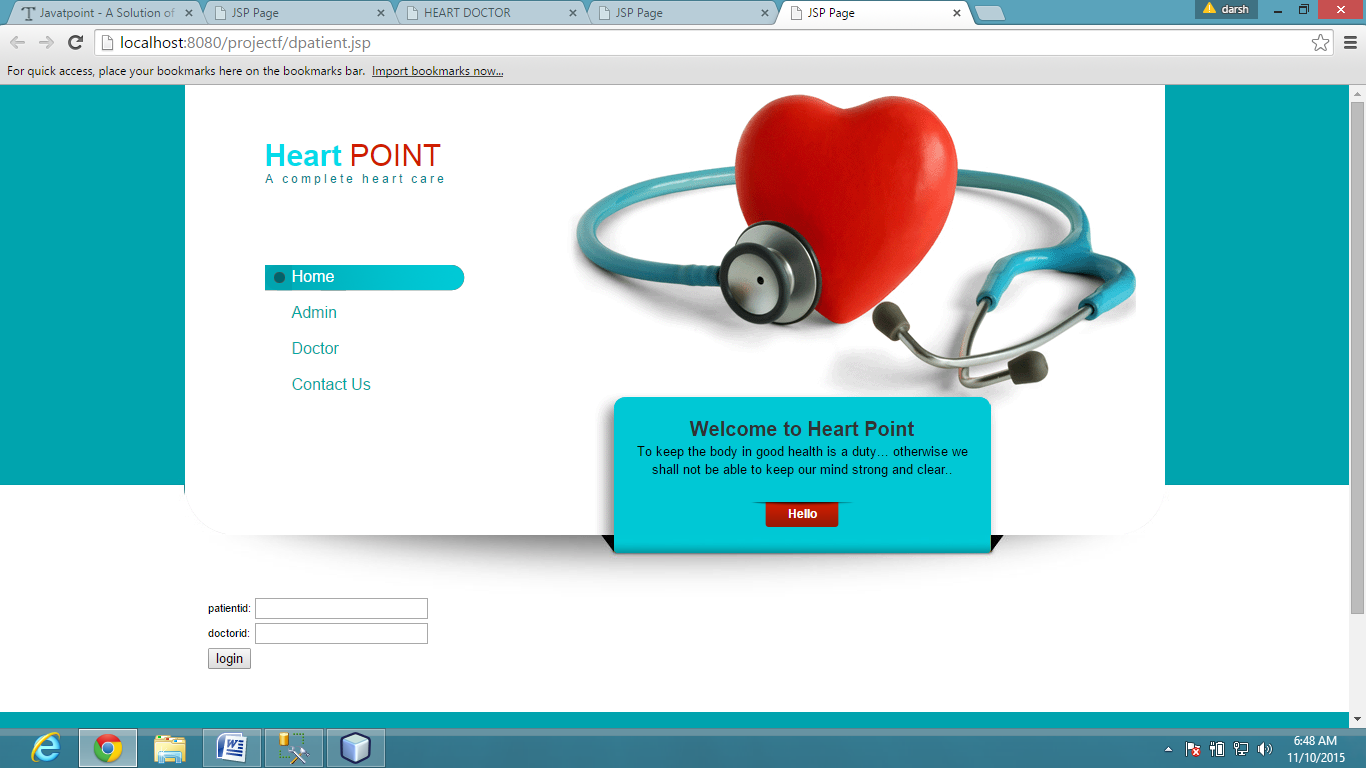
1. insert medicine and disease



5)Delete doctor detail from database



6)delete patient detail form



**6 System Testing**

## 6.1 Testing Method

### Black-Box Testing

The technique of testing without having any knowledge of the interior workings of the application is called black-box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, while performing a black-box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

### White-Box Testing

White-box testing is the detailed investigation of internal logic and structure of the code. White-box testing is also called **glass testing** or **open-box testing**. In order to perform**white-box** testing on an application, a tester needs to know the internal workings of the code.The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

### Grey-Box Testing

Grey-box testing is a technique to test the application with having a limited knowledge of the internal workings of an application. In software testing, the phrase the more you know, the better carries a lot of weight while testing an application.Mastering the domain of a system always gives the tester an edge over someone with limited domain knowledge. Unlike black-box testing, where the tester only tests the application's user interface; in grey-box testing, the tester has access to design documents and the database. Having this knowledge, a tester can prepare better test data and test scenarios while making a test plan.

## 

## 6.2 Testing Strategies

The choice of **test approaches** or **test strategy** is one of the most powerful factor in the success of the test effort and the accuracy of the test plans and estimates. This factor is under the control of the testers and test leaders.

Let’s survey the major types of test strategies that are commonly found:

* **Analytical:** Let us take an example to understand this. The risk-based strategy involves performing a risk analysis using project documents and stakeholder input, then planning, estimating, designing, and prioritizing the tests based on risk. Another analytical test strategy is the requirements-based strategy, where an analysis of the requirements specification forms the basis for planning, estimating and designing tests. Analytical test strategies have in common the use of some formal or informal analytical technique, usually during the requirements and design stages of the project.
* **Model-based:** Let us take an example to understand this. You can build mathematical models for loading and response for e commerce servers, and test based on that model. If the behavior of the system under test conforms to that predicted by the model, the system is deemed to be working. Model-based test strategies have in common the creation or selection of some formal or informal model for critical system behaviors, usually during the requirements and design stages of the project.
* **Methodical:** Let us take an example to understand this. You might have a checklist that you have put together over the years that suggests the major areas of testing to run or you might follow an industry-standard for software quality, such as ISO 9126, for your outline of major test areas. You then methodically design, implement and execute tests following this outline. Methodical test strategies have in common the adherence to a pre-planned, systematized approach that has been developed in-house, assembled from various concepts developed inhouse and gathered from outside, or adapted significantly from outside ideas and may have an early or late point of involvement for testing.
* **Process – or standard-compliant:** Let us take an example to understand this. Alternatively, you might adopt one of the agile methodologies such as Extreme Programming. Process- or standard-compliant strategies have in common reliance upon an externally developed approach to testing, often with little – if any – customization and may have an early or late point of involvement for testing.
* **Dynamic:**Let us take an example to understand this. You might create a lightweight set of testing guide lines that focus on rapid adaptation or known weaknesses in software. Dynamic strategies, such as **exploratory testing,**have in common concentrating on finding as many defects as possible during test execution and adapting to the realities of the system under test as it is when delivered, and they typically emphasize the later stages of testing.
* **Consultative or directed:** Let us take an example to understand this. You might ask the users or developers of the system to tell you what to test or even rely on them to do the testing. Consultative or directed strategies have in common the reliance on a group of non-testers to guide or perform the testing effort and typically emphasize the later stages of testing simply due to the lack of recognition of the value of early testing.
* **Regression-averse:** Let us take an example to understand this. You might try to automate all the tests of system functionality so that, whenever anything changes, you can re-run every test to ensure nothing has broken. Regression-averse strategies have in common a set of procedures – usually automated – that allow them to detect regression defects. A regression-averse strategy may involve automating functional tests prior to release of the function, in which case it requires early testing, but sometimes the testing is almost entirely focused on testing functions that already have been released, which is in some sense a form of post release test involvement.

## 6.3 Testing Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Test Cases name | Description | Input | Expected outcome |
| 1)Admin Login | Verify that admin login with valid username and password | Username=xyz and password=123 | On clicking of the login button admin home page appears |
| 2)Doctor Registration | Verify that admin will add all the details in registration page | Username=xyz, [email=xyz@gmail.com](mailto:email=xyz@gmail.com), | On clicking of the submit button home page appears. |
| 3)Doctor Login | Verify that doctor login with valid username and password | Username=xyz, password=123 | On clicking of the login button doctro home page appears. |
| 4)Patient Registration | Verify that doctor will add all the details in registration page | Username=xyz, [email=xyz@gmail.com](mailto:email=xyz@gmail.com), address=Ahmedabad, mob no.-942960530 | On clicking of the submit button home page appears. |

**7 conclusion**

This system is vrery useful to doctor and pharma company.

8 Bibliography

http://www.javatpoint.com/ajax-tutorial

http://www.javatpoint.com/javascript-tutorial