**E-Tender Management System**

**INTRODUCTION:**

E-Tenders management System is an automated method for manual tendering process where tendering process can be done through online process. In present trend manual methods are used for tendering process where users need to submit documents and participate on location for attending tendering process. This is a time taking process and there will be a problem for users who are away from location. In order to solve this problem I proposed a novel approach to submit tenders through online.

Here users need to register their details using Registration form. They have to give email and password in that form which is also used as login details in further. After Completing Registration, user must login to see the latest tender updates. If they want to submit the tender, then go to the tender form and fill that form. The person who quotes the less quotation will obviously get the tender rights. The administrator will have all the rights to approve or Reject the tender.

**SYSTEM ANALYSIS**

**EXISTING SYSTEM:**

In the existing system, manual methods are used for tendering process where users need to submit documents and participate on location for attending tendering process. This is a time taking process and there will be a problem for users who are away from location.

**PROPOSED SYSTEM:**

In this project, I proposed a novel approach to submit the Tenders through online. It is the most easiest and secured process. Everything will be done through online only. It is very useful for Government as well as private firms to take tender process very easily. By using this we can store all the details of the people who participated in tender process in separate Database. That means we don’t loss any data regarding to that particular tender in any case. No one can access anyone’s account, because every user has their own login id and password. It gives a good protection to their confidential data like quote amount etc.

**MODULES:**

1. Administrator
2. Client

**Administrator Module**:

Administrator will play a key role in this application. First Administrator will publish the Tender Details. Then he will verify all the tender forms which are submitted by the applicants. He has the right to Accept/Reject the tender form. Administrator will announce the final result about the tender.

**Client Module:**

Here, Client is a person who wants to puts a tender. He has to read the tender which is published by Administrator. Then he needs to register by providing appropriate information. By the registration we can generate a login id and password. By giving those id and password we can submit the tender form by filling the required fields. He can also verify the status of his tender.

**SYSTEM REQUIREMENTS**

**SOFTWARE REQUIREMENTS:**

* Operating System : Windows XP and higher versions
* User Interface : HTML, CSS
* Client-side Scripting : JavaScript
* Programming Language : Java
* Web components : JSP
* Other technologies : JDBC,JAVA MAIL
* IDE/Workbench : Net Beans8.1
* SERVER : APACHE TOMCAT 8.0.28
* Database : MYSQL

**HARDWARE REQUIREMENTS**

* Hard Disk : 40GB
* RAM : 1GB
* Processor : Core2duo And Above

**UML DIAGRAMS:**

The Unified Modeling Language (UML) is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software intensive system under development. UML describes the real time systems it is very important to make a conceptual model and then proceed gradually. Conceptual model of UML can be mastered by learning the following three major elements:

1. UML building blocks
2. Rules to connect the building blocks
3. Common mechanisms of UML

The building blocks of UML can be defined as:

* a) Things
* b) Relationships
* c) Diagrams

# (a) Things:

Things are the most important building blocks of UML. Things can be:

* 1.Structural
* 2.Behavioral
* 3.Grouping
* 4.Annotational

**1. Structural things**: The Structural things define the static part of the model. They represent physical and conceptual elements. Following are the brief descriptions of the structural things.

## A) Class:

Class represents set of objects having similar responsibilities.

class

## B) Interface:

Interface defines a set of operations which specify the responsibility of a class.

Interface

## C) Collaboration:

Collaboration defines interaction between elements.

Collaboration

## D) Use case:

Use case represents a set of actions performed by a system for a specific goal.

Use case

## E) Component:

Component describes physical part of a system.

Component

## F) Node:

A node can be defined as a physical element that exists at run time.



# 2. Behavioral things: A behavioral thing consists of the dynamic parts of UML models. Following are the behavioral things:

## A) Interaction:

Interaction is defined as a behavior that consists of a group of messages exchanged among elements to accomplish a specific task.

Interaction

## B) State machine:

State machine is useful when the state of an object in its life cycle is important. It defines the sequence of states an object goes through in response to events. Events are external factors responsible for state change.



# 3. Grouping things: Grouping things can be defined as a mechanism to group elements of a UML model together. There is only one grouping thing available:

## Package:

Package is the only one grouping thing available for gathering structural and behavioral things.



# 4. Annotational things: Annotational things can be defined as a mechanism to capture remarks, descriptions, and comments of UML model elements. Note is the only one Annotational thing available.

## Note:

A note is used to render comments, constraints etc of an UML element.

Note

# (2) Relationships:

Relationship is another most important building block of UML. It shows how elements are associated with each other and this association describes the functionality of an application.

There are four kinds of relationships available.

## Dependency:

Dependency is a relationship between two things in which change in one element also affects the other one.

Dependency

## Association:

Association is basically a set of links that connects elements of an UML model. It also describes how many objects are taking part in that relationship.

Association

## Generalization:

Generalization can be defined as a relationship which connects a specialized element with a generalized element. It describes inheritance relationship in the world of objects.Generalization

## Realization:

Realization can be defined as a relationship in which two elements are connected. One element describes some responsibility which is not implemented and the other one implements them. This relationship exists in case of interfaces.

Realization

**UML Diagrams**:

UML includes the following nine diagrams and the details are described in the following chapters.

1. **Class diagram:** It describes the structure of a system by showing the system's classes, their attributes, and the relationships among the classes.
2. **Object diagram:** It shows a complete or partial view of the structure of a modelled system at a specific time.
3. **Use case diagram:** It shows the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases.
4. **Sequence diagram:** It shows how objects communicate with each other in terms of a sequence of messages. Also indicates the lifespan of objects relative to those messages
5. **Collaboration diagram:** It is a specific type of interaction diagram, where the focus is on timing constraints.
6. **Activity diagram:** It represents the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.
7. **State chart diagram:** It is standardized notation to describe many systems, from computer programs to business processes.
8. **Deployment diagram**: It serves to model the hardware used in system implementations, and the execution environments and artifacts deployed on the hardware.
9. **Component diagram:** It depicts how a software system is split up into components and shows the dependencies among these components.

**Use case Diagram:**

**User:**

****

**Administrator:**

****

**Sequence Diagram:**

****

**Collaboration Diagram:**

****

**Activity Diagram:**

****

**Deployment Diagram:**

****

**Class Diagram:**

****

**TECHNOLOGY DESCRIPTION**

**Software Description:**

* **HTML:**

Hyper Text Markup Language (HTML) is a language used to create hypertext documents that have hyper links embedded in them. It consists of tags embedded in the text of a document with HTML. We can build web pages or web documents. It is basically a formatting language and not a programming language. The browser reading the document interprets mark up tags to help format the document for subsequent display to a reader. HTML is a language for describing structured documents. HTML is a platform independent. WWW (World Wide Web) pages are written using HTML. HTML tags control in part the representation of the WWW page when view with web browser. The browser interprets HTML tags in the web document and displays it. Different browsers show data differently. Examples of browsers used to be web pages include:

1. Netscape
2. Internet Explorer

* **JAVA:**

Java was developed by James Gosling, Patrick Naught on, Chris Wrath, Ed Frank and Mike Sheridan at SUN Micro Systems Incorporation in 199I.It took 18 months to develop the first working version .This language initially called "OAK", but was renamed " JAVA " In 1995.Before the initial implementation of OAK in 1992 and the public announcement of Java in 1995, many more contributed to the design and evolution of the language.

### The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

* + - * Simple
      * Architecture neutral
      * Object oriented
      * Portable
      * Distributed
      * High performance
      * Interpreted
      * Multithreaded
      * Robust
      * Dynamic
      * Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. The following figure illustrates how this works.



*Fig Java Flow*

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



*Fig. Some OS Supporting Java*

### The Java Platform

A platform is the hardware or software environment in which a program runs. We’ve already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and Mac OS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

* The Java Virtual Machine (JVM)
* The Java Application Programming Interface (Java API)

You’ve already been introduced to the Java VM. It’s the base for the Java platform and is ported onto various hardware-based platforms. The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages. The following figure depicts a program that’s running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware



*Fig 4.1.3. Java Platform*

Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

Java is a powerful but lean object oriented programming language. Lt has generated a lot of excitement because it makes it possible to program for internet by creating applets programs that can be embedded in a web page. The context of an apple is limited only by one’s imagination. Java is more than programming language for writing applets. It is being used more and more for writing standalone application as well.

**Database Description:**

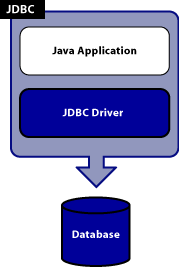
* **JAVA Database Connectivity**: (JDBC)

JDBC is an API developed by Sun Microsystems that provides a standard way to access data using the Java programming language. Using JDBC, an application can access a variety of databases and run on any platform with a Java Virtual Machine. It isn’t necessary to write separate applications to access different database systems, JDBC allows you to write one application that can send SQL statements to different database systems.

* **How Does JDBC Work?**

JDBC makes it possible to do the following things within a Java application:

* Establish a connection with a data source
* Send queries and update statements to the data source
* Process the results



*Fig 4.2.1. JDBC Architecture*

## *Features of JDBC*

* Retrieval of auto-generated keys
* Multiple open result sets

**JDBC Performance**

* Retrieving only required data
* Selecting functions that optimize performance
* Managing connections and updates

### Establishing a Connection

The first thing to do, of course, is to install Java, JDBC and the DBMS on your working machines. Since we want to interface with a MySql database, we would need a driver for this specific database as well. To access a database, a connection must be opened between our program (client) and the database (server). This involves a step:

Class.forName("com.mysql.jdbc.Driver");

**Creating JDBC Statements**

A JDBC Statement object is used to send your SQL statements to the DBMS, and should not to be confused with an SQL statement. A JDBC Statement object is associated with an open connection, and not any single SQL Statement. You can think of a JDBC Statement object as a channel sitting on a connection, and passing one or more of your SQL statements to the DBMS. The following code snippet, using our Connection object con, does it for you:

Statement stmt = con.createStatement();

## *What is JSP?*

Java Server Pages (JSP) is a technology that lets you mix regular, static HTML with dynamically-generated HTML. Many Web pages that are built by CGI programs are mostly static, with the dynamic part limited to a few small locations. But most CGI variations, including servlets, make you generate the entire page via your program, even though most of it is always the same. JSP lets you create the two parts separately

### Handling Errors with Exceptions

The truth is errors always occur in software programs. Often, database programs are critical applications, and it is imperative that errors be caught and handled gracefully. Programs should recover and leave the database in a consistent state. Rollback-s used in conjunction with Java exception handlers is a clean way of achieving such a requirement. The client (program) accessing a server (database) needs to be aware of any errors returned from the server. JDBC give access to such information by providing two levels of error conditions: SQLException and SQLWarning. SQLExceptions are Java exceptions, which, if not handled, will terminate the application. SQLWarnings are subclasses of SQLException, but they represent nonfatal errors or unexpected conditions, and as such, can be ig

**JAVA SERVER PAGES (JSP)**

A server side technology, Java Server Pages are an extension to the Java servlet technology that was developed by Sun. JSPs have dynamic scripting capability that works in tandem with HTML code, separating the page logic from the static elements – the actual design and display of the page – to help make the HTML more functional (i.e. dynamic database queries).

A JSP is translated into Java servlet before being run and it processes HTTP requests and generates responses like any servlet. However, JSP technology provides a more convenient way to code a servlet. Translation occurs the first time the application is run. A JSP translator is triggered by the .jsp file name extension in a URL. JSPs are fully interoperable with servlets. You can include output from a servlet or forward the output to a servlet and a servlet can include output from a JSP or forward output to a JSP.

## *Advantages of JSP*

* Vs. Active Server Pages (ASP). ASP is a similar technology from Microsoft. The advantages of JSP are twofold. First, the dynamic part is written in Java, not Visual Basic or other MS-specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and non-Microsoft Web servers.
* Vs. Pure Servlets. JSP doesn’t give you anything that you couldn’t in principle do with a servlet. But it is more convenient to write (and to modify!) regular HTML than to have a zillion println statements that generate the HTML. Plus, by separating the look from the content you can put different people on different tasks: your Web page design experts can build the HTML, leaving places for your servlet programmers to insert the dynamic content.
* Vs. JavaScript. JavaScript can generate HTML dynamically on the client. This is a useful capability, but only handles situations where the dynamic information is based on the client’s environment. With the exception of cookies, HTTP and form submission data is not available to JavaScript. And, since it runs on the client, JavaScript can’t access server-side resources like databases, catalogs, pricing information, and the like.
* Vs. Static HTML. Regular HTML, of course, cannot contain dynamic information. JSP is so easy and convenient that it is quite feasible to augment HTML pages that only benefit marginally by the insertion of small amounts of dynamic data. Previously, the cost of using dynamic data would preclude its use in all but the most valuable instances.

**JSP Directives**

JSP pages use JSP directives to pass instructions to the JSP engine. These may include the following:

* JSP page directives communicate page-specific information, such as buffer and thread information or error handling.
* Language directives specify the scripting language, along with any extensions.
* The include directive (shown in the example above) can be used to include an external document in the page. A good example is a copyright file or company information, file – it is easier to maintain this file in one central location and include it in several pages than to update it in each JSP page. However, the included file can also be another JSP file.
* A taglib directive indicates a library of custom tags that the page can invoke.

**JSP Tags:**

Most JSP processing will be implemented through JSP-specific XML-based tags. JSP 1.0 includesa number of standard tags, referred to as the core tags. These include:

**jsp:useBean:** This tag declares the usage of an instance of a JavaBeans component. If the Bean does not already exist, then the JavaBean component instantiates and registers the tag.

**Jsp:setProperty:** This sets the value of a property in a Bean.

**Jsp:getProperty:** This tag gets the value of a Bean instance property, converts it to a string, and puts It in the implicit object “out”.

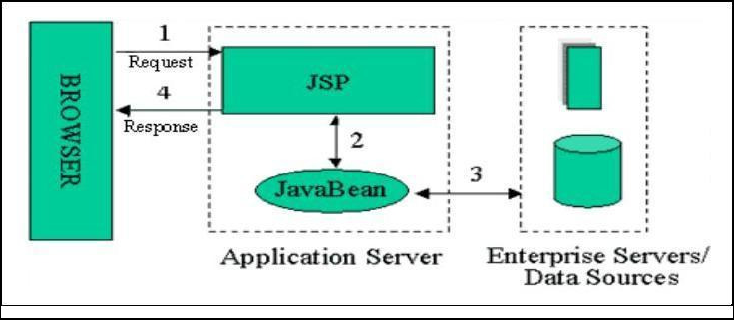
**MVC-1 ARCHITECTURE:**

The main aim of the MVC architecture  is to separate the business logic and application data from the presentation data to the user.

1) **Model:** The model object knows about all the data that need to be displayed. It is model who is aware about all the operations that can be applied to transform that object. It only represents the data of an application. The model represents enterprise data and the business rules that govern access to and updates of this data. Model is not aware about the presentation data and how that data will be displayed to the browser.

2) **View:** The view represents the presentation of the application. The view object refers to the model. It uses the query methods of the model to obtain the contents and renders it. The view is not dependent on the application logic. It remains same if there is any modification in the business logic. In other words, we can say that it is the responsibility of the of the view’s to maintain the consistency in its presentation when the model changes.

3) **Controller:**  Whenever the user sends a request for something then it always go through the controller. The controller is responsible for intercepting the requests from view and passes it to the model for the appropriate action. After the action has been taken on the data, the controller is responsible for directing the appropriate view to the user. In  GUIs, the views and the controllers often work closely together.

****

*Fig 4.2.3: MVC1 Architecture*

Apache Tomcat Server:

Introduction

For administrators and web developers alike, there are some important bits of information you should familiarize yourself with before starting out. This document serves as a brief introduction to some of the concepts and terminology behind the Tomcat container. As well, where to go when you need help.

Apache Tomcat is an open source software implementation of the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed under the Java Community Process. Apache Tomcat powers numerous large-scale, mission-critical web applications across a diverse range of industries and organizations. Apache Tomcat includes tools for configuration and management, but can also be configured by editing [XML](http://en.wikipedia.org/wiki/XML) configuration files.

**Directories and Files**

Throughout the docs, you’ll notice there are numerous references to **$CATALINA\_HOME**. This represents the root of your Tomcat installation. When we say, “This information can be found in your $CATALINA\_HOME/README.txt file” we mean to look at the README.txt file at the root of your Tomcat install

* These are some of the key tomcat directories, all relative to $CATALINA\_HOME:
* /bin – Start-up, shutdown, and other scripts. The \*.sh files (for Unix systems) are functional duplicates of the \*.bat files (for Windows systems). Since the Win32 (10) command-line lacks certain functionality, there are some additional files in here.
* /conf – Configuration files and related DTDs. The most important file in here is server.xml. It is the main configuration file for the container.
* /logs – Log files are here by default.
* /webapps – This is where your webapps go with their root folder.

**Components of Tomcat:**

### Catalina

Catalina is Tomcat’s servlet container. Catalina implements [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems)’ specifications for [servlet](http://en.wikipedia.org/wiki/Java_servlet) and Java Server (JSP). In Tomcat, a Realm element represents a “database” of usernames, passwords, and roles (similar to [Unix groups](http://en.wikipedia.org/w/index.php?title=Unix_group&action=edit&redlink=1)) assigned to those users. Different implementations of Realm allow Catalina to be integrated into environments where such authentication information is already being created and maintained, and then use that information to implement Container Managed Security as described in the Servlet Specification.

* **Coyote**

Coyote is Tomcat’s HTTP Connector component that supports the HTTP1.1 protocol for the web server or application container. Coyote listens for incoming connections on a specific TCP port on the server and forwards the request to the Tomcat Engine to process the request and send back a response to the requesting client.

* **Jasper**

Jasper is Tomcat’s JSP Engine. Tomcat 5.x uses Jasper 2, which is an implementation of the Sun Microsystems’s Java Server Pages 2.0 specification. Jasper parses JSP files to compile them into Java code as servlets (that can be handled by Catalina). At runtime, Jasper detects changes to JSP files and recompiles them.

* **Jasper 2**

From Jasper to Jasper 2, important features were added:

JSP Tag library pooling – Each tag markup in JSP file is handled by a tag handler class. Tag handler class objects can be pooled and reused in the whole JSP servlet.

Background JSP compilation – While recompiling modified JSP Java code, the older version is still available for server requests. The older JSP servlet is deleted once the new JSP servlet has finished being recompiled.

Recompile JSP when included page changes – Pages can be inserted and included into a JSP at runtime. The JSP will not only be recompiled with JSP file changes but also with included page changes.

JDT Java compiler – Jasper 2 can use the Eclipse JDT (Java Development Tools) Java compiler instead of Ant and javac.

**Deployment Descriptor:**

Java web applications use a deployment descriptor file to determine how URLs map to servlets, which URLs require authentication, and other information. This file is named web.xml, and resides in the app’s WAR under the WEB-INF/ directory. Web.xml is part of the servlet standard for web applications.

A web application’s deployment descriptor describes the classes, resources and configuration of the application and how the web server uses them to serve web requests. When the web server receives a request for the application, it uses the deployment descriptor to map the URL of the request to the code that ought to handle the request.

Here is a simple web.xml example that maps all URL paths (/\*) to the servlet class ‘mysite.server.ComingSoonServlet’ :

<web-app xmlns=”http://java.sun.com/xml/ns/javaee” version=”2.5”>

<servlet>

<servlet-name>comingsoon</servlet-name>

<servlet-class>mysite.server.ComingSoonServlet</servlet-class>

</servlet>

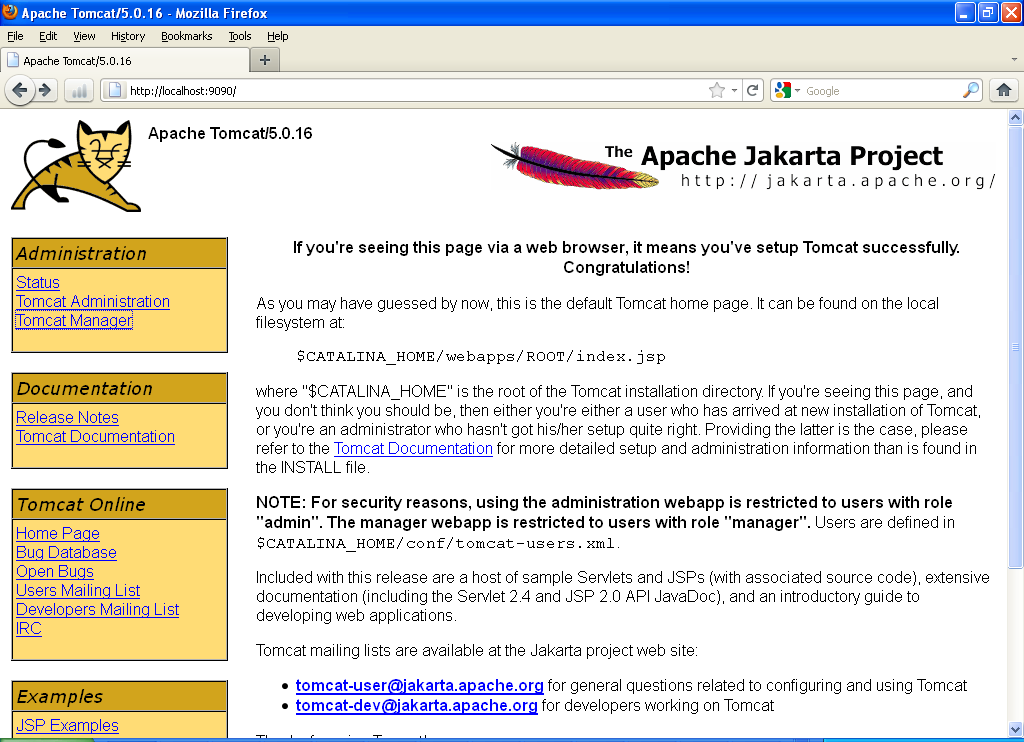
<servlet-mapping>

<servlet-name>comingsoon</servlet-name>

<url-pattern>/\*<url-pattern>

</servlet-mapping>

</web-app>



*Fig 2.5.5.1: Apache Tomcat Server*

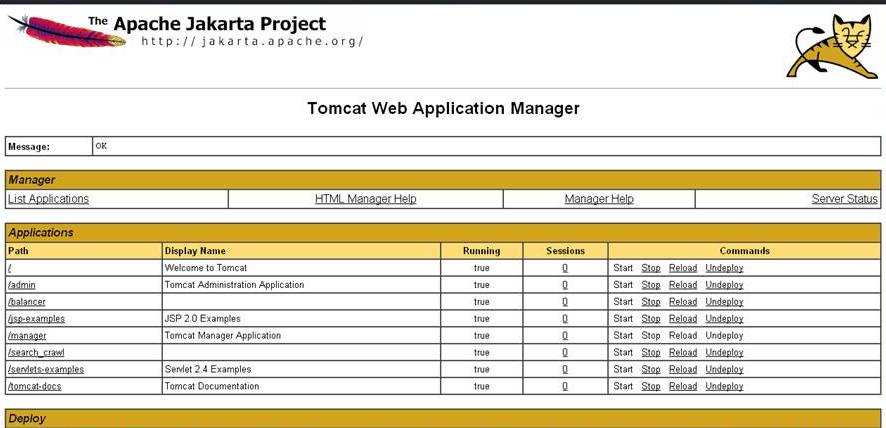
**

Fig 2.5.5.2: Tomcat Application Manager

**SOURCE CODE:**

**Index.jsp:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Home Page</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/customestyle.css" rel="stylesheet">

<link href="fonts/glyphicons-halflings-regular.svg" rel="stylesheet">

<link href="js/custom.js">

<link href='//fonts.googleapis.com/css?family=Advent Pro' rel='stylesheet'>

</head>

<body style="margin:0px;padding:0px;background-color:white;font-family: 'Advent Pro';font-size:17px;">

<div class="jumbotron text-center">

<h1>E tenderS</h1>

</div>

<div class="container" style="background-color:white">

<ul class="nav nav-tabs">

<li class="active"><a href="index.jsp">home</a></li>

<li><a href="PublishTenderDetails.jsp">View Tenders</a></li>

<li><a href="TenderApprovals.jsp"> Tender approvals</a></li>

<li><a href="successStory.jsp">Success Stories</a></li>

<li><a href="Register.jsp"> Register</a></li>

<li><a href="Login.jsp"> Login</a></li>

<li><a href="AboutUs.jsp">About US</a></li>

<li><a href="ContactUs.jsp"> Contact Us</a></li>

</ul>

<div class="row">

<div class="col-md-3">

<img src="images/1.png" style="width:1300px;">

</div>

</div>

</div>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**Register.jsp**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Home Page</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/customestyle.css" rel="stylesheet">

<link href="fonts/glyphicons-halflings-regular.svg" rel="stylesheet">

<link href="js/custom.js">

<link href='//fonts.googleapis.com/css?family=Advent Pro' rel='stylesheet'>

<script>

function validateform()

{

var name=document.registration.name.value;

if(name==null||name==""){

window.alert("Name can't be Blank");

return false;

}

var password=document.registration.password.value;

if(password.length<6){

window.alert("Password must be atleast 6 characters long");

return false;

}

var pinCode=document.registration.pinCode.value;

if(pinCode.length!=6){

window.alert("PIN Code must be 6 Digits Only");

return false;

}

var mobileNumber=document.registration.mobileNumber.value;

if(mobileNumber.length!=10){

window.alert("Mobile Number must be 10 Digits");

return false;

}

var mobileNumber=document.registration.mobileNumber.value;

if(isNaN(mobileNumber)){

window.alert("Mobile Number must contain Digits only");

return false;

}

var estYear=document.registration.estYear.value;

var text=/^[0-9]+$/;

if(estYear !=0){

if ((estYear != "") && (!text.test(estYear))) {

window.alert("Establishment Year is not proper. Please check");

return false;

}

if (estYear.length != 4) {

window.alert("Establishment Year is not proper. Please check");

return false;

}

var current\_year=new Date().getFullYear();

if((estYear > current\_year))

{

window.alert("Incompatible Establishment Year! Please enter Correct Year");

return false;

}

return true;

}

}

</script>

</head>

<body style="margin:0px;padding:0px;background-color:white;font-family: 'Advent Pro';font-size:17px;">

<div class="jumbotron text-center">

<h1>E tenderS</h1>

</div>

<div class="container" style="background-color:white">

<ul class="nav nav-tabs">

<li><a href="index.jsp">home</a></li>

<li><a href="PublishTenderDetails.jsp">View Tenders</a></li>

<li><a href="TenderApprovals.jsp"> Tender approvals</a></li>

<li><a href="successStory.jsp">Success Stories</a></li>

<li class="active"><a href="Register.jsp"> Register</a></li>

<li><a href="Login.jsp"> Login</a></li>

<li><a href="AboutUs.jsp">About US</a></li>

<li><a href="ContactUs.jsp"> Contact Us</a></li>

</ul>

<form name="registration" action="DoRegister.jsp" method="post" enctype="multipart/form-data" onsubmit="return validateform()">

<div class="form-group row">

<div class="tab-pane fade in active">

<h3> Registration Form </h3>

</div>

</div>

<div>

<%

String msg=request.getParameter("msg");

if(msg==null)

{

msg="";

}

%>

<div class="form-group row">

<div class="form-group col-md-3 text-success">

<h3> <%=msg%> </h3>

</div>

</div>

</div>

<div class="form-group row ">

<div class="form-group col-md-6">

<label>Name</label>

<input type="text" name="name" class="form-control" required="">

</div>

</div>

<div class="form-group row ">

<div class="form-group col-md-6">

<label>Company Name</label>

<input type="text" name="companyName" class="form-control" required="">

</div>

</div>

<div class="form-group row ">

<div class="form-group col-md-6">

<label>Register Id</label>

<input type="text" name="registerId" class="form-control" required="">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<label>Mobile Number</label>

<input type="text" name="mobileNumber" class="form-control" required="">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<label>City</label>

<select class="form-control" name="city" required="">

<option> Select City </option>

<option> Vijayawada </option>

<option> Hyderabad </option>

<option> Bangalore </option>

<option> Chennai </option>

<option> Mumbai </option>

<option> Kolkata </option>

<option> Goa </option>

<option> Kochi </option>

</select>

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<label> State </label>

<select class="form-control" name="state" required="">

<option> Select State </option>

<option>ANDHRA PRADESH</option>

<option>TELANGANA</option>

<option>KARNATAKA</option>

<option>TAMILANADU</option>

<option>MAHARASHTRA</option>

<option>WESTBENGAL</option>

<option>GOA</option>

<option>KERALA</option>

</select>

</div>

</div>

<div class="form-group row ">

<div class="form-group col-md-6">

<label>PIN Code</label>

<input type="text" name="pinCode" class="form-control" required="">

</div>

</div>

<div class="form-group row ">

<div class="form-group col-md-6">

<label>PAN Number</label>

<input type="text" name="panNumber" class="form-control" required="">

</div>

</div>

<div class="form-group row ">

<div class="form-group col-md-6">

<label>Establishment Year</label>

<input type="text" name="estYear" class="form-control" required="">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<label> Type Of Bidder </label>

<input type="text" readonly="" name="radioopt" value="indian" class="form-control" required="">

</div>

</div>

<div class="form-group row" >

<div class="form-group col-md-6">

<label>Is Your Company ISO Certified?</label><br><br>

<label class="radio-inline col-md-2">

<input type="radio" name="radioopt1" id="yes" value="yes" onclick="document.getElementById('upload').style.display='block'">YES

</label>

<label class="radio-inline">

<input type="radio" name="radioopt1" id="no" value="no" onclick="document.getElementById('upload').style.display='none'">NO

</label>

</div>

</div>

<div class="form-group row" id="upload">

<div class="form-group col-md-6">

<label>Upload ISO Certificate</label>

<input type="file" class="form-control" name="file1" value="noImage">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<label>Email</label>

<input type="email" name="email" class="form-control" required="">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<label>Password</label>

<input type="password" name="password" class="form-control" required="">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<div class="checkbox">

<h4>If the filled details was fake then we will take judicial action</h4>

<label><input type="checkbox" name="check" required="">I agree</label>

</div>

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<input type="submit" value="Submit" class="btn btn-success">

<input type="reset" value="Reset" class="btn btn-warning">

</div>

</div>

</form>

</div><!--

-->

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**Login.jsp**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Home Page</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/customestyle.css" rel="stylesheet">

<link href="fonts/glyphicons-halflings-regular.svg" rel="stylesheet">

<link href="js/custom.js">

<link href='//fonts.googleapis.com/css?family=Advent Pro' rel='stylesheet'>

</head>

<body style="margin:0px;padding:0px;background-color:white;font-family: 'Advent Pro';font-size:17px;">

<div class="jumbotron text-center">

<h1>E tenderS</h1>

</div>

<div class="container" style="background-color:white">

<ul class="nav nav-tabs">

<li><a href="index.jsp">home</a></li>

<li><a href="PublishTenderDetails.jsp">View Tenders</a></li>

<li><a href="TenderApprovals.jsp"> Tender approvals</a></li>

<li><a href="successStory.jsp">Success Stories</a></li>

<li><a href="Register.jsp"> Register</a></li>

<li class="active"><a href="Login.jsp"> Login</a></li>

<li><a href="AboutUs.jsp">About US</a></li>

<li><a href="ContactUs.jsp"> Contact Us</a></li>

</ul>

<div class="container">

<form action="LoginCheck.jsp" method="post">

<div class="form-group row">

<div class=" tab-pane fade in active">

<h3>Login Details</h3>

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-4">

<label>Email</label>

<input type="email" class="form-control" name="email">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-4">

<label>Password</label>

<input type="password" class="form-control" name="password">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<input type="submit" value="Submit" class="btn btn-default">

<input type="reset" value="Reset" class="btn btn-default">

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-6">

<%

String msg=request.getParameter("msg");

if(msg==null)

{

msg=" ";

}

else{

%>

<div class="form-group row">

<div class="form-group col-md-3 text-danger">

<h4> <%=msg%> </h4>

</div>

</div>

<%

}

%>

</div>

</div>

</form>

</div>

</div>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**TenderAnnouncement.jsp**

<!DOCTYPE html>

<%

String msg=request.getParameter("msg");

if(msg==null)

{

msg="";

}

%>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Home Page</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/customestyle.css" rel="stylesheet">

<link href="fonts/glyphicons-halflings-regular.svg" rel="stylesheet">

<link href="js/custom.js">

<link href='//fonts.googleapis.com/css?family=Advent Pro' rel='stylesheet'>

</head>

<body style="margin:0px;padding:0px;background-color:white;font-family: 'Advent Pro';font-size:17px;">

<div class="jumbotron text-center">

<h1>E tenderS</h1>

</div>

<div class="container" style="background-color:white">

<ul class="nav nav-tabs">

<li><a href="AdminHome.jsp">home</a></li>

<li class="active"><a href="TenderAnnouncement.jsp">Publish Tender Details</a></li>

<li><a href="AdminApprovalTenders.jsp"> Tender approvals</a></li>

<li><a href="Logout.jsp">Logout </a></li>

</ul>

<form action="DoTenderAnnouncement.jsp" method="post" name="tenderAnnounce" enctype="multipart/form-data" onsubmit="return DateCheck()">

<div class="container">

<div class="form-group row">

<div class="form-group col-md-3 text-success">

<h4><%=msg%></h4>

</div>

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label> Tender Id</label>

<input type="text" class="form-control" name="tenderId" required="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label> Start Date </label>

<input type="date" class="form-control" name="startDate" required="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label> End Date </label>

<input type="date" class="form-control" name="endDate" required="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label> Upload Tender Document </label>

<input type="file" class="form-control" name="file" required="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label> Department</label>

<input type="text" class="form-control" name="department" required="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<input type="submit" value="Submit" class="btn btn-success">

<input type="reset" value="Reset" class="btn btn-warning" >

</div>

</div>

</form>

</div>

<script>

function DateCheck()

{

var startDate= document.tenderAnnounce.startDate.value;

var endDate= document.tenderAnnounce.endDate.value;

//

// alert(startDate);

// alert(endDate);

var eDate = new Date(endDate);

var sDate = new Date(startDate);

if(startDate!=''&& sDate>eDate)

{

alert("Please ensure that the End Date is greater than Start Date.");

return false;

}

var startDate=document.tenderAnnounce.startDate.value;

//alert(startDate);

var todaydate = new Date();

var day = todaydate.getDate();

var month = todaydate.getMonth() + 1;

var year = todaydate.getFullYear();

var datestring = year + "-" + month + "-" + day;

//alert(datestring);

// dateString i.e today

// startDate i.e tender start Date

if( new Date(startDate) < new Date(datestring)){

alert("Please Enter Valid Start Date");

return false;

}

//return false;

}

</script>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**TenderApplication.jsp:**

<%@page import="java.sql.ResultSet"%>

<%@page import="mts.DAO"%>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Home Page</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/customestyle.css" rel="stylesheet">

<link href="fonts/glyphicons-halflings-regular.svg" rel="stylesheet">

<link href="js/custom.js">

<link href='//fonts.googleapis.com/css?family=Advent Pro' rel='stylesheet'>

</head>

<body style="margin:0px;padding:0px;background-color:white;font-family: 'Advent Pro';font-size:17px;">

<div class="jumbotron text-center">

<h1>E tenderS</h1>

</div>

<div class="container" style="background-color:white">

<ul class="nav nav-tabs">

<li><a href="UserLogin.jsp">home</a></li>

<li class="active"><a href="TenderApplication.jsp">Tender Form</a></li>

<li><a href="ShareInfo.jsp">Share Your Story</a></li>

<li><a href="ViewStatus.jsp"> Status</a></li>

<li><a href="Logout.jsp">Logout </a></li>

</ul>

<div class="container">

<%

String msg=request.getParameter("msg");

if(msg==null)

{

msg=" ";

}

%>

<%

String mail=(String)session.getAttribute("email");

//out.println(mail);

DAO dao=new DAO();

ResultSet rs=dao.getPublishedData();

ResultSet ra=dao.getDetails(mail);

while(ra.next())

{

%>

<div >

<form action="RegTenderForm.jsp" name="tenderForm" method="post" onsubmit="return validate(this);">

<div class="form-group row">

<div class="tab-pane fade in active">

<h3> Tender Application </h3>

</div>

</div>

<div class="form-group row">

<div class="form-group col-md-3 text-success">

<h4> <%=msg%> </h4>

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>Tender Id</label>

<select name="tenderId" class="form-control" required="">

<option value="-1" selected="selected">Select a Tender Id</option>

<% while(rs.next())

{

%>

<option><%=rs.getString(2)%></option>

<%

}

%>

</select>

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>Company Name</label>

<input type="text" name="companyName" value="<%=ra.getString(2)%>" required="" class="form-control" readonly="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>Company Register Number</label>

<input type="text" name="companyRegisterNumber" value="<%=ra.getString(3)%>" required="" class="form-control" readonly="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>Location</label><br>

<input type="text" name="location" value="<%=ra.getString(5)%>" required="" class="form-control" readonly="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>Pincode</label>

<input type="text" name="pincode" value="<%=ra.getString(7)%>" required="" class="form-control" readonly="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>PAN Number</label>

<input type="text" name="panNumber" value="<%=ra.getString(8)%>" required="" class="form-control" readonly="">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>Total Experience</label>

<input type="text" name="experience" required="" class="form-control">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<label>Quote Amount</label>

<input type="text" name="quoteAmount" required="" class="form-control">

</div>

</div>

<div class="form-group row">

<div class="col-md-6">

<input type="submit" value="Submit" class="btn btn-default btn-lg">

<input type="reset" value="Clear" class="btn btn-default btn-lg">

</div>

</div>

</form>

<%

}

%>

</div>

</div>

</div>

<script>

function validate()

{

if( document.tenderForm.tenderId.value == "-1" )

{

alert( "Please select TenderId!" );

return false;

}

var experience=document.tenderForm.experience.value;

if(isNaN(experience)){

alert("Total Experience must be in digits only");

return false;

}

var quoteAmount=document.tenderForm.quoteAmount.value;

if(isNaN(quoteAmount)){

alert("Quote Amount must be in digits only");

return false;

}

}

</script>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**PublishTenderDetails:**

<%@page import="java.sql.ResultSet"%>

<%@page import="mts.DAO"%>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Home Page</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/customestyle.css" rel="stylesheet">

<link href="fonts/glyphicons-halflings-regular.svg" rel="stylesheet">

<link href="js/custom.js">

<link href='//fonts.googleapis.com/css?family=Advent Pro' rel='stylesheet'>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

</head>

<body style="margin:0px;padding:0px;background-color:white;font-family: 'Advent Pro';font-size:17px;">

<div class="jumbotron text-center">

<h1>E tenderS</h1>

</div>

<div class="container" style="background-color:white">

<ul class="nav nav-tabs">

<li><a href="index.jsp">home</a></li>

<li class="active"><a href="PublishTenderDetails.jsp">View Tenders</a></li>

<li><a href="TenderApprovals.jsp"> Tender approvals</a></li>

<li><a href="successStory.jsp">Success Stories</a></li>

<li><a href="Register.jsp"> Register</a></li>

<li><a href="Login.jsp"> Login</a></li>

<li><a href="AboutUs.jsp">About US</a></li>

<li><a href="ContactUs.jsp"> Contact Us</a></li>

</ul>

<%

DAO dao=new DAO();

ResultSet rs=dao.getPublishedData();

%>

<div>

<h3>Tender Announcements</h3> <br>

<%

while(rs.next()){

%>

<div class="row" style="margin-bottom:20px;">

<div class="btn col-md-2 col-md-offset-1 btn-warning btn-lg">

<span data-toggle="tooltip" title="Tender Id"><%=rs.getString(6)+"/"+rs.getString(2)%></span>

</div>

<div class="btn col-md-offset-1 col-md-2 btn-info btn-lg">

<span data-toggle="tooltip" title="Date Of Announcement "> <%=rs.getDate(3)%></span>

</div>

<div class="col-md-offset-1 btn col-md-2 btn-danger btn-lg">

<span data-toggle="tooltip" title="Date Of Close"> <%=rs.getDate(4)%> </span>

</div>

<div class="col-md-offset-1 btn col-md-2 btn-success btn-lg">

<a style="color:white;text-decoration:none" href="uploads\\<%=rs.getString(5)%>" target="\_blank">download </a>

</div>

</div>

<%

}

%>

</div>

</div>

<script>

$(document).ready(function(){

$('[data-toggle="tooltip"]').tooltip();

});

</script>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**AdminHome.jsp**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Home Page</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/customestyle.css" rel="stylesheet">

<link href="fonts/glyphicons-halflings-regular.svg" rel="stylesheet">

<link href="js/custom.js">

<link href='//fonts.googleapis.com/css?family=Advent Pro' rel='stylesheet'>

</head>

<body style="margin:0px;padding:0px;background-color:white;font-family: 'Advent Pro';font-size:17px;">

<div class="jumbotron text-center">

<h1>E tenderS</h1>

</div>

<div class="container" style="background-color:white">

<ul class="nav nav-tabs">

<li class="active"><a href="AdminHome.jsp">home</a></li>

<li><a href="TenderAnnouncement.jsp">Publish Tender Details</a></li>

<li><a href="AdminApprovalTenders.jsp">Tender approvals</a></li>

<li><a href="Logout.jsp">Logout </a></li>

</ul>

<div class="container">

<div class="tab-pane fade in active">

<div class="row">

<div class="col-md-3">

<img src="images/admin.jpg" style="width:600px;">

</div>

</div>

</div>

</div>

</div>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**LoginCheck.jsp**

<%@page import="mts.DAO"%>

<%@page import="mts.LoginBean"%>

<%

String email = request.getParameter("email");

session.setAttribute("email", email);

String password = request.getParameter("password");

LoginBean loginBean=new LoginBean(email, password,"");

DAO dao = new DAO();

LoginBean loginBean1=dao.loginCheck(loginBean);

String desig=loginBean1.getDesig();

if(desig!=null)

{

if (desig.equalsIgnoreCase("admin"))

{

response.sendRedirect("AdminHome.jsp?welcome Admin");

}

else if(desig.equalsIgnoreCase("user"))

{

response.sendRedirect("UserLogin.jsp?Wecome");

}

else{

response.sendRedirect("Login.jsp?msg=Invalid Login");

}

}else{

response.sendRedirect("Login.jsp?msg=Invalid Login");

}

%>

**DAO.java**

package mts;

import static java.lang.System.out;

import java.math.BigDecimal;

import java.sql.\*;

import java.util.ArrayList;

import javax.management.Query;

import javax.sql.\*;

public class DAO {

private Connection con;

public DAO()

{

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/tenders", "root", "");

if (con != null){

System.out.print("Connection Established .....");

} else {

System.out.print("Connection not established...");

}

} catch (ClassNotFoundException cne) {

System.out.println("Driver is not Loaded");

cne.printStackTrace();

} catch (SQLException se) {

System.out.println("Connection is not established");

se.printStackTrace();

}

}

public boolean doRegister(DTORegister dtoRegister) {

boolean flag = false;

try {

String qry = "insert into register values(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";

PreparedStatement pstmt = con.prepareStatement(qry);

pstmt.setString(1,dtoRegister.getName());

pstmt.setString(2,dtoRegister.getCompanyName());

pstmt.setString(3,dtoRegister.getRegisterId());

pstmt.setString(4,dtoRegister.getMobileNumber());

pstmt.setString(5,dtoRegister.getCity());

pstmt.setString(6,dtoRegister.getState());

pstmt.setString(7,dtoRegister.getPinCode());

pstmt.setString(8,dtoRegister.getPanNumber());

pstmt.setString(9,dtoRegister.getEstYear());

pstmt.setString(10,dtoRegister.getRadioopt());

pstmt.setString(11,dtoRegister.getRadioopt1());

pstmt.setString(12,dtoRegister.getFile1());

pstmt.setString(13,dtoRegister.getEmail());

pstmt.setString(14,dtoRegister.getPassword());

pstmt.setString(15,dtoRegister.getCheck());

pstmt.setString(16,"user");

int r = pstmt.executeUpdate();

if (r == 1) {

flag = true;

}

} catch (SQLException se) {

se.printStackTrace();

}

return flag;

}

public boolean doTenderForm(DTOTenderForm dtoTenderForm) {

System.out.println(dtoTenderForm);

boolean flag = false;

try {

String qry = "insert into tenderform values(?,?,?,?,?,?,?,?,?,?)";

PreparedStatement pstmt = con.prepareStatement(qry);

pstmt.setString(1, dtoTenderForm.getTenderId());

pstmt.setString(2, dtoTenderForm.getCompanyName());

pstmt.setString(3, dtoTenderForm.getCompanyRegisterNumber());

pstmt.setString(4, dtoTenderForm.getLocation());

pstmt.setInt(5, dtoTenderForm.getPincode());

pstmt.setString(6, dtoTenderForm.getPanNumber());

pstmt.setInt(7, dtoTenderForm.getExperience());

pstmt.setDouble(8, dtoTenderForm.getQuoteAmount());

pstmt.setString(9,"no");

pstmt.setTimestamp(10,new Timestamp(System.currentTimeMillis()));

System.out.println(qry);

int r = pstmt.executeUpdate();

if (r == 1) {

flag = true;

}

} catch (SQLException se) {

se.printStackTrace();

}

return true;

}

public LoginBean loginCheck(LoginBean loginbean) {

boolean flag = false;

LoginBean loginBean=new LoginBean();

try {

String qry = "select \* from register where email=? and password=?";

PreparedStatement pstmt = con.prepareStatement(qry);

pstmt.setString(1, loginbean.getEmail());

pstmt.setString(2, loginbean.getPassword());

ResultSet rs = pstmt.executeQuery();

if (rs.next()) {

String desig=rs.getString("Desig");

loginBean.setDesig(desig);

}

} catch (SQLException se) {

se.printStackTrace();

}

return loginBean;

}

public ResultSet getTendersData(){

ResultSet rs =null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from tenderform where clear='no'");

rs=pstmt.executeQuery();

} catch (Exception e) {

e.getCause();

}

return rs;

}

public int ConformedTenders(String companyRegisterNumber,String pancard,BigDecimal quoteAmount,Timestamp AcceptedTime,String tenderId){

int i=0;

try {

PreparedStatement pstmt=con.prepareStatement("insert into AcceptedTenders values(?,?,?,?,?,?)");

pstmt.setInt(1,0);

pstmt.setString(2,companyRegisterNumber);

pstmt.setString(3,pancard);

pstmt.setBigDecimal(4,quoteAmount);

pstmt.setTimestamp(5,AcceptedTime);

pstmt.setString(6,tenderId);

i=pstmt.executeUpdate();

} catch (Exception e) {

e.printStackTrace();

}

return i;

}

public int tenderAnnounce(String tenderId,String startDate,String endDate,String fileName,String department){

int i=0;

try {

PreparedStatement pstmt=con.prepareStatement("insert into tenderannouncement values(?,?,?,?,?,?)");

pstmt.setInt(1,0);

pstmt.setString(2,tenderId);

pstmt.setString(3,startDate);

pstmt.setString(4,endDate);

pstmt.setString(5,fileName);

pstmt.setString(6,department);

i=pstmt.executeUpdate();

} catch (Exception e) {

e.printStackTrace();

}

return i;

}

public ResultSet getPublishedData(){

ResultSet rs=null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from tenderannouncement");

rs=pstmt.executeQuery();

} catch (Exception e) {

e.printStackTrace();

}

System.out.println("\*\*\*\*\*\*\* publish tender details" +rs);

return rs;

}

public ResultSet getTenderIdFromAcceptedTenders(){

ResultSet rs =null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from acceptedtenders");

rs=pstmt.executeQuery();

} catch (Exception e) {

e.printStackTrace();

}

System.out.println("\*\*\*\*\*\*\*" +rs);

return rs;

}

public ResultSet checkTenderIdExistancy(String tenderId){

ResultSet rs =null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from tenderannouncement where tenderId='"+tenderId+"'");

rs=pstmt.executeQuery();

} catch (Exception e) {

e.printStackTrace();

}

return rs;

}

public ResultSet getConformTenders(String tenderId){

ResultSet rs =null;

String qry="select \* from acceptedtenders where tenderId='"+tenderId+"'";

try {

PreparedStatement pstmt=con.prepareStatement(qry);

rs=pstmt.executeQuery();

} catch (Exception e) {

e.printStackTrace();

}

System.out.println("++++++++++++++" +qry);

return rs;

}

public ResultSet getConformAllTenders(){

ResultSet rs =null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from acceptedtenders");

rs=pstmt.executeQuery();

} catch (Exception e) {

e.printStackTrace();

}

return rs;

}

public int updateRecords(String tenderId){

int i=0;

try{

PreparedStatement pstmt=con.prepareStatement("update tenderform set clear='yes' where tenderId='"+tenderId+"'");

i=pstmt.executeUpdate();

}catch(Exception e){

e.printStackTrace();

}

return i;

}

public int tenderCount(String tenderId){

int i=0;

ResultSet rs=null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from tenderform where TenderId=?");

pstmt.setString(1,tenderId);

rs=pstmt.executeQuery();

while(rs.next()) {

i=i+1;

}

} catch (Exception e) {

e.printStackTrace();

}

System.out.println("tenders count \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"+i);

return i;

}

public ArrayList<String> getEmails(){

ArrayList<String> al=new ArrayList<String>();

ResultSet rs=null;

try{

PreparedStatement pstmt=con.prepareStatement("select email from register where Desig=?");

pstmt.setString(1,"user");

rs=pstmt.executeQuery();

while (rs.next()) {

String email = rs.getString(1);

al.add(email);

}

}catch(Exception e){

e.printStackTrace();

}

return al;

}

public ArrayList<String> getEmails(String registerId){

ArrayList<String> al=new ArrayList<String>();

ResultSet rs=null;

try{

PreparedStatement pstmt=con.prepareStatement("select \* from register where registerId=?");

pstmt.setString(1,registerId);

rs=pstmt.executeQuery();

while (rs.next()) {

String email = rs.getString("email");

al.add(email);

}

}catch(Exception e){

e.printStackTrace();

}

return al;

}

public boolean getSuccessStories(DTOShareInfo dtoShareInfo){

boolean flag=false;

dtoShareInfo.toString();

try {

String qry="insert into Info values(?,?,?)";

PreparedStatement pstmt=con.prepareStatement(qry);

pstmt.setInt(1,0);

pstmt.setString(2,dtoShareInfo.getCompanyRegisterNumber());

pstmt.setString(3,dtoShareInfo.getStory());

int i=pstmt.executeUpdate();

if(i>0)

{

flag=true;

}

} catch (Exception e) {

e.printStackTrace();

}

return flag;

}

public ResultSet getAllSuccessStories()

{

ResultSet rs=null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from Info");

rs=pstmt.executeQuery();

} catch (Exception e) {

e.printStackTrace();

}

return rs;

}

public ArrayList<String> getFileFromRegister()

{

ArrayList<String> al=new ArrayList<String>();

ResultSet ra=null;

try {

PreparedStatement pstmt=con.prepareStatement("select \* from register where Desig!='admin' and file1!='noImage'" );

ra=pstmt.executeQuery();

while(ra.next()){

String fileName=ra.getString("file1");

al.add(fileName);

out.println(fileName);

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* first Time \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

}

} catch (Exception e) {

e.printStackTrace();

}

return al;

}

public ResultSet getDetails(String email)

{

ResultSet rs=null;

try {

String qry="select \* from register where email='"+email+"'";

PreparedStatement pstmt=con.prepareStatement(qry);

rs=pstmt.executeQuery();

} catch (Exception e) {

e.printStackTrace();

}

return rs;

}

public String getFileNameUsingregisterId(String registerId){

String fileName="NO-Certificate-Available";

ResultSet rs=null;

String query="select \* from register where registerId='"+registerId+"'";

try {

PreparedStatement pstmt=con.prepareStatement(query);

rs=pstmt.executeQuery();

if(rs.next()){

fileName=rs.getString(12);

}

} catch (Exception e) {

}

System.out.println("File Name......."+query);

System.out.println("File Name......."+fileName);

return fileName;

}

}

**DTORegister.java**

package mts;

public class DTORegister {

private String name;

private String companyName;

private String registerId;

private String mobileNumber;

private String city;

private String state;

private String pinCode;

private String panNumber;

private String estYear;

private String radioopt;

private String radioopt1;

private String file1;

private String email;

private String password;

private String check;

private String desig;

public DTORegister() {

}

public DTORegister(String name, String companyName, String registerId, String mobileNumber, String city, String state, String pinCode, String panNumber, String estYear, String radioopt, String radioopt1, String file1, String email, String password, String check, String desig) {

this.name = name;

this.companyName = companyName;

this.registerId = registerId;

this.mobileNumber = mobileNumber;

this.city = city;

this.state = state;

this.pinCode = pinCode;

this.panNumber = panNumber;

this.estYear = estYear;

this.radioopt = radioopt;

this.radioopt1 = radioopt1;

this.file1 = file1;

this.email = email;

this.password = password;

this.check = check;

this.desig = desig;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCompanyName() {

return companyName;

}

public void setCompanyName(String companyName) {

this.companyName = companyName;

}

public String getRegisterId() {

return registerId;

}

public void setRegisterId(String registerId) {

this.registerId = registerId;

}

public String getMobileNumber() {

return mobileNumber;

}

public void setMobileNumber(String mobileNumber) {

this.mobileNumber = mobileNumber;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

public String getState() {

return state;

}

public void setState(String state) {

this.state = state;

}

public String getPinCode() {

return pinCode;

}

public void setPinCode(String pinCode) {

this.pinCode = pinCode;

}

public String getPanNumber() {

return panNumber;

}

public void setPanNumber(String panNumber) {

this.panNumber = panNumber;

}

public String getEstYear() {

return estYear;

}

public void setEstYear(String estYear) {

this.estYear = estYear;

}

public String getRadioopt() {

return radioopt;

}

public void setRadioopt(String radioopt) {

this.radioopt = radioopt;

}

public String getRadioopt1() {

return radioopt1;

}

public void setRadioopt1(String radioopt1) {

this.radioopt1 = radioopt1;

}

public String getFile1() {

return file1;

}

public void setFile1(String file1) {

this.file1 = file1;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getCheck() {

return check;

}

public void setCheck(String check) {

this.check = check;

}

public String getDesig() {

return desig;

}

public void setDesig(String desig) {

this.desig = desig;

}

@Override

public String toString() {

return "DTORegister{" + "name=" + name + ", companyName=" + companyName + ", registerId=" + registerId + ", mobileNumber=" + mobileNumber + ", city=" + city + ", state=" + state + ", pinCode=" + pinCode + ", panNumber=" + panNumber + ", estYear=" + estYear + ", radioopt=" + radioopt + ", radioopt1=" + radioopt1 + ", file1=" + file1 + ", email=" + email + ", password=" + password + ", check=" + check + ", desig=" + desig + '}';

}

}

**DTOTenderform.java**

package mts;

public class DTOTenderForm

{

private String tenderId;

private String companyName;

private String companyRegisterNumber;

private String location;

private int pincode;

private String panNumber;

private int experience;

private double quoteAmount;

public DTOTenderForm() {

}

public DTOTenderForm(String tenderId, String companyName, String companyRegisterNumber, String location, String panNumber, int experience, double quoteAmount) {

this.tenderId = tenderId;

this.companyName = companyName;

this.companyRegisterNumber = companyRegisterNumber;

this.location = location;

this.panNumber = panNumber;

this.experience = experience;

this.quoteAmount = quoteAmount;

}

public String getTenderId() {

return tenderId;

}

public void setTenderId(String tenderId) {

this.tenderId = tenderId;

}

public String getCompanyName() {

return companyName;

}

public void setCompanyName(String companyName) {

this.companyName = companyName;

}

public String getCompanyRegisterNumber() {

return companyRegisterNumber;

}

public void setCompanyRegisterNumber(String companyRegisterNumber) {

this.companyRegisterNumber = companyRegisterNumber;

}

public String getLocation() {

return location;

}

public void setLocation(String location) {

this.location = location;

}

public int getPincode() {

return pincode;

}

public void setPincode(int pincode) {

this.pincode = pincode;

}

public String getPanNumber() {

return panNumber;

}

public void setPanNumber(String panNumber) {

this.panNumber = panNumber;

}

public int getExperience() {

return experience;

}

public void setExperience(int experience) {

this.experience = experience;

}

public double getQuoteAmount() {

return quoteAmount;

}

public void setQuoteAmount(double quoteAmount) {

this.quoteAmount = quoteAmount;

}

@Override

public String toString() {

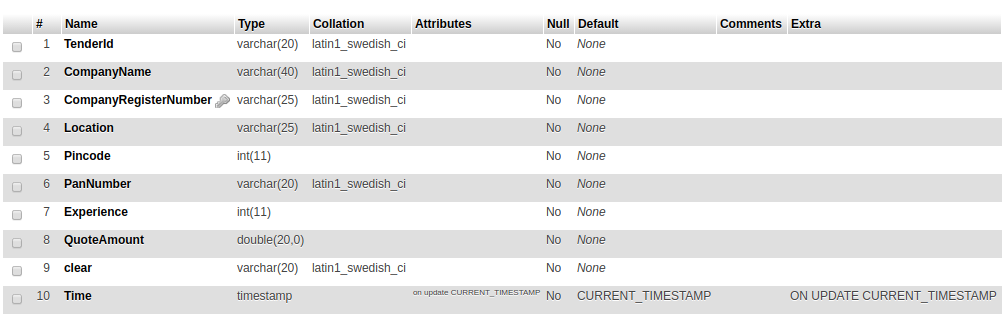
return "DTOTenderForm{" + "tenderId=" + tenderId + ", companyName=" + companyName + ", companyRegisterNumber=" + companyRegisterNumber + ", location=" + location + ", pincode=" + pincode + ", panNumber=" + panNumber + ", experience=" + experience + ", quoteAmount=" + quoteAmount + '}';

}

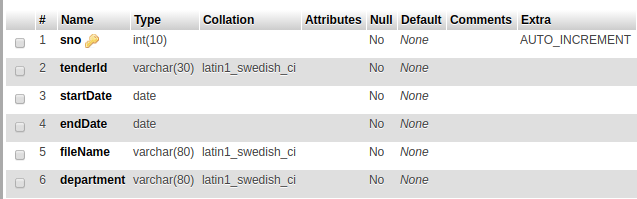
}

**Database**:

**Tender table:**

****

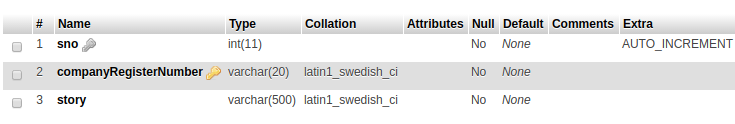
**Tender Announcement:**

****

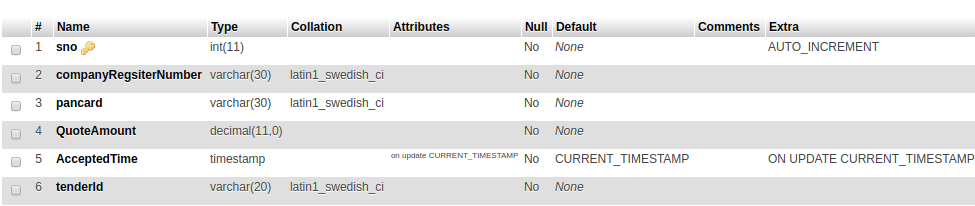
**Register Table:**

****

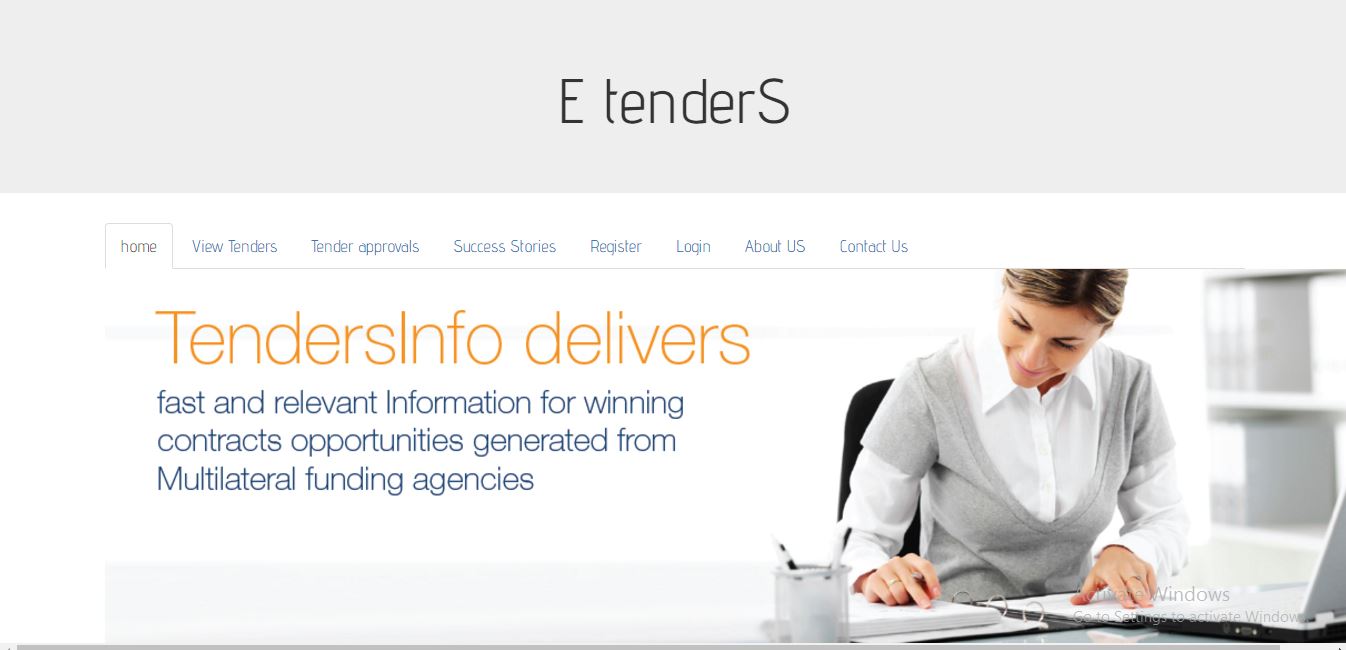
**Tender Info Table:**

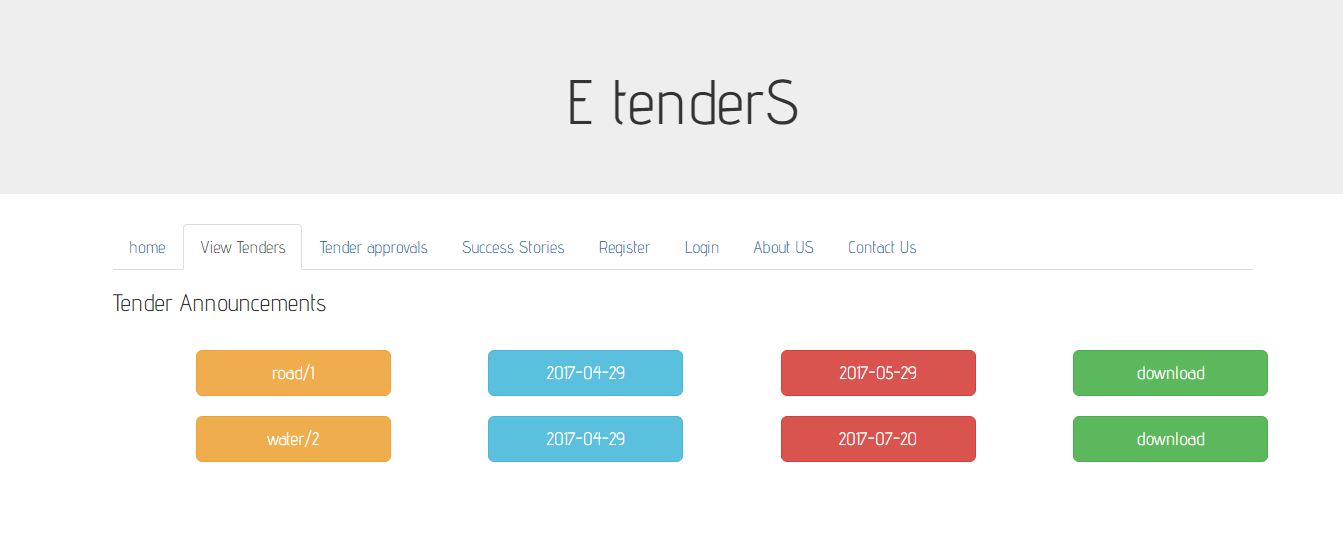
****

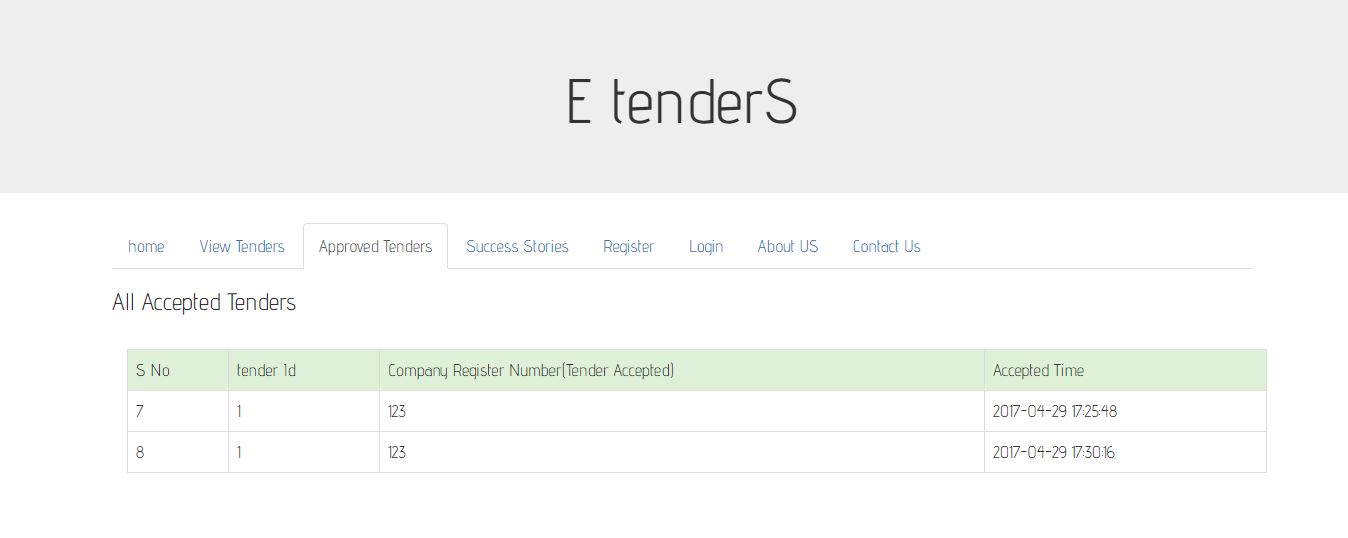
**Accepted Tenders:**

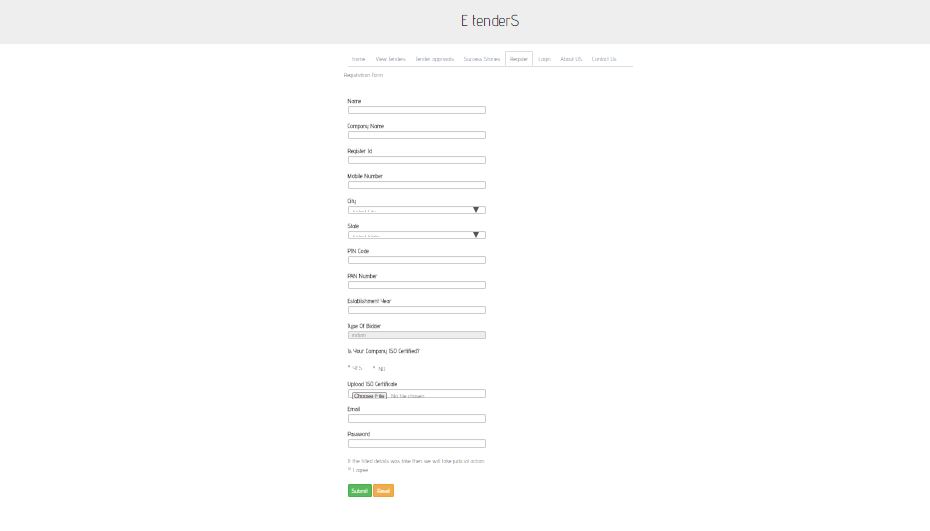
****

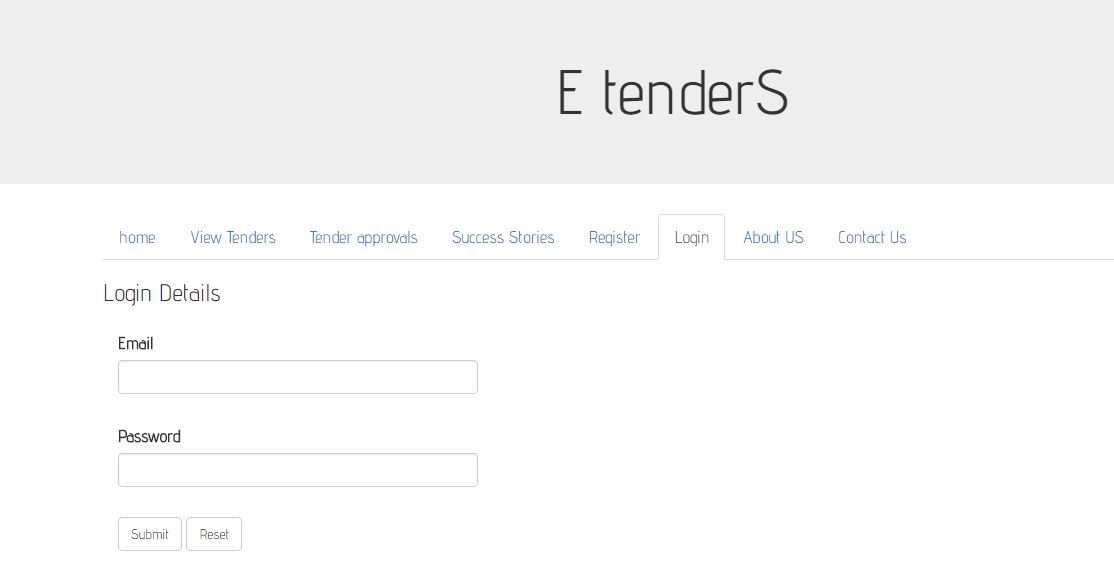
**Screen Shots:**

****

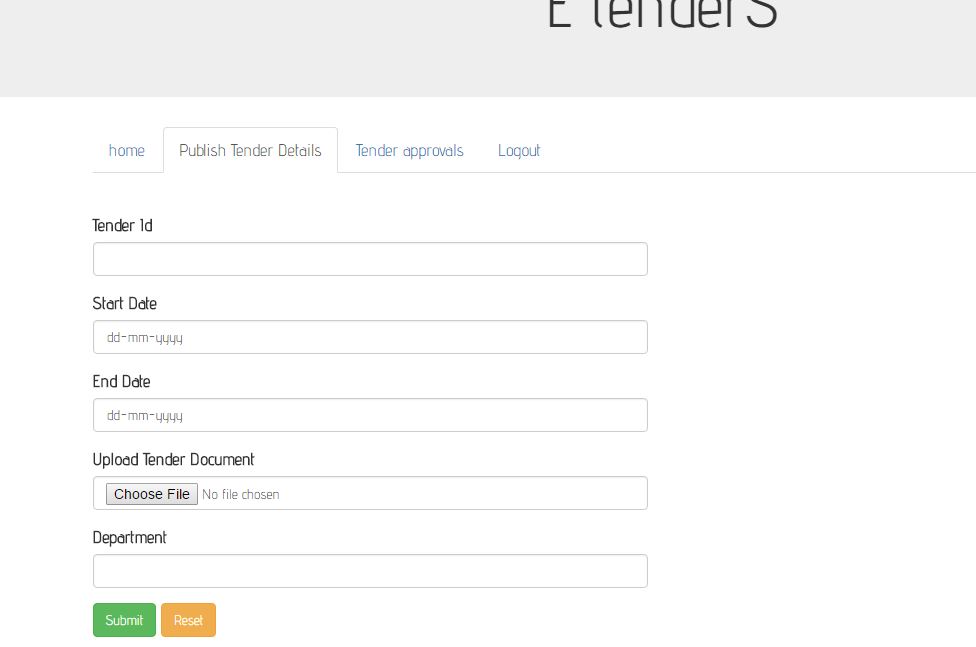
****

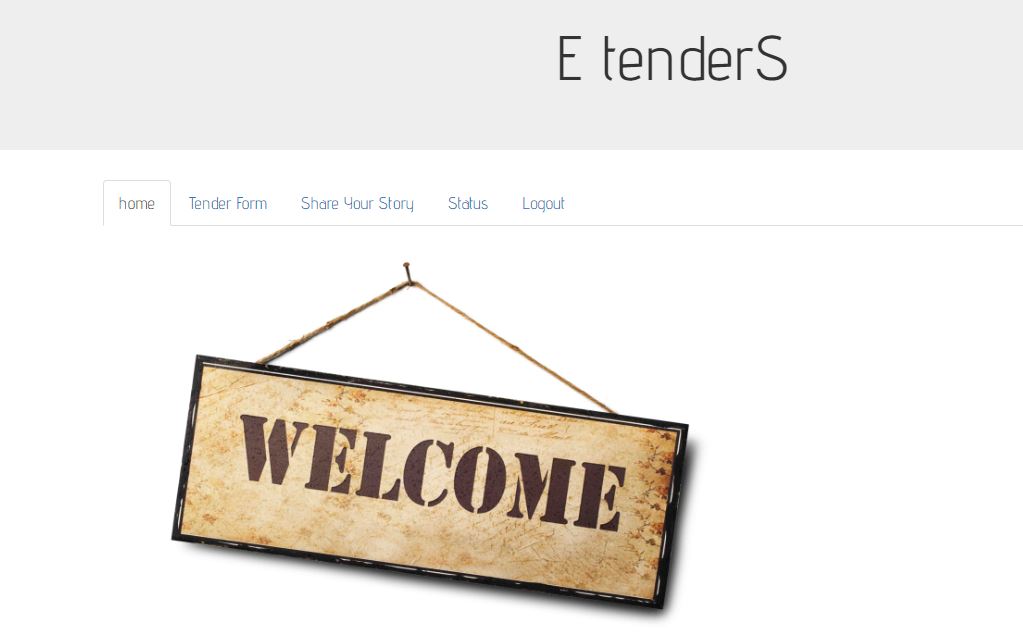
****

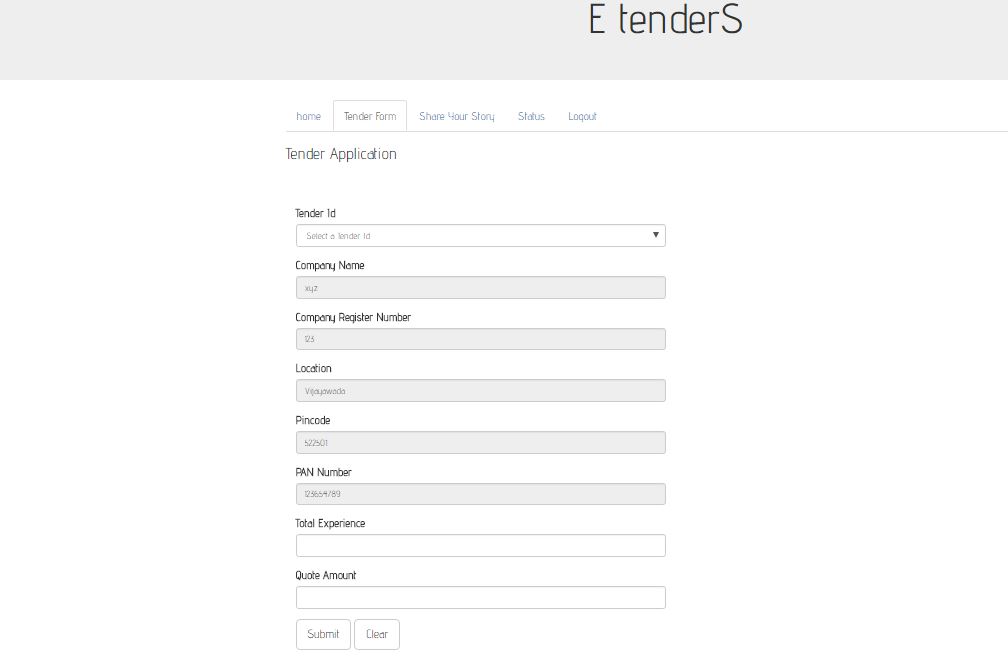
****

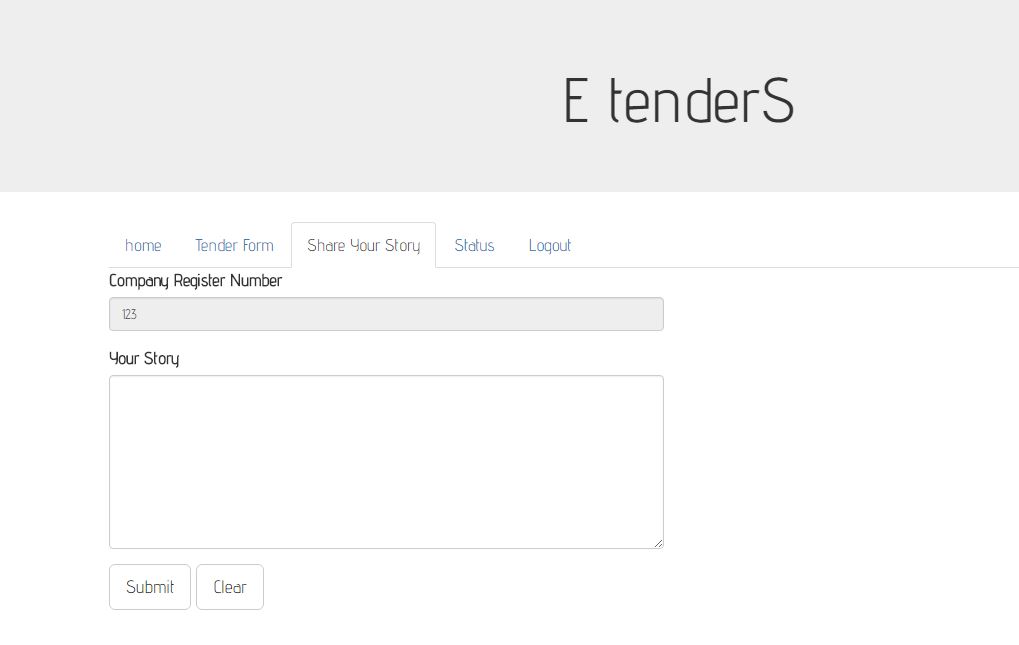
****

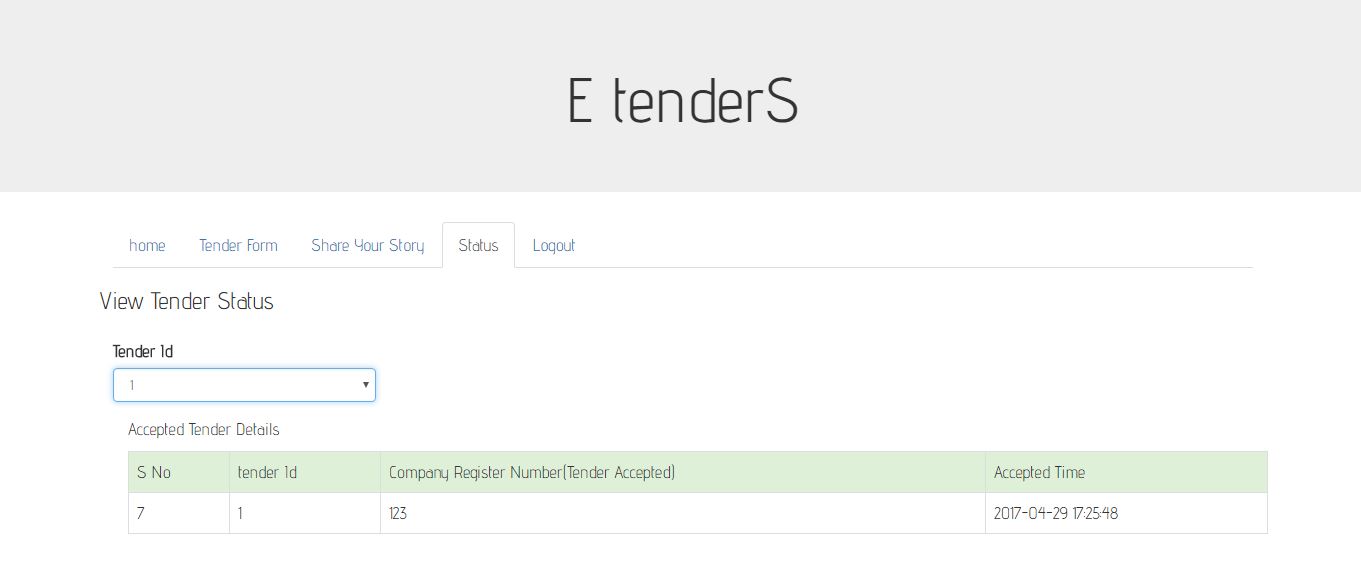
****

****

****

****

****

****

BibloGraphy:

W3Schools-https://www.w3schools.com/

JavatPoint- https://www.javatpoint.com/

TutorialsPoint- https://www.tutorialspoint.com/java/

Conclusion:

The research was finding solutions for current manual tender processing system of RDA. The problems that group has found in the existing manual system are - take lot of time to process, date are in many hard file, many records, less security, data retrieving delay, high overhead cost, lack of information for top management and middle management to analyze, more human resources etc,so our question was how to overcome these problems ,so introducing a web based e-Tendering system will do it.