**Inspection Code**

**By**

**Pseudo Mavericks**

**Starting the Project:**

The project “Employee Management System” was developed using the languages such as PHP, HTML, CSS, Bootstrap, JQuery, Javascript, MySQL. The IDE used in the development was Eclipse/Netbeans.

1. **Setting up the DB File:**

The below code represents the code written in PHP for establishing a connection with the database. The file name is ‘db.php’ which is placed in root directory. The ‘username’ and ‘password’ in the code represents the db user credentials of local(if used for running locally) or server(if deployed on server). This code creates a PDO object which can be used throughout the project by including this file.

<?php

function connect() {

$username = 'root';

$password = '';

$mysqlhost = 'localhost';

$dbname = 'employee\_ms';

$pdo = new PDO('mysql:host='.$mysqlhost.';dbname='.$dbname.';charset=utf8', $username, $password);

if($pdo){

//make errors throw exceptions

$pdo->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

return $pdo;

}else{

die("Could not create PDO connection.");

}

}

$sql = connect();

?>

1. **Authorizing user:**

The below code is from the file ‘auth.php’ which is in the root directory. It helps to start the session and get the employee data based on the ID. The user is authorized based on the ID and the employee record is fetched from database by taking the employee id from session variable. This code also has functionality to stop the session.

<?php

ob\_start();

$status = session\_status();

if($status == PHP\_SESSION\_NONE){

session\_start();

}

else if($status == PHP\_SESSION\_DISABLED){

//Sessions are not available

}else if($status == PHP\_SESSION\_ACTIVE){

//Destroy current and start new one

session\_destroy();

session\_start();

}

if(!isset($\_SESSION['employee'])){

header('Location: login.php');

exit();

}

else

{

include 'db.php';

$employeeid = $\_SESSION['employee'];

$stmt = $sql->prepare("select \* from employees where id=?");

$stmt->bindParam(1, $employeeid, PDO::PARAM\_INT);

$stmt->execute();

$employeedata = $stmt->fetch();

}

?>

1. **Index File:**

The below code is from file ‘index.php’ which is in the root directory. This is the landing page of employee after login. This file will be used in other php files after logging in.

<body>

<?php $page="index"; $tab=""; include "includes/nav.php"; ?>

<div class="main-content" id="panel">

<?php include 'includes/header.php'; ?>

<div class="header bg-primary pb-6">

<div class="container-fluid">

<div class="header-body">

<div class="row align-items-center py-4">

<div class="col-lg-12">

<h6 class="h2 text-white d-inline-block mb-0">

Welcome, <?php echo $employeedata['firstname'].' '.$employeedata['lastname']; ?>

</h6>

</div>

</div>

</div>

</div>

</div>

</div>

<?php include "includes/scripts.php"; ?>

</body>

1. **Helper files:**

These files are included in the ‘includes’ folder in the main folder. It contains files such as ‘head.php’, ‘header.php’, ‘nav.php’, ‘scripts.php’. The head.php file just contains the stylesheets and meta data. This code is included in all the presentation files throughout the project. The ‘header.php’ file contains the header code which will be included in all the presentation files.

The below code represents the code for navigation of employee after logged in.

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link <?php echo $page=='index'?'active':''; ?>" href="index.php">

<i class="fa fa-home text-primary"></i>

<span class="nav-link-text">Dashboard</span>

</a>

</li>

<li class="nav-item">

<a class="nav-link <?php echo $page==''?'active':''; ?>" href="#!">

<i class="fa fa-cog text-primary"></i>

<span class="nav-link-text">Payroll</span>

</a>

</li>

<li class="nav-item">

<a class="nav-link <?php echo $page==''?'active':''; ?>" href="#!">

<i class="fa fa-cog text-primary"></i>

<span class="nav-link-text">Billing Hours</span>

</a>

</li>

<li class="nav-item">

<a class="nav-link <?php echo $page==''?'active':''; ?>" href="#!">

<i class="fa fa-cog text-primary"></i>

<span class="nav-link-text">Leave Requests</span>

</a>

</li>

<li class="nav-item">

<a class="nav-link <?php echo $page==''?'active':''; ?>" href="logout.php">

<i class="fa fa-power-off text-primary"></i>

<span class="nav-link-text">Logout</span>

</a>

</li>

</ul>

The scripts.php file contains the javascript required for the project. For instance the below code represents the alert for the user if there are any wrong details while logging in.

function notify(type, message) {

if(type=='success'){

var title = 'Success'

}else if(type=='danger'){

var title = 'Error';

}

$.notify({

icon: 'ni ni-bell-55',

title: title,

message: message,

url: ''

}, {

element: 'body',

type: type,

allow\_dismiss: true,

placement: {

from: 'top',

align: 'center'

},

offset: {

x: 15, // Keep this as default

y: 15 // Unless there'll be alignment issues as this value is targeted in CSS

},

spacing: 10,

z\_index: 1080,

delay: 2500,

timer: 25000,

url\_target: '\_blank',

mouse\_over: false,

animate: {

enter: 'animated fadeInDown',

exit: 'animated fadeOutUp'

},

template: '<div data-notify="container" class="alert alert-dismissible alert-{0} alert-notify" role="alert">' +

'<span class="alert-icon" data-notify="icon"></span> ' +

'<div class="alert-text"</div> ' +

'<span class="alert-title" data-notify="title">{1}</span> ' +

'<span data-notify="message">{2}</span>' +

'</div>' +

// '<div class="progress" data-notify="progressbar">' +

// '<div class="progress-bar progress-bar-{0}" role="progressbar" aria-valuenow="0" aria-valuemin="0" aria-valuemax="100" style="width: 0%;"></div>' +

// '</div>' +

// '<a href="{3}" target="{4}" data-notify="url"></a>' +

'<button type="button" class="close" data-notify="dismiss" aria-label="Close"><span aria-hidden="true">&times;</span></button>' +

'</div>'

});

}

// Show notify

<?php if(isset($alert)){ echo $alert; } ?>

1. **Login Page:**

The code for login is in ‘login.php’ file and is in the root directory. This file contains the UI code for login. After submitting the details it starts a session to store the employee id in the session variable.

The below code represents the UI code for login.

<div class="row justify-content-center">

<div class="col-xl-5 col-lg-6 col-md-8 px-5">

<h1 class="text-white">Welcome back!</h1>

<p class="text-lead text-white">Please login with your credentials below.</p>

</div>

</div>

<form role="form" method="post">

<div class="form-group mb-3">

<?php if(isset($msg)){ echo $msg; } ?>

<div class="input-group input-group-merge input-group-alternative">

<div class="input-group-prepend">

<span class="input-group-text"><i class="ni ni-email-83"></i></span>

</div>

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

</div>

</div>

<div class="form-group mb-3">

<div class="input-group input-group-merge input-group-alternative">

<div class="input-group-prepend">

<span class="input-group-text"><i class="ni ni-lock-circle-open"></i></span>

</div>

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

</div>

</div>

<div class="text-center mb-0">

<button type="submit" name="submit" class="btn btn-primary">Sign in</button>

</div>

</form>

-The below code starts the session after logging in after validating the user credentials.

if(isset($\_POST['submit'])){

$\_email = $\_POST['email'];

$\_password = $\_POST['password'];

$stmt = $sql->prepare("select \* from employees where email = ?");

$stmt->bindParam(1, $\_email, PDO::PARAM\_STR);

$stmt->execute();

$row = $stmt->fetch();

$total = $stmt->rowCount();

if($total>0){

$db\_email = $row['email'];

$db\_password = $row['password'];

if($\_email == $db\_email && password\_verify($\_password, $db\_password)){

$\_SESSION['employee'] = $row['id'];

ob\_start();

header('location:index.php');

exit();

}else {

$error = "<div class='alert alert-danger'>Email or Password is incorrect</div>";

}

}else {

$error = "<div class='alert alert-danger'>Email or Password is incorrect</div>";

}

}

1. **Register the Employee**

The file ‘register.php’, which is in the root directory has the code to register the employee. The UI code takes the required fields from the user and submits it to the database.

The below form contains the code for collecting the employee data.

<form role="form" method="post">

<div class="form-group mb-3">

<?php if(isset($error)){ echo $error; } ?>

<div class="input-group input-group-merge input-group-alternative">

<div class="input-group-prepend">

<span class="input-group-text"><i class="ni ni-single-02"></i></span>

</div>

<input required class="form-control" name="firstname" placeholder="First Name" type="text">

</div>

</div>

<div class="form-group mb-3">

<div class="input-group input-group-merge input-group-alternative">

<div class="input-group-prepend">

<span class="input-group-text"><i class="ni ni-single-02"></i></span>

</div>

<input required class="form-control" name="lastname" placeholder="Last Name" type="text">

</div>

</div>

<div class="form-group mb-3">

<div class="input-group input-group-merge input-group-alternative">

<div class="input-group-prepend">

<span class="input-group-text"><i class="ni ni-email-83"></i></span>

</div>

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

</div>

</div>

<div class="form-group mb-3">

<div class="input-group input-group-merge input-group-alternative">

<div class="input-group-prepend">

<span class="input-group-text"><i class="ni ni-lock-circle-open"></i></span>

</div>

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

</div>

</div>

<div class="form-group mb-3">

<div class="input-group input-group-merge input-group-alternative">

<div class="input-group-prepend">

<span class="input-group-text"><i class="ni ni-lock-circle-open"></i></span>

</div>

<input required class="form-control" name="cpassword" placeholder="Confirm Password" type="password">

</div>

</div>

<div class="text-center mb-0">

<button type="submit" name="submit" class="btn btn-primary">Register</button>

</div>

</form>

The below code validates the data and inserts the data into employee table.

if(isset($\_POST['submit'])){

$stmt = $sql->prepare("select \* from employees where email=?");

$stmt->bindParam(1, $\_POST['email'], PDO::PARAM\_STR);

$stmt->execute();

$result = $stmt->fetch();

if($stmt->rowCount()>0){

$error = "<div class='alert alert-danger'>Sorry, this email address already exists.</div>";

}elseif($\_POST['password'] != $\_POST['cpassword']){

$error = "<div class='alert alert-danger'>Password and Confirm Password doesn't match.</div>";

}else{

$created\_at = date('Y-m-d');

$options = [ 'cost' => 11];

$password = password\_hash($\_POST['password'], PASSWORD\_BCRYPT, $options);

$tutorrole = 'tutor';

$query = "INSERT into employees (firstname, lastname, email, password, created\_at) VALUES (?, ?, ?, ?, ?)";

$stmt = $sql->prepare($query);

$stmt->bindParam(1, $\_POST['firstname'], PDO::PARAM\_STR);

$stmt->bindParam(2, $\_POST['lastname'], PDO::PARAM\_STR);

$stmt->bindParam(3, $\_POST['email'], PDO::PARAM\_STR);

$stmt->bindParam(4, $password, PDO::PARAM\_STR);

$stmt->bindParam(5, $created\_at, PDO::PARAM\_STR);

$stmt->execute();

$\_SESSION['message'] = "<div class='alert alert-success'>Account created successfully. You can now login to your account.</div>";

header("location:login.php");

}

}

1. **Database:**

The file ‘employee\_ms.sql’ has the sql queries to create the tables required for the project. These tables are used for other phases too in the project.

-- Table structure for table `billing\_hours`

--

CREATE TABLE `billing\_hours` (

`id` int(11) NOT NULL,

`user\_id` int(11) NOT NULL,

`hours` int(11) NOT NULL,

`date` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--

-- Table structure for table `employees`

--

CREATE TABLE `employees` (

`id` int(11) NOT NULL,

`firstname` varchar(255) NOT NULL,

`lastname` varchar(255) NOT NULL,

`email` varchar(255) NOT NULL,

`password` varchar(255) NOT NULL,

`phone` varchar(255) NOT NULL,

`created\_at` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--

-- Table structure for table `employee\_project`

--

CREATE TABLE `employee\_project` (

`id` int(11) NOT NULL,

`employee\_id` int(11) NOT NULL,

`project\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--

-- Table structure for table `hr\_users`

--

CREATE TABLE `hr\_users` (

`id` int(11) NOT NULL,

`firstname` varchar(255) NOT NULL,

`lastname` varchar(255) NOT NULL,

`email` varchar(255) NOT NULL,

`password` varchar(255) NOT NULL,

`phone` varchar(255) NOT NULL,

`created\_at` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--

-- Table structure for table `leave\_requests`

--

CREATE TABLE `leave\_requests` (

`id` int(11) NOT NULL,

`user\_id` int(11) NOT NULL,

`leave\_start\_date` varchar(255) NOT NULL,

`leave\_end\_date` varchar(255) NOT NULL,

`leave\_request\_date` varchar(255) NOT NULL,

`reason` text NOT NULL,

`status` varchar(255) NOT NULL,

`action\_by` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- --------------------------------------------------------

--

-- Table structure for table `projects`

--

CREATE TABLE `projects` (

`id` int(11) NOT NULL,

`title` varchar(255) NOT NULL,

`start\_date` date NOT NULL,

`finish\_date` date NOT NULL,

`status` varchar(255) NOT NULL,

`created\_at` date NOT NULL,

`hr\_id` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

.