Task: 1 Date:	Develop a simple web site including all the information using HTML 5 and CSS 3

Aim:

To develop a simple web site including all the information using HTML5 and CSS3

Procedure:

1. Install MS Visual studio code
2. Create HTML File
3. Create CSS File
4. Link CSS File in HTML File using link tag
5. Write necessary code

Program:

index.html

```
<!DOCTYPE html> <!-- The new doctype -->
<html>
   <head>
         <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
         <title>Task 1</title>
         <link rel="stylesheet" type="text/css" href="styles.css" />
      </head>
   <body>
         <section id="page">
         <header>
         <hgroup>
               <img src="logo.png" width=30%>
               <h3>and a fancy slogan</h3>
            </hgroup>
         <nav class="clear"> <!-- The nav link semantically marks your main site navigation --
>
               <ul>
                  <li><a href="#article1">About Us</a></li>
                  <li><a href="#article2">CSE</a></li>
                  <li><a href="#article3">Placement</a></li>
               </ul>
            </nav>
         </header>

         <section id="articles">
         <div class="line"></div>
            <article id="article1">
               <h2>Photoshoot Effect</h2>
```

```
        <div class="line"></div>
                <div class="articleBody clear">
                   <figure>
                            <a href=""><img src="download.jpg" width="620" height="340"
/></a>
                   </figure>
                   <p>Vel Tech is well-known for its renowned educational practices, which has
been recognized and endowed with several awards.
                   </p>
                </div>
             </article>
<div class="line"></div>

             <article id="article2">
                <h2>Sweet AJAX Tabs</h2>
        <div class="line"></div>
                <div class="articleBody clear">
                   <figure>
                            <img src="download1.jpg" width="620" height="340" /></a>
                   </figure>
                   <p><b>We are Passionate. Doers in Innovative Engineering
Education</b></p>
                   <p> About Vel Tech University
                  </p>
                </div>
             </article>
                     </section>
       <footer> <!-- Marking the footer section -->
        <div class="line"></div>
<p>Copyright 2023 - veltech.edu.in</p>
<a href="#" class="up">Go UP</a>
</footer>
          </section>
       </body>
</html>
```

styles.css

```
*{
	/* Universal selector: */
	margin:0;
	padding:0;
}

header,footer,
article,section,
hgroup,nav,
figure{
	/* Giving a display value to the HTML5 rendered elements: */
	display:block;
}

body{
	/* Setting the default text color, size, page background and a font stack: */
	font-size:0.825em;
	color:#fcfcfc;
	background-color:#355664;
	font-family:Arial, Helvetica, sans-serif;
}

/* Hyperlink Styles: */

a, a:visited {
	color:#0196e3;
	text-decoration:none;
	outline:none;
}

a:hover{
	text-decoration:underline;
}

a img{
	border:none;
}

/* Headings: */

h1,h2 {
	font-family:"Myriad Pro","Helvetica Neue",Helvetica,Arial,Sans-Serif;
	text-shadow:0 1px 1px black;
}

h1{
	/* The logo text */
	font-size:3.5em;
	padding:0.5em 0 0;
	text-transform:uppercase;
}
```

```
	h2{
		font-size:2.2em;
		font-weight:normal;
		letter-spacing:0.01em;
		text-transform:uppercase;
	}

	p{
		line-height:1.5em;
		padding-bottom:1em;
	}

	.line{
		/* The dividing line: */
		height:1px;
		background-color:#24404c;
		border-bottom:1px solid #416371;
		margin:1em 0;
		overflow:hidden;
	}

	/* Article styles: */

	#page{
		width:960px;
		margin:0 auto;
		position:relative;
	}

	article{
		background-color:#213E4A;
		margin:3em 0;
		padding:20px;

		text-shadow:0 2px 0 black;
	}

	figure{
		border:3px solid #142830;
		float:right;
		height:300px;
		margin-left:15px;
		overflow:hidden;
		width:500px;
	}

	figure:hover{
		-moz-box-shadow:0 0 2px #4D7788;
		-webkit-box-shadow:0 0 2px #4D7788;
		box-shadow:0 0 2px #4D7788;
	}

	figure img{
		margin-left:-60px;
}
```

```
footer{



}

footer p{
        marg
        in-
        botto
        m:-
        2.5e
        m;
        posit
        ion:r
        elati
        ve;
}
```

Output:

Result:

Thus the above program was executed successfully and the output was verified.

Task: 2 Date:	Create home page, sign up and login page for clinic management service using Bootstrap Framework

Aim:

To create home page, sign up and login page for clinic management service using Bootstrap Framework.

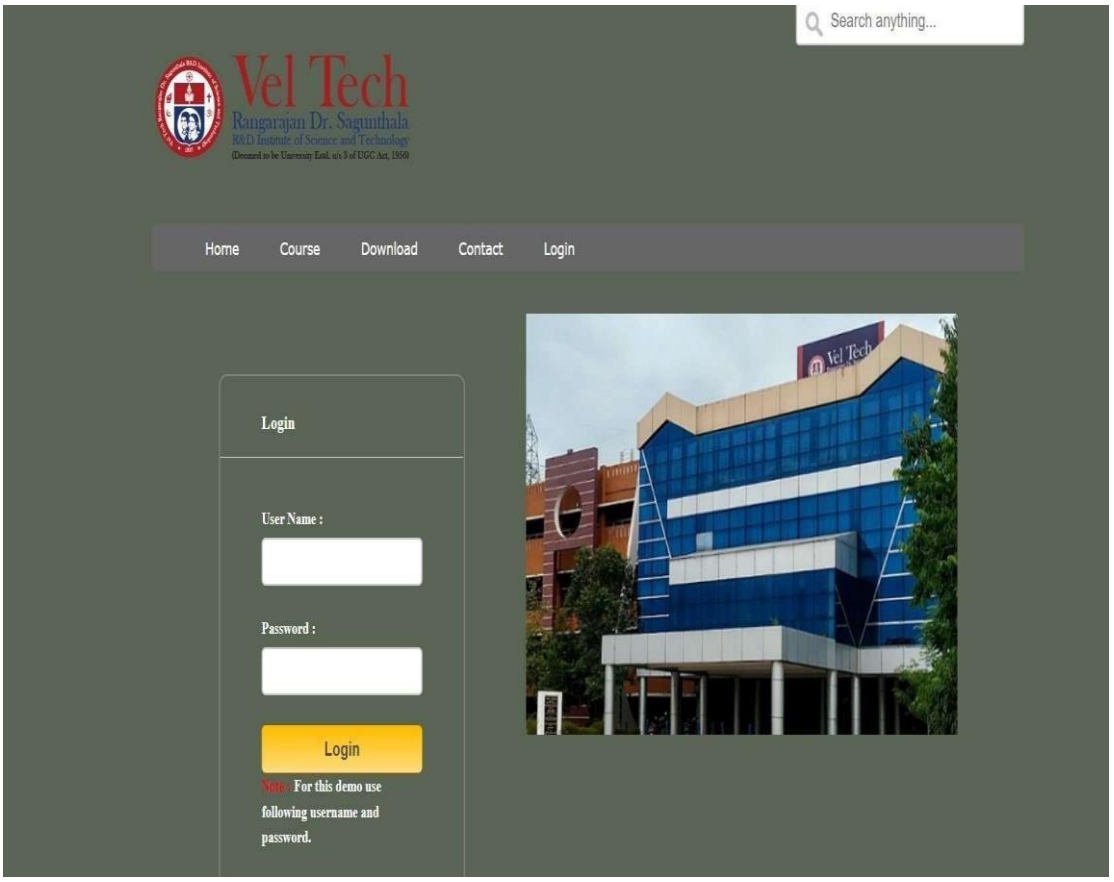
Procedure:

1. Install MS Visual studio code
2. Create HTML File
3. Create CSS File
4. Link CSS File in HTML File using link tag
5. Write necessary code and include bootstrap framework

Program:

```
<div class="container">
<div class="main">
<h3>Login </h3><hr/>
<form id="form_id" method="post" name="myform">
<strong>User Name :</strong></br>
<input type="text" name="username" id="username"/></br>
<strong>Password :</strong></br>
<input type="password" name="password" id="password"/></br>
<input type="button" value="Login" id="submit" onclick="validate()"/>
</form>
 <span><b class="note">Note : </b><strong>For this demo use following username
and password. <br/><b class="valid"><h4>User Name : Vel Tech<br/>Password :
CSE</b></h4></span></strong>
</div>
<div class="fugo">
<a href="images/Image.png"><img src="images/Image.png" /></a>
</div>
</div>
```

Output:



Result:

Thus the above program was executed successfully and the output was verified.

Task: 3 Date:	Validate the Registration, user login, user profile and payment by credit card pages using JavaScript.

Aim:

To validate the Registration, user login, user profile and payment by credit card pages using JavaScript.

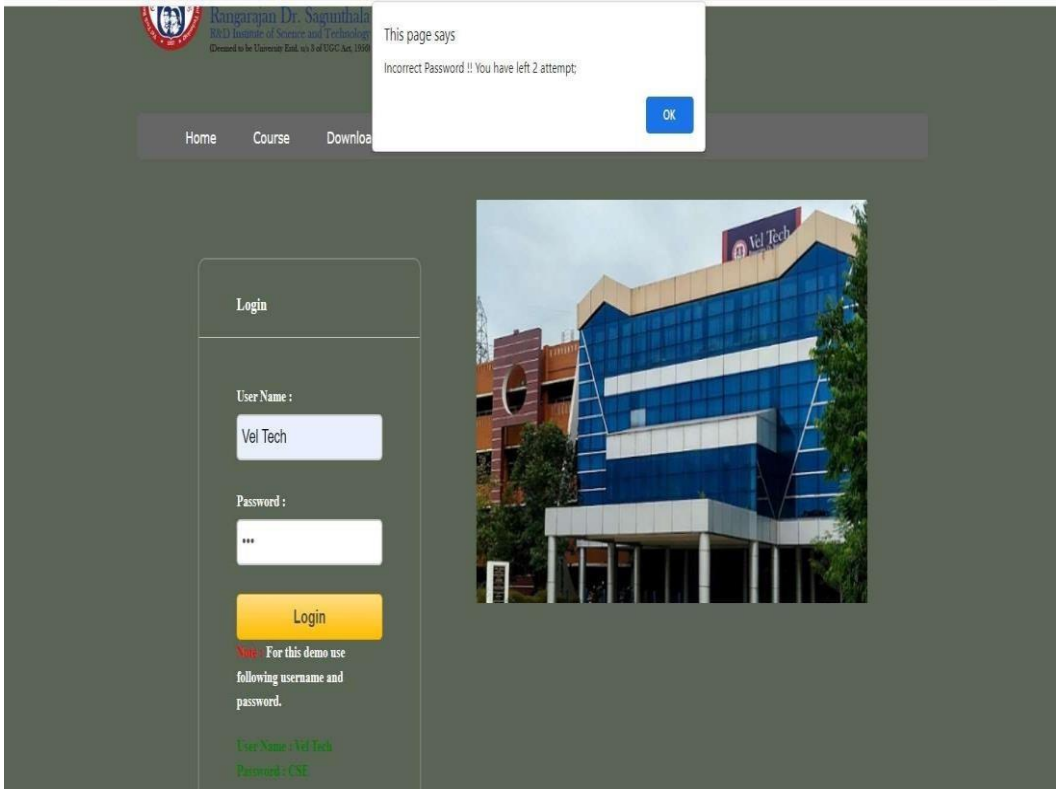
Procedure:

1. Install MS Visual studio code
2. Create HTML File
3. Create CSS File
4. Link CSS File in HTML File using link tag
5. Write necessary code and include bootstrap framework
6. Create Javascript file for validation of login page

Program:

```
var attempt = 3; //Variable to count number of attempts
//Below function Executes on click of login button
function validate(){
	var username = document.getElementById("username").value;
	var password = document.getElementById("password").value;
	if ( username == "Vel Tech" && password == "CSE"){
		alert ("Login successfully");
		window.location = "success.html"; //redirecting to other page
		return false;
	}
	else{
		attempt --;//Decrementing by one
		alert("Incorrect Password !! You have left "+attempt+" attempt;");
		//Disabling fields after 3 attemptsif( attempt== 0){
			document.getElementById("username").disabled = true;
			document.getElementById("password").disabled = true;
			document.getElementById("submit").disabled = true;
			return false;
		}
	}
}
```

Output:



Result:

Thus the above program was executed successfully and the output was verified.

Task: 4 Date:	Parse the web page to get the required information using JQuery and DOM Traversing.

Aim:

To parse the web page to get the required information using JQuery and DOM Traversing.

Procedure:

1. Install MS Visual studio code
2. Create HTML File
3. Add jQuery link
4. Write necessary code to traverse the element
5. Save and run the program.

Program:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery.parseHTML demo</title>
<script src="https://code.jquery.com/jquery-3.5.0.js"></script>
</head>
<body>
<div id="log">
<h3>Content:</h3>
</div>
<script>
var $log = $( "#log" ),
str = "hello, <b>my name is</b> jQuery.", html = $.parseHTML( str ),
nodeNames = [];
// Append the parsed HTML
$log.append( html );
// Gather the parsed HTML's node names
$.each( html, function( i, el ) {
nodeNames[ i ] = "<li>" + el.nodeName + "</li>";
});
// Insert the node names
$log.append( "<h3>Node Names:</h3>" );
$( "<ol></ol>" )
 .append( nodeNames.join( "" ) )
.appendTo( $log );
</script>

</body>
</html>
```

Output:

Content:

hello, **my name is** jQuery.

Node Names:

1. #text
2. B
3. #text

Result:

Thus the above program was executed successfully and the output was verified.

Task: 5 Date:	**Implement a server-side logic using PHP to create three-tier applications for conducting online examination for displaying student mark list. Assume that student information is available in a database which has been stored in a database server.**

Aim:

To implement a server-side logic using PHP to create three-tier applications for conducting online examination for displaying student mark list.

Procedure:

- Environment Setup
  1. Install XAMPP Web Server
  2. Open the XAMPP Control Panel.
  3. Start the Apache server by clicking on the Start button.
  4. Start the MySQL by clicking on the Start button.
  5. Create all the files needed for login.
  6. Create login table in the database using phpMyAdmin in XAMPP.
- Creation of Necessary Files
  1. index.html - This file is created for the GUI view of the login page and empty field validation.
  2. style.css - This file is created for the attractive view of the login form.
  3. connection.php - Connection file contains the connection code for database connectivity.
  4. authentication.php - This file validates the form data with the database which is submitted bythe user

Program:

Connect to database:

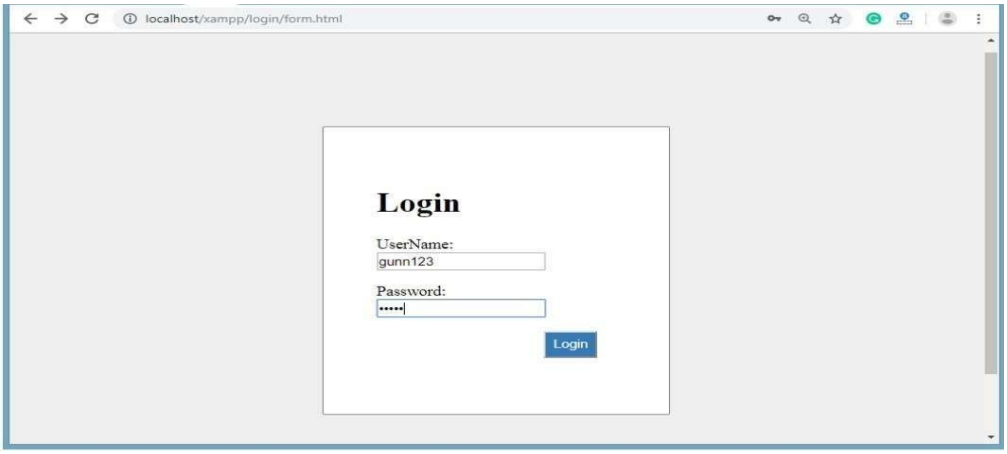
```
<?php
    $host = "localhost";
    $user = "root";
    $password = '';
    $db_name = "CSE";
    $con = mysqli_connect($host, $user, $password, $db_name);
    if(mysqli_connect_errno()) {
        die("Failed to connect with MySQL: ". mysqli_connect_error());
    }
?>
```

Authenticating the Database

```
<?php
    include('connection.php');
    $username = $_POST['user'];
    $password = $_POST['pass'];
        $username = stripcslashes($username);
        $password = stripcslashes($password);
```

```
$username = mysqli_real_escape_string($con, $username);
$password = mysqli_real_escape_string($con, $password);
$sql = "select *from login where username = '$username' and password =
'$password'";
$result = mysqli_query($con, $sql);
$row = mysqli_fetch_array($result, MYSQLI_ASSOC);
$count =
mysqli_num_rows($re
sult); if($count == 1){
   echo "<h1><center> Login successful </center></h1>";
}
else{
   echo "<h1> Login failed. Invalid username or password.</h1>";
}
?>
```

Output:



Result:

Thus the above program was executed successfully and the output was verified.

Task:6 Date:	Create a simple HTTP web server using Node.js to generate a dynamic response

Aim: To create a simple HTTP web server using Node.js to generate a dynamic response

Procedure:

1. Install Node Js
2. Create the server using http.createServer(function (request, response) {});
3. Send the response
4. Server will be listening at port 3000
5. Run index.js file using below command:
    a. nodeindex.js

Program:

```
moduleconst http=
require("http")
constserver=http.createServer((req,res) =>{
res.write("This is the response from the server")
   res.end();
})
server.listen((3000), ()=> {
   console.log("ServerisRunning");
})
```

Output:

```
$ node index.js
Server is Running
```

Result: Thus the above program was executed and output was verified successfully.

Task:7 Date:	Create a three-tier application using Node.js and MySQL data bas

Aim: To create a three-tier application using Node.js and MySQL data base

Procedure:

1. Install Node.js
2. Include Npm registry
3. Use any TextEditor like VSCode or notepad.
4. Postman:this will allow to test your API(GET, POST,PUT,DELETE,etc.)

Program:

```
$mkdir project && mkdirproject/server && mkdirproject/client && cdserver/
$touch package.json
```

- Scripts to start application

```
{
  "name":"server",
  "version":"1.0.0","pr
  ivate":
  true,"scripts":{
  "start": "node -r esm
  app.js","dev":"nodemon-
  resmapp.js"
 },
}
```

- Installation:

```
$npx install-peerdeps—deveslint-config-airbnb
```

- BuildServer:

```
//Import all dependencies & middleware here
import express from 'express';

//Init an Express App.This later starts a server and put all dependencies into your project to use
constapp =express();
//Use your dependencies here

//use all controllers(APIs)here
app.get('/',(req,res) => {
  res.status(200).json({sta
    tus:'success'
  });
});

// Start Anything here
```

```
app.listen(8080,()=>{
  console.log('Example app listening on port 8080!');
});
```

- Startserver:

```
$npmstart
```

- CreateRESTfulAPIs

```
Import express from'express';
Const user
Controller=express.Router();userControlle
r.get('/', (req,res)=> {
  res.status(200).json({status:'
    success'
  });
});
Export tdefault userController;
```

- Install & Start

```
MongoDBapp.listen(8080,(
)=> {
  console.log(`Started successfully server at port
  ${port}`);mongoose.connect('mongodb://localhost/test').then
  (()=>{
    console.log(`Conneted to mongoDB at port27017`);
  });
});
```

Output:

UsePOST/methodand enterlocalhost:8080/add-user.Thiswillcallthe“/add-user”API.

```
{
 'email':’example@gmail.com’,'passw
 ord':'123456789'
}
```

Result: Thus the above program was executed and output was verified successfully.

Task:8 **Date:**	**Create a simple single web page chat bot's application using Angular for Bike Rental System.**

Aim: To create a simple single web page chat bot's application using Angular for Bike Rental System

Procedure:

1. Install angular
2. This application has only one extra dependency – the Dialog Flow JavaScript SDK. It is written in TypeScript, so we can install it to the dev dependencies.
3. We need to add the Angular Forms Module to the imports and add the Chat Dialog Component to exports.
4. Then import the chat module into the app module
5. Now we have to call this app Component where ever we want the functionality to been included. Currently in our application, we used it in index.html

Program:

```
import{NgModule}from'@angular/core';
import{CommonModule}from'@angular/common';impo
rt { FormsModule } from
'@angular/forms';import{ChatService}from
'../chat.service';
import{ChatDialogComponent}from'./chat-dialog/chat-dialog.component';
<b>@NgModule</b>({im
ports: [Common Module,
Forms Module
],
declarations: [Chat
Dialog Component
],
exports:[Chat Dialog Component],//&lt;--export here
providers:[Chat Service]
})
Export class ChatModule{}
```

Output:

Result: Thus the above program was executed and output was verified successfully.

Task:9 **Date:**	**Develop a micro service for finding what people think by asking 500 people's opinion for any consumer product in Node.js using Seneca Toolkit.**

Aim: To develop a micro service for finding what people think by asking 500 people's opinion for any consumer product in Node.js using Seneca Toolkit.

Procedure:

1. `seneca.add` method adds a new action pattern to the Seneca instance
2. Pattern property is used to match in any JSON messages that the Seneca instance receives.
3. Action function is used to execute when a pattern matches a message.
4. To initialize a plugin, you add a special action pattern: init:<plugin-name>

Program:

```
this.add({
        role:
        "movement",cmd:"ra
        wMoves",
    }, (msg, reply) =>{var
        err=null;
        var rawMoves=[];
        varpos=msg.piece.position;

        switch (msg.piece.piece)
        {case'R':
            rawMoves =
            rankAndFile(pos);break;
        case'B':
            rawMoves =
            diagonal(pos);break;
        case'Q':
            rawMoves=rankAndFile(pos)
                .concat(diagonal(pos));br
            eak;
        case'K':
            rawMoves=rankAndFile(pos,1)
                .concat(diagonal(pos,
            1))break;
        default:
            err = "unhandled " +
            msg.piece;break;
        };
```

```
        reply(err,rawMoves);
    });
```

Output:

```
[{file:‘c’,rank:‘4’},
 {file:‘d’,rank:‘5’},
 {file:‘e’,rank:‘4’},
 {file:‘d’,rank:‘3’},
 {file:‘b’,rank:‘4’},
 {file:‘d’,rank:‘6’},
```

Result: Thus the above program was executed and output was verified successfully.

Task:10 **Date:**	**Develop a simple micro service for E-Payment service in Node.js using Seneca toolkit.**

Aim: To develop a simple micro service for E-Payment service in Node.js using Seneca toolkit.

Procedure:

1. `.add` method adds a new action pattern to the Seneca instance
2. Pattern property is used to match in any JSON messages that the Seneca instance receives.
3. Action function is used to execute when a pattern matches a message.
4. To initialize a plugin, you add a special action pattern: init:<plugin-name>

Program:

```
Var seneca=require('seneca')();
seneca
  .use('basic')
  .use('entity');

seneca.add({"role": "product", "cmd": "create"}, (args, done) =>
  {var product =seneca.make$("Product");
  product.name
  =args.name;product.description =
  args.description;product.price
  =args.price;product.save$((err,
  savedProduct) =>
  {done(err,savedProduct);
  });
});

// Listen for messages in the specified transport type and
port.seneca.listen({
    "type":"http","p
    ort":8080
});
```

Output:

Result: Thus the above program was executed and output was verified successfully.

Usecase:

HOSPITAL REGISTRATION FORM

Aim:

To implement hospital Management System using PHP and MySQL.

Procedure:

- Environment Setup
  1. Install XAMPP Web Server
  2. Open the XAMPP Control Panel.
  3. Start the Apache server by clicking on the Start button.
  4. Start the MySQL by clicking on the Start button.
  5. Create all the files needed for login.
  6. Create login table in the database using phpMyAdmin in XAMPP.
- Creation of Necessary Files
  - ✓ index.html - This file is created for the GUI view of the login page.
  - ✓ Welcome.php - this file contains the connection code for database connectivity and inserts form data into the database after submission.

Program:

home.html

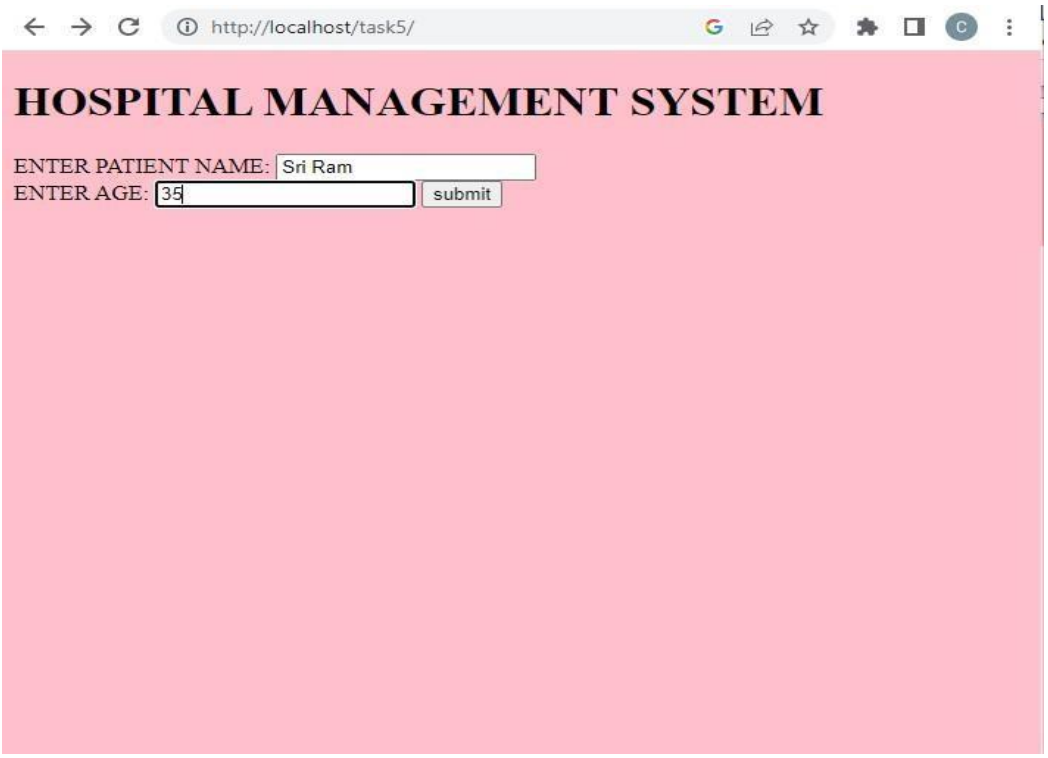
```
<!DOCTYPE html>
<html>
<body BGCOLOR="PINK">

<h1>HOSPITAL MANAGEMENT SYSTEM </h1>

<form action="welcome.php" method="POST">

 ENTER PATIENT NAME: <input type="text" name="name" ><br>

 ENTER AGE: <input type="text"  name="age" >
<input type="submit" value="submit">

</form>
</body>
</html>
```

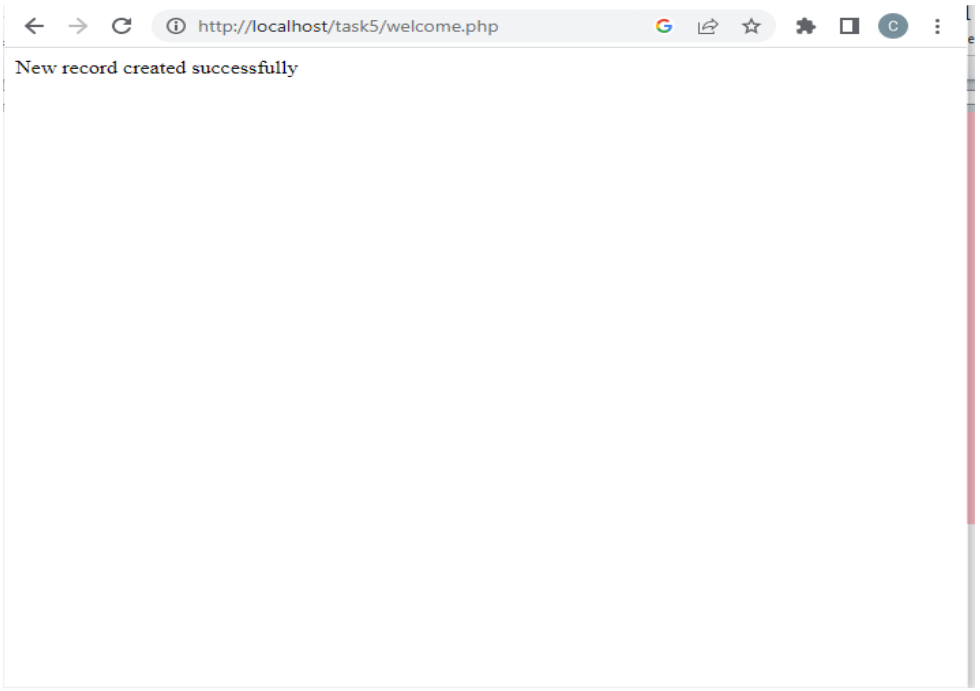
Welcome.php

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "sample";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
```

```
if ($conn->connect_error)
 {
  die("Connection failed: " . $conn->connect_error);
}
$name=$_POST["name"];
$age=$_POST["age"];

$sql = "INSERT INTO patient (name, age) VALUES ('$name', '$age')";

if ($conn->query($sql) === TRUE)
 {
        echo "New record created successfully";
}
else
 {
        echo "Error: " . $sql . "<br>" . $conn->error;
}


$conn->close();
?>
```

Output:





Result:

Thus the above program was executed successfully and the output was verified.

Usecase:

LIBRARY MANAGEMENT SYSTEM

Aim:

To implement library management system using PHP and MySQL.

Procedure:

- Environment Setup
  - ✓ Install XAMPP Web Server
  - ✓ Open the XAMPP Control Panel.
  - ✓ Start the Apache server by clicking on the Start button.
  - ✓ Start the MySQL by clicking on the Start button.
  - ✓ Create all the files needed for login.
  - ✓ Create login table in the database using phpMyAdmin in XAMPP.
- Creation of Necessary Files
  - ✓ index.html - This file is created for the GUI view of the login page.
  - ✓ Welcome.php - this file contains the connection code for database connectivity and inserts form data into the database after submission.

Home.html

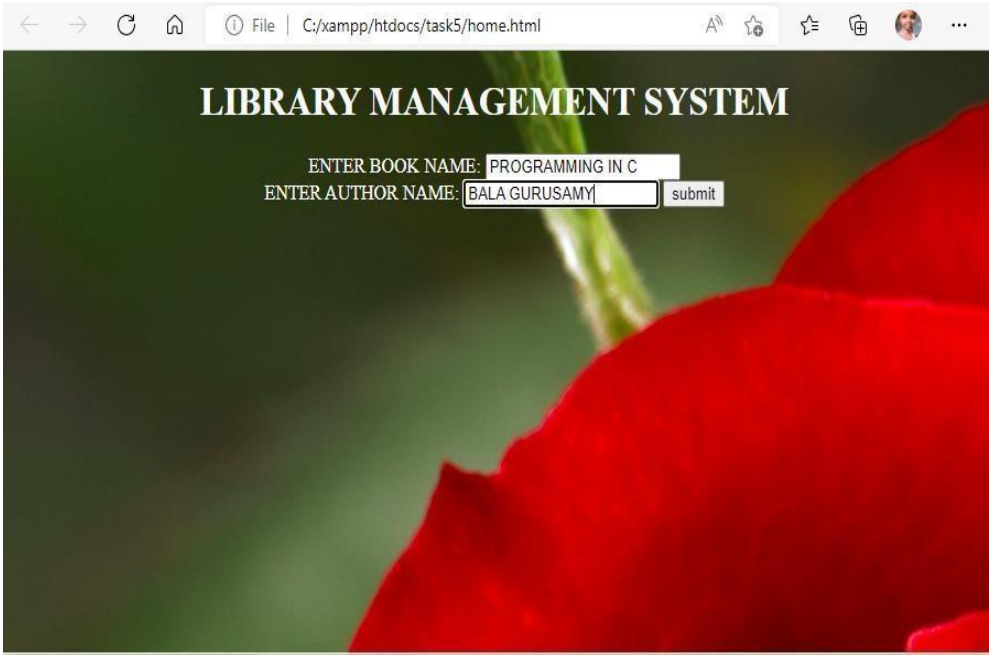
```
<!DOCTYPE html>
<html>
<body background="C:\Users\NIVAASHINI\Desktop\rose.jpg">
<p >
<center>
<h1 style="color:white">LIBRARY MANAGEMENT SYSTEM </h1>

<form action="welcome.php" method="POST">

<label style="color:white">ENTER BOOK NAME:</label> <input type="text"
name="bname" ><br>

<label style="color:white"> ENTER AUTHOR NAME: </label><input type="text"
name="aname" >
<input type="submit" value="submit">
</center>
</form>
</body>
</html>
```

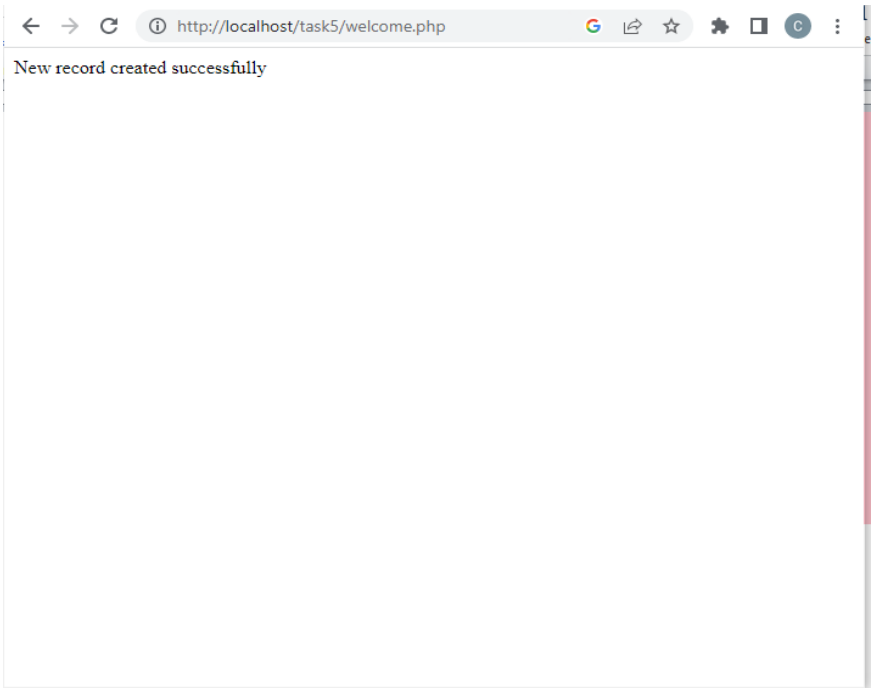
Welcome.php

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "sample";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
```

```
// Check connection
if ($conn->connect_error)
 {
  die("Connection failed: " . $conn->connect_error);
}
$bname=$_POST["name"];
$aname=$_POST["age"];

$sql = "INSERT INTO book (bookname, authorname) VALUES ('$bname', '$aname')";

if ($conn->query($sql) === TRUE)
 {
        echo "New record created successfully";
}
else
 {
        echo "Error: " . $sql . "<br>" . $conn->error;
}


$conn->close();
?>
```

Output:





Result:

Thus the above program was executed successfully and the output was verified.

Usecase:

RAILWAY TICKET MANAGEMENT SYSTEM LOGIN PAGE

Aim:

To implement railway ticket management system using PHP and MySQL.

Procedure:

- Environment Setup
  - ✓ Install XAMPP Web Server
  - ✓ Open the XAMPP Control Panel.
  - ✓ Start the Apache server by clicking on the Start button.
  - ✓ Start the MySQL by clicking on the Start button.
  - ✓ Create all the files needed for login.
  - ✓ Create login table in the database using phpMyAdmin in XAMPP.
- Creation of Necessary Files
  - ✓ index.html - This file is created for the GUI view of the login page.
  - ✓ Welcome.php - this file contains the connection code for database connectivity and inserts form data into the database after submission.

home.html

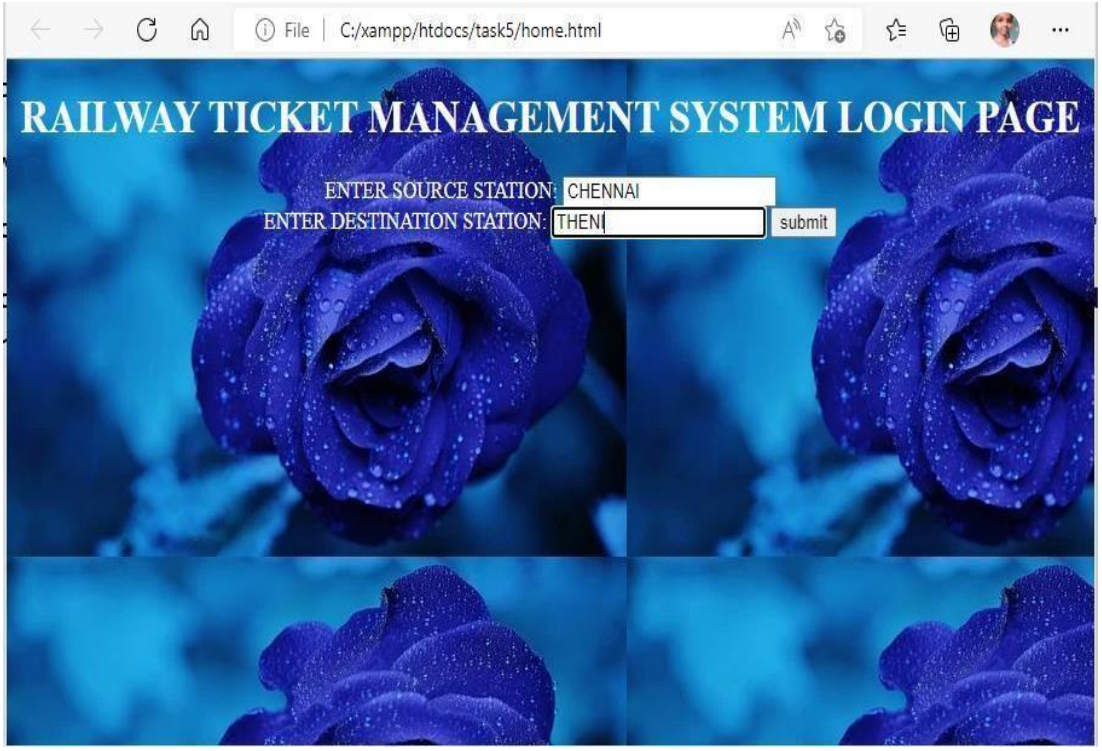
```
<!DOCTYPE html>
<html>
<body background="C:\Users\NIVAASHINI\Desktop\rose1.jpg">
<p >
<center>
<h1 style="color:white">RAILWAY TICKET MANAGEMENT SYSTEM  LOGIN PAGE
</h1>

<form action="welcome.php" method="POST">

<label style="color:white">ENTER SOURCE STATION:</label> <input type="text"
name="source" ><br>

<label style="color:white"> ENTER DESTINATION STATION: </label><input type="text"
name="destination" >
<input type="submit" value="submit">
</center>
</form>
</body>
</html>
```

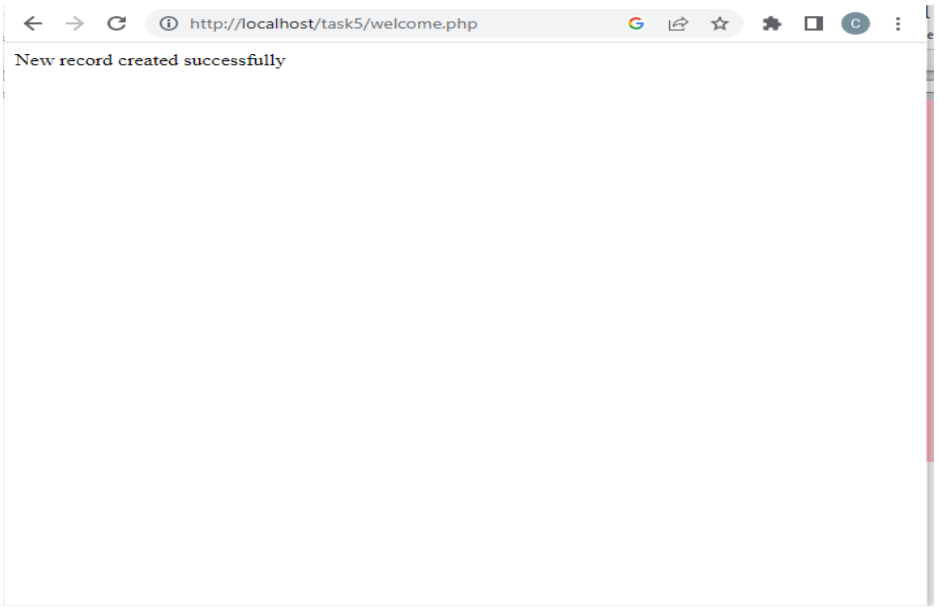
Welcome.php

```
<?php
$servername = "localhost";
```

```
$username = "root";
$password = "";
$dbname = "sample";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error)
 {
  die("Connection failed: " . $conn->connect_error);
}
$source=$_POST["source"];
$destination=$_POST["destination"];

$sql = "INSERT INTO train (source, destination) VALUES ('$source', '$destination')";

if ($conn->query($sql) === TRUE)
 {
        echo "New record created successfully";
}
else
 {
        echo "Error: " . $sql . "<br>" . $conn->error;
}


$conn->close();
?>
```

Output:





Result:

Thus the above program was executed successfully and the output was verified.