Bug Fixing Assistance Portal

Group -7

PROJECT REPORT

Submitted by

REG NO	NAME
CB.EN.U4CSE17008	ARUL JAYANTH M
CB.EN.U4CSE17009	ARVIND KUMAR
CB.EN.U4CSE17044	P RAJUKUMAR
CB.EN.U4CSE17051	SAI SUDHA PANIGRAHI
CB.EN.U4CSE17052	SAI SURAJ

in partial fulfilment of the requirements for the COURSE - 15CSE376(NET CENTRIC PROGRAMMING)

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMRITA SCHOOL OF ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE - 641112

NOVEMBER 2020

BONAFIDE CERTIFICATE

This is to certify that the report entitled "**Bug Fixing Assistance Portal**" submitted by **Group -7** in partial fulfilment of the requirements for the award of grade in course - 15CSE376 (NET CENTRIC PROGRAMMING is a bonafide record of the work carried out at Amrita School of Engineering, Coimbatore.

Evaluated on: 7/12/2020

Course Faculty :Mrs.G.Ramya

ACKNOWLEDGEMENT

I wish to record my deep sense of gratitude and profound thanks to my course faculty **Mrs.G.Ramya** , designation, Computer Science and Engineering Department, Amrita School of Engineering, Coimbatore, for his/her keen interest, inspiring guidance, constant encouragement with my work during all stages, to bring this report into fruition.

Abstract

HTML Validator is a debugging tool that also aids in the fixing the various bugs that might be encountered in the HTML pages. Validating and finding the bugs in the code is very important because:

- Help Improve Rankings in Search Engines.
- Validation helps teach best practices
- Improved Website User Experience
- Make Website Browsers Friendly
- Multiple Device Accessibility
- Validation help for easy Coding and Maintenance

For questions other than validation, we provide a forum where a developer could seek answers from the huge ever-growing web developers' community.

Technologies used -Term 1

HTML Introduction

- **Hypertext Markup Language** (**HTML**) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

- Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

- HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as `<img />` and `<input />` directly introduce content into the page. Other tags such as `<p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

- HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content.

A Simple HTML Document

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h1>My First Heading</h1>
<p>My first paragraph.</p>
</body>
</html>
```

CSS Introduction

Cascading Style Sheets, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

CSS provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

Advantages of CSS

- CSS saves time − You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- Pages load faster − If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- Easy maintenance − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML − CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Multiple Device Compatibility − Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Global web standards − Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

JavaScript Introduction

JavaScript is a cross-platform, object-oriented scripting language used to make webpages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.).

There are also more advanced **server side versions** of JavaScript such as Node.js, which allow you to add more functionality to a website than simply downloading files. Inside a host environment (for example, a web browser), JavaScript can be connected to the objects of its environment to provide programmatic control over them.

JavaScript contains a standard library of objects, such as **`Array`**, **`Date`**, and **`Math`**, and a core set of language elements such as operators, control structures, and statements. Core JavaScript can be extended for a variety of purposes by supplementing it with additional objects; for example:

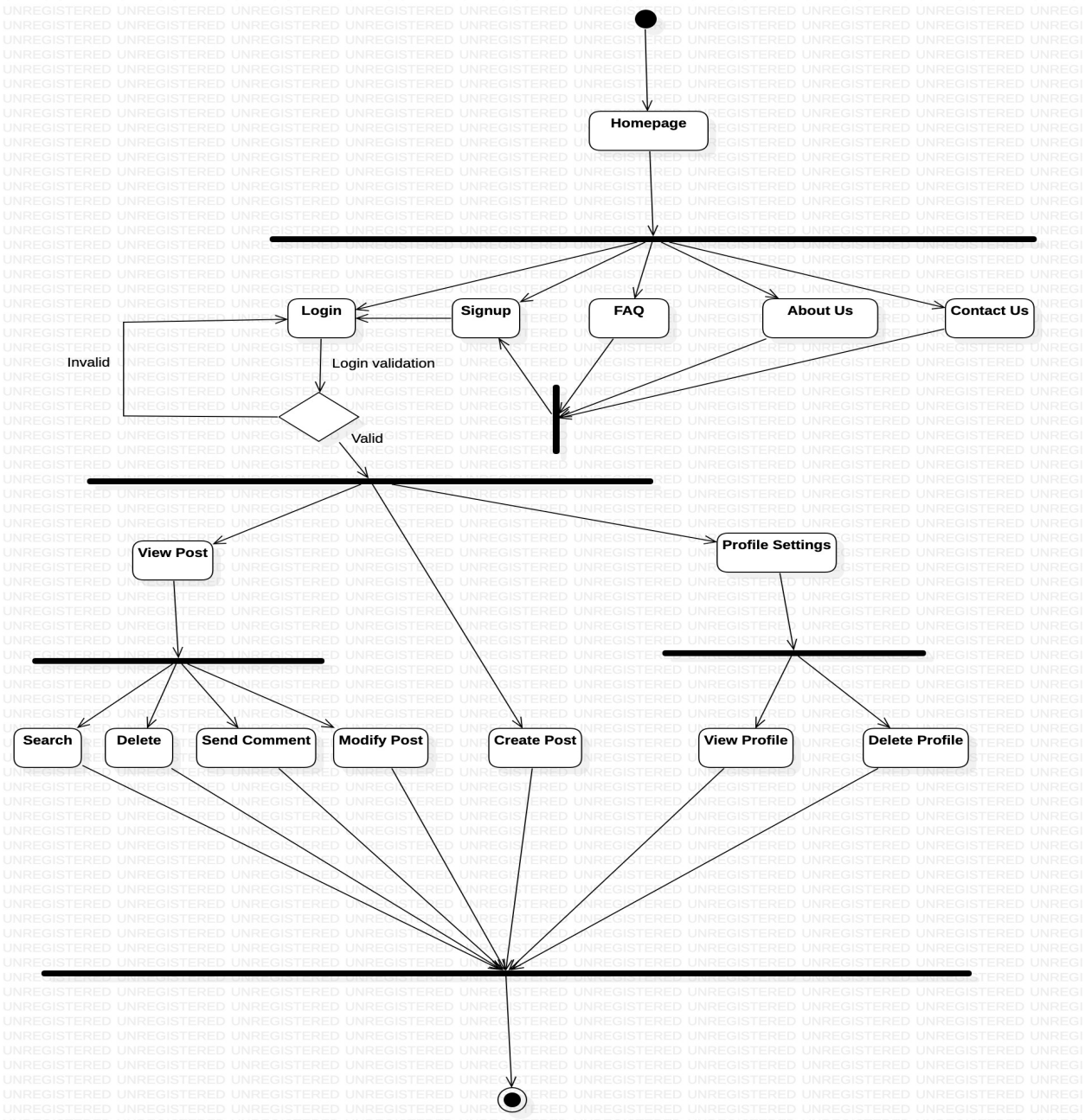
- *Client-side JavaScript* extends the core language by supplying objects to control a browser and its *Document Object Model* (DOM). For example, client-side extensions allow an application to place elements on an HTML form and respond to user events such as mouse clicks, form input, and page navigation.
- *Server-side JavaScript* extends the core language by supplying objects relevant to running JavaScript on a server. For example, server-side extensions allow an application to communicate with a database, provide continuity of information from one invocation to another of the application, or perform file manipulations on a server.

This means that in the browser, JavaScript can change the way the webpage (DOM) looks. And, likewise, Node.js JavaScript on the server can respond to custom requests from code written in the browser.

System Requirements Specifications

- A computer with any browser that supports new versions of HTML5, CSS, JS etc. (Ex. Chrome, Mozilla FireFox, Safari).
- A code editor like Visual Studio Code or Notepad++ for coding.

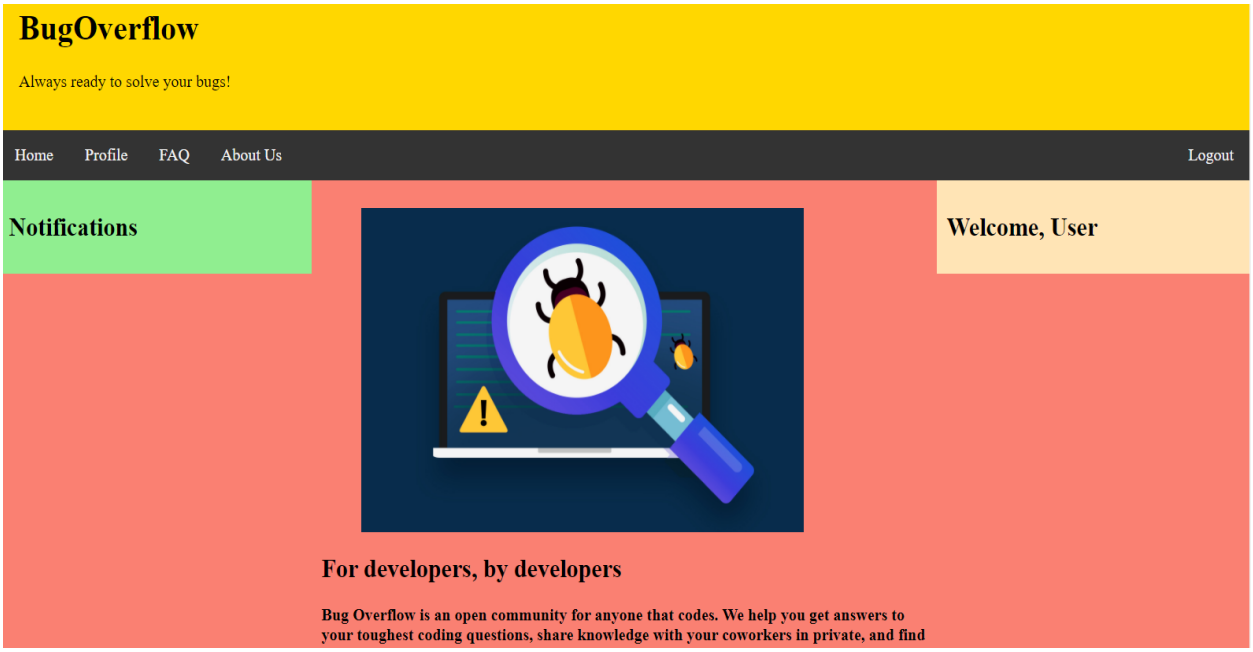
Activity Diagram:-


Homepage
Login
Signup
FAQ
About Us
Contact Us
Invalid
Login validation
Valid
View Post
Profile Settings
Search
Delete
Send Comment
Modify Post
Create Post
View Profile
Delete Profile

Implementations

1. Home page

Screenshot:



Description:

Helps in navigation to other pages

Elements used:

Navigation bar

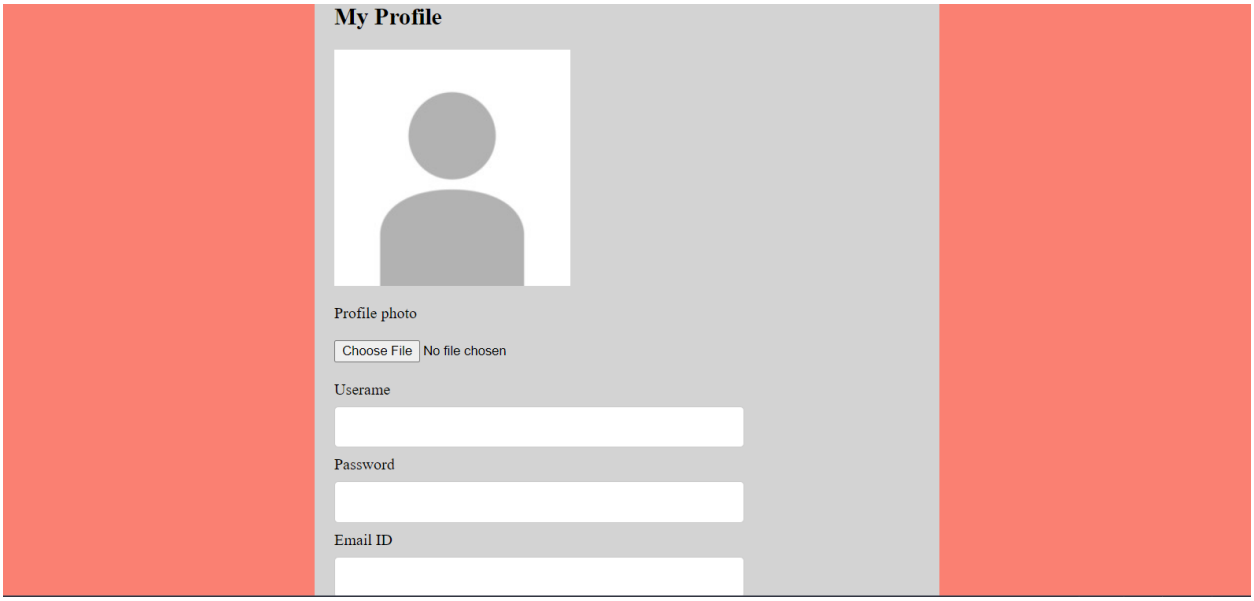
Code Snippet:

```
 <div class="header">
  <header>

  <h1>BugOverflow</h1>
  <p>Always ready to solve your bugs!</p>
    </header>

</div>

<div class="topnav">
  <nav>
  <a href="#" id="home_tab">Home</a>
  <a  href="#" id="profile_tab">Profile</a>
```

```
  <a href="#">FAQ</a>
    <a href="#">About Us</a>
   <a href="#"
      style="float:right">Logout</a>
    </nav>
</div>

<div id="home_page">
  <div class="column left">
    <h2>Notifications</h2>
  </div>

  <div class="column middle">
    <figure>
    <img src="https://blog.hyperiondev.com/wp-content/uploads/2019/01/Blog-Avoid-Bugs-
Programming.jpg"width="450" height="300">
</figure>
    <section>
    <h2>For developers, by developers
</h2>
    <p><b>Bug Overflow is an open community for anyone that codes. We help you get answers to your
toughest coding questions, share knowledge with your coworkers in private, and find your next dream
job.</b></p>
   <section>
  </div>

  <div class="column right">
    <article>
    <h2>Welcome, User</h2>
      </article>
  </div>
```

2. Profile page

Screenshot:



Description:
For viewing and editing profile details

Elements used:
Form, text fields, input type and pattern validation

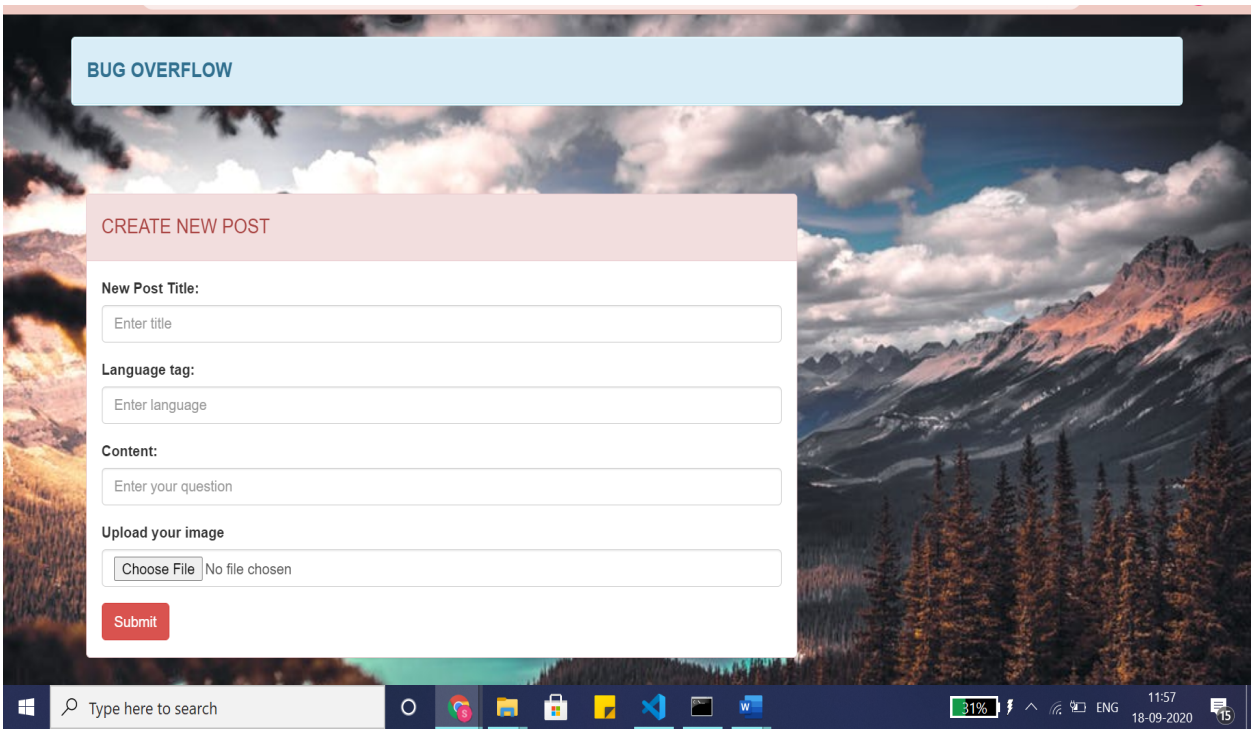
Code Snippet:

```
<div id="profile_page">
     <h2>My Profile</h2>

<div>
  <form onsubmit="return false"> <br>
    <label for="pic">Profile photo</label> <br><br>
    <input type="file" id="pic" name="pic"> <br><br>

    <label for="uname">Userame</label> <br>
    <input type="text" id="uname" name="uname" required> <br>

    <label for="pw">Current Password</label> <br>
    <input type="password" id="pw" name="pw" required > <br>

    <label for="npw">New Password</label> <br>
    <input type="password" id="npw" name="npw" pattern="[A-Za-z0-9]{8,}" title="at least 8 characters" >
<br>

 <label for="eid">Email ID</label> <br>
```

```
    <input type="email" id="eid" name="eid" required > <br>

    <label for="mno">Mobile No </label> <br>
    <input type="tel" id="mno" name="mno" pattern="[0-9]{10}" title="10 digit no." required > <br>

    <input type="submit" value="Save Changes">
  </form>
</div>
```

3. Create Post Page

Screenshot:



Description:

This page enables the user to create a new post in bug overflow.

Elements used:

Form, Image insertion, text fields, input type and pattern validation.

Code Snippets:

HTML

```
<form role="form" method="post" enctype="multipart/form-data">
            <div class="form-group">
              <label for="name">New Post Title:</label>
              <input type="text" class="form-control" name="n" id="name" placeholder="Enter title"
required>
            </div>
            <div class="form-group">
              <label for="lang">Language tag:</label>
              <input type="email" class="form-control" name="e" id="email" placeholder="Enter language"
required>
            </div>
            <div class="form-group">
              <label for="content">Content:</label>
              <input type="password" class="form-control" name="p" id="pwd" placeholder="Enter your
question" pattern=".{8,}" title="length of password has to be atleast 8" required>
            </div>

            <div class="form-group">
                        <label for="pwd">Upload your image</label>
                        <input type="file" class="form-control" id="file" name="img" >
                      </div>

            <button type="submit" class="btn btn-danger">Submit</button>
          </form>
```

Java Script

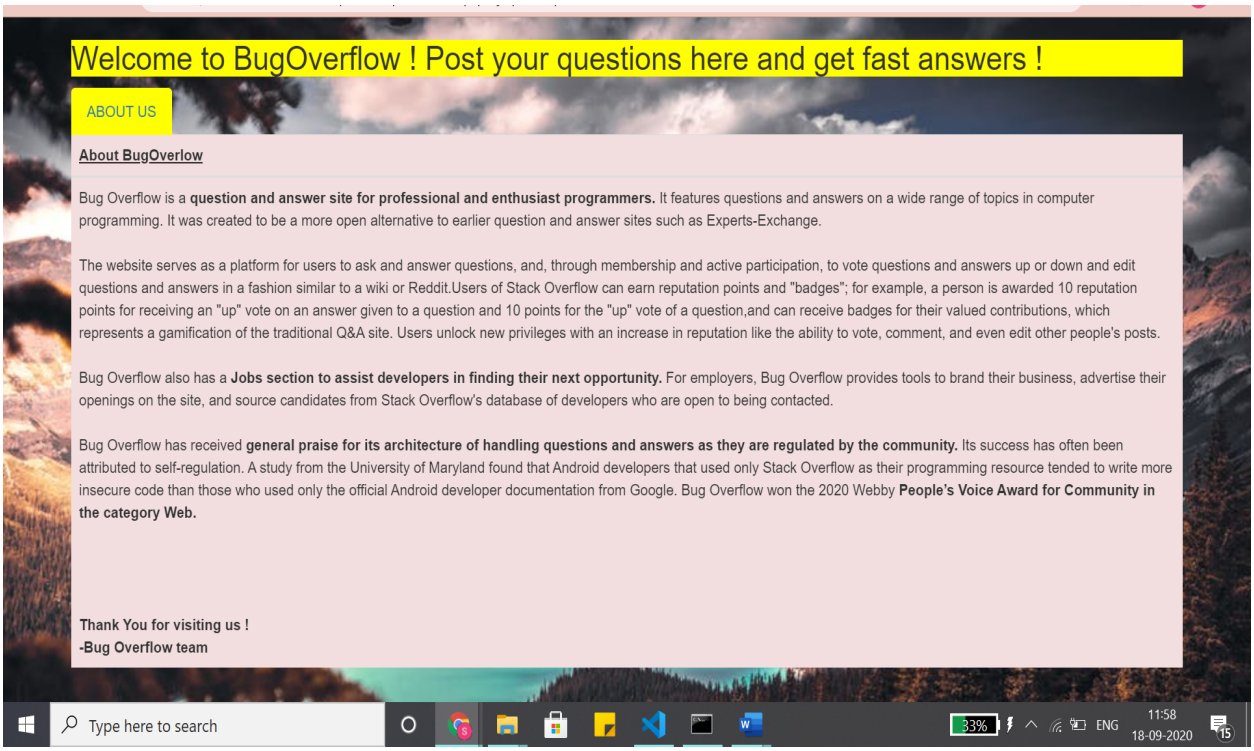
```
<script>
                    function validateForm() {
                      var x = document.forms["create_post"]["title"].value;
                      var y = document.forms["create_post"]["lang"].value;
                      var z = document.forms["create_post"]["content"].value;

                    if (x == "" ||y==""|| z=="") {
                              alert("This field must be filled out");
                    }
                    if (! isNaN(x) ||! isNaN(y) || ! isNaN(z)) {
                                        alert("This field cannot be a number");

          }
                              return false;

                    }
                    </script>
```

CSS

```
.bod{

  background-image: url('747964.jpeg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;


}

.bodyLight{

  background-image: url('light_theme.jpg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;


}
```

4. About Bug Overflow Page

Screenshot:



Description:

This page enables the user to know more about bug overflow.

Elements used:

HTML tags, CSS styling, semantic tags

Code Snippet:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Home page</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <link rel="stylesheet" href="style.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>
<body class="bod">
<div class="container">
  <h2 class="hi">Welcome to BugOverflow! Post your questions here and get fast answers !</h2>
  <ul class="nav nav-tabs">
    <li><a data-toggle="tab" href="#menu1" class="hi">ABOUT US</a></li>
  </ul>
```

```
    <table class="table" >
  <thead>
    <tr class="danger">
      <th><u>About BugOverlow</u></th>
    </tr>
  </thead>
  <tbody>
  <tr class="danger">
    <td>
    <section>
    Bug Overflow is a <b>question and answer site for professional and enthusiast programmers.</b> It features questions and answers on a wide range of topics in computer programming.
    It was created to be a more open alternative to earlier question and answer
    sites such as Experts-Exchange.
    <br>
    <br>
    </section>
    <section>
    The website serves as a platform for users to ask and answer questions, and, through membership and active participation, to vote questions and answers up or down and edit questions
    and answers in a fashion similar to a wiki or Reddit.Users of Stack Overflow can earn reputation points and "badges"; for example, a person is awarded 10 reputation points for receiving an "up" vote on an
    answer given to a question and 10 points for the "up" vote of a question,and can receive badges for their valued contributions, which represents a gamification of the traditional Q&A site.
    Users unlock new privileges with an increase in reputation like the ability to vote, comment, and even edit other people's posts.
    <br>
    </section>
```

Java Script

```
<script>
        function mOver(obj) {
          obj.className = "info"
        }

        function mOut(obj) {
          obj.className = "danger"
        }
 </script>
```

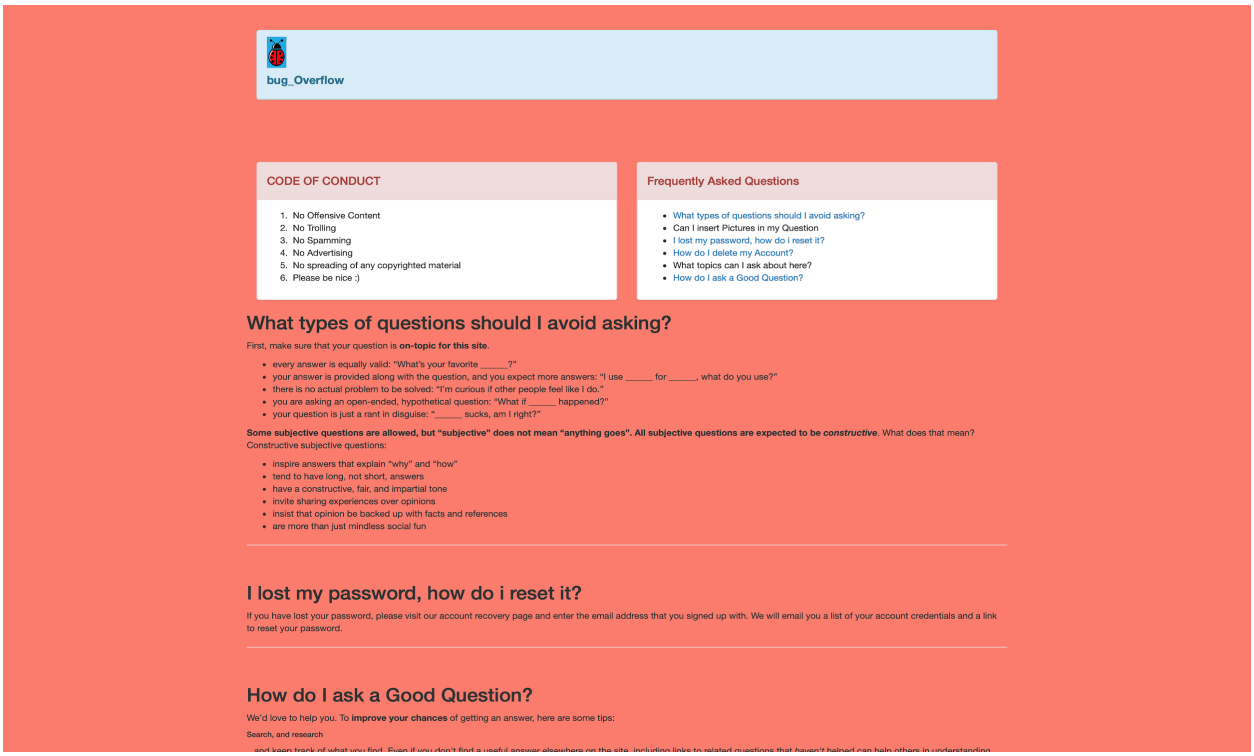
CSS

```
.bod{

  background-image: url('747964.jpeg');
```

```
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;


}

.bodyLight{

  background-image: url('light_theme.jpg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;


}
```

5. FAQ page

Screenshot:-



Description:

This page enables the user to know about the frequently asked questions and code of conduct in bugoverflow.

Elements used:

HTML tags, CSS styling, semantic tags

Code Snippet:

```
<article id="one">
<h2>What types of questions should I avoid asking?</h2>
<p>First, make sure that your question is <b>on-topic for this site</b>.</p>
<ul>.....</ul>
<p><b>Some subjective questions are allowed, but “subjective” does not mean “anything goes”. All subjective questions are expected to be <i>constructive</i></b>. What does that mean? Constructive subjective questions:</p>
<ul>……...</ul>
</article>
<hr>
<br>
<aritcle id="three">
<h2>I lost my password, how do i reset it?</h2>
<p>If you have lost your password, please visit our account recovery page and enter the email address that you signed up with. We will email you a list of your account credentials and a link to reset your password.</p>
```

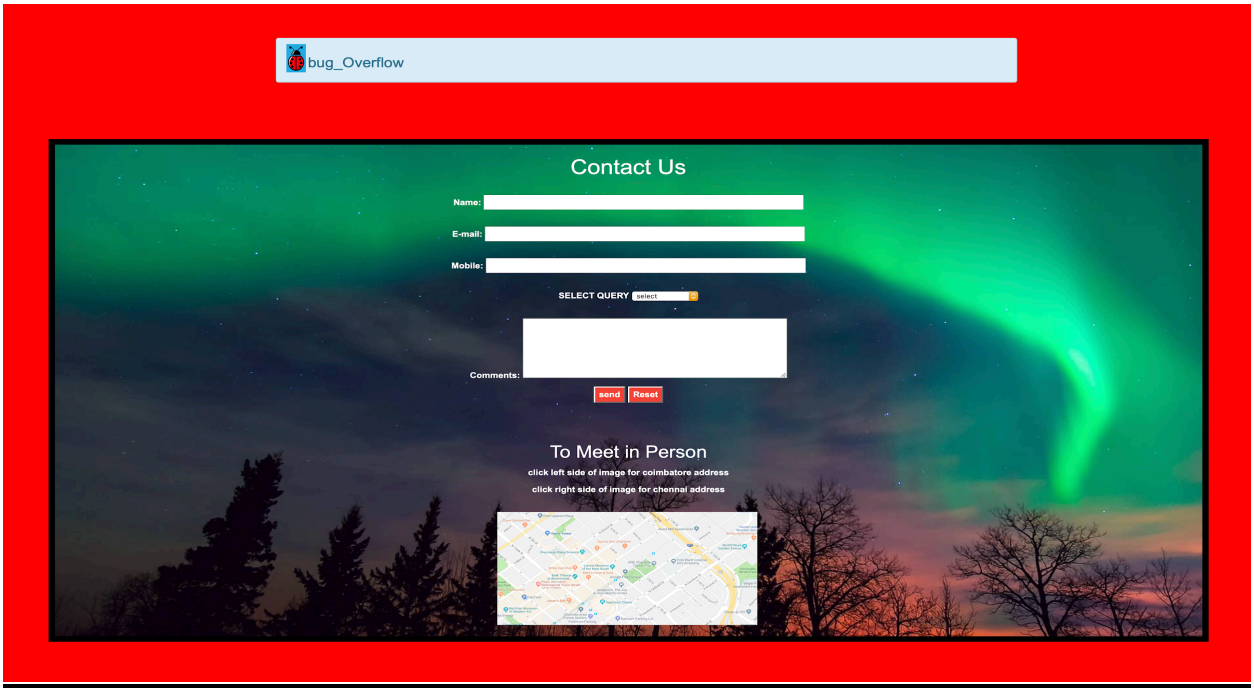
</aritcle>
<hr>

<h6>Write a title that summarizes the specific problem</h6>

The title is the first thing potential answerers will see, and if your title isn't interesting, they won't read the rest. So make it count:
.....
Examples:
.....
<h6>Introduce the problem before you post any code</h6>
.....
<h6>Help others reproduce the problem</h6>
y to get you in trouble if you're posting your employer's code, it likely includes a lot of irrelevant details that readers will need to ignore when trying to reproduce the problem. Here are some guidelines:

6. Contact Us page

Screenshot:



Description:

This page enables the user to know about the frequently asked questions and code of conduct in bug overflow. The user can report abuse content, enquire about career, can report bugs in website,enquire about advertising and sponsoring events, etc.

Elements used:

HTML tags(Form, text fields, input type), CSS styling, Javascript Methods for Form Validation and background change.

Code Snippet:

1) **Javascript Functions:-**

```
function mouseOver() {
  document.getElementById("backd").style.backgroundColor= "red";
}
function mouseOut() {
  document.getElementById("backd").style.backgroundColor= "yellow";
}
function verify()
{ var name=document.forms["ContactForm"]["Name"];
            var email=document.forms["ContactForm"]["EMail"];
            var mobile=document.forms["ContactForm"]["Mobile"];
            var what=document.forms["ContactForm"]["Subject"];
  var comment=document.getElementById("Comment");
if (name.value == "")                                                    { …….}
  else if(/^[a-zA-Z-,]+(\s{0,1}[a-zA-Z-, ])*$/.test(name.value)==false)  {.........}
if (email.value == "")                                                   { …….}
else if(/^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@+(?:\.[a-zA-Z0-9-]+)*$/.test(email.value)==false)  {.......}
if (mobile.value == "")                                  { …….}
  else if(/^\d+$/.test(mobile.value)==false)             {……}
  else if(/^\d{10}$/.test(mobile.value)==false)          {......}
            if (what.selectedIndex < 1)                  { ……. }
    if (comment.value == "")                             { ……. }
  if(comment.value.length>500)                           {...........}
  return true; }
 function yesnoCheck(that) {
    if (that.value == "other") {
        document.getElementById("ifYes").style.display = "block";
    } else {
        document.getElementById("ifYes").style.display = "none";
    }}
```

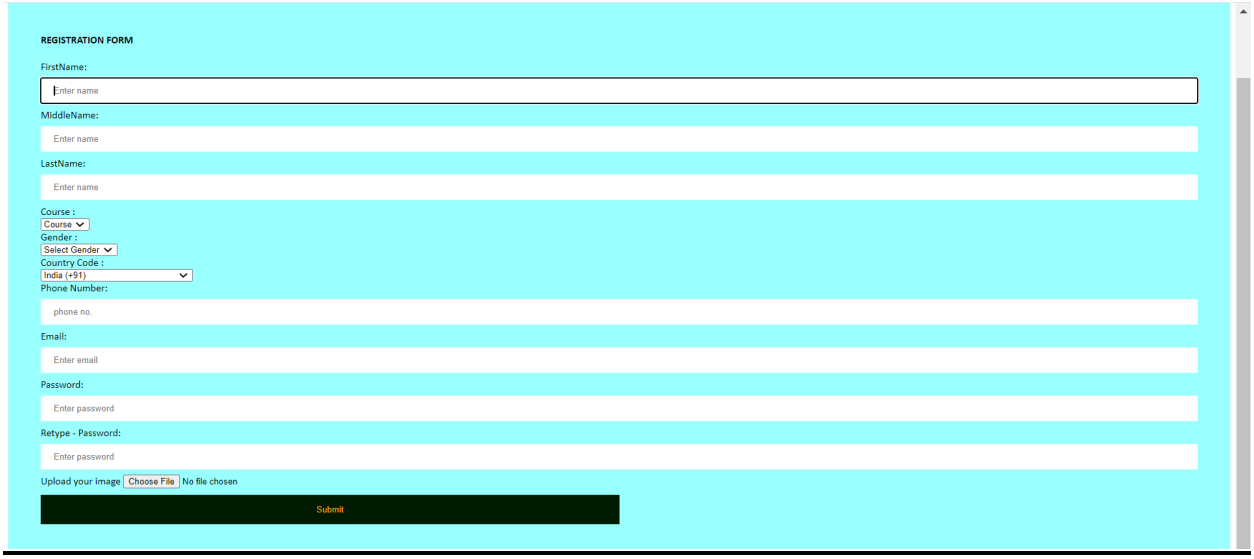
2) **Forms**:-

```
<form name="ContactForm" onsubmit="return verify()" method="post">
            <p>Name: <input type="text" size=65 name="Name"> </p><br>
            <p>E-mail: <input type="text" size=65 name="EMail"> </p><br>
            <p>Mobile: <input type="text" size=65 name="Mobile"> </p><br>
            <p>SELECT QUERY
                        <select type="text" value="" name="Subject" onchange="yesnoCheck(this);">
      <option default>select</option>
      <option>Bug Bounty</option>
                                    <option>Report Abuse</option>
                                    <option>Careers</option>
                                    <option>Advertising</option>
                                    <option>Sponsorship</option>
      <option value="other">Other</option></select></p>
  <div id="ifYes" style="display: none;">
 <label for="car">If selected other, please mention</label> <input type="text" id="car" name="car" /><br />
</div><br>
```

```
<p>Comments: <textarea cols="55" rows="5" name="Comment" id="Comment"></textarea></p>
        <p><input type="submit" value="send" name="Submit" class="button">
                <input type="reset" value="Reset" name="Reset" class="button"> </p>
</form>
```

7. Registration page:

Screenshot:



Description:

This page enables the user to create a new Account in bug overflow.

Elements used:

Form,text fields, input type and Password verification and upload image..

Code Snippet:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Signup</title>
</head>
        </div>
        <div class="col-sm-6">
          <div class="panel panel-danger">
            <div class="panel-heading"><h4>REGISTRATION FORM</h4></div>
                <div class="panel-body">

        <form role="form"  method="post" action="signup_sub.php" enctype="multipart/form-data">
                                <div class="form-group">
                                  <label for="name">FirstName:</label>
                                <input type="text" required pattern="![0-9]" class="form-control"
name="n" id="name" placeholder="Enter name" required>
```

```
                                    </div><div class="form-group">
                                      <label for="name">MiddleName:</label>
                                      <input type="text" required pattern="![0-9]"class="form-control"
name="n" id="name" placeholder="Enter name" required>
                                    </div><div class="form-group"><div class="form-group">
                                      <label for="name">LastName:</label>
                                      <input type="text" class="form-control" name="n" id="name"
placeholder="Enter name" required>
                                    </div>              <label for="name">Course :</label>
            </label><br>
              <select>     ……...
            </select>
            </div> <div>   <label>
<label for="name">Gender :</label>
</label><br>
<select>
            <option value="Select Gender">Select Gender</option>
            <option value="Male">Male</option>
            <option value="Female">Female</option>
            <option value="Other">Other</option>
            <option value="Don't Mention">Don't Mention</option>
            </select>  </div>
```

8.Viewpost Page:

Screenshot:

Description:

This page enables users to Search for a particular Bug.

Elements used:

HTML (Forms,Text fields and input type), CSS Styling, Javascript Validation Functions.

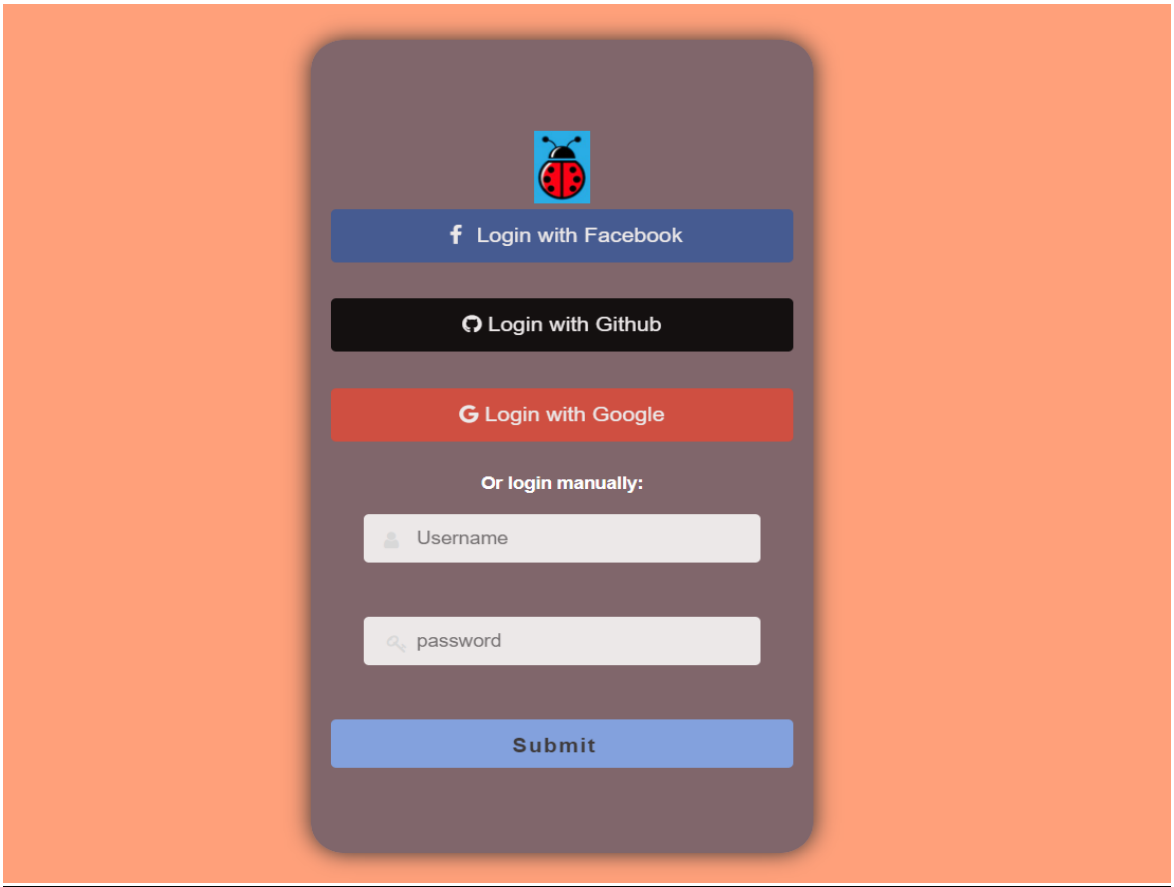
Code Snippet:

```
<!DOCTYPE html>
<html>
```

```
<head>
<meta name="viewpost" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
}
#myInput {.....}
#myUL {.....}
#myUL li a {……}
#myUL li a:hover:not(.header) {
  background-color: #eee;
}
</style>
</head>
<body>
<h2>view post</h2>
<input type="text" id="myInput" onkeyup="myFunction()" placeholder="Search for bug types.." title="Type
in a name">
<ul id="myUL">.....</ul>
<script>
function myFunction() {
    var input, filter, ul, li, a, i, txtValue;
    input = document.getElementById("myInput");
    filter = input.value.toUpperCase();
    ul = document.getElementById("myUL");
    li = ul.getElementsByTagName("li");
    for (i = 0; i < li.length; i++) {
        a = li[i].getElementsByTagName("a")[0];
        txtValue = a.textContent || a.innerText;
        if (txtValue.toUpperCase().indexOf(filter) > -1) {
            li[i].style.display = "";
        } else {
            li[i].style.display = "none";        }    }}
</script>
</body>
</html>
```

9.Login Page

Screenshot:



Description:

This page enables user to login into to website using username and password.

Elements used:

Form, text fields, input type and pattern validation

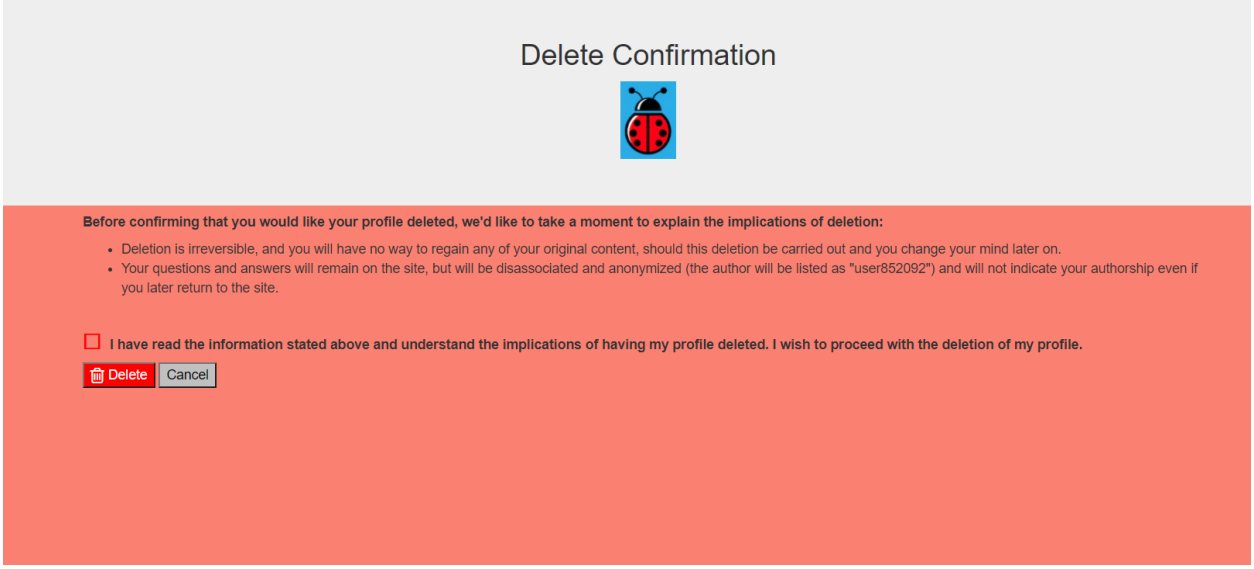
Code Snippet:

```
myInput.onfocus = function() {
  document.getElementById("message").style.display = "block";
}
myInput.onblur = function() {
  document.getElementById("message").style.display = "none";
}
myInput.onkeyup = function() {
  var lowerCaseLetters = /[a-z]/g;
  if(myInput.value.match(lowerCaseLetters)) {  …….  }
else { ……..}
 var upperCaseLetters = /[A-Z]/g;
if(myInput.value.match(upperCaseLetters)) {
    ……..}
else { ……..}
  var numbers = /[0-9]/g;
  if(myInput.value.match(numbers)) { ……..}
else { ……..}
```

```
 if(myInput.value.length >= 8) {  ……...}
else { …….. }
}
```

10. Delete Account

Screenshot:



Description:

This page enables user delete their account

Elements used:

Check box, buttons,Pop ups

Code Snippet:

```
<div class="jumbotron text-center" style="margin-bottom:0">
  <h1>Delete Confirmation</h1> </div>
<div class="container">
  <div class="row">
    <h5> <b>Before confirming that you would like your profile deleted, we'd like to take a moment to explain the implications of deletion:</b></h5>
<ul>.....</ul><br>
 <input type="checkbox" required name="confirm" id="confirm"> <label for="confirm"> I have read the information stated above and understand the implications of having my profile deleted. I wish to proceed with the deletion of my profile.</label>  <br>
    <button type="button" class="delete" onclick="document.getElementById('id01').style.display='block'"><i class="fa fa-trash"></i> Delete</button></div></div>
<div id="id01" class="modal">
  <span onclick="document.getElementById('id01').style.display='none'" class="close" title="Close Modal">×</span>
  <form class="modal-content">
    <div class="container">
      <h1>Delete Account</h1>
```

```
<p>Are you sure you want to delete your account?</p>
<div class="clearfix">
  <button type="button" </form>
```

Testing:-

Field	Input Given	Success/Failure	Reason for Failure
email	abcgmail.com	Failure	wrong format - no '@' present
email	pqr@gmail.com	Success	Field criteria is satisfied
password	Xyz12345	Success	Field criteria is satisfied
password	abcd6338	Failure	no Uppercase letter
password	abcdE8	Failure	Less than 8 characters
password	abcdEfhg	Failure	No digits
Name	12ab	Failure	Only alphabet and space after first word is applicable

Name	Null	Failure	Name field should be filled
Name	Abd efr	Success	Field criteria is satisfied
Mobile	abc	Failure	Only digits are accepted
Mobile	124945	Failure	Mobile should have 10 digits
Mobile	Null	Failure	Mobile Field should be filled
Mobile	8754526092	Success	Field criteria is satisfied
Select Query	Null	Failure	An option in dropdown menu should be selected
Select Query	Other	Success	A separate form input is generated below the field for the Other option
Comment	Null	Failure	Comment field should be filled

Comment	>500 characters	Failure	Character limit is 500
Comment	<500 characters	Success	Field Criteria is satisfied

Evaluation Sheet

Roll No	Technology	Max Marks	Marks Awarded
CB.EN.U4CSE17008	HTML CSS JS Viva	10 10 10 10	
CB.EN.U4CSE17009	HTML CSS JS Viva	10 10 10 10	
CB.EN.U4CSE17044	HTML CSS JS Viva	10 10 10 10	

CB.EN.U4CSE17051	HTML CSS JS Viva	10 10 10 10	
CB.EN.U4CSE17052	HTML CSS JS Viva	10 10 10 10	
	Project Documentation	10	
	Total	50	

Technologies used -Term 2

XML – Extensible Markup Language

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification of 1998 and several other related specifications all of them free open standards define XML.

There are three important characteristics of XML that make it useful in a variety of systems and

solutions:

- **XML is extensible:** XML allows you to create your own self-descriptive tags, or language, that suits your application.
- **XML carries the data, does not present it:** XML allows you to store the data irrespective of how it will be presented.
- **XML is a public standard:** XML was developed by an organization called the World Wide Web Consortium (W3C) and is available as an open standard.

XML is not similar to HTML. XML was created so that richly structured documents could be used over the web. XML differs from HTML in following ways:

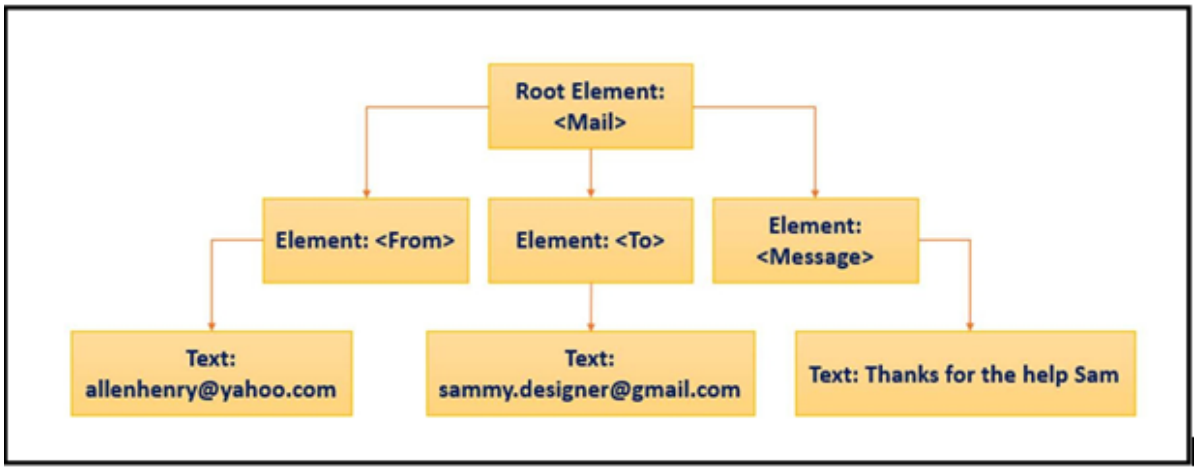
HTML	XML
HTML Document formats and displays web page data.	XML Document carry data along with their description.
HTML uses predefined tags like <h1>,<p>,etc.	XML uses user-defined tags (meta language).
HTML is not Case Sensitive.	XML is case sensitive.
It is directly viewable in browser.	Viewable if proper stylesheet provided.

Some important concepts in XML:

- **XML DOM:** It defines a standard way of accessing and manipulating XML Documents. It presents an XML document as a tree structure.
- **XPath:** It can be used to navigate through elements and attributes in an XML document.
- **XSLT:** It is the recommended style sheet language for XML. It is far more sophisticated than CSS. It uses XPath to find information in an XML Document.
- **XQuery:** XQuery is the language for querying XML Data.
- **XML Schema** or **XML DTD** can be used to validate an XML Document.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Mail>
    <From>allenhenry@yahoo.com</From>
    <To>sammy.designer@gmail.com</To>
    <Message>Thanks for the help Sam</Message>
</Mail>
```

Example XML Document



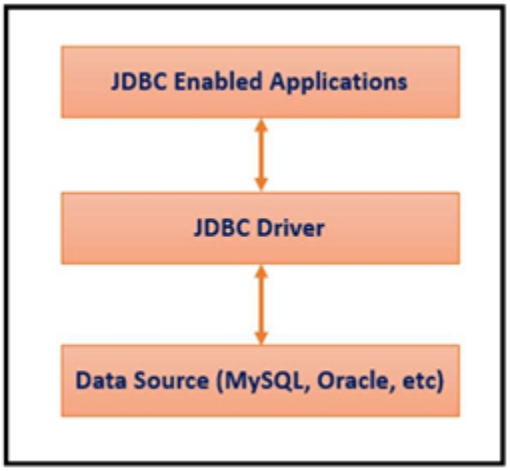
Example DOM Tree Structure

JDBC – Java Database Connectivity

Java Database Connectivity (JDBC) is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database. The JDBC data standard defines a set of Java interfaces to enable application developers abstract data access functionality including:

- Establish Connection with a data source
- Execute SQL Queries
- Process Result Sets

The architecture of JDBC-based data connectivity is as follows:



- **JDBC Enabled Application:** This is any application written in Java Programming Language.
- **JDBC Driver:** Commercial JDBC Drivers, or a framework to easily build a JDBC Driver, are available from vendors like Simba Technologies.

- There are four types of JDBC Drivers in use: JDBC-ODBC Bridges, Native API/Partly Java Driver, Pure Java Driver, Native Protocol Java Driver.
- **Data Source:** A data source can be a file or a particular database on a DBMS (such as Oracle, MySQL, etc).

JDBC drivers are available for popular databases such as PostgreSQL, MySQL, MariaDB, SQL Server, DB2, Oracle, H2, Derby, etc. If you are using Maven or Gradle in your project, then you would add the JDBC driver as a dependency to your project instead of adding the .jar file manually. For example, in the case of MySQL, you would have to add the following tag to your pom.xml file:

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.20</version>
</dependency>
```

The steps for connecting to a database with JDBC are as follows:

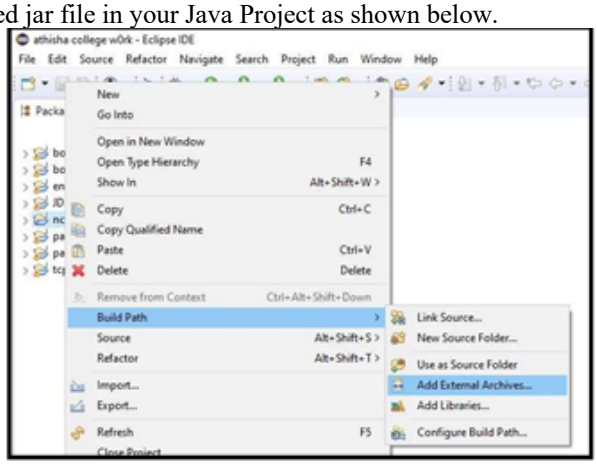
1. Install or locate the database you want to access.
2. Include the JDBC library.
3. Ensure the JDBC driver you need is on your classpath.
4. Use the JDBC library to obtain a connection to the database.
5. Use the connection to issue SQL commands.
6. Close the connection when you're finished.

Table Specifications

S.no	Table Name	Attributes/Features
1.	user_profile	username varchar(20) - profile username password varchar(20) - profile password email varchar(20) - registered email id phone number20) - mobile number
2.	login	username varchar(20) - profile username Password varchar(20) - profile password
3.	contactus	id varchar(7) - Unique token id to identify the enquiry/complaint name varchar(20) - Name of the person email varchar(30) - Email id of the person mobile int -Mobile number of the person query varchar (15) - Query category other varchar(20) - If query chosen is “Other” comment varchar(500) - Details of the enquiry/complaint in 500 characters
4.	Registration Page	Firstname varchar(20)-Name of the user Middlename varchar(20)-User middle name Lastname varchar(20)-User surname Phone varchar(11)-Contact info of user email varchar(30)-User mail id password varchar(20)-User password RetypePassword varchar(20)-User password
5.	Create Post Page	Title varchar(20)-Title of the post Language varchar(20)-Programming language used Content varchar(20)-The question to be asked

Steps in Database Connectivity

1. Download the corresponding JDBC jar file for your database. We have used JDBC connector for MySQL.
2. Include the downloaded jar file in your Java Project as shown below.



3. Import the java sql package in your class:

```
import java.sql.*;
```

4. Set Class Name:

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

5. Create Database Connection:

   Here url, username and password are string variables which contain the credentials of the respective database.

```
Connection con=DriverManager.getConnection(url,username,password);
```

6. Create a Statement Variable to execute the queries:

```
Statement st=con.createStatement();
```

7. Execute Query:

   - executeQuery() is used to execute a SQL query.
   - executeUpdate() is used to execute an Update Query such as Create, etc.

```
st.executeQuery("USE new_schema;");
st.executeUpdate(sql);
```

8. Close Statement and Connection Object.

```
st.close();
con.close()
```

user_profile

XML Code

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE user SYSTEM "user.dtd">
  <user>
  <person>
      <username>raju</username>
      <password>21331</password>
      <email>adas@gmail.com</email>
      <phone>9824521721</phone>
  </person>
  <person>
          <username>go123</username>
      <password>ramkum19%^</password>
      <email>das@outlook.com</email>
      <phone>8340613052</phone>
  </person>
   <person>
          <username>kola</username>
      <password>Kehk89!$687</password>
      <email>hmb@fsad.com</email>
      <phone>9065123665</phone>
  </person>
</user>
```

XML DTD – Document Type Definition

```
<?xml encoding="UTF-8"?>
<!ELEMENT user (person)+>
<!ATTLIST user xmlns CDATA #FIXED ">
<!ELEMENT person (username,password,email,phone)>
<!ATTLIST person xmlns CDATA #FIXED ">
<!ELEMENT username (#PCDATA)>
<!ATTLIST username xmlns CDATA #FIXED ">
<!ELEMENT password (#PCDATA)>
<!ATTLIST password xmlns CDATA #FIXED ">
<!ELEMENT email (#PCDATA)>
<!ATTLIST email xmlns CDATA #FIXED ">
<!ELEMENT phone (#PCDATA)>
<!ATTLIST phone xmlns CDATA #FIXED ">
```

XML Schema Specifications

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xs:element name="user">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="person"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="person">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="username"/>
        <xs:element ref="password"/>
        <xs:element ref="email"/>
        <xs:element ref="phone"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="username" type="xs:NCName"/>
  <xs:element name="password" type="xs:string"/>
  <xs:element name="email" type="xs:string"/>
  <xs:element name="phone" type="xs:integer"/> </xs:schema>
```

JDBC Connection:-

```
public class db {
        // JDBC driver name and database URL
           static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
           static final String DB_URL = "jdbc:mysql://localhost/db";
           //  Database credentials
           static final String USER = "root";
           static final String PASS = "root";
        public static void main(String[] args) {
                // TODO Auto-generated method stub
                  Connection conn = null;
                   Statement stmt = null;
                   try{
                      //STEP 2: Register JDBC driver
                      Class.forName("com.mysql.jdbc.Driver");
                      //STEP 3: Open a connection
                      System.out.println("Connecting to database...");
                      conn = DriverManager.getConnection(DB_URL, USER, PASS);
                      File file = new File("user.xml");
                    //an instance of factory that gives a document builder
                    DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
                    //an instance of builder to parse the specified xml file
                    DocumentBuilder db = dbf.newDocumentBuilder();
                    Document doc = db.parse(file);
                    doc.getDocumentElement().normalize();
                    System.out.println("Root element: " +
```

```
doc.getDocumentElement().getNodeName());
                    NodeList nodeList = doc.getElementsByTagName("person");
                  // nodeList is not iterable, so we are using for loop
                    for (int itr = 0; itr < nodeList.getLength(); itr++)
                    {
                    Node node = nodeList.item(itr);
                    if (node.getNodeType() == Node.ELEMENT_NODE)
                    {
                    Element eElement = (Element) node;
String username=
eElement.getElementsByTagName("username").item(0).getTextContent() ;
String password= eElement.getElementsByTagName("password").item(0).getTextContent()
String email= eElement.getElementsByTagName("email").item(0).getTextContent();
String phone= eElement.getElementsByTagName("phone").item(0).getTextContent();
                     //STEP 4: Execute a query
stmt = conn.createStatement();
String sql = "INSERT INTO user VALUES
('"+username+"','"+password+"','"+email+"','"+phone+"');";
                    // sql = "DELETE FROM user;";
                      stmt.executeUpdate(sql);
                    }
                    }
                      System.out.println("Query executed successfully...");
                   }catch(SQLException se){
                      //Handle errors for JDBC
                      se.printStackTrace();
                   }catch(Exception e){
                      //Handle errors for Class.forName
                      e.printStackTrace();
                   }finally{
                      //finally block used to close resources
                      try{
                         if(stmt!=null)
                            stmt.close();
                      }catch(SQLException se2){
                      }// nothing we can do
                      try{
                         if(conn!=null)
                            conn.close();
                      }catch(SQLException se){
                         se.printStackTrace();
                      }//end finally try
                   }//end try }
```

XSLT - eXtensible Stylesheet Language

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:template match="/">
<html>
<head>
<title>user details</title>
<style type="text/css"> .a{ text-align : center; color: black; } .clear { clear: both; } body { font-family:
Arial, Helvetica, sans-serif; font-height: 20px; background-color: salmon; } table { font-family: Arial,
```

```
Helvetica, sans-serif; border-collapse: collapse; width: 100%; font-weight: normal; } td, th { border: 3px
solid black; text-align: left; padding: 8px; font-weight: normal; } </style>
</head>
<body>
<h1 class="a">User Details</h1>
<table>
<tr>
<td><strong>Username</strong></td>
<td><strong>Password</strong></td>
<td><strong>Email</strong></td>
<td><strong>Phone</strong></td>
</tr>
<div class="clear"/>
<xsl:for-each select="user/person">
<tr>
<th><xsl:value-of select="username"/></th>
<th><xsl:value-of select="password"/></th>
<th><xsl:value-of select="email"/></th>
<th><xsl:value-of select="phone"/></th>
</tr>
<div class="clear"/>
</xsl:for-each>
</table></body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Output

User Details

Username	Password	Email	Phone
raju	21331	adas@gmail.com	9824521721
asd2312	87453Sjhf*	xfa@yahoo.com	7293442694
loki	janav995$%	kond@hotmail.in	9347187225
vasta	pass1234	adjasd@dia.com	9452264520
go123	ramkum19%^	das@outlook.com	8340613052
nithish19	Gbg86^*$10	kljl@kdjd.in	9523493617
aav007	Rav17@@ghd	asd@g.com	8674132105
kola	Kehk89!$687	hmb@fsad.com	9065123665
neeru43	YGH28*@)	asda@gmh.com	8354231531
surkesh99	UDI82*92	asda@gsaf.in	9432168645

REGISTRATION PAGE CODES:

Xml with dtd code:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="raj.xsl"?>
<!DOCTYPE signup
[
<!ELEMENT user (Firstname,Middlename,Lastname,Phone,email,password,RetypePassword)>
<!ELEMENT Firstname (#PCDATA)>
<!ELEMENT Middlename (#PCDATA)>
<!ELEMENT Lastname (#PCDATA)>
<!ELEMENT Phone (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT password (#PCDATA)>
<!ELEMENT RetypePassword (#PCDATA)>
]>
<signup xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="raju.xsd">
    <user>
    <Firstname>Ravi</Firstname>
    <Middlename>Kumar</Middlename>
    <Lastname>Reddy</Lastname>
    <Phone>950-2552046</Phone>
    <email>ravikumarreddy18@gmail.com</email>
    <password>Ravi1234</password>
    <RetypePassword>Ravi1234</RetypePassword>
    </user>
    </signup>
```

Xsd code:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:simpleType name="nametype">
             <xs:restriction base="xs:string">
             <xs:pattern value="[A-Z][a-z]+"/>
             </xs:restriction>
             </xs:simpleType>
<xs:simpleType name="phn" >
             <xs:restriction base="xsd:string">
             <xs:pattern value="[0-9]{3}-[0-9]{7}"/>
             </xs:restriction>
             </xs:simpleType>
              <xs:simpleType name="pass">
             <xs:restriction base="xs:string">
             <xs:pattern value="[a-zA-Z0-9]{8}"/>
             </xs:restriction>
             </xs:simpleType>
<xs:complexType name="registertype">
  <xs:sequence>
```

```
    <xs:element type="nametype" maxoccurs="1" name="Firstname"/>
    <xs:element type="nametype" maxoccurs="1" name="Middlename"/>
    <xs:element ttype="nametype" maxoccurs="1" name="Lastname"/>
    <xs:element type="phn" maxOccurs="1" name="Phone"/>
    <xs:element type="xs:string" name="email"/>
     <xs:element type="pass" maxoccurs="1" name="password"/>
     <xs:element type="pass" maxoccurs="1" name="RetypePassword"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="signup">
  <xs:complexType>
    <xs:sequence>
        <xs:element name="user" minOccurs="1" type="registertype" maxOccurs="20"/>
  </xs:sequence>
  </xs:complexType>
  </xs:element>
</xs:schema>
```

Xsl code:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html>
  <head>
  <style> …….. </style>
  </head>
  <body >
  <h2>Login Users Data</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Firstname</th>
      <th>Middlename</th>
      <th>Lastname</th>
      <th>Phone</th>
      <th>email</th>
      <th>password</th>
      <th>RetypePassword</th>
    </tr>
     <xsl:for-each select="signup/user"><tr>
      <td><xsl:value-of select="Firstname"/></td>
      <td><xsl:value-of select="Middlename"/></td>
      <td><xsl:value-of select="Lastname"/></td>
      <td><xsl:value-of select="Phone"/></td>
      <td><xsl:value-of select="email"/></td>
      <td><xsl:value-of select="password"/></td>
      <td><xsl:value-of select="RetypePassword"/></td></tr>
    </xsl:for-each>
  </table>
  </body>
```

```
  </html>
</xsl:template>
</xsl:stylesheet>
```

OUTPUT:

Login Users Data

Firstname	Middlename	Lastname	Phone	email	password	RetypePassword
Ravi	Kumar	Reddy	950-2552046	ravikumarreddy18@gmail.com	Ravi1234	Ravi1234
Abhi	Ram	Reddy	950-2552047	abhiramreddy@19	Abhi1234	Abhi1234

Jdbc code:

```
public class SignUp
{public static void main(String[] args) throws Exception
  {    Scanner in = new Scanner(System.in);
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/ncp_casestudy","root","admin");
    File file=new File("reg.xml");
    Statement st = con.createStatement();
    DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
    DocumentBuilder builder = factory.newDocumentBuilder();
    Document xml=builder.parse(file);
    NodeList weatherlength= xml.getElementsByTagName("user");
    javax.xml.xpath.XPath xpath = XPathFactory.newInstance().newXPath();
    Object res = xpath.evaluate("/signup/user",xml,XPathConstants.NODESET);
    System.out.println("Check MySql");
    PreparedStatement stmt = con.prepareStatement("insert into signup values(?,?,?,?,?,?,?)");
    for (int i=0;i<weatherlength.getLength();i++)
    {   Node node = weatherlength.item(i);
        List<String> columns = Arrays.asList(
          getTextContent(node, "Firstname"),
            getTextContent(node, "Middlename"),
            getTextContent(node, "Lastname"),
          getTextContent(node, "Phone"),
          getTextContent(node, "email"),
        getTextContent(node, "password"),
          for (int n = 0 ; n <columns.size() ; n++)
        {stmt.setString(n+1, columns.get(n));}
        stmt.execute();}}
  static private String getAttrValue(Node node,String attrName)
  { if (!node.hasAttributes())    return "";
     NamedNodeMap nmap = node.getAttributes();
     if(nmap==null)    return "";
     Node n=nmap.getNamedItem(attrName);
     if (n==null)      return "";
     return n.getNodeValue();}
  static private String getTextContent(Node parentNode,String childName)
  { NodeList nlist = parentNode.getChildNodes();
     for (int i = 0;i<nlist.getLength();i++)
     {Node n = nlist.item(i);
       String name = n.getNodeName();
       if(name != null && name.equals(childName))
         return n.getTextContent();}
     return "";}}
```

Contact Us page:-

XML code:-

```
<?xml version="1.0" encoding="utf-8" ?>
<contactUs>
<token id="dcd12">
<name>Arul</name>
<email>arul@gmail.com</email>
<mobile>2910564738</mobile>
<query>Other</query>
<other>Regarding Logo</other>
<comment>The Logo is bad, please update it</comment>
</token>
<token id="dcsfvrw">
<name>Jayanth</name>
<email>jayanth@gmail.com</email>
<mobile>6758493021</mobile>
<query>Careers</query>
<other/>
<comment>Iam a final year student looking for entry level position</comment>
</token>
<token id="sfdvcsd12">
<name>Muthu</name>
<email>muthu@gmail.com</email>
<mobile>1234567890</mobile>
<query>Advertising</query>
<other/>
<comment>We are looking for sponsorship for our college techfest</comment>
</token>
</contactUs>
```

DTD code:-

```
!ELEMENT contactUs (token*)>
<!ELEMENT token (name,email,mobile,query,other,comment)>
<!ATTLIST token id CDATA #REQUIRED>
<!ELEMENT name (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT mobile (#PCDATA)>
<!ELEMENT query (#PCDATA)>
<!ELEMENT other (#PCDATA)*>
<!ELEMENT comment (#PCDATA)>
```

XSD code:-

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="contactUs">
  <xsd:complexType>
   <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="token">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="name" type="xsd:string" />
                <xsd:element name="email" type="xsd:string" />
                <xsd:element name="mobile" type="xsd:integer" />
                <xsd:element name="query" type="xsd:string" />
                <xsd:element name="other" type="xsd:string" />
                <xsd:element name="comment" type="xsd:string" />
        </xsd:sequence>
        <xsd:attribute name="id" type="xsd:string"/>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence> </xsd:complexType> </xsd:element> </xsd:schema>
```

Output for DTD and XSD validation:-

```
(base) Aruls-MacBook-Pro:XML1 aruljayanth$ xmllint --noout --dtdvalid storeDTD.dtd storeDTD.xml
(base) Aruls-MacBook-Pro:XML1 aruljayanth$ xmllint --noout --schema storeXSD.xsd storeXSD.xml
storeXSD.xml validates
```

JDBC code:-

```
public class xmlexcel {
public static void main(String[] args) throws FilloException, SAXException, IOException,
ParserConfigurationException {
    Fillo fillo = new Fillo();
    com.codoid.products.fillo.Connection connection = fillo.getConnection("./Book1.xlsx");
   System.out.println("Existing record in the excel sheet");
   String strQuery = "Select * from Sheet1";
   Recordset recordset = connection.executeQuery(strQuery);
   DocumentBuilderFactory docBuilderFactory = DocumentBuilderFactory.newInstance();
      DocumentBuilder docBuilder = docBuilderFactory.newDocumentBuilder();
      Document doc = docBuilder.parse(new File("./database.xml"));
      doc.getDocumentElement().normalize();
System.out.println("Root element of doc “+doc.getDocumentElement().getNodeName());
      NodeList listOfPersons = doc.getElementsByTagName("token");
      for (int s = 0; s < listOfPersons.getLength(); s++) {
         Node firstPersonNode = listOfPersons.item(s);
         if (firstPersonNode.getNodeType() == Node.ELEMENT_NODE) {
            Element firstPersonElement = (Element) firstPersonNode;
        NodeList idList = (NodeList) firstPersonElement.getAttributes().getNamedItem("id");
```

```
            String id = ((Node) idList).getFirstChild().getNodeValue().trim();
            NodeList nameList = firstPersonElement.getElementsByTagName("name");
            Element nameElement = (Element) nameList.item(0);
            NodeList text2List = nameElement.getChildNodes();
            String name = ((Node) text2List.item(0)).getNodeValue().trim();
     NodeList queryList = firstPersonElement.getElementsByTagName("query");
            Element queryElement = (Element) queryList.item(0);
            NodeList text5List = queryElement.getChildNodes();
            String query = ((Node) text5List.item(0)).getNodeValue().trim();
            String other="";
 if(query.equals("Other")){
                NodeList otherList = firstPersonElement.getElementsByTagName("other");
                Element otherElement = (Element) otherList.item(0);
                NodeList text6List = otherElement.getChildNodes();
                other+= ((Node) text6List.item(0)).getNodeValue().trim(); }
NodeList commentList = firstPersonElement.getElementsByTagName("comment");
Element commentElement = (Element) commentList.item(0);
NodeList text7List = commentElement.getChildNodes();
String comment = ((Node) text7List.item(0)).getNodeValue().trim();
connection.executeUpdate("insert into Sheet1(id,name,email,mobile,query,other,comment)
values('" + id + "','" + name + "','" + email +"','" + mobile +"','" + query +"','" + other +"','" +
comment +"')"); } }
      System.out.println("\nContents after inserting data to excel file");
      strQuery = "Select * from Sheet1";
      recordset = connection.executeQuery(strQuery);
      recordset.close();
      connection.close(); }}
```

Output:-

```
/Library/Java/JavaVirtualMachines/jdk-10.0.1.jdk/Contents/Home/bin/java "-javaagent:/Applications/IntelliJ IDEA CE.app/Contents/lib/idea_rt.jar=53536:/Applications/IntelliJ IDEA CE
Existing record in the excel sheet
id: dcd12 name: Arul email: arul@gmail.com mobile:  2910564738 query:   Other other:    Regarding Logo comment: The Logo is bad, please update it
id: dcsfvrw name: Jayanth email:    jayanth@gmail.com mobile:   6758493021 query:   Careers other:   comment:   Iam a final year student looking for entry level position

Root element of the doc is contactUs
7 columns(s) affected
7 columns(s) affected
7 columns(s) affected

Contents after inserting data to excel file
id: dcd12 name: Arul email: arul@gmail.com mobile:  2910564738 query:   Other other:    Regarding Logo comment: The Logo is bad, please update it
id: dcsfvrw name: Jayanth email:    jayanth@gmail.com mobile:   6758493021 query:   Careers other:   comment:   Iam a final year student looking for entry level position
id: dcd12 name: Arul email: arul@gmail.com mobile:  2910564738 query:   Other other:    Regarding Logo comment: The Logo is bad, please update it
id: dcsfvrw name: Jayanth email:    jayanth@gmail.com mobile:   6758493021 query:   Careers other:   comment:   Iam a final year student looking for entry level position
id: sfdvcsd12 name: Muthu email:    muthu@gmail.com mobile: 1234567890 query:   Advertising other:   comment:   We are looking for sponsorship for our college techfest

Process finished with exit code 0
```

XSLT code:-

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="contactUs">
<html>  <head>
<style type="text/css"> ….. </style>  </head>
<body><center><h1>Contact Us</h1>
 <table border="2"> <tr bgcolor="#2acd22">  ……..  </tr>
  <xsl:for-each select="token">  <tr>
    <td><xsl:value-of select="@id"/></td>
    <td><xsl:value-of select="name"/></td>
    <td><xsl:value-of select="email"/></td>
    <td><xsl:value-of select="mobile"/></td>
    <td><xsl:value-of select="query"/></td>
    <td><xsl:value-of select="other"/></td>
    <td><xsl:value-of select="comment"/></td>  </tr>
  </xsl:for-each>  </table>
</center>  </body>  </html>
</xsl:template>  </xsl:stylesheet>
```

OUTPUT:-

LOGIN PAGE:

XML CODE

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE form SYSTEM "login.dtd">
<form>
<?xml-stylesheet href="loginhtml.xsl" type="text/xsl"?>
   <!--                    <form                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="./login.xsd"> -->
<article lang="login">
   <para>
      <Name>.....</Name>
      <Password>.....</Password>
   </para>
   </article>
</form>
```

DTD CODE

```
<!ELEMENT form (article*)>
<!ELEMENT article (para*)>
<!ATTLIST article lang CDATA #REQUIRED>
<!ELEMENT para (Name,Password)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Password (#PCDATA)>
```

XSD CODE

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="form">
   <xsd:complexType>
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
     <xsd:element name="article">
        <xsd:complexType>
           <xsd:sequence minOccurs="0" maxOccurs="unbounded">
              <xsd:element name="para">
                 <xsd:complexType>
                    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                       <xsd:element name="Name" type="xsd:string" />
                       <xsd:element name="Password" type="xsd:string" />
                    </xsd:sequence>
                 </xsd:complexType>
              </xsd:element>
           </xsd:sequence>
           <xsd:attribute name="lang" type="xsd:string"/>
        </xsd:complexType>
     </xsd:element>
    </xsd:sequence>
   </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

XSLT CODE

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="form">
   <xsl:for-each select="article">
      <h1>Login: <xsl:value-of select="@login"></xsl:value-of></h1>
         <xsl:for-each select="para">
            <div style="display:inline-block;width:25%;margin-left:6%;vertical-align:top;">
               <h3><xsl:value-of select="Name"></xsl:value-of> </h3>
               <h3><xsl:value-of select="Password"></xsl:value-of></h3>
            </div>
   </xsl:for-each>
   <hr/>
   </xsl:for-each>
</xsl:template>
<xsl:output method="html" version="1.0"
encoding="UTF-8" indent="yes"/>
</xsl:stylesheet>
```

Output:

Login Users Data

Firstname	Middlename	Lastname	Phone	email	password
Ravi	Kumar	Reddy	950-2552046	ravikumarreddy18@gmail.com	Ravi1234
Abhi	Ram	Reddy	950-2552047	abhiramreddy@19	Abhi1234

JDBC CODE

```
public class login {
  static final String drive="com.mysql.jdbc.Driver";
  static final String url="jdbc:mysql://localhost:3306/ncp";
  static final String user="root";
  static final String pass="";
    public static void main(String[] args) {
       Scanner sc=new Scanner(System.in);
    System.out.println("JDBC Connection");
    Connection conn=null;
    Statement stmt=null;
    ResultSet rs=null;
       String sql="";
    Scanner scan=new Scanner(System.in);
       switch(op) {
       case 1:
         System.out.println("Contents in xml");
   DocumentBuilderFactory docBuilderFactory = DocumentBuilderFactory.newInstance();
       DocumentBuilder docBuilder = docBuilderFactory.newDocumentBuilder();
```

```
      Document doc = docBuilder.parse(new File("login.xml"));
      doc.getDocumentElement().normalize();
      System.out.println("Root element of the doc is " + doc.getDocumentElement().getNodeName());
      NodeList listOfMessage = doc.getElementsByTagName("para");
      for (int s = 0; s < listOfMessage.getLength(); s++) {
        Node messageNode = listOfMessage.item(s);
        if (messageNode.getNodeType() == Node.ELEMENT_NODE) {
          Element messageElement = (Element) messageNode;
          NodeList nameList = messageElement.getElementsByTagName("Name");
          Element nameElement = (Element) nameList.item(0);
          NodeList textFNList = nameElement.getChildNodes();
          String Name = ((Node) textFNList.item(0)).getNodeValue().trim();
          NodeList pwd = messageElement.getElementsByTagName("Password");
          Element pwdlist = (Element) pwd.item(0);
          NodeList textLNList = pwdlist.getChildNodes();
          String Password = ((Node) textLNList.item(0)).getNodeValue().trim();
          System.out.println("--------------------");
          System.out.println("Name: "+Name);
          System.out.println("Password: "+Password);   }}
        break;
        case 2:
          stmt=conn.createStatement();
          sql="select * from login";
          rs=stmt.executeQuery(sql);
          boolean isEmpty=true;
          while(rs.next()) {
          isEmpty=false;
          System.out.println("---------------------------");
          System.out.println("Name: "+rs.getString("username"));
          System.out.println("Password: "+rs.getString("password"));
          System.out.println("---------------------------"); }
        if(isEmpty) {
          System.out.println("Table has no data");}
        break; }
    scan.close();
     sc.close();
      System.exit(0);}
  catch(Exception e){
    System.out.println(e);}}}
```

Create Post Page

XML Code

```
<?xml version="1.0" encoding="utf-8" ?>
<gadgetList>
<postDetails>
<title>NCP review code</title>
```

```
<lang>xml</lang>
<content>what is xml ? Can someone clear this doubt?</content>
</postDetails>
<postDetails>
<title>computational Intelligence review code</title>
<lang>python</lang>
<content>what are the imports required for basic data
preprocessing?</content>
</postDetails>
<postDetails>
<title>MLDM review code</title>
<lang>Python</lang>
<content>What is difference between Data mining and pattern
recognition</content>
</postDetails>
<postDetails>
<title>Final year review code</title>
<lang>Linux</lang>
<content>How to implement tcp simulation in Linux kernel?</content>
</postDetails>
<postDetails>
<title>SICP year review code</title>
<lang>Racket</lang>
<content>How to install racket?</content>
</postDetails>
<postDetails>
<title>Final year review code</title>
<lang>Linux</lang>
<content>How to implement tcp simulation in Linux kernel?</content>
</postDetails>
</gadgetList>
```

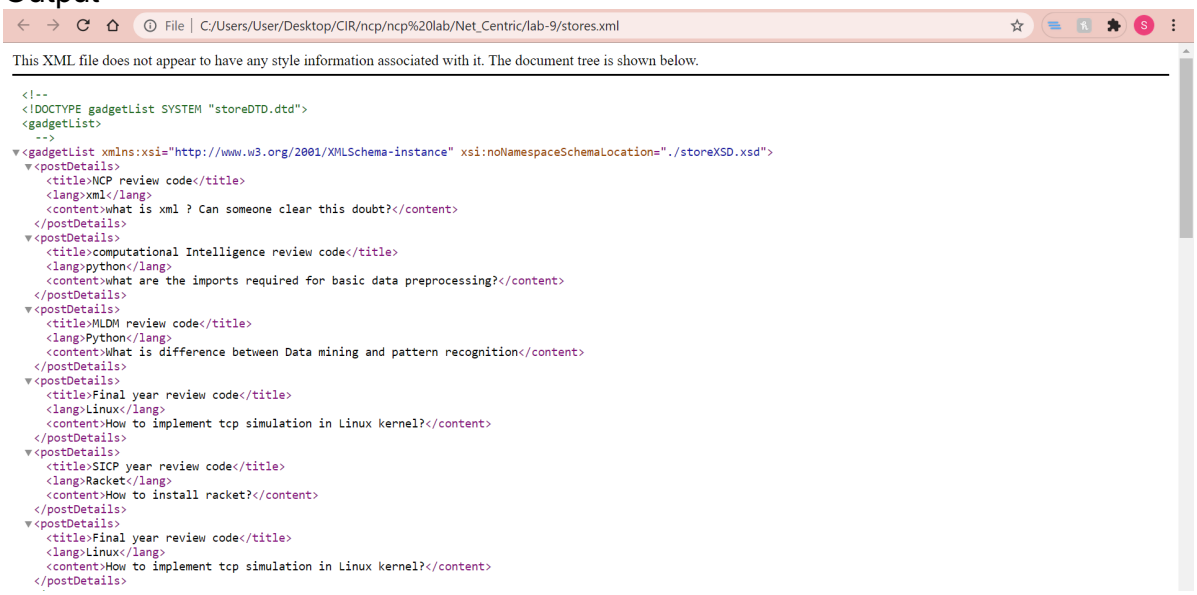
XML DTD – Document Type Definition

```
<!ELEMENT gadgetList (productDetails*)>
<!ELEMENT productDetails (title,lang,content)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT lang (#PCDATA)>
<!ELEMENT content (#PCDATA)>
```

XML Schema Specifications

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="gadgetList">
    <xsd:complexType>
      <xsd:sequence minOccurs="0" maxOccurs="unbounded">


            <xsd:complexType>
                <xsd:sequence minOccurs="0" maxOccurs="unbounded">

                    <xsd:element name="postDetails">

                        <xsd:complexType>
                            <xsd:sequence minOccurs="0"
maxOccurs="unbounded">
                                <xsd:element name="title" type="xsd:string"
/>
                                <xsd:element name="lang" type="xsd:string" />
                                <xsd:element name="content" type="xsd:string"
/>
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
                <xsd:attribute name="brand" type="xsd:string"/>
            </xsd:complexType>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Output



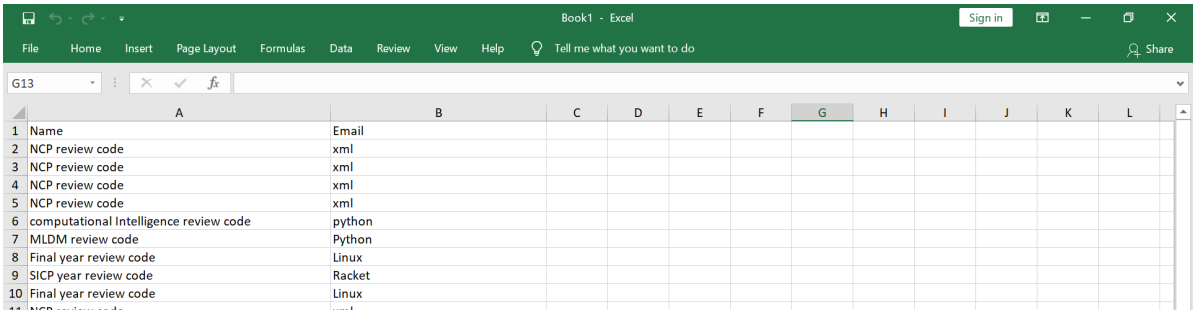
JDBC Connection:-

```
package jbdcXML;

import java.io.File;
import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

import com.codoid.products.exception.FilloException;
import com.codoid.products.fillo.Fillo;
import com.codoid.products.fillo.Recordset;

public class jdbcXML {

    public static void main(String[] args) throws FilloException,
SAXException, IOException, ParserConfigurationException {
        // TODO Auto-generated method stub
        Fillo fillo = new Fillo();
        com.codoid.products.fillo.Connection connection =
```

```
fillo.getConnection("C:/Users/User/Desktop/CIR/ncp/ncp lab/Net_Centric/lab-
11/lab-11/jbdcXML/Book1.xlsx");
        System.out.println("Existing record in the excel sheet");
        String strQuery = "Select * from Sheet1";
        Recordset recordset = connection.executeQuery(strQuery);
        while (recordset.next()) {
            System.out.println("name: " + recordset.getField("name") + "
email: " + recordset.getField("email"));
        }
        System.out.println();
        DocumentBuilderFactory docBuilderFactory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder docBuilder = docBuilderFactory.newDocumentBuilder();
        Document doc = docBuilder.parse(new
File("C:/Users/User/Desktop/CIR/ncp/ncp lab/Net_Centric/lab-11/lab-
11/jbdcXML/database.xml"));
        doc.getDocumentElement().normalize();
        System.out.println("Root element of the doc is " +
doc.getDocumentElement().getNodeName());
        NodeList listOfPersons = doc.getElementsByTagName("postDetails");
        for (int s = 0; s < listOfPersons.getLength(); s++) {
            Node firstPersonNode = listOfPersons.item(s);
            if (firstPersonNode.getNodeType() == Node.ELEMENT_NODE) {
                Element firstPersonElement = (Element) firstPersonNode;
                NodeList nameList =
firstPersonElement.getElementsByTagName("title");
                Element nameElement = (Element) nameList.item(0);
                NodeList textFNList = nameElement.getChildNodes();
                String title = ((Node)
textFNList.item(0)).getNodeValue().trim();
                NodeList emailList =
firstPersonElement.getElementsByTagName("lang");
                Element emailElement = (Element) emailList.item(0);
                NodeList textLNList = emailElement.getChildNodes();
                String lang = ((Node)
textLNList.item(0)).getNodeValue().trim();
                connection.executeUpdate("insert into Sheet1(Name,email)
values('" + title + "','" + lang + "')");
                System.out.println("record inserted");
            }
        }
        System.out.println("\nContents after inserting data to excel file");
```

```
        strQuery = "Select * from Sheet1";
        recordset = connection.executeQuery(strQuery);
        while (recordset.next()) {
            System.out.println("title: " + recordset.getField("name") + "
lang: " + recordset.getField("email"));
        }
        recordset.close();
        connection.close();
    }
}
```

Output:



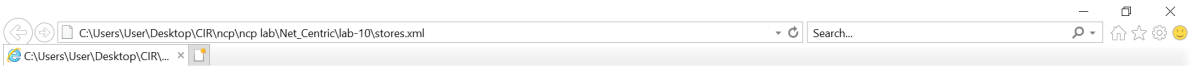
	A	B
1	Name	Email
2	NCP review code	xml
3	NCP review code	xml
4	NCP review code	xml
5	NCP review code	xml
6	computational Intelligence review code	python
7	MLDM review code	Python
8	Final year review code	Linux
9	SICP year review code	Racket
10	Final year review code	Linux

XSLT - eXtensible Stylesheet Language

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:transform version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="gadgetList">
            <xsl:for-each select="postDetails">
                <xsl:value-of select="title"></xsl:value-of> ...
                ------------------------------------------
                <xsl:value-of select="lang"></xsl:value-of>
                ------------------------------------------
                <xsl:value-of select="content"></xsl:value-of>
            ------------------------------------------------------------
        <xsl:text>&#xa;</xsl:text>
        <xsl:text>&#xd;</xsl:text>
            </xsl:for-each>
    xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
</xsl:template>
<xsl:output method="text" version="1.0"
encoding="UTF-8" indent="yes"/>
</xsl:transform>
```

Output

HTML



NCP review code

xml

what is xml ? Can someone clear this doubt?

computational Intelligence review code

python

what are the imports required for basic data preprocessing?

MLDM review code

Python

What is difference between Data mining and pattern recognition

Final year review code

Linux

How to implement tcp simulation in Linux kernel?

SICP year review code

Racket

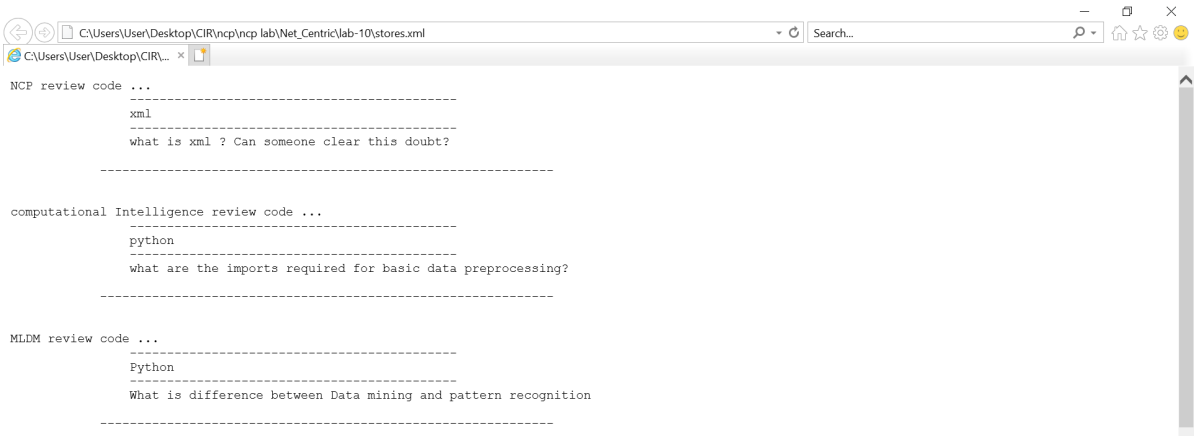
How to install racket?

Final year review code

Linux

How to implement tcp simulation in Linux kernel?

TEXT



Evaluation Sheet

Registration Number	Technology/Max Marks	Marks Awarded
CB.EN.U4CSE17008	XML(15) JDBC(15) VIVA(10) Total	
CB.EN.U4CSE17009	XML(15) JDBC(15) VIVA(10) Total	
CB.EN.U4CSE17044	XML(15) JDBC(15) VIVA(10) Total	
CB.EN.U4CSE17051	XML(15) JDBC(15) VIVA(10) Total	
CB.EN.U4CSE17052	XML(15) JDBC(15) VIVA(10) Total	
	Documentation(10)	

Technologies used -Term 3

Servlets

Servlet technology is used to create a web application (resides at server side and generates a dynamic web page). Servlet technology is robust and scalable because of java language. Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language. However, there were many disadvantages to this technology. We have discussed these disadvantages below. There are many interfaces and classes in the Servlet API such as Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse, etc.

What is a Servlet?

Servlet can be described in many ways, depending on the context.

- Servlet is a technology which is used to create a web application.
- Servlet is an API that provides many interfaces and classes including documentation.
- Servlet is an interface that must be implemented for creating any Servlet.
- Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
- Servlet is a web component that is deployed on the server to create a dynamic web page.

JSP

Java Server Pages (JSP) is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. This tutorial will teach you how to use Java Server Pages to develop your web applications in simple and easy steps.

Why to Learn JSP?

JavaServer Pages often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But JSP offers several advantages in comparison with the CGI.

- Performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself instead of having separate CGI files.
- JSP are always compiled before they are processed by the server unlike CGI/Perl which requires the server to load an interpreter and the target script each time the page is

requested.
● JavaServer Pages are built on top of the Java Servlets API, so like Servlets, JSP also has access to all the powerful Enterprise Java APIs, including JDBC, JNDI, EJB, JAXP, etc.
● JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.
Finally, JSP is an integral part of Java EE, a complete platform for enterprise class applications. This means that JSP can play a part in the simplest applications to the most complex and demanding.

Applications of JSP

As mentioned before, JSP is one of the most widely used languages over the web. Few of them are:

- **JSP vs. Active Server Pages (ASP)**
  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.
- **JSP vs. Pure Servlets**
  It is more convenient to write (and to modify!) regular HTML than to have plenty of println statements that generate the HTML.
- **JSP vs. JavaScript**
  JavaScript can generate HTML dynamically on the client but can hardly interact with the web server to perform complex tasks like database access and image processing etc.

Amazon Web Services (AWS) - Cloud Computing Services

Amazon Web Services (AWS) is the world's most comprehensive and broadly adopted cloud platform, offering over 175 fully featured services from data centers globally. Millions of customers—including the fastest-growing startups, largest enterprises, and leading government agencies—are using AWS to lower costs, become more agile, and innovate faster.

Most functionality

AWS has significantly more services, and more features within those services, than any other cloud provider–from infrastructure technologies like compute, storage, and databases–to emerging technologies, such as machine learning and artificial intelligence, data lakes and analytics, and Internet of Things. This makes it faster, easier, and more cost effective to move your existing applications to the cloud and build nearly anything you can imagine.

AWS also has the deepest functionality within those services. For example, AWS offers the widest variety of databases that are purpose-built for different types of applications so you can choose the right tool for the job to get the best cost and performance.

Most secure

AWS is architected to be the most flexible and secure cloud computing environment available today. Our core infrastructure is built to satisfy the security requirements for the military, global banks, and other high-sensitivity organizations. This is backed by a deep set of cloud security tools, with 230 security, compliance, and governance services and features. AWS supports 90 security standards and compliance certifications, and all 117 AWS services that store customer data offer the ability to encrypt that data.

Fastest pace of innovation

With AWS, you can leverage the latest technologies to experiment and innovate more quickly. We are continually accelerating our pace of innovation to invent entirely new technologies you can use to transform your business. For example, in 2014, AWS pioneered the serverless computing space with the launch of AWS Lambda, which lets developers run their code without provisioning or managing servers. And AWS built Amazon SageMaker, a fully managed machine learning service that empowers everyday developers and scientists to use machine learning–without any previous experience.

Steps for J2EE DEPLOYMENT

Requirements

- Java Development Kit, which can be downloaded from here.
- IntelliJ Idea Ultimate Edition, download it here. The ultimate edition is paid, but comes with a 30-day trial license. However, as long as you are a student, you can claim an educational license for free. More about it here.
- Heroku account. If you don't have, create one for free here.
- Heroku CLI. Find the instructions here.
- Tomcat server. You need to download the same from this link, under Binary Distributions > Core > zip. After downloading, make sure to extract the contents to any folder of your wish.
- If you're on Linux/macOS, you'll need to enable executable permission for

the tomcat binaries by running the following command on your terminal.

```
sudo chmod -R +x <path-to-tomcat-extracted-folder>/bin
```

J2EE Project Creation

- Open IntelliJ Idea Ultimate and create a new project.
- Create a Java Enterprise project, with Maven as the build tool, JUnit as the test runner, and Java as the language.
- In the next window, under Specifications, mark the checkbox for Servlet and ignore the rest.
- Regarding the project name and properties, you can fill in as you wish, and click on Finish!
- Now, the creation of the project is complete and the IDE might take sometime to load all the necessary plugins/dependencies.
- If you look at the project directory, there will be a pom.xml at the root of the file. This is the file which Maven uses to maintain the dependencies of your project. Essentially, it's similar to a package.json in NodeJS (or) composer.json in PHP.

Incase you ever want to add a library, say MySQL connector, you'll need to get its dependency XML tag from here and paste inside your pom.xml. After which, you'll see a tiny dialog on your top-right with a refresh button which on clicking will install the newly added package.

JSP

- Let's create a JSP file. Right click on Project Folder > src > main > webapp from the sidebar, and create a JSP file named hello.jsp, and fill in some JSP content.
- Click on Add configuration at the center top. After the window opens, click on the + button as instructed, and choose Tomcat server > Local.
- Give it a name as you wish, and near the Application server input box, click on the configure button. Now, enter the folder path where you extracted the zip of the Tomcat server.
- If you see a warning at the bottom saying No artifacts marked for

deployment, click the Fix button near the warning and choose ProjectName:war exploded. Then, click Okay.

- Now, you'll see a neat Run button at the top. Click on it and wait until Java compiles your web app and opens your default browser.
- Your browser would have opened the URL of your application context, similar to localhost:8080/J2EEDemo_war_exploded. This is the root of your web application. Append /hello.jsp to your current URL (it becomes http://localhost:8080/J2EEDemo_war_exploded/hello.jsp) and enter. You'll see the contents you wrote in your hello.jsp.

```
<div id="profile_page">

<h2>MyProfile</h2>                                                    <img
src="https://t3.ftcdn.net/jpg/00/64/67/80/240_F_64678017_zUpiZFjj04cnLri7oADnyMH0XBYyQghG.jpg">
<br>      <%  String driverName = "com.mysql.jdbc.Driver";

StringconnectionUrl=                                    "jdbc:mysql://bugoverflow.clhuzfls9udn.us-east-
1.rds.amazonaws.com:3306/bugoverflow?autoReconnect=true&useSSL=false";

String userId = "root";

String password = "bugoverflow";

String email =null;

 String phone =null;
```

Servlet

- Let's create a Servlet now. Right-click on Project Folder > src > main > java from the sidebar and create a new Java Servlet. Enter the name and package as you wish.
- Under the doGet() method, paste the following code :

```
@WebServlet(name = "LoginServlet")
public class LoginServlet extends HttpServlet {
    protected      void      doGet(HttpServletRequest      request,
HttpServletResponse     response)     throws     ServletException,
IOException {
```

```
        PrintWriter out = response.getWriter();
        out.println("You are at login page");}}
```

- Next thing to do is assign an URL to this servlet. Open src/main/webapp/WEB-INF/web.xml and paste the following :

Make sure you change the servlet-class according to your package and class name.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
         version="4.0">
    <servlet>
<servlet-name>LoginServlet</servlet-name>                <servlet-
class>in.co.aruljayanth.LoginServlet</servlet-class>
    </servlet> <servlet-mapping>
        <servlet-name>LoginServlet</servlet-name>
        <url-pattern>/login</url-pattern>
    </servlet-mapping> </web-app>
```

- Now, click on the Run button again and select Restart server.
- Visit http://localhost:8080/J2EEDemo_war_exploded/login to get the contents from the servlet.

Implementation

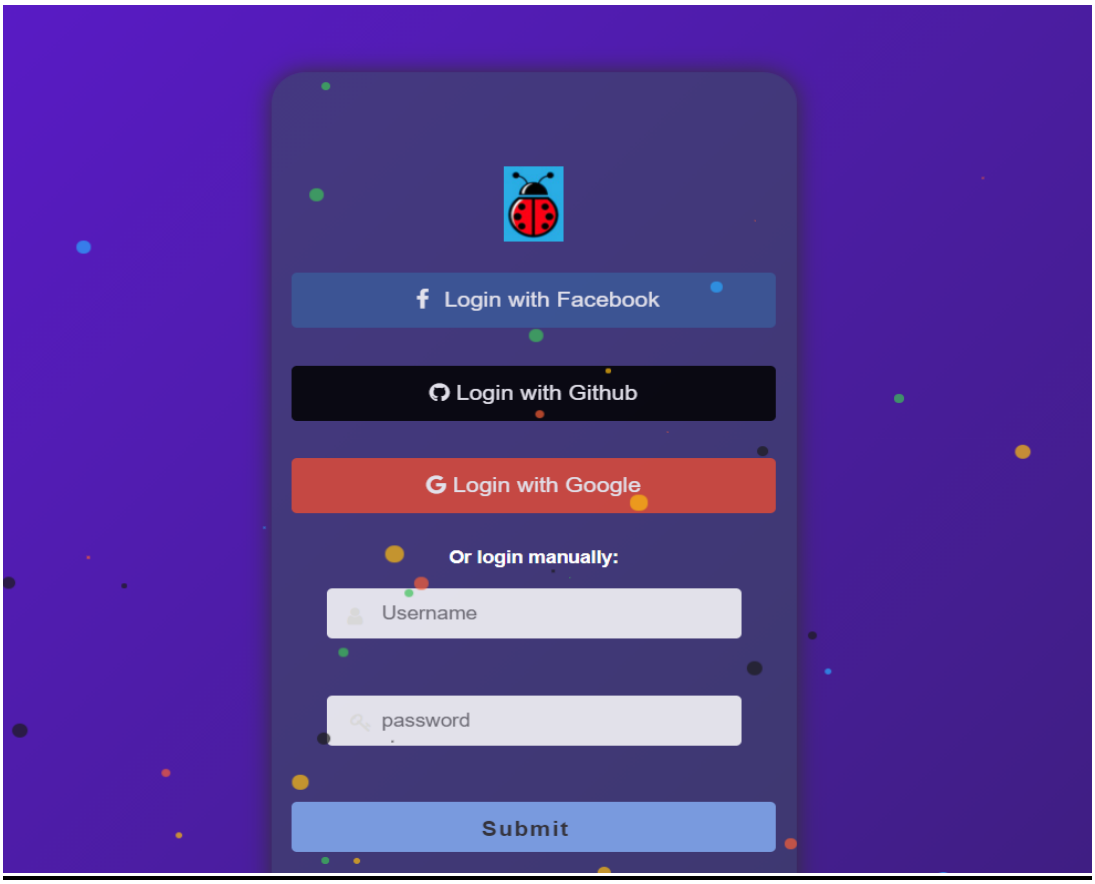
Login Servlet

```
public class FirstServlet extends HttpServlet {
public void doPost(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
   response.setContentType("text/html");
   PrintWriter out = response.getWriter();
   String n =request.getParameter("username");
   String p =request.getParameter("psw");
   HttpSession session = request.getSession(true);
```

```
   if(LoginDao.validate(n,p)){
      RequestDispatcher rd=request.getRequestDispatcher("welcome.jsp");
      rd.forward(request,response);}
   else{
      out.print("Sorry username or password is wrong");
      RequestDispatcher rd=request.getRequestDispatcher("login.html");
      rd.include(request,response);}
   out.close();}}
```

Welcome JSP

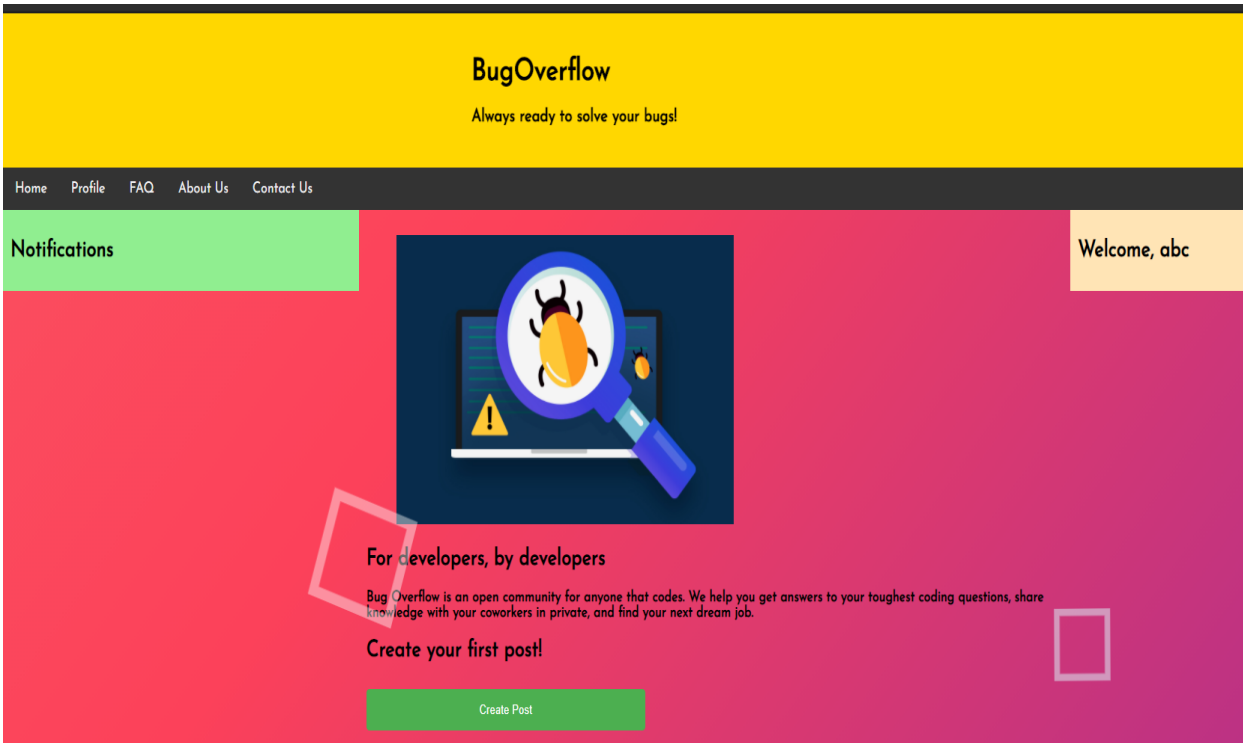
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<meta http-equiv="refresh"
      content="5; url = login.html" />
<title>Logged In</title>
</head>
<style type="text/css">
            body {
  background-color: lightsalmon ;
}
  </style>
<body>
            <table style="with: 50%">
            <tr><td>
            <% String username = request.getParameter("username"); %>
<a style="text-align:center;color:#3C4663">Welcome    <% out.println(username); %> User!!!! You have logged
in.</a>
</table>
</body>
</html>
```



Delete Servlet

```
public class SecondServlet extends HttpServlet {
public void doPost(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
   response.setContentType("text/html");
   PrintWriter out = response.getWriter();
   String n =request.getParameter("username");
   String p =request.getParameter("psw");
   HttpSession session = request.getSession(true);
   if(LoginDao.validate(n,p)){
         DelDao.validate(n);
      RequestDispatcher rd=request.getRequestDispatcher("deleted.html");
      rd.forward(request,response);}
   else{
      out.print("Sorry username or password is wrong");
      RequestDispatcher rd=request.getRequestDispatcher("delete-profile.html");
      rd.include(request,response);}
   out.close();}}
```

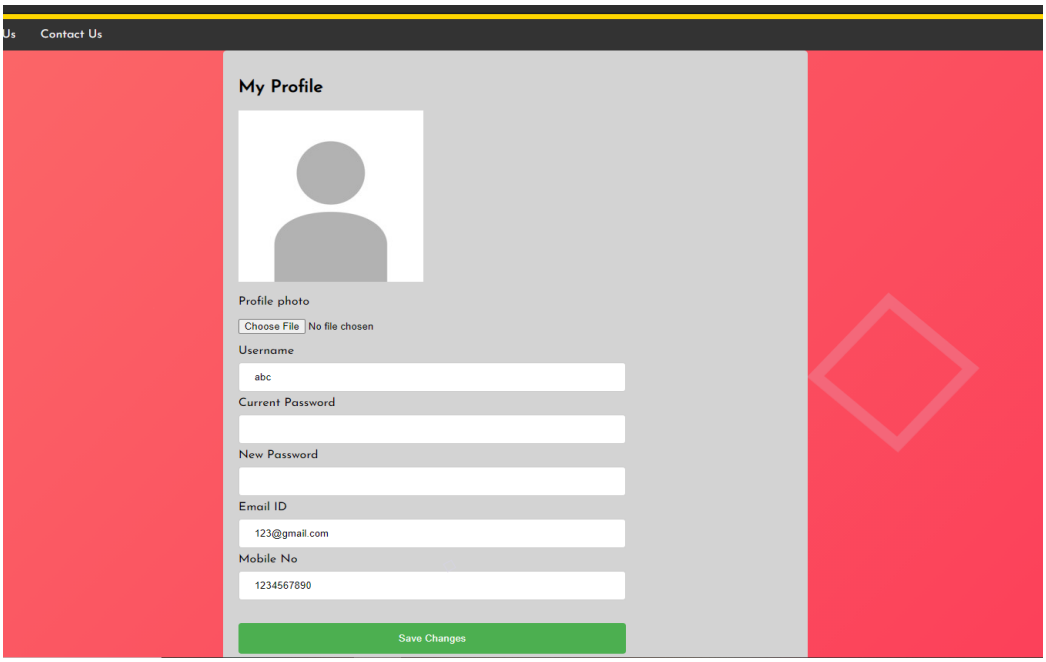
Home Jsp




```
<%
String driverName = "com.mysql.jdbc.Driver";
String connectionUrl = "jdbc:mysql://bugoverflow.clhuzfls9udn.us-east-1.rds.amazonaws.com:3306/bugoverflow?autoReconnect=true&useSSL=false";
String userId = "root";
String password = "bugoverflow";
String email =null;
  String phone =null;
try {
Class.forName(driverName);
} catch (ClassNotFoundException e) {
e.printStackTrace();
}

Connection connection = null;
Statement statement = null;
ResultSet rs = null;

try {
connection = DriverManager.getConnection(
connectionUrl , userId, password);
PreparedStatement ps=connection.prepareStatement(
"select * from profile where username=?");
ps.setString(1,username);
```

```
rs=ps.executeQuery();
rs.next();
email=rs.getString("email");
phone=rs.getString("phone");

}

catch (Exception e) {
e.printStackTrace();
}
%>
```

Profile Servlet




```
@WebServlet(urlPatterns = {"/profile"})
public class profile extends HttpServlet {
  static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
            static final String DB_URL = "jdbc:mysql://bugoverflow.clhuzfls9udn.us-east-
1.rds.amazonaws.com:3306/bugoverflow?autoReconnect=true&useSSL=false";

            //  Database credentials
            static final String USER = "root";
            static final String PASS = "bugoverflow";

   protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
     response.setContentType("text/html;charset=UTF-8");
     Connection conn = null;
                  Statement stmt = null;
     try ( PrintWriter out = response.getWriter()) {
```

```
        /* TODO output your page here. You may use following sample code. */

        String username = request.getParameter("uname");
          String password = request.getParameter("pw");
      String newpass = request.getParameter("npw");
          String email = request.getParameter("eid");
     String phone = request.getParameter("mno");

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

Admin Servlet

```
@WebServlet(urlPatterns = {"/profile"})
public class profile extends HttpServlet {
  static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
Static final String DB_URL = "jdbc:mysql://bugoverflow.clhuzfls9udn.us-east-1.rds.amazonaws.com:3306/bugoverflow?autoReconnect=true&useSSL=false";
            //  Database credentials
            static final String USER = "root";
            static final String PASS = "bugoverflow";
   protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
     response.setContentType("text/html;charset=UTF-8");
     Connection conn = null;
                    Statement stmt = null;
     try ( PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        String username = request.getParameter("uname");
          String password = request.getParameter("pw");
      String newpass = request.getParameter("npw");
          String email = request.getParameter("eid");
     String phone = request.getParameter("mno");
        Class.forName("com.mysql.jdbc.Driver");
```

Contact Servlet

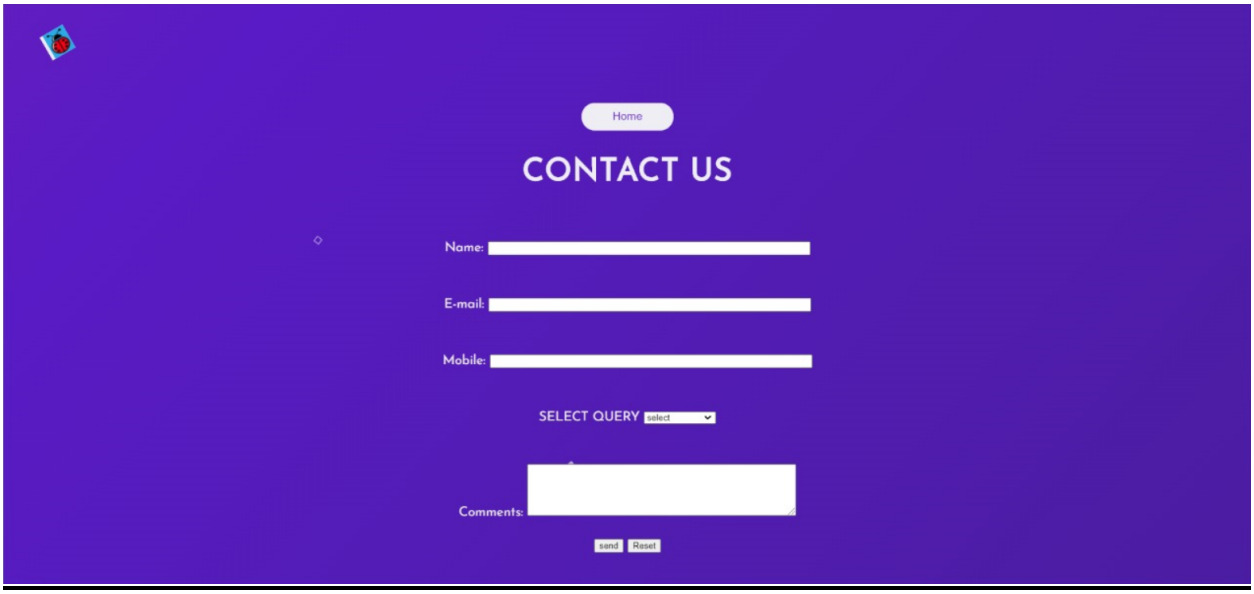
```
@WebServlet(urlPatterns = {"/ContactServlet"})
public class ContactServlet extends HttpServlet {
  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
      PrintWriter out = response.getWriter();
      out.println("<html>");
      out.println("<head>");
      out.println("<title>CONTACT US</title>");
      out.println("</head>");
      out.println("<center><h1>" + "ACKNOWLEDGEMENT" + "</h1></center>");
      out.println("");
```

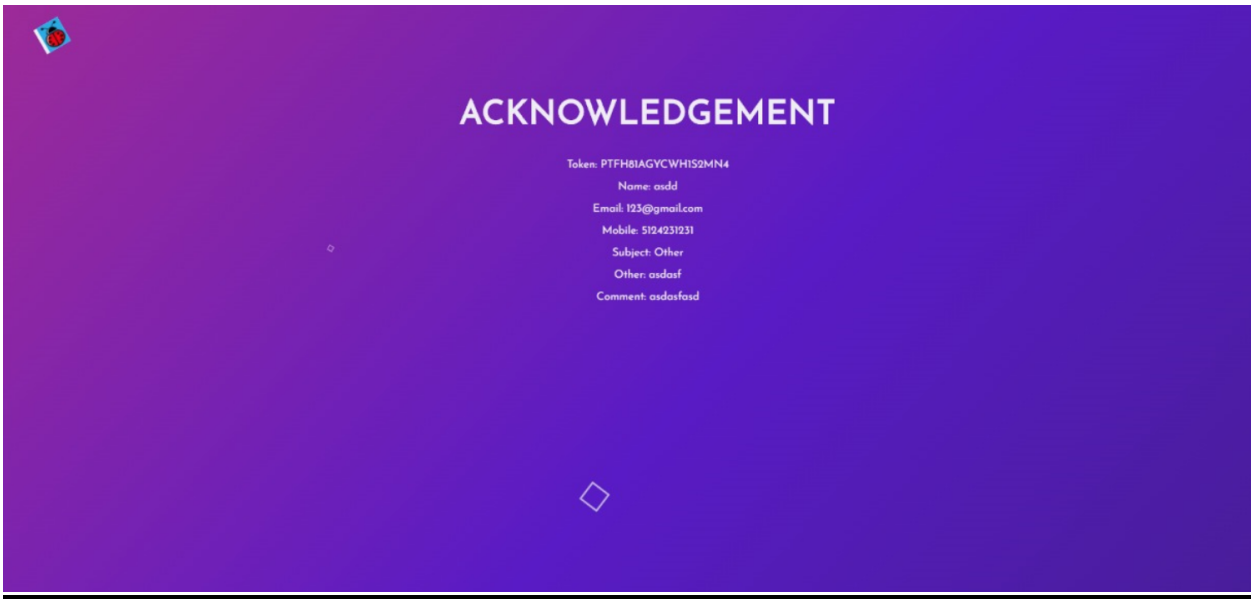
```
String name=request.getParameter("Name");
String email=request.getParameter("EMail");
String mobile=request.getParameter("Mobile");
String subject=request.getParameter("Subject");
String other="";
String comment=request.getParameter("Comment");
LocalDate myObj = LocalDate.now();
String SALTCHARS = "ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890";
StringBuilder salt = new StringBuilder();
Random rnd = new Random();
while (salt.length() < 18) { // length of the random string.
   int index = (int) (rnd.nextFloat() * SALTCHARS.length());
   salt.append(SALTCHARS.charAt(index));}
String token = salt.toString();
if(subject.equals("Other")){
   other=request.getParameter("car");
   out.println("Other: "+other+ "<br>"+ "<br>");
}
```

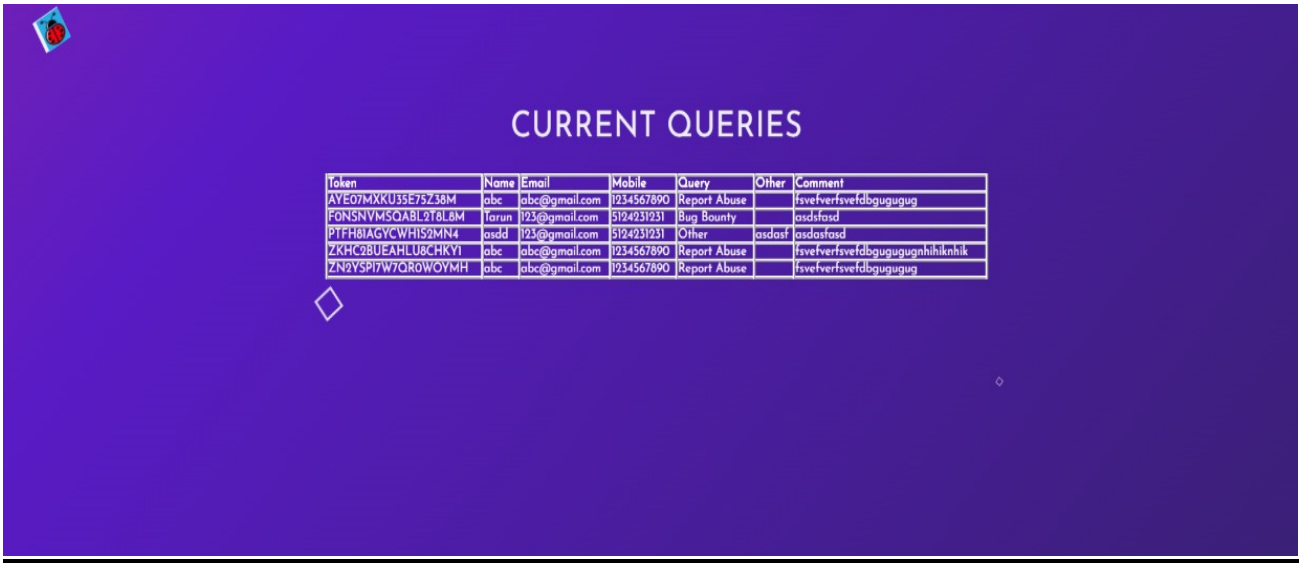
Contact JSP

```
<link href="https://fonts.googleapis.com/css?family=Josefin+Sans" rel="stylesheet">
<div class="container-fluid">
  <div class="background">
    <div class="cube"></div>
    <div class="cube"></div>
    <div class="cube"></div>
    <div class="cube"></div>
    <div class="cube"></div>
  <header>
    <div class="logo"><span><img src="logo.png" width="35px" height="40px"></span></div>
    <section class="header-content">
        <button style="margin:auto;display:block;" onclick="window.location.href='home.jsp'">Home</button
      <div id="one" onmouseover="mouseOver()" onmouseout="mouseOut()">
        <center>
          <h1 align=center>Contact Us</h1>
        </center>
        <br>
        <form action="ContactServlet" name="ContactForm" onsubmit="return verify()" method="post">
          <p>Name: <input type="text" size=65 name="Name"> </p><br>
          <p>E-mail: <input type="text" size=65 name="EMail"> </p><br>
          <p>Mobile: <input type="text" size=65 name="Mobile"> </p><br>
          <p>SELECT QUERY
            <select type="text" value="" name="Subject" onchange="yesnoCheck(this);">
              <option default>select</option>
              <option>Bug Bounty</option>
              <option>Report Abuse</option>
              <option>Careers</option>
              <option>Advertising</option>
```

```
                <option>Sponsorship</option>
                <option value="Other">Other</option>
              </select></p>
            <div id="ifYes" style="display: none;">
              <label for="car">If selected other, please mention</label> <input type="text" id="car" name="car" /><br />
            </div>
            <br>
            <p>Comments: <textarea cols="55" rows="5" name="Comment" id="Comment"></textarea></p>
            <p><input type="submit" value="send" name="Submit" class="button">
              <input type="reset" value="Reset" name="Reset" class="button">
            </p>
          </form>
          <br>
          <h2> To Meet in Person </h2>
          <p> click left side of image for coimbatore address</p>
          <p> click right side of image for chennai address</p>
          <br>
          <img src="https://miro.medium.com/max/4064/1*qYUvh-EtES8dtgKiBRiLsA.png" alt="map" width="400" height="200" usemap="#google">
          <map name="google">
            <area shape="rect" coords="201,0,399,199" href="https://www.google.com/maps/place/Chennai,+Tamil+Nadu/@13.0474878,80.0689276,11z/data=!3m1!4b1!4m5!3m4!1s0x3a5265ea4f7d3361:0x6e61a70b6863d433!8m2!3d13.0826802!4d80.2707184?hl=en" alt="chennai">
          </map>
          <br><br>
        </div>
```



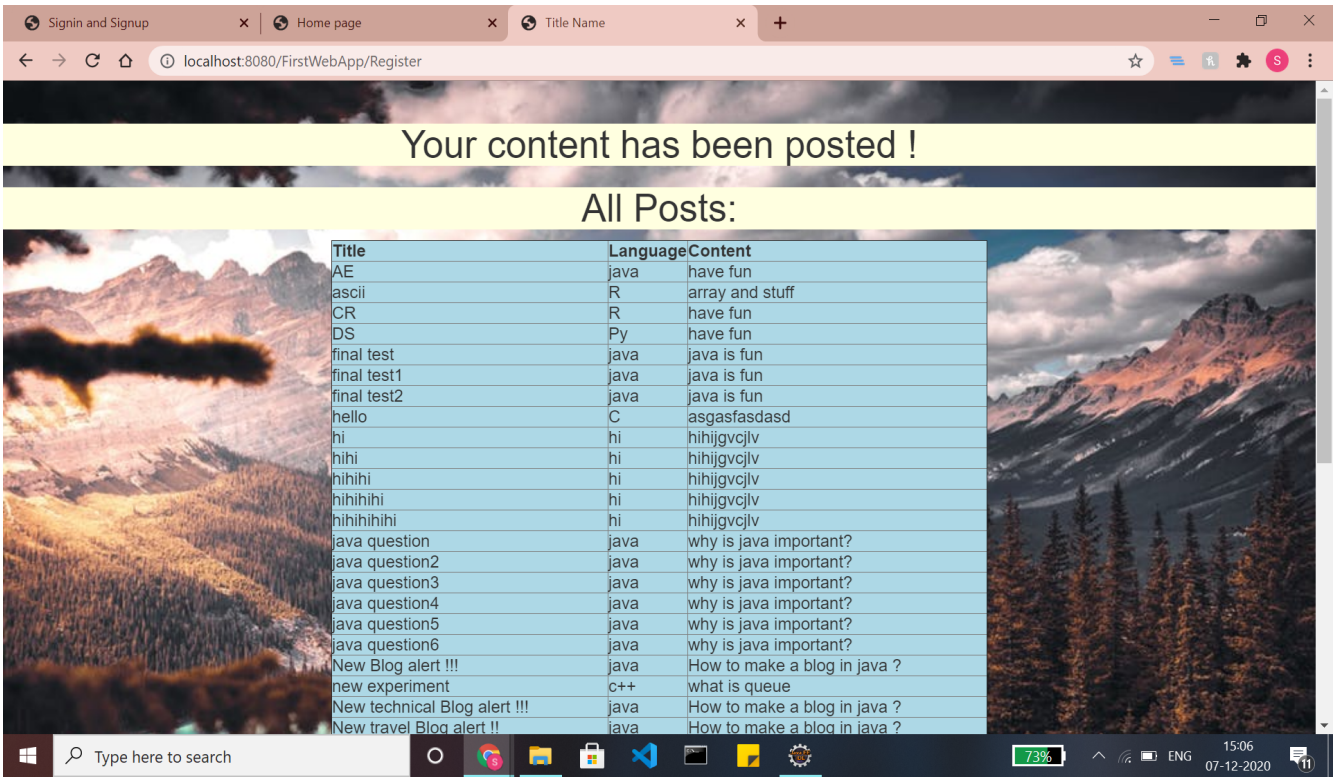




Create Post Servlet

```
import java.io.*;
import java.sql.*;
import javax.servlet.ServletException;
import javax.servlet.http.*;

public class Register extends HttpServlet {
/**
		 *
		 */
		private static final long serialVersionUID = 1L;
```

```
public void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {

response.setContentType("text/html");
PrintWriter out = response.getWriter();
String n=request.getParameter("title");
String p=request.getParameter("lang");
String e=request.getParameter("content");
String drive = "com.mysql.jdbc.Driver";
Connection conn=null;
String                                                    url="jdbc:mysql://bugoverflow.clhuzfls9udn.us-east-
1.rds.amazonaws.com:3306/bugoverflow?characterEncoding=latin1";
String user="root";
String pass="bugoverflow";
          try {
            Class.forName(drive);
                    System.out.println("Connecting to DB....");
                    conn=DriverManager.getConnection(url,user,pass);
                    System.out.println("Connection successful.... \n");

PreparedStatement ps=conn.prepareStatement(
"insert into post values(?,?,?)");
ps.setString(1,n);
ps.setString(2,p);
ps.setString(3,e);
//ps.setString(4,c);
ps.executeUpdate("use bugoverflow;");
int i=ps.executeUpdate();
if(i>0)
//out.print("You have posted successfully...");
response.setContentType("text/html");
PrintWriter out1 = response.getWriter();
//out1.println("Included HTML block:");
request.getRequestDispatcher("show").include(request, response);
out1.close();
}catch (Exception e2) {System.out.println(e2);}
out.close();
} }
```



Create Post JSP

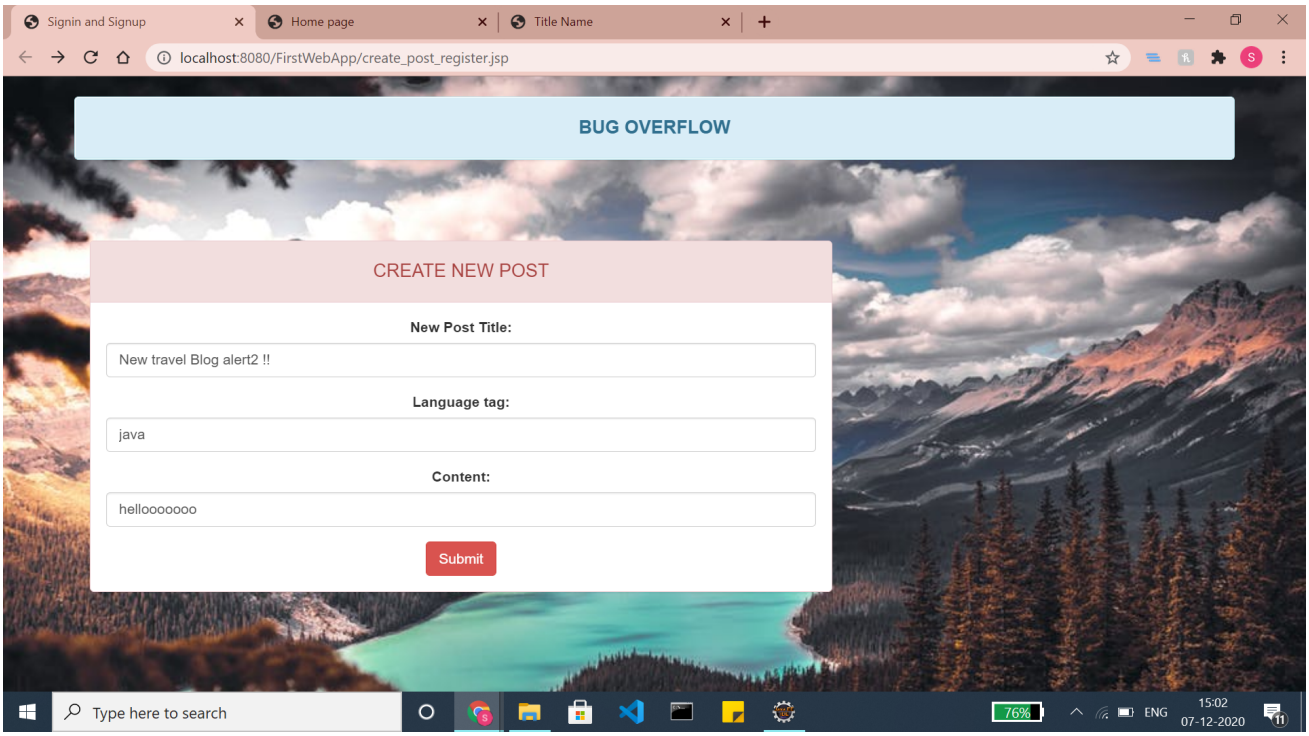
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Signin and Signup</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <link rel="stylesheet" href="style.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>
<body class="bod">

        <script>
                function validateForm() {
                  var x = document.forms["create_post"]["title"].value;
```

```
  var y = document.forms["create_post"]["lang"].value;
  var z = document.forms["create_post"]["content"].value;

if (x == "" ||y==""|| z=="") {
        alert("This field must be filled out");
}
if (! isNaN(x) ||! isNaN(y) || ! isNaN(z)) {
                alert("This field cannot be a number");

}
        return false;

}
</script>
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
id="WebApp_ID" version="3.1">

 <servlet>
 <servlet-name>abc</servlet-name>
 <servlet-class>add_servlet</servlet-class>

 </servlet>
 <servlet-mapping>
 <servlet-name>abc</servlet-name>
 <url-pattern>/add</url-pattern>
 </servlet-mapping>

 <servlet>
 <servlet-name>abc2</servlet-name>
 <servlet-class>Register</servlet-class>

 </servlet>
 <servlet-mapping>
 <servlet-name>abc2</servlet-name>
 <url-pattern>/Register</url-pattern>

 </servlet-mapping>
 <servlet>
```

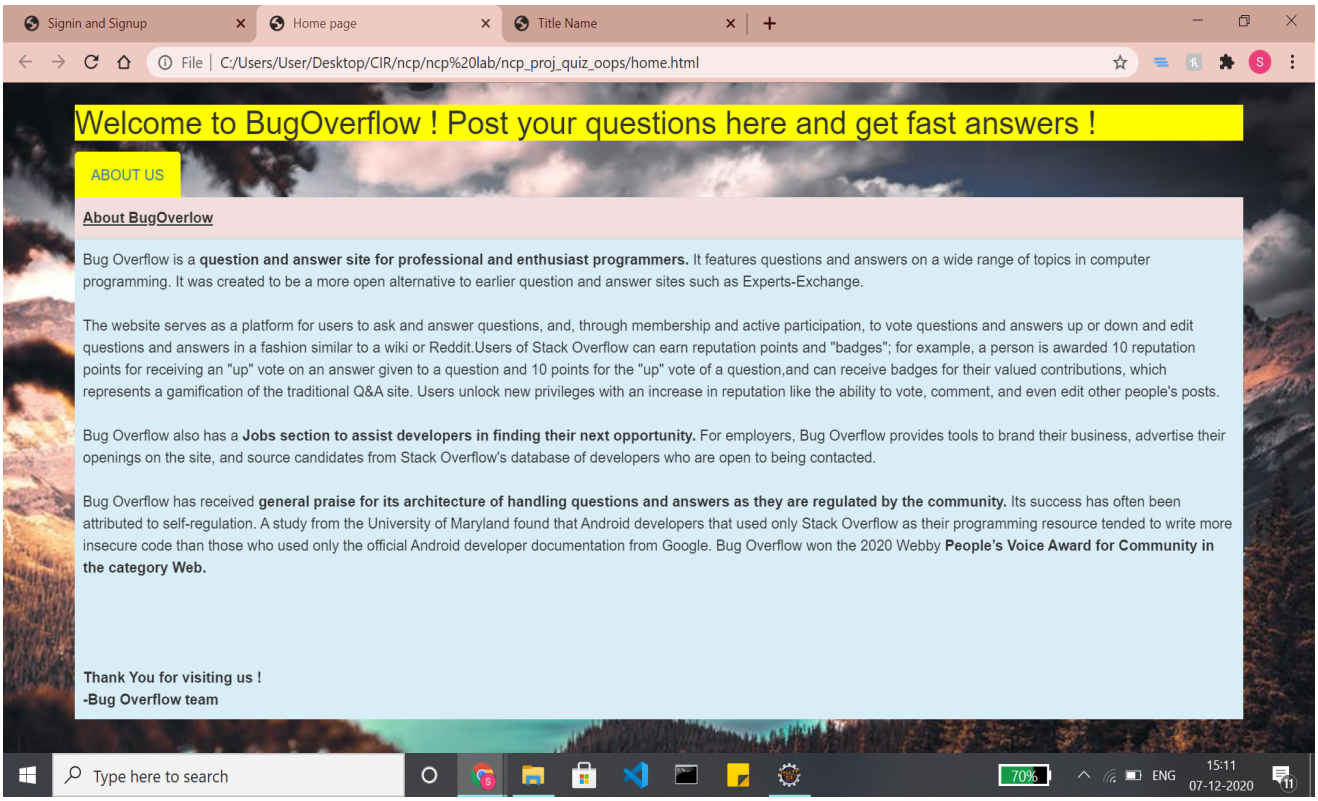
```
  <servlet-name>abc3</servlet-name>
  <servlet-class>show</servlet-class>
  </servlet>
  <servlet-mapping>
  <servlet-name>abc3</servlet-name>
  <url-pattern>/show</url-pattern>
  </servlet-mapping>
</web-app>
```



About us page

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Home page</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
  <link rel="stylesheet" href="style.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>
<body class="bod">
```

```
<div class="<div class="container">">



 <h2 class="hi">Welcome to BugOverflow <?php echo $name;?>! Post your questions here and get fast answers !</h2>
 <ul class="nav nav-tabs">

  <li><a data-toggle="tab" href="#menu1" class="hi">ABOUT US</a></li>

 </ul>

            <table class="table" >
  <thead>
   <tr class="danger">
    <th><u>About BugOverlow</u></th>

   </tr>
  </thead>
  <tbody>
                  <script>
                        function mOver(obj) {
                         obj.className = "info"
                        }

                        function mOut(obj) {
                         obj.className = "danger"
                        }
                        </script>


        <tr class="danger" onmouseover="mOut(this)" onmouseout="mOver(this)">
         <td>
         <section>
         Bug Overflow is a <b>question and answer site for professional and enthusiast programmers.</b> It features questions and answers on a wide range of topics in computer programming.
         It was created to be a more open alternative to earlier question and answer
         sites such as Experts-Exchange.
         <br>
         <br>
         </section>
```



Registration Servlet

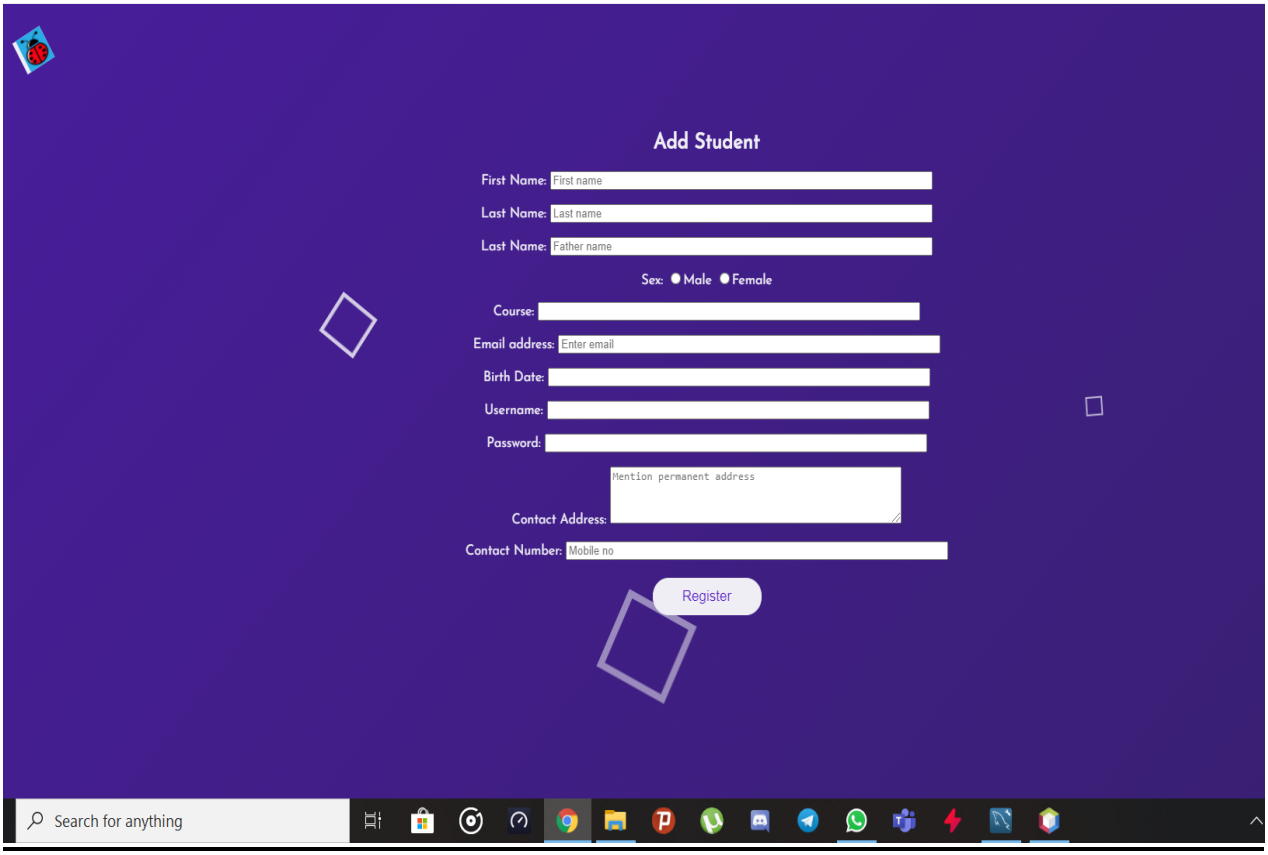
```
public class newstudent extends HttpServlet
{
public void doPost(HttpServletRequest req, HttpServletResponse res)
{
      try
                        {
            PrintWriter out = res.getWriter();
            String firstname=req.getParameter("fname");
            String un=req.getParameter("uname");
            String pas=req.getParameter("pass");
            String ob=req.getParameter("obj");
                        String lastname=req.getParameter("lname");
                        String fathername=req.getParameter("fathername");
                        String se=req.getParameter("sex");
                        String add=req.getParameter("Address");
                        String em=req.getParameter("emailid");
                        String db=req.getParameter("dob");
                        Long  phoneno=Long.parseLong(req.getParameter("mobileno"));
            //out.print("insert into register values
('"+fn+"','"+ln+"','"+reg+"','"+usern+"','"+passw+"','"+passw+"','"+dob+"','"+phno+"','"+lia+"')");
            Class.forName("com.mysql.jdbc.Driver");
          Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/ncp_casestudy","root","admin");
          Statement stmt=con.createStatement();
          String sql1 = "insert into newstudent values
('"+firstname+"','"+un+"','"+pas+"','"+ob+"','"+lastname+"','"+fathername+"','"+se+"','"+add+"','"+em+"','"+db+"','"+phone
no+"')";
```

```
        stmt.executeUpdate(sql1);
        String sql2= "insert into login values ('"+un+"','"+pas+"')";
       System.out.println("inserted");
        res.sendRedirect("intoadmin.jsp");
        con.close();
     }catch(Exception e){System.out.println(e);}}}
```

Register JSP

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
 <%@ page import="javax.servlet.http.*,javax.servlet.*,java.sql.*" %>
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Registration</title>
    <meta charset="utf-8">
    <link rel="stylesheet" href="final.css">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <!-- jQuery library -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <!-- Popper JS -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
    <!-- Latest compiled JavaScript -->
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>
  </head>
  <body class="fluid-container bg-secondary">
    <div class="container bg-dark">
      <h2>Add Student</h2>
      <form name="register" action="newstd" method="POST" >
        <div class="form-group">
          <label for="fname">First Name:</label>
       <input type="text" class="form-control" id="fname" name="fname" pattern="^[A-Z][a-z]+" placeholder="First name" title="Enter First as capital letter followed by small letters(exclude numbers and specialcharacters except space)" required>
        </div>
        <div class="form-group">
          <label for="lname">Middle Name:</label>
          <input type="text" class="form-control" id="lname" name="lname" pattern="^[A-Z][a-z]+" placeholder="Last/Middle name" title="Enter First as capital letter followed by small letters(exclude numbers and specialcharacters except space)" required>
        </div>
        <div class="form-group">
          <label for="fathername">Last Name:</label>
          <input type="text" class="form-control" id="fathername" name="fathername" pattern="^[A-Z][a-z]+" placeholder="Father/Surname name" title="Enter First as capital letter followed by small letters(exclude numbers and specialcharacters except space)" required>
```

```
        </div>
        <div class="form-group">
          <label for="sex">Sex:</label>
          <input type="radio" name="sex" value="male" size="10" >Male
          <input type="radio" name="sex" value="Female" size="10" >Female
        </div>
         <div class="form-group">
            <label for="Objective">Course:</label>
            <input type="text" name="obj" class="form-control"  required>
        </div>
 <div class="form-group">
            <label for="email">Email address:</label>
            <input type="text" class="form-control" id="email" name="emailid" pattern="^(([-\w\d]+)(\.[-
\w\d]+)*@([-\w\d]+)(\.[-\w\d]+)*(\.([a-zA-Z]{2,5}|[\d]{1,3})){1,2})$" aria-describedby="emailHelp"
placeholder="Enter email" title="Your email address" required>
            <small id="emailHelp" class="form-text">We'll never share your email with anyone else.</small>
        </div>
        <div class="form-group">
            <label for="DOB">Birth Date:</label>
            <input type="text" class="form-control" id="DOB" name="dob" required>
        </div>
        <div class="form-group">
            <label for="Username">Username:</label>
            <input type="text" class="form-control" id="Username" name="uname" required>
        </div>
        <div class="form-group">
            <label for="password">Password:</label>
            <input type="password" class="form-control" id="password" name="pass" required>
        </div>
        <div class="form-group">
            <label for="address">Contact Address:</label>
            <textarea class="form-control" id="address" name="Address" rows="5" placeholder="Mention
permanent address" title="Enter your permanent address" required></textarea>
        </div>
        <div class="form-group">
            <label for="mobileno">Contact Number:</label>
            <input type="text" class="form-control" id="mobileno" name="mobileno" placeholder="Mobile no"
title="Enter your mobile number" required>
        </div>
        <button type="submit" class="btn btn-outline-light btn-md">Register</button>
      </form>
    </div>
  </body>
</html>
```


Add Student
First Name: First name
Last Name: Last name
Last Name: Father name
Sex: Male Female
Course:
Email address: Enter email
Birth Date:
Username:
Password:
Contact Address: Mention permanent address
Contact Number: Mobile no
Register
Search for anything

TESTING

Field	Input Given	Success/Failure	Reason
Login page	Username,Password	Success	Data verified successfully
Sign up page	Username,Password, email,phone	Success	Data inserted successfully
Profile page	Password, email,phone	Failure	Password mismatch
Create Post page	Title ,language,content	Failure	Title already exists
Contact us page	Name, mobile, email, query, comment	Passed	Data Inserted Successfully

Evaluation Sheet

Registration Number	Technology/Max Marks	Marks Awarded
CB.EN.U4CSE17008	Servlet (15) JSP(15) Integration(10) Report (10) Total(50)	
CB.EN.U4CSE17009	Servlet (15) JSP(15) Integration(10) Report (10) Total(50)	
CB.EN.U4CSE17044	Servlet (15) JSP(15) Integration(10) Report (10) Total(50)	
CB.EN.U4CSE17051	Servlet (15) JSP(15) Integration(10) Report (10) Total(50)	
CB.EN.U4CSE17052	Servlet (15) JSP(15) Integration(10) Report (10) Total(50)	