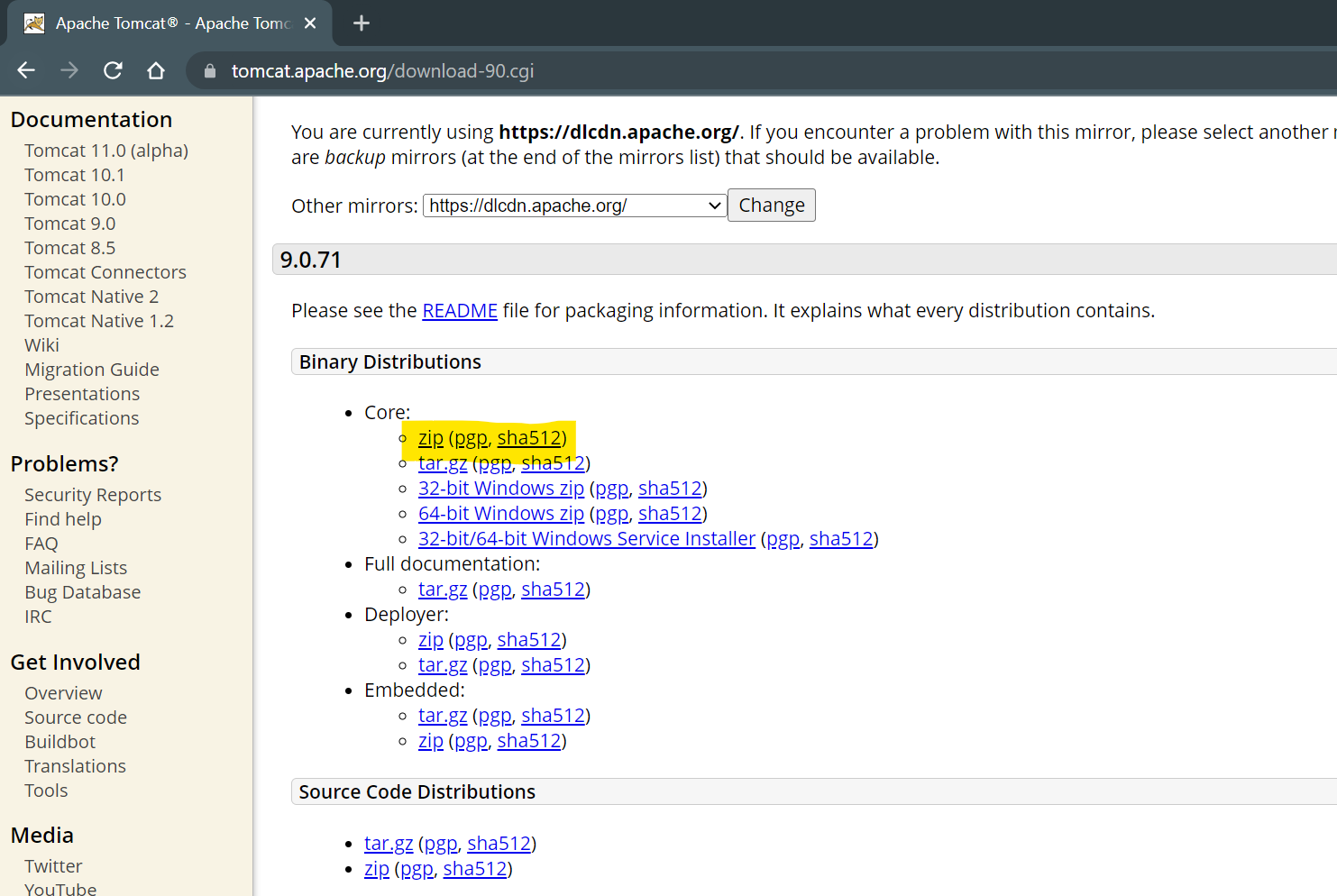
Tomcat Server Setup

1. Download tomcat

<https://tomcat.apache.org/download-90.cgi>

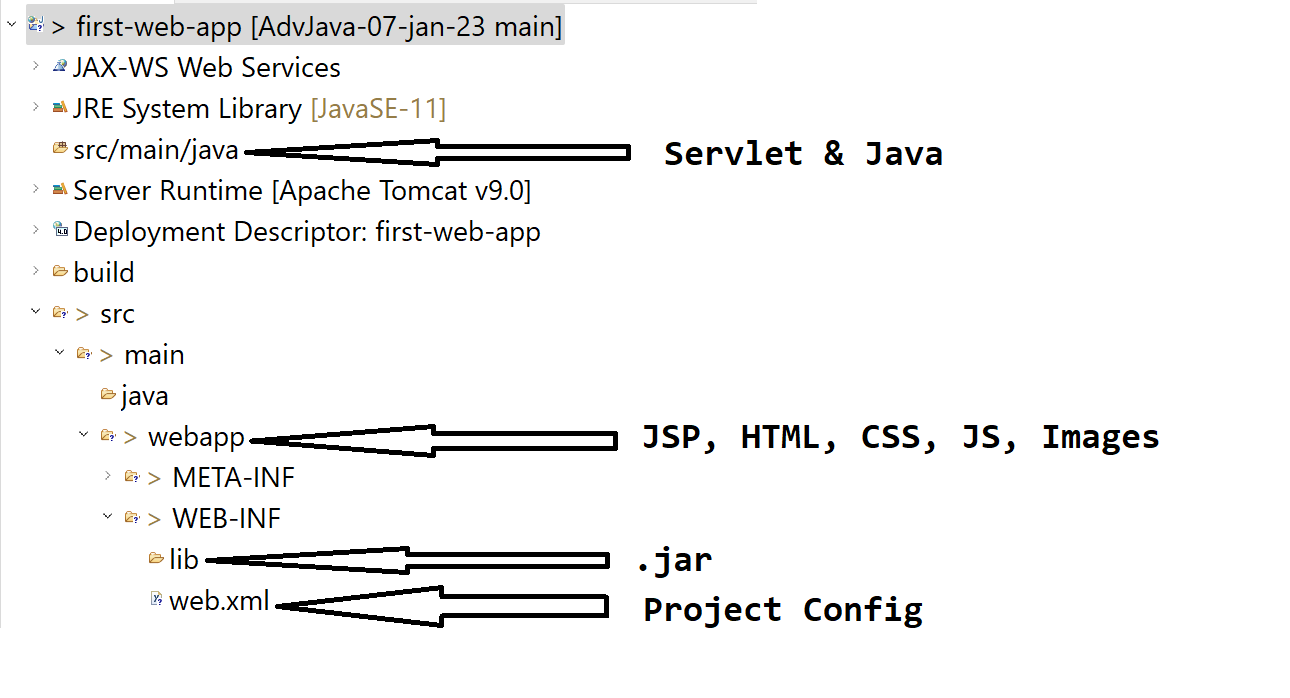


1. Copy and paste the zip file into specific location.
2. Extract the Zip File
3. Server Setup in Eclipse
   1. Set the “**JavaEE”** perspective
   2. Select “Servers” tab from the bottom for the window
   3. Click on the link to add new server.
   4. Expand “Apache” option from the list in the new window.
   5. Select a version which you downloaded
   6. Click on “Next”
   7. Browse and set the path of the location where you extracted the server zip file.
      1. Path of the parent folder of bin,lib, config etc.
   8. Click on Next and Finish

**Steps to create Web application**

1. Go “File” Menu -> “New” -> select “Dynamic Web Project”
2. Provide Name of the application and **make sure that Target Runtime** is selected
3. Click on “Next” -> click On “Next”
4. **Make sure that “generate Web.xml Deployment Descriptor” option is checked/selected**

Project Structure



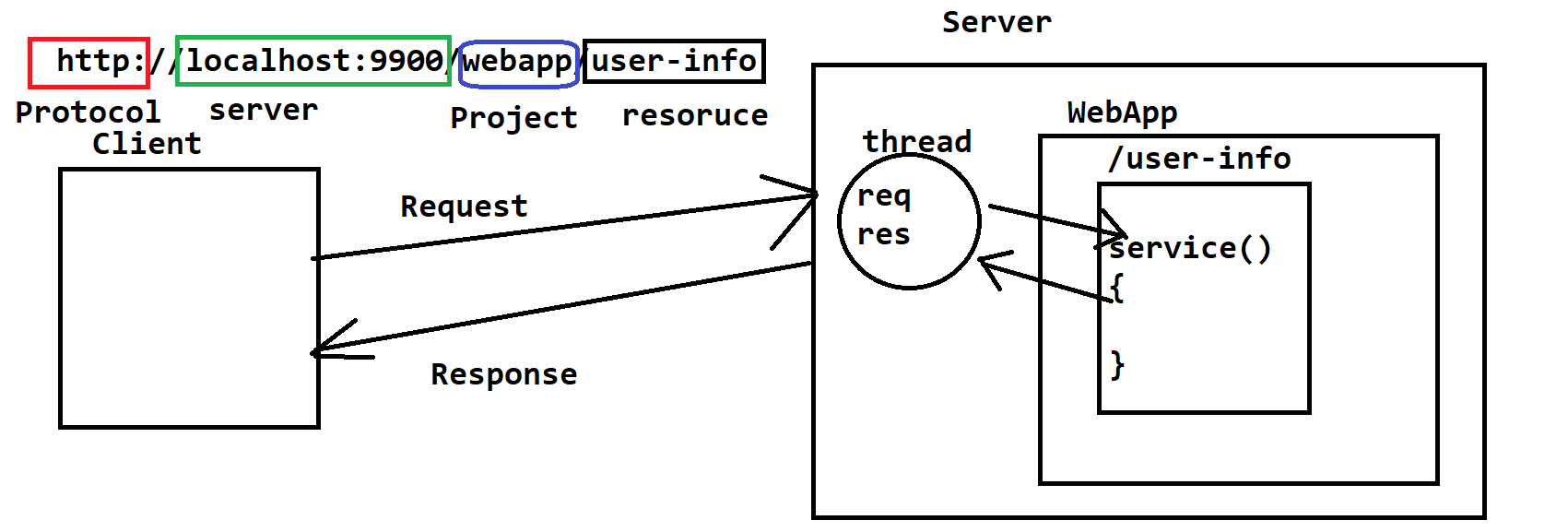
**Servlet**

1. Servlet is a java class.
2. In servlet main method us not used.
3. Servlet will be execute by the server (Servlet Container).
4. Servlet are execute and can be access by using URL.
5. Servlet are use to create dynamic web pages.
6. In the servlet you can use java code along with HTML (HTML in JAVA)
7. Servlet has to create inside src/main/java
8. Servlet are use to get request, process a request and generate the response

**Ways To Create Servlet**

1. Create java class and use Java EE APIs to create servlet
2. Implements Class with **Servlet interface**
3. Extends class with **GenericServlet abstract class**
4. Extends class with **HttpServlet abstract class**
5. Override the service method to work with request and repones.
6. Provide the URL for the servlet using @WebServlet annotation
   1. Use @WebServlet Annotation on class
   2. Provide the URL of the servlet which must be starts with “/”

Web Application Execution



**Response to a user**

1. To return a response you need an HttpServletResponse
2. To Write a response to a client you needs to set the type of response which is also knows as MIME type.

[**https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics\_of\_HTTP/MIME\_types/Common\_types**](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Common_types)

1. To set the response type you have to use method  
   **response.setContentType(“MIME TYPE”);**
2. To return a response you have to use the following java APIs

**PrintWriter**: is use for text type of response

**OutputStream**: Is use to return the binary response such as pdf, docx, images, videos

**Steps to create servlet**

1. Right click on src/main/java ->”New” Option -> click on “servlet” option
2. Provide class name. -> click “Next”
3. You can edit URL or keep same -> Click “Next”
4. Select “service” check box from the method list -> click on “Finish”

**Parameter**

1. Parameter is a user Data
2. Parameters are always pass from the request. Either it can be pass from the URL or from the request body
3. If it is pass from the url then you can see this parameter after ‘?’
4. Every parameter has 2 parts
   1. Key: always in string format
   2. Value: always in string format
5. Multiple parameter can be there in URL, which will be separated by ‘&’
6. To get the parameters you can use the request object

**request.getParameter(“Key”) : Value**

**Attribute**

1. Attribute is the user data
2. This data will aways pass internally from request.
3. Attribute is always in object form.
4. Attributes are always in key and values pair.
5. Where key is in string format and value is in object format.
6. To set and get the attribute you can use following methods

**Request.setAttribute(“Key”, Object);**

**Request.getAttribute(“Key”) : Object**

**Add Jar files into web application**

1. Copy a jar file from the file system
2. Paste the jar file into project

src/main/webapp/WEB-INF/lib

**JSP**

1. JSP is Java Server Pages
2. Mainly JSP use to create a dynamic web pages and for the designing part of the application.
3. On JSP you can use the HTML, CSS, JS code directly like a HTML file.
4. On top of it you can also use Java code in JSP pages.
5. For JSP page you do not have to specify the URL explicitly. By default every JSP page has a url as /pagename.jsp
6. JSP file extension is .jsp.

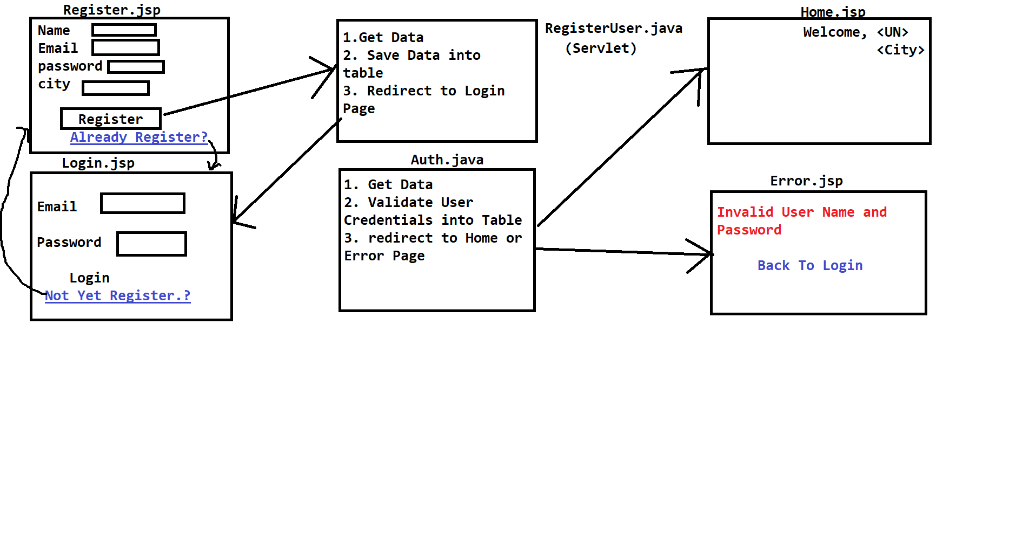
**Implicit Objects**

1. Implicit object are the by default available on every JSP page.
2. These Objects are only present inside service method.
3. There are 9 implicit objects present in JSP

|  |  |
| --- | --- |
| **Object Name** | **Class/Interface name** |
| request | HttpServletRequest |
| response | HttpServletResponse |
| session | HttpSession |
| out | JspWriter |
| application | ServletContext |
| exception | Throwable |
| page | this keyword in java |
| pageContext | PageContext |
| config | ServletConfig |

**NOTE: All these implicit objects only accessible inside scriptlet and expression tag**

**Task-1**

****

**Query To Check User credentials**

**Select \* form <tableName> where email=? AND password=?**

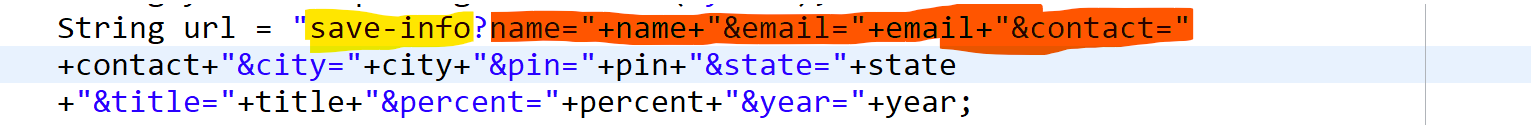
**Set name and City into request attribute and redirect to Home Page.**

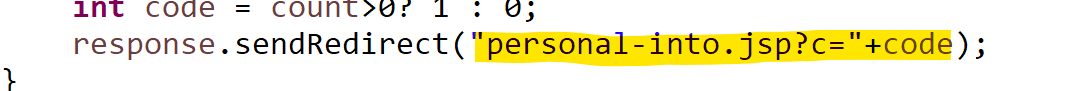
**Session Tracking techniques**

1. To retain the old request details into new request.
2. You can achieve this using 4 techniques
   1. Hidden form field
      1. To carry the old request data into new request which is generate by form field and submit.
      2. To Use this you have to use hidden field
      3. You can only carry the string type of data, Object cannot to transfer using this techniques.
      4. Syntax:



* 1. URL rewriting
     1. To carry the old request into new request generated by anchor (<a>) or sendRedirect
     2. It a way using which you can add the parameters manually inside.
     3. You can only carry the string type of data, Object cannot to transfer using this techniques.
     4. It can be use for a less number of parameter but it not good approach to use it for more number of parameter.
     5. Syntax:





* 1. Cookie
     1. Cookie use to store the user info into Client Side (Browser).
     2. Every Cookie will be added inside request and pass to server side with every request.
     3. There can be maximum 40-41 cookies store at client side.
     4. Syntax:

Cookie ck = new Cookie(“KEY”, “VALUE”);

* 1. HttpSession 
     1. Is use to store User information at server side

**MVC**

1. Model View Controller
2. Is a structure way of creating application
3. **Model** is use to add logical code and DB connection code
4. **View** is use to create UI/UX
5. **Controller** to handle the action of the view, to redirect into view and connecting View with model.

**MVC Application:**

**ToDO Application**

**1. Create New Task**

**2. View All Create Task**

**3. Search Task By Status and Scheduled Date**

**4. Update Task details and Status of the Task**



**5. Delete the task.**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| **id (PK) auto\_increment(sequence)** | **Int** |
| **title** | **Varchar** |
| **Status (Open, Inprogress, Completed)** | **Varchar** |
| **scheduledDate** | **Date** |
| **updatedDate** | **Date** |

mysql> create table task(

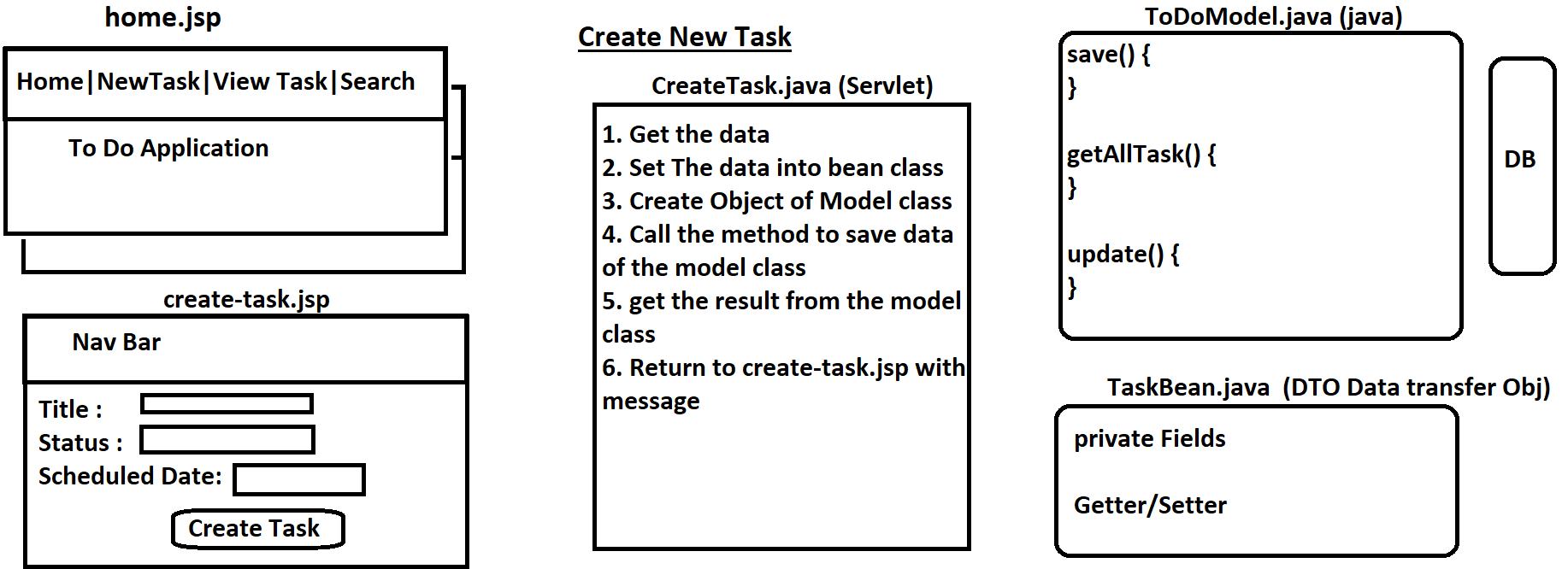
-> id int primary key auto\_increment,

-> title varchar(100),

-> status varchar(15),

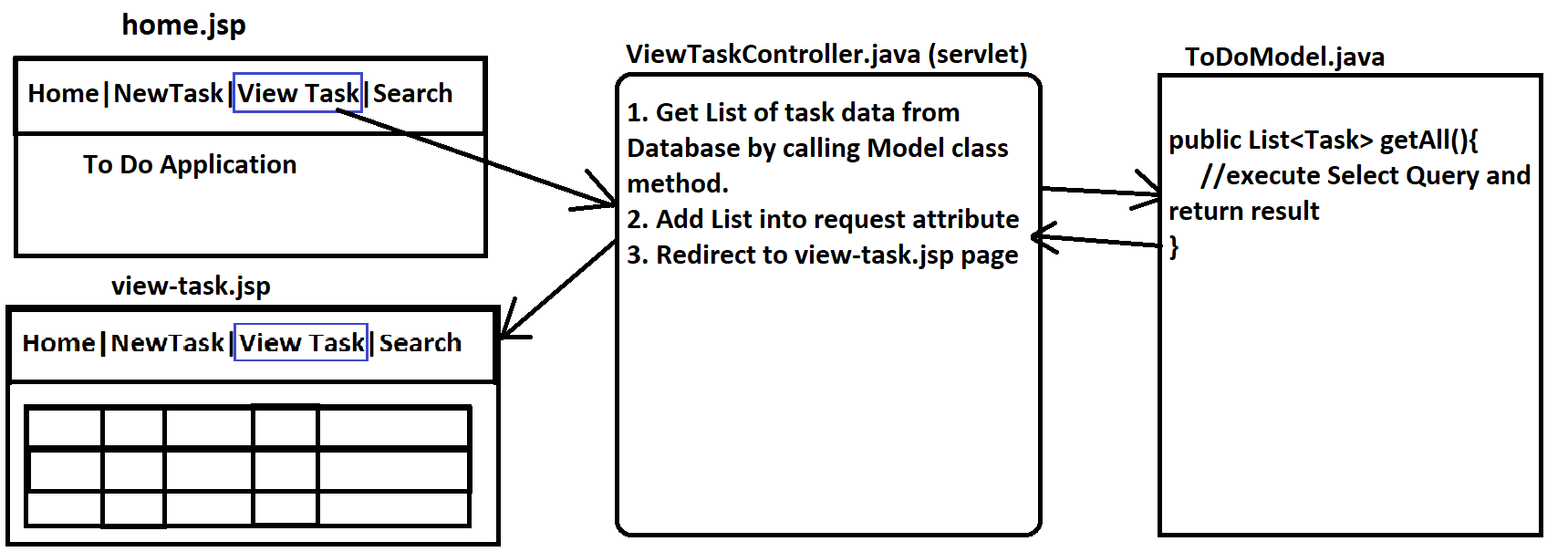
-> scheduledOn varchar(15),

-> updatedOn varchar(15));

1. **Create New Task** 

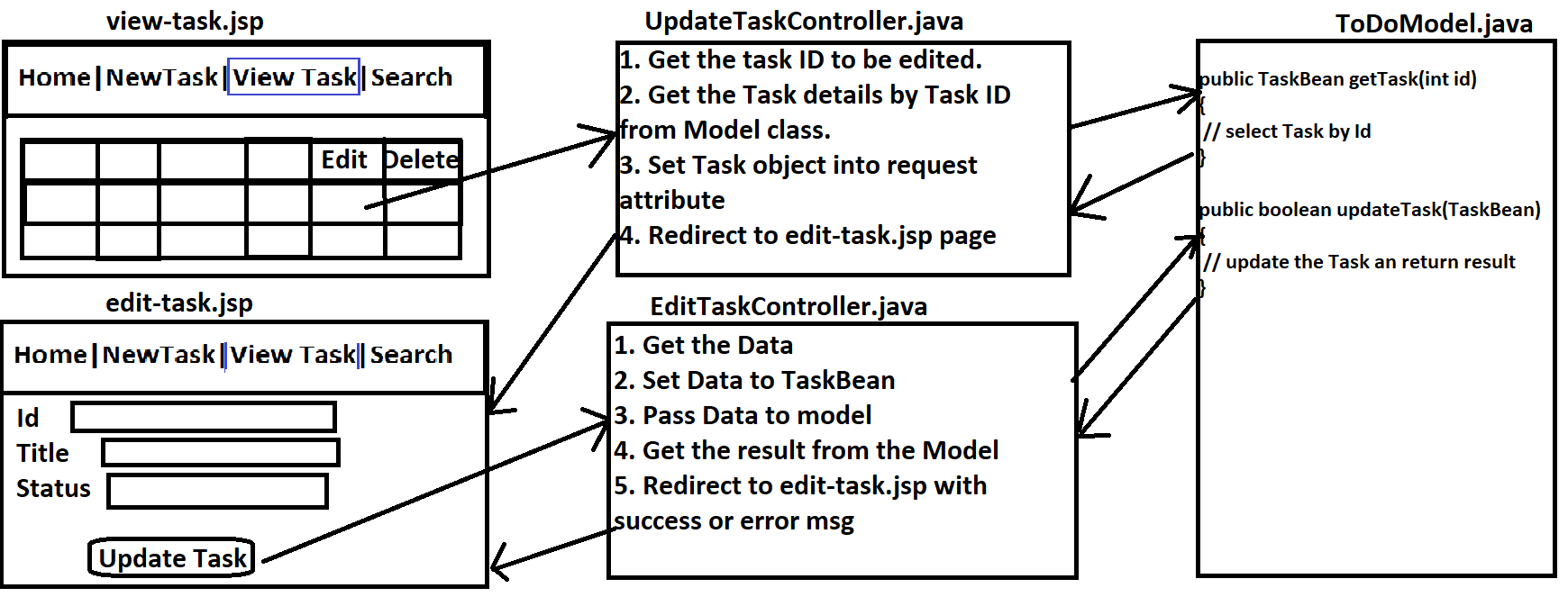


1. **View Task**



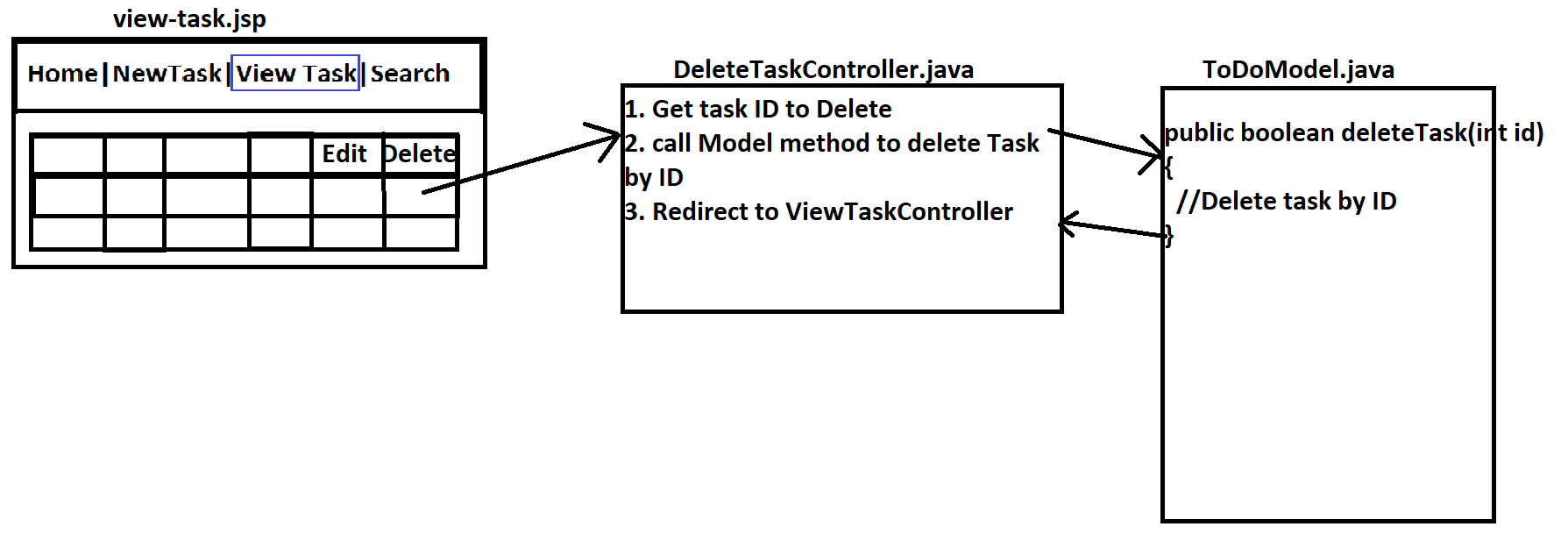


1. **Update Task**

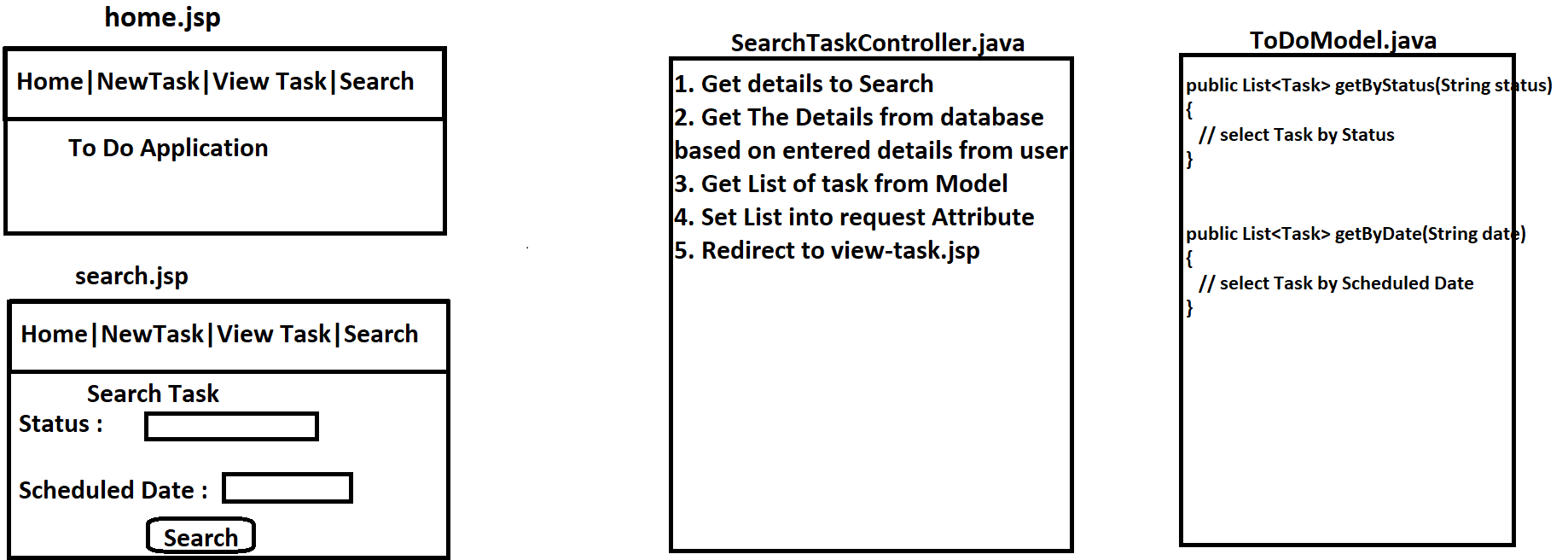




1. **Delete Task**



1. **Search Task**





**Contact Management Application**

User can manage the contact number using this application. Functionalities

1. **User can add new contact number into phone book**
2. User can edit the existing contact number
3. **User can get the list of all the contact number**
4. User can search the contact by name or number
5. Can delete the contact

**Data To be manage**

Id: int PK auto\_incrment

First name: varchar

Middle name : varchar

Last name: varchar

Email: varchar

Dob: Date