University of Bahrain
College of Information Technology
Department of Information Systems

جامعة البحرين
كلية تقنية المعلومات
قسم نظم المعلومات

ITIS345 Web Design and Development II

Course Project

Developing a PHP Web Application

Travel

Group:		**21**	**Section:**	**01**
No.	**ID**	**NAME**		
1	20192408	Fatema Hussain Habib		
2	20193627	Naba Yahya		
3	202006306	Manar Eyad Ahmed		
4	202005915	Mariam Jawad		

Table of Contents

Introduction

PHP is a language that combines input and output. It contains many features that are necessary for developing web applications. It is also considered a text programming language that includes functions, which are a series of instructions that will be executed on other programs or applications. It is useful for everyone to follow the latest technology trends, as it ranks sixth among the most common languages. There are two types: one for the server side and one for the client side.

- Server-side: The server receives a script, processes the request, and sends the output to the web browser in HTML format with the help of the database that stores the information.
- Client-side: Uses web browsers to run scripts without seeing the code.

Characterizing the PHP language:

1. Easy to use: It is preferred for beginners as it is easy to learn the language compared to other languages and modify HTML code.
2. Multifunctional: It is used in many functions, such as creating blogs, desktop applications, and e-commerce. It can also send emails using PHP.
3. Fast: It works to make websites load faster, especially on the server side, compared to the Python language and improves search engine rankings.

In this project, we will rely on the PHP language to design the website, in addition to other languages. This is to create the most important pages for users, allowing them to browse the site safely, see weather details, and control the design. We aim to use what we learned during the class and learn more about the language.

Database Connection

In this part, we created a config file and wrote all the details for the database server name "localhost", user name "rootp", password "projects", and database name "project_345". In the database, we created a "projects" table and considered the user name, email, day of birth, password, and confirmed password that the user entered. Still, other fields like ID are auto-increment, and salts are random, so users should not enter them.

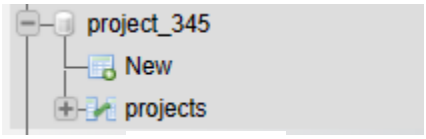

Figure 1

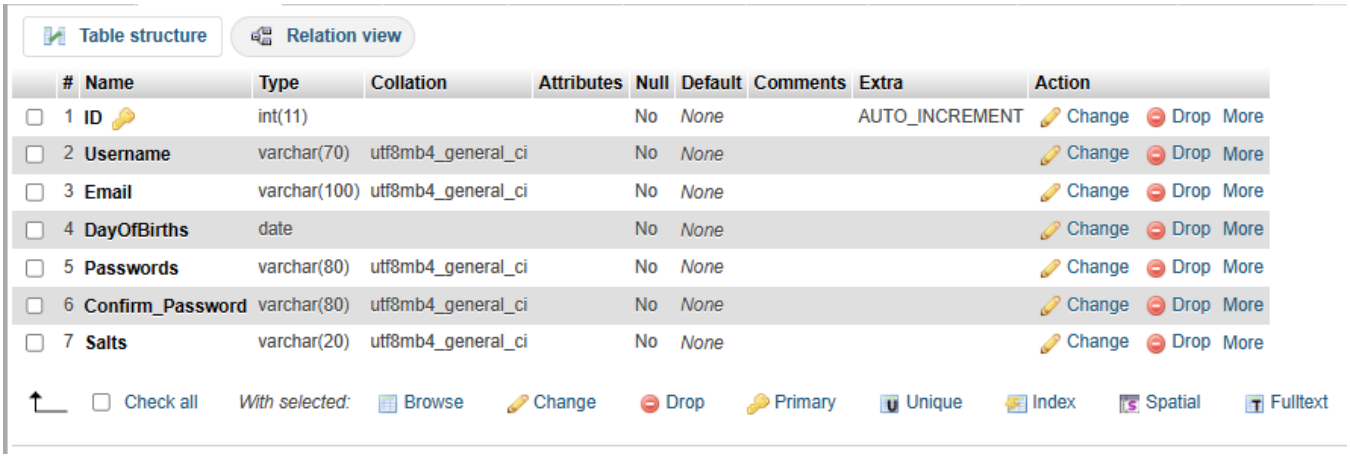

Figure 2

Sign up page

- Description:

For this page, we created two functions. First, "Success Alter" is used when the user's full input with correct information, displaying a success pop-up message, and moving to the log-in page. Second, a "Random String" is used for salt "$salt = generateRandomString()" to help in function md5(). $salt It is applied during the hashing process, as it is an additional part of the data to make hacking difficult, and the use of salt makes it impossible to find the resulting hash, so we used with password field.

Then we connected with the database. We wrote down some conditions. For example, the password condition states that if the password does not match the confirmed password, the user cannot create an account. The email condition checks whether or not the email exists in the database; if it does, the user cannot use the same email. The username condition is to check if usernames exist in the database or not; if they exist, users can't use the same username. Third, we inserted the all-user information in the table (projects). In addition, we use the function md5(). md5() is a function built into the PHP language that is used for hashing and is popular in various applications because it is fast and simple. It consists of two entries, one of which is the string value, and the other is the password field. It is used to always store the password securely, even if the system is hacked " $password_md5 = md5($password. $salt)". The user can't ignore any field because all are required, and the password must be 8 characters or more for better security.

- Final Interface:

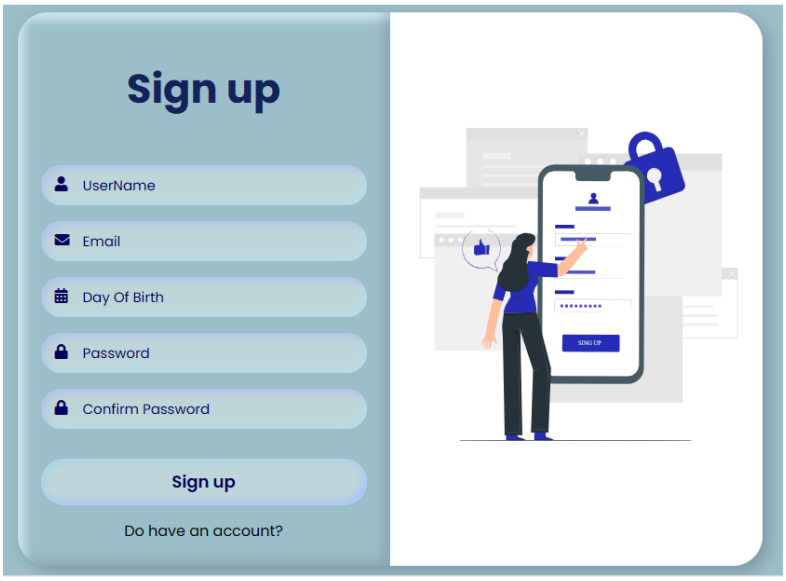

Figure 3

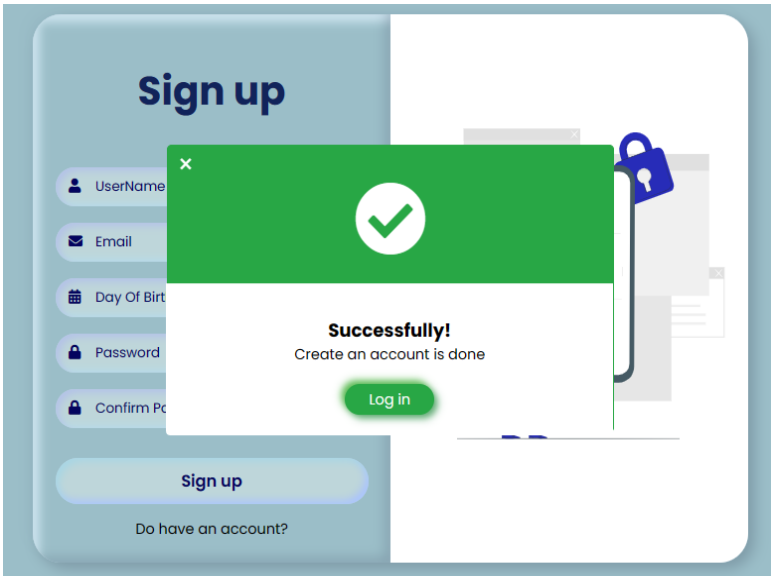

Figure 4

←T→	ID	Username	Email	DayOfBirths	Passwords	Confirm_Password	Salts
Edit Copy Delete	2	f44	uobyear@gmail.com	2001-06-06	1cd6c5cfb7a87853b81b607fe101f06b	1cd6c5cfb7a87853b81b607fe101f06b	4F5aDC

Figure 5

Log in Page

- Description:

In the log-in form, we wrote conditions to check two things. First, ensure that the username is correct in the same database as the user; otherwise, the user does not have an account, so the display will be altered to tell the user that the user that the "username not found!" and go back to the log-in page. If the password is not the same as the one in the database, display the incorrect message and go back to the log-in page. If you successfully log in, the session will begin and take you to the home page. "session_start()" is the creation of an open session, or when used, the save is called for the open and read session, where the stored data can be retrieved every time one logs in.

- Final Interface:

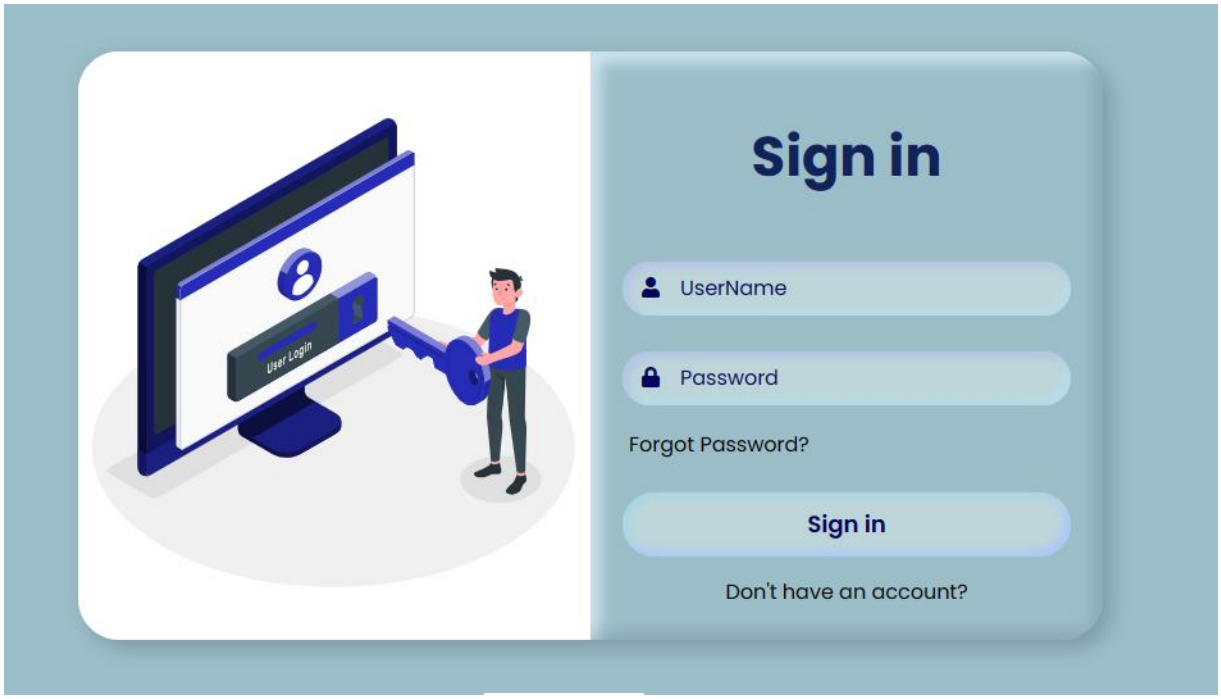

Figure 6

Figure 7

Figure 8

Reset Password Page:

- Description:

On this page, when the user forgets the password and wants to reset it, First, we check whether the email is in the database or not; if not, we display an error to tell you, "Email does not exist.". Then, if the email is correct, the user writes an 8-character or more new password, confirms the password, and checks if there is a similar password in two fields, and we use salt with function md5. After that, insert a new password in the database. If correct, reset the display pop-up message to move to the log-in page.

- Final Interface:

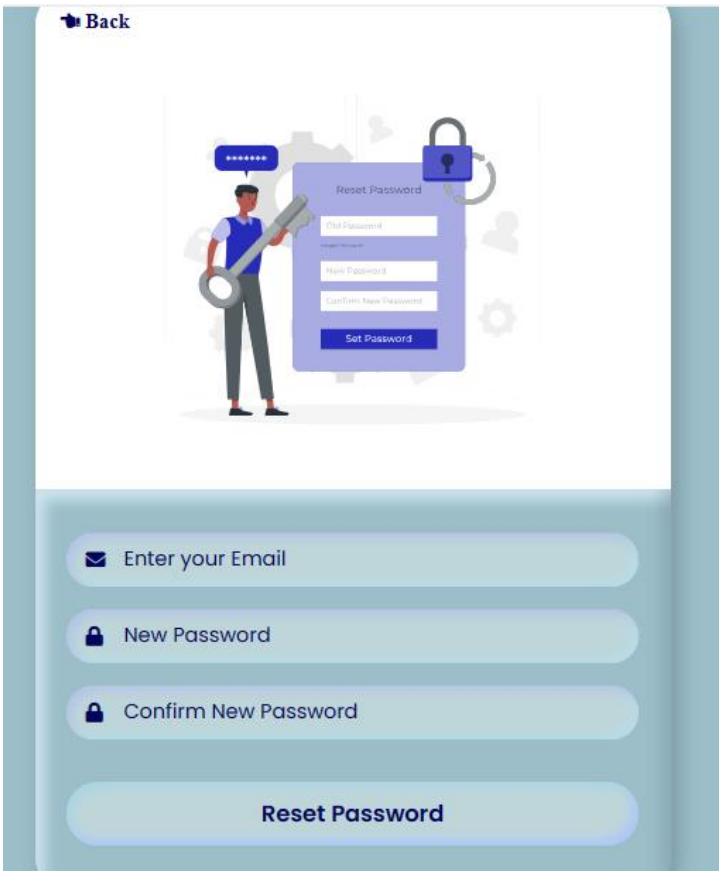

Figure 9

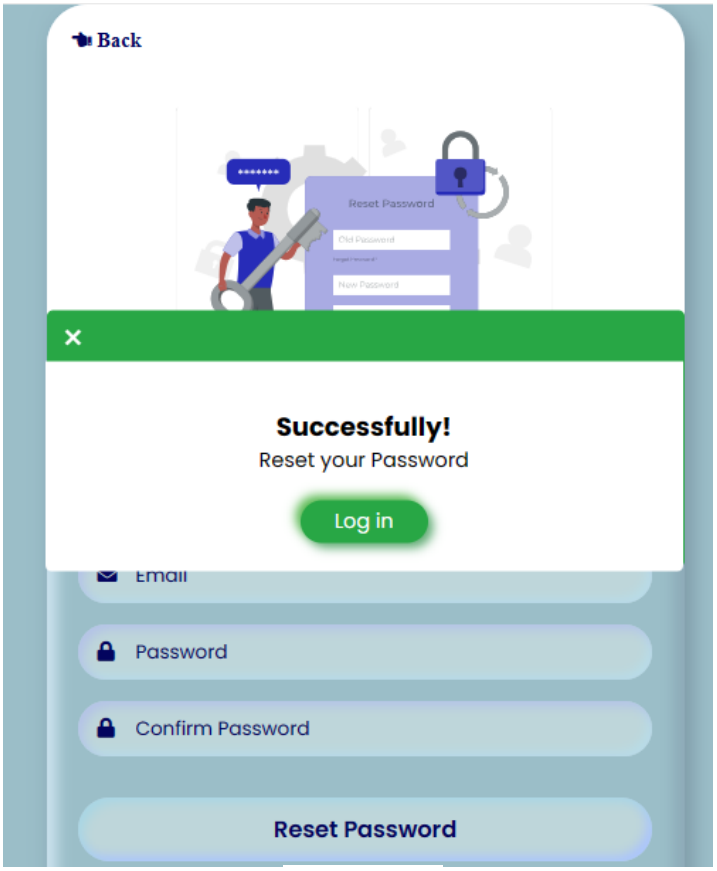

Figure 10

Figure 11

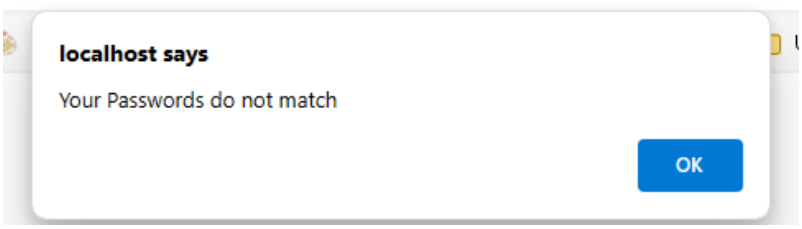

Figure 12

Home Page

- Description:

After the user logs in, he will be transferred directly to the homepage. This page has been designed to suit all the user's needs, as it contains all the information related to the company, such as a simple overview and the services it provides. The user can change the theme of the page, such as the background color, font size, and color. To provide this distinctive feature to the user, we used cookies, which are a small piece of data that is stored inside the browser and the site can retrieve it later. The site allows saving some data related to the user, such as a user preference, and the site can retrieve any information. Or previous preparations made by the user during his previous visits.

On the next page, we will discuss how to use cookies to change the user preference via the settings page.

- Final Interface:

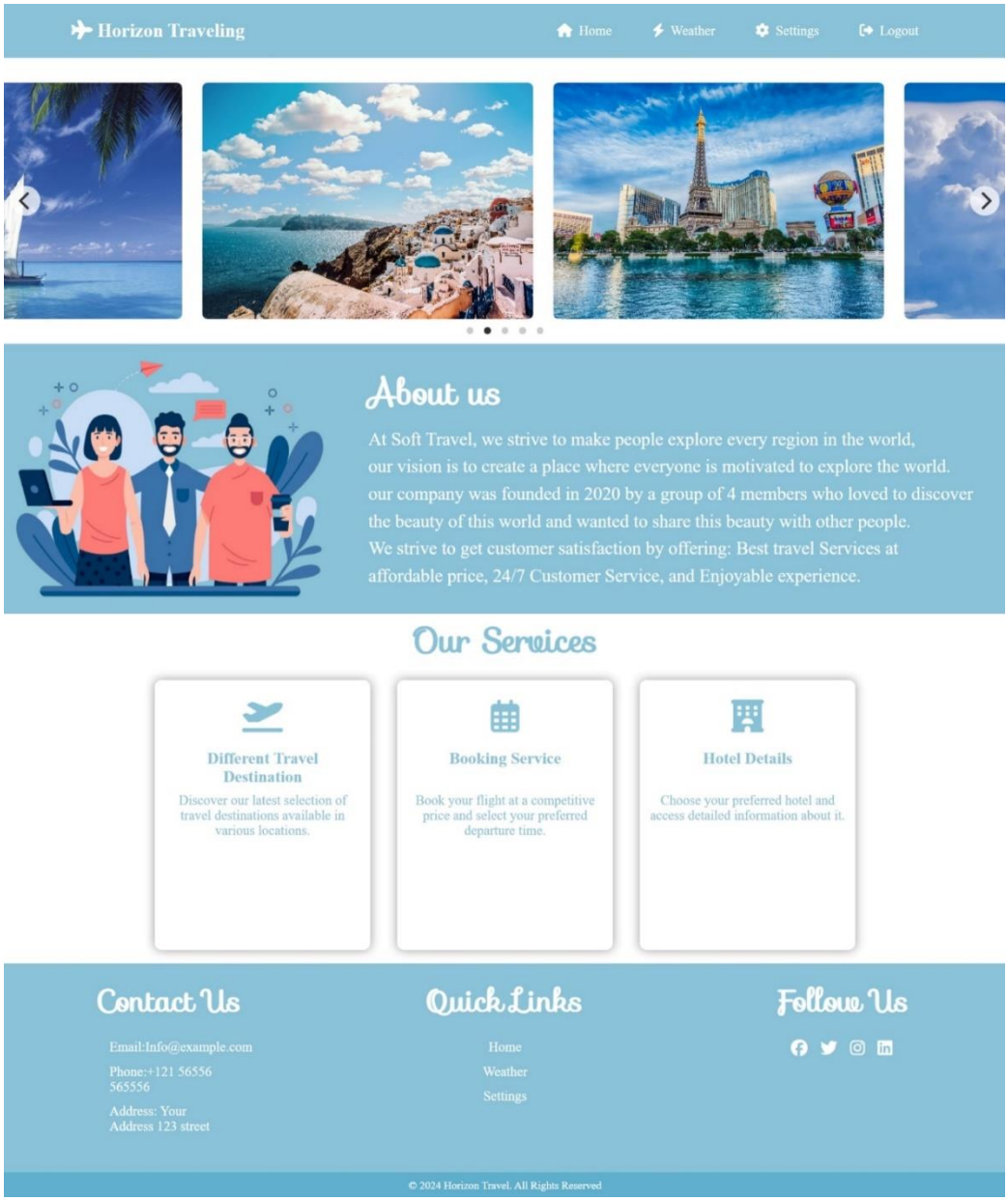

Figure 13

Settings Page

- Description:

In settings.php, The user can easily design the home page based on his preferences. The page consists of a form that has 3 lists: font size list, font color list, and background color list. The user should select his preference from three lists above then click on customize button. If the user chooses from the three lists then the cookie will be created using setcookie() function and has expiry time for 24 hours. Which means that the cookie will expire after one day. The cookie will be stored in the browser so the user can access website later. After clicking on customize button it'll show the following message "Your settings have been saved! You can go to the home page to see the results" and it'll take you directly to the home page to see changes.

- Final Interface:

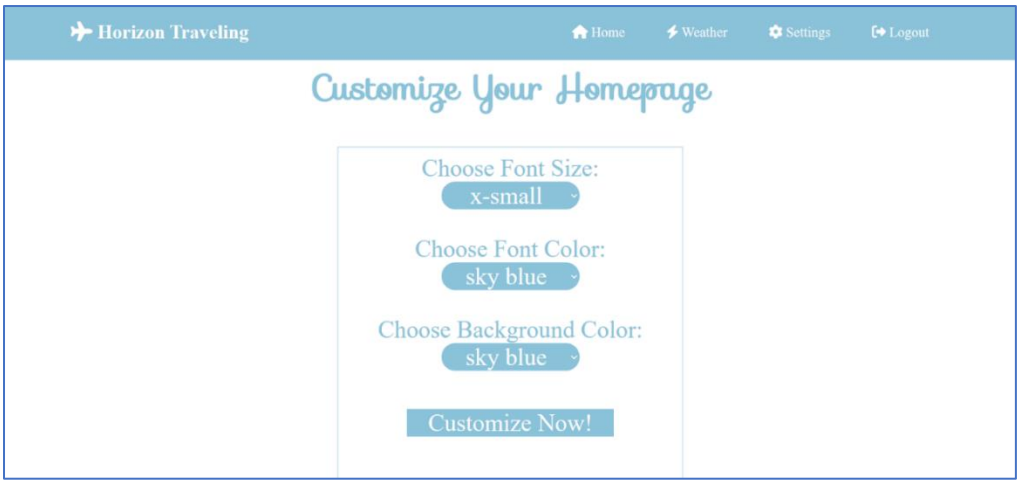

Figure 14

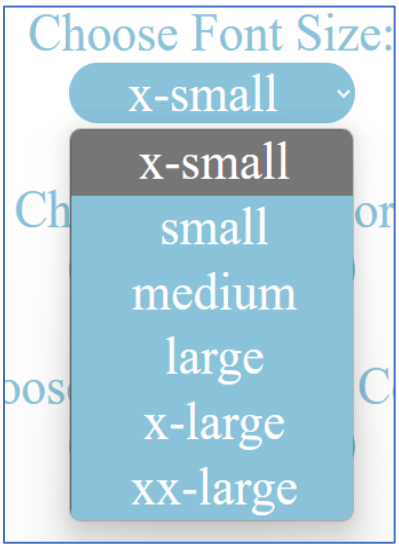

Figure 15

Figure 16

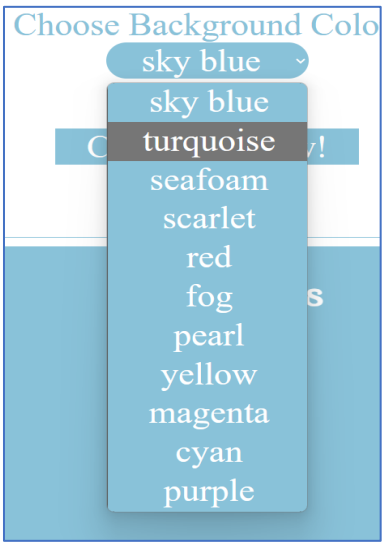

Figure 17

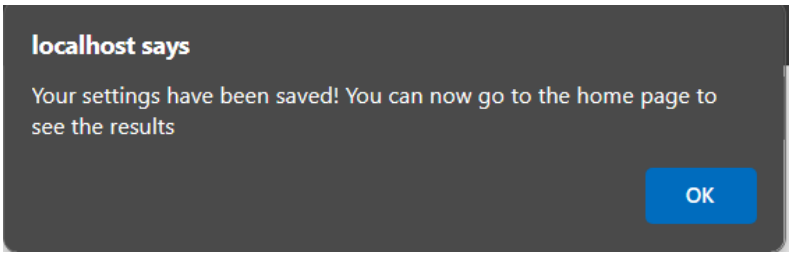

Figure 18

- Output of cookies in home page:

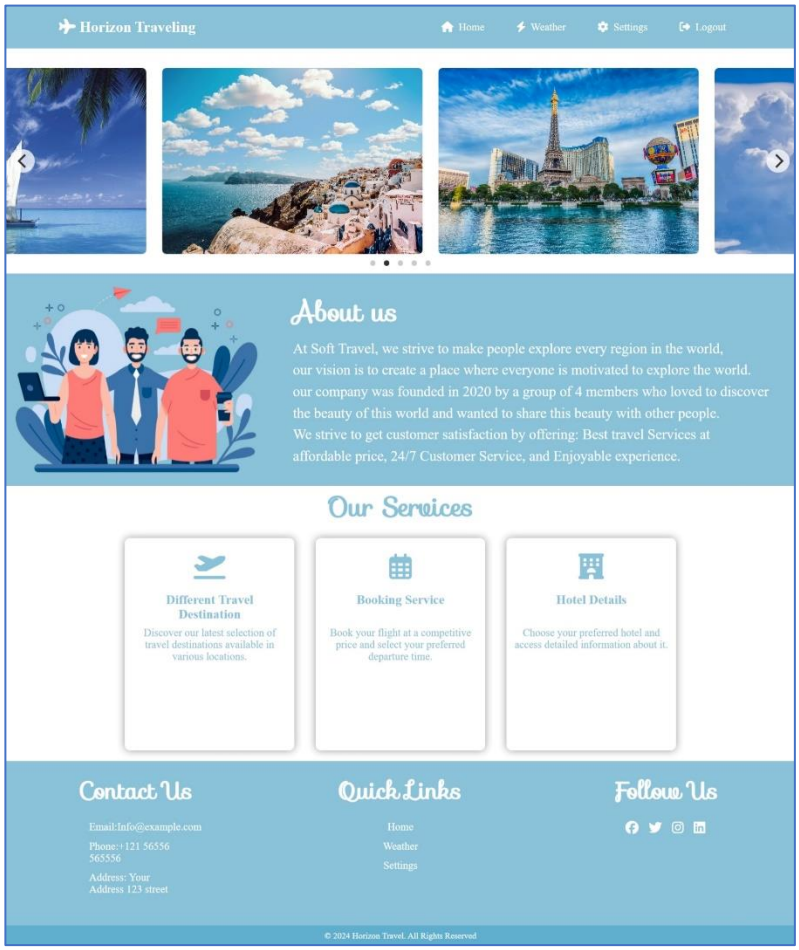

Figure 19

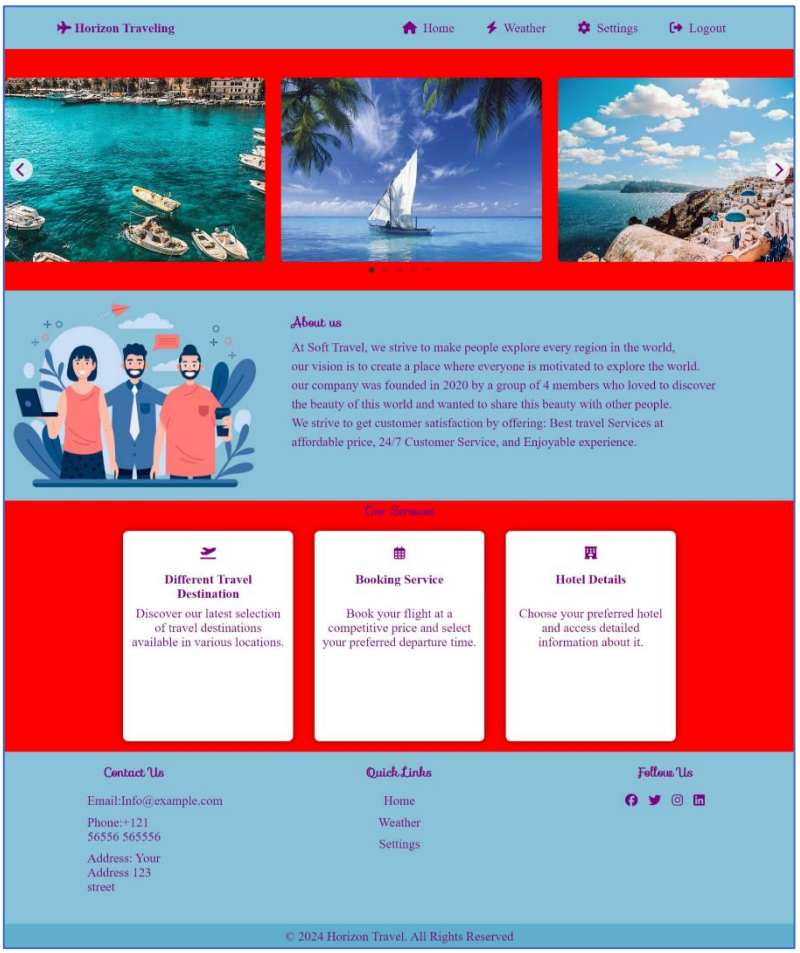

Figure 20

Before applying the cookie

After applying the cookie

With the following format:

Font size: X-large

Font color: purple

Background color: red

Weather Pages (Web Service with API Key Registration):

(the API key owner is **Mariam Jawad**, who do the Sign up with academic email to the OpenWeatherMap and get the API key)

API key:

```
4fd753e86220f1947a861721bea1ccce
```

Figure 21

Here we have two weather pages:

The Two Weather Pages allow us to have casual conversations about the weather with visitors to our website. Gathering weather data from OpenWeatherMap by registration is the first step, which is similar to asking a weather specialist for the most recent updates.

Weather1 page

- Description:

In this PHP code segment, we created an array named `$cities` to store the names of various cities. The array is then sorted alphabetically using the sort() function to ensure a consistent order. Through a `foreach` loop, each city name is iterated over. Within the loop, substr() extracts the first letter of each city name, while strtoupper() ensures uniformity by converting it to uppercase. These functions aid in grouping cities by their initial letters. A conditional check is employed to determine if the current letter being processed differs from the previous one. If so, a disabled <option> element is appended to the dropdown list, acting as a visual divider for cities starting with that letter. This organizational structure enhances user experience by facilitating easier navigation through the dropdown list. Finally, each city name is dynamically added as an <option> within the dropdown, allowing users to select their desired city. The code's benefits lie in its ability to present cities in a sorted manner and visually organize them by their initial letters, thereby improving usability. The chosen city is subsequently sent to "weather2.php" after clicking the "Search" button.

- Final Interface:

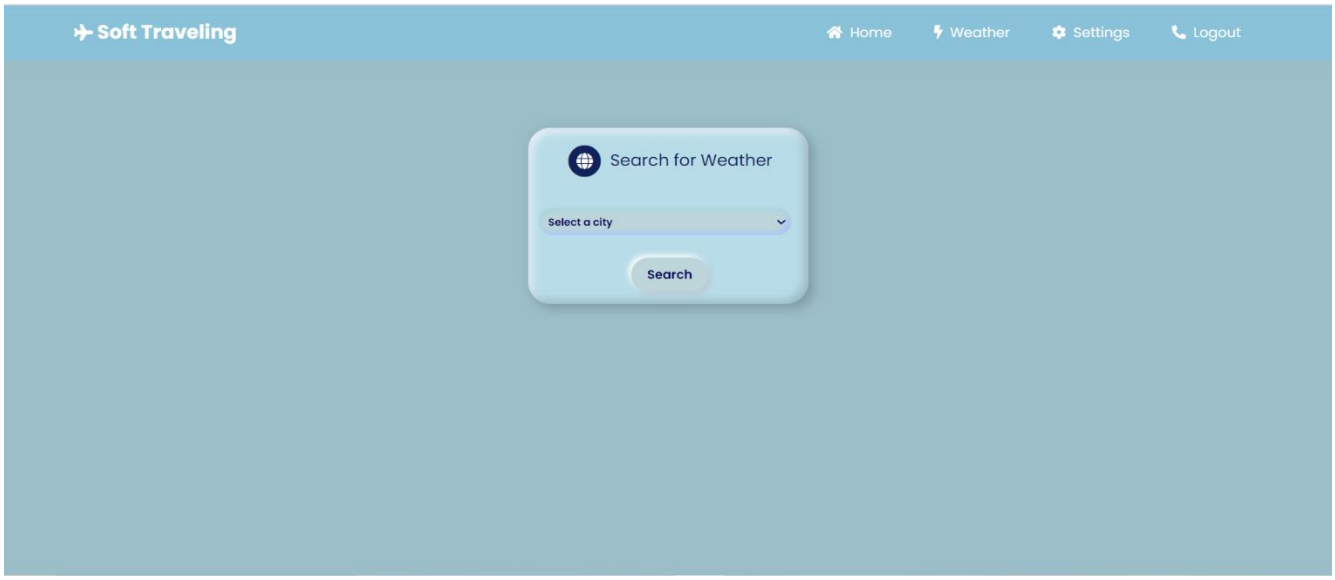

Figure 22

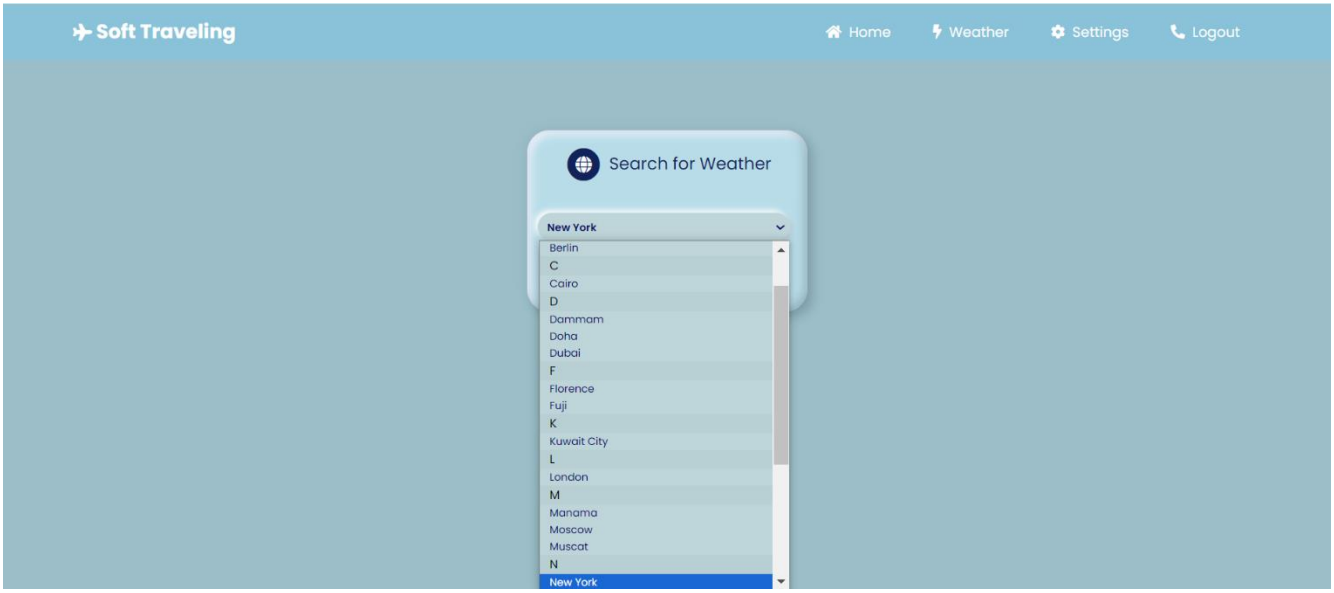

Figure 23

Weather2 page

- Description:

This PHP code fetches and displays a weather forecast for the selected city in weather1 from the form. giving a brief overview of the local weather in various locations, Displays the forecast for each day, including temperature, humidity, and wind speed as required. To facilitate simple navigation, we provided a "Back" button to return to the previous page (weather1) if the user wants to select other cities.

It's an easy-to-use method of disseminating current weather information so that everyone can access and comprehend it.

An API key is a special identification that permits access to an API (Application Programming Interface). Software programs can connect with one another through the use of APIs, which are collections of guidelines and protocols.

An API key is issued when registering for services such as OpenWeatherMap. As a credential, this key enables you to authenticate and access the information and features that the API has to offer.

Integrating an API key, like the one given by OpenWeatherMap, with our API_URL as part of our Weather Page has several benefits:

Access Control: Strict control over who has access to our Weather Page's API is ensured by the use of API keys. This feature ensures that weather data retrieval and action execution are limited to authorized users only.

Usage Monitoring: We can track and examine how often the Weather Page's API is used thanks to API keys. This lets us adjust our service by monitoring use rates and the amount of requests received, for example.

Enhanced Security: The security of interactions between our application and the OpenWeatherMap API is strengthened in large part by the use of API keys. We protect against unwanted access and preserve the integrity of our data by using API keys to authenticate requests and encrypt data transmissions.

Rate Limiting: With the use of API keys, we may set rate limitations that stop us from receiving too many requests in a given amount of time from specific users or apps. This feature makes sure that our Weather Page's API doesn't become overloaded by too much traffic, so it stays responsive and available to all visitors.

We improved the user experience, maximized performance, and maintained the security and dependability of our service by adding these capabilities to the API integration for our Weather Page.

- Final Interface:

Figure 24

Logout page

- Description:

The logout page in PHP is typically used to end a user's current session and log them out of the system. After the logout process, the session data is erased, and the user is returned to the login page or another website.

we deal with two distinct forms. First, it performs several actions on the supplied model. The first form redirects the visitor to the login page, and the second form redirects the user to the home page.

On the logout page, the user's current privacy information is removed. Session data may include the user's name, permissions, and other session-specific information. This data is erased by either clearing the usage to utilize the private data or resetting it to its default value, after eliminating the session data, the session must be properly terminated. This can be accomplished by executing the'session_destroy()' method, which deletes and flushes all session data.

a condition has been added if (isset($_POST['submit']). This means that this condition is met. If we click inside this condition, the steps to log out of the session and redirect to the login page are performed.

Added elseif condition (isset($_POST['submit1'])) It should be noted that logging out does not result in an error condition. If an error occurs during logout, the error message will be saved in the $error variable and will be presented to the user on the website. This is accomplished by using an echo code to send the error message.

The code in the second section is formatted by clicking the "Submit 1" button on the checkout page. When this button is pressed, the user is directed to the "home.php" page via the header() function, which can redirect the version to the URL being searched for. The Exit() command is then used to terminate the execution of any other code on the page and prevent any other code from being executed.

- Final Interface:

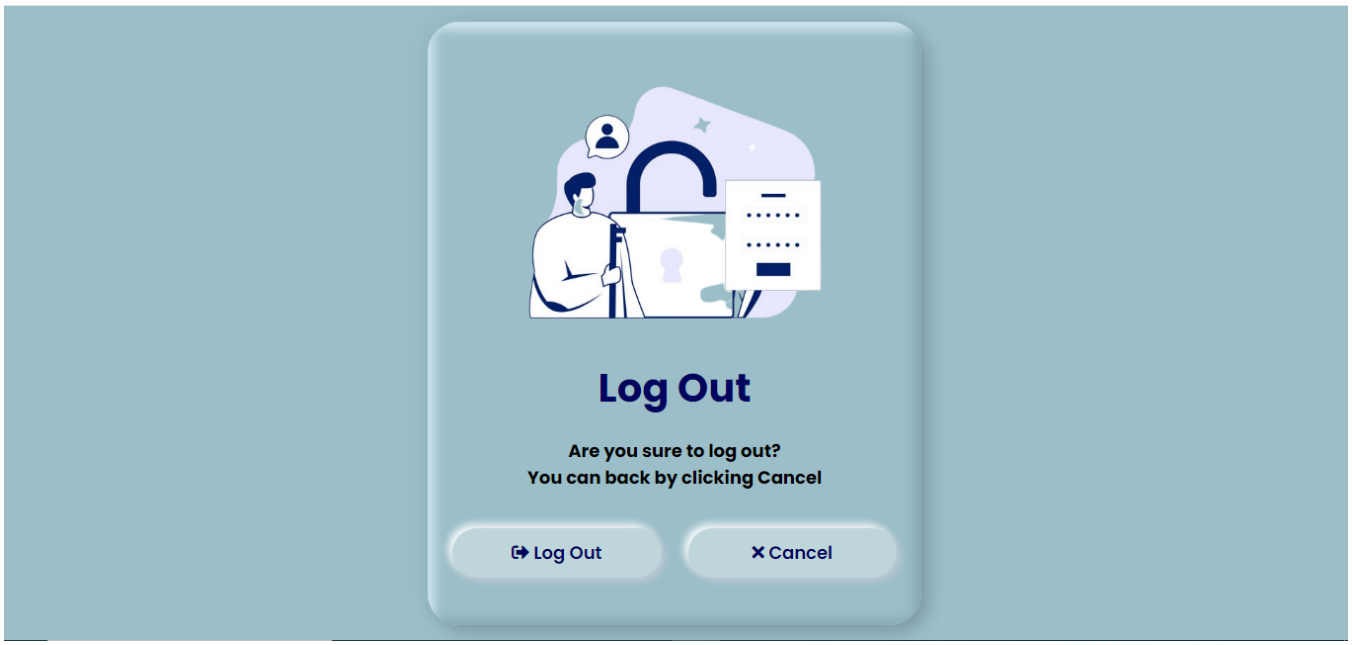

Figure 25

Conclusion

PHP is a popular server-side scripting language that is essential to utilize for web development due to its simplicity, scalability, and flexibility. We gained many benefits and knowledge from using it on our website and learned how important it is to use it on any website. If you want a dynamic website that is free, simple to use, easily integrated with a database, and has good performance, we strongly suggest utilizing PHP.

Fatema Hussain Habib – 20192408	**php codes** (log in, sign up, reset password, weather 1, weather 2), **report** (log in, sign up, reset password), introduction
Manar Eyad - 202006306	**php codes & report** (home, settings), conclusion
Naba Yahya - 20193627	**php codes & report** (log out)
Mariam Jawad - 202005915	**php codes** (weather 2), **report** (weather 1, weather 2)

References

1. Astari S, “What Is PHP: Understanding the Scripting Language,” Hostinger Tutorials, Mar. 01, 2024. [Online]. Available: https://www.hostinger.com/tutorials/what-is-php/
2. "PHP: Password Hashing – Manual," php.net [Online]. Available: https://www.php.net/manual/en/faq.passwords.php#faq.passwords.salt
3. "PHP’s md5() Function: Understanding Hashes," CloudDevs. [Online]. Available: https://clouddevs.com/php/md5-function/
4. "PHP: session_start – Manual," php.net. [Online]. Available: https://www.php.net/manual/en/function.session-start.php
5. Postman API Platform. "API Authentication." Accessed May 16, 2024. [Online]. Available: https://www.postman.com/api-platform/api-authentication/
6. API Dog. "A Comprehensive Guide to API Keys." Accessed May 16, 2024. [Online]. Available: https://apidog.com/blog/a-comprehensive-guide-to-api-keys/
7. F5. "API Security Best Practices." Accessed May 16, 2024. [Online]. Available: https://www.f5.com/pdf/ebooks/api-security-best-practices.pdf
8. OpenWeatherMap. "OpenWeatherMap Guide." Accessed May 16, 2024. [Online]. Available: https://openweathermap.org/guide
9. PHP: header - Manual.” ," php.net. [Online]. Available: https://www.php.net/manual/en/function.header.php
10. PHP: Basic usage - Manual.” ," php.net. [Online]. Available: https://www.php.net/manual/en/session.examples.basic.php

Source Code

Config Page:

```
<?php

$servername = "localhost";
$username = "rootp";
$password = "projectss";
$dbname = "project_345";

?>
```

Sign-up Page:

```
<?php

require_once "config.php";

function successAlter($bold, $message, $linkText, $url) {

   echo '<div class="popup" style="position: fixed; top: 50%; left: 50%; transform: translate(-50%, -50%); z-index: 9999;">

   <div class="alert alert-success fade in" style="background-color: #28a745; border: 2px solid #28a745; color: #000; padding: 15px; border-radius: 5px; width: 500px; text-align: center;">

        <a href="#" class="close" data-dismiss="alert" aria-label="close" style="position: absolute; top: 1px; left: 12px; font-size: 28px; color: #fff; text-decoration: none;">&times;</a><br>

        <i class="fa fa-check-circle" style="font-size: 80px; color: #fff; margin-bottom: 30px;"></i><br>

        <div class="model-content" style="background:#fff; padding:10px;width: 500px; margin-left: -17px; margin-bottom: -19px; border-radius: 0 0 5px 5px; border: 3px solid #fff;"><br>

        <strong style="font-weight: bold; font-size: 20px;">'.$bold.'</strong><br>'.$message.' <br>

        <button style="border-radius:18px; margin-top: 20px; margin-bottom: 10px; padding: 5px;  background:#28a745; border: none; outline: none; width: 100px; box-shadow: inset 4px 4px 8px #34D058, inset -4px -4px 8px #34D058; box-shadow:4px 4px 8px rgb(95, 171, 115), -4px -4px 8px rgb(88, 186, 79);">

        <a href="'.$url.'" style="text-decoration: none; padding:5px; font-size:16px; color: #fff;">'.$linkText.'</a></button>

        </div>  </div>  </div> </div>'; }

function generateRandomString($length = 6) {

   $Random = '%ABCDEFGHIJKLMNOPQRSTUVWXYZ_0123456789*abcdefghijklmnopqrstuvwxyz';

   $charactersLength = strlen($Random);

   $randomString = '';

   for ($i = 0; $i < $length; $i++) {

      $randomString .= $Random[rand(0, $charactersLength - 1)];  }

   return $randomString; }

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

   try {

      $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

      $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

      $user_name = $_POST['user_name'];

      $email = $_POST['email'];

      $DayOfBirth = $_POST['DayOfBirth'];

      $password = $_POST['password'];

      $c_password = $_POST['c_password'];

      if ($password != $c_password) {

         echo "<script>alert('Your Password does not match'); window.location = 'signup.php';</script>";  }
```

```
        $salt = generateRandomString();
        $password_md5 = md5($password. $salt);
        $sql = "SELECT COUNT(email) AS num FROM projects WHERE email = :email";
        $stmt = $conn->prepare($sql);
        $stmt->bindValue(':email', $email);
        $stmt->execute();
        $row = $stmt->fetch(PDO::FETCH_ASSOC);

        $sql = "SELECT COUNT(Username) AS nam FROM projects WHERE Username = :Username";
        $stmt = $conn->prepare($sql);
        $stmt->bindValue(':Username', $user_name);
        $stmt->execute();
        $raw1 = $stmt->fetch(PDO::FETCH_ASSOC);

if ($row['num'] > 0) {
            echo "<script>alert('Email already exists'); window.location = 'signup.php';</script>";
 }     elseif ($raw1['nam'] > 0) {
            echo "<script>alert('Username already exists'); window.location = 'signup.php';</script>";
 } else {
            $stmt = $conn->prepare("INSERT INTO projects (Username, Email, DayOfBirths, Passwords, Confirm_Password,Salts)
              VALUES(:Username, :Email, :DayOfBirths, :Passwords, :Confirm_Password, :Salts)"
);

            $stmt->bindValue(':Username', $user_name);
            $stmt->bindValue(':Email', $email);
            $stmt->bindValue(':DayOfBirths', $DayOfBirth);
            $stmt->bindValue(':Passwords', $password_md5);
            $stmt->bindValue(':Confirm_Password', $password_md5);
            $stmt->bindValue(':Salts', $salt);

            if ($stmt->execute()) {
               successAlter("Successfully!","Create an account is done","Log in",  "login.php");   } else {
               echo "<script>alert('Invalid Create Account');</script>";
 } }
} catch(PDOException $e) {
        $error = "Error: " . $e->getMessage();
        echo '<script>alert("' . $error . '");</script>'; }
}
?>
```

Log in Page:

```php
<?php
session_start();
require_once "config.php";
if (isset($_POST['submit'])) {
    try {
        $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
        $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $user_name = $_POST['user_name'];
        $password = $_POST['password'];
        $stmt = $conn->prepare("SELECT * FROM projects WHERE Username = :Username");
        $stmt->bindParam(':Username', $user_name);
        $stmt->execute();
        $check = $stmt->fetch(PDO::FETCH_ASSOC);

        if ($check) {
            $s_password = $check['Passwords'];
            $s_salt = $check['Salts'];
            $password_md5 = md5($password . $s_salt);
            if ($s_password === $password_md5) {
                // Username and password are correct
                header("Location: home.php");
            } else {
                // Password is incorrect
                $_SESSION['message'] = "Incorrect password!";
                echo '<script>alert("' . $_SESSION['message'] . '");</script>';
            }
        } else {
            // Username does not exist
            $_SESSION['message'] = "Username not found!";
            echo '<script>alert("' . $_SESSION['message'] . '");</script>';
        }
    } catch(PDOException $e) {
        $error = "Error: " . $e->getMessage();
        echo "An error occurred: $error";
    }
}
?>
```

Reset Password:

```
<?php
require_once 'config.php';
$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

function generateRandomString($length = 6){
    $characters = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
    $randomString = '';
    $charactersLength = strlen($characters);
    for ($i = 0; $i < $length; $i++) {
        $randomString .= $characters[rand(0, $charactersLength - 1)];
    }
 return $randomString;
}

if (isset($_POST['send_pass'])) {
    try {
        $newPass = $_POST['password'];
        $confirmPass = $_POST['c_password'];
        $email = $_POST['email'];
        $query = $conn->prepare('SELECT * FROM projects WHERE email = ?');
        $query->execute([$email]);
        $row = $query->fetch();

        if ($row) {
            if ($newPass != $confirmPass) {
            echo "<script>alert('Your Passwords does not match');window.location = 'reset_password.php';</script>";
}
        $salt = generateRandomString();
        $password_md5 = md5($newPass . $salt);
        $Query = $conn->prepare('UPDATE projects SET Passwords = ?, Confirm_Password = ?, Salts = ? WHERE Email = ?');
        $Query->execute([$password_md5, $password_md5, $salt, $email]);
```

```
 function successAlter($bold, $message, $linkText, $url)  {

      echo '<div class="popup" style="position: fixed; top: 50%; left: 50%; transform: translate(-50%, -50%); z-index: 9999;">

        <div class="alert alert-success fade in" style="background-color: #28a745; border: 2px solid #28a745; color: #000; padding: 15px; border-radius: 5px; width: 500px; text-align: center;">

            <a href="reset_password.php" class="close" data-dismiss="alert" aria-label="close" style="position: absolute;  top: 1px; left: 12px; font-size: 28px; color: #fff; text-decoration: none;">&times;</a><br>

            <div class="model-content" style="background:#fff; padding:10px;width: 500px; margin-left: -17px; margin-bottom: -19px; border-radius: 0 0 5px 5px; border: 3px solid #fff;"><br>

            <strong style="font-weight: bold; font-size: 20px;">'.$bold.'</strong><br>'.$message.' <br>

            <button style="border-radius:18px; margin-top: 20px; margin-bottom: 10px; padding: 5px; background:#28a745;   border: none; outline: none; width: 100px; box-shadow: inset 4px 4px 8px #34D058, inset -4px -4px 8px #34D058; box-shadow:4px 4px 8px rgb(95, 171, 115), -4px -4px 8px rgb(88, 186, 79);">

            <a href="'.$url.'" style="text-decoration: none; padding:5px; font-size:16px; color: #fff;">'.$linkText.'</a></button>

            </div>

            </div>

        </div>

      </div>';

      }

      successAlter("Successfully!","Reset your Password","Log in",  "login.php");

 } else {

      echo "<script>alert('Email does not exist'); window.location = 'reset_password.php';</script>";

      }

      }catch(PDOException $e) {

        $error = "Error: " . $e->getMessage();

        echo '<script>alert("' . $error . '");</script>';

      }

}

?>
```

Settings Page:

```
<?php
if (isset($_POST['fontSize']) && isset($_POST['fontColor']) && isset($_POST['backColor'])) {

    $expireTime=time()+60+60*24;
    setcookie('fontSize', $_POST['fontSize'], $expireTime);
    setcookie('fontColor', $_POST['fontColor'], $expireTime);
    setcookie('backColor', $_POST['backColor'], $expireTime);

    $msg = "<script>alert('Your settings have been saved! You can now go to the home page to see
the results'); window.location = 'home.php';</script>";
}
?>
```

```
  <?php
    if(isset($_POST['submit'])){
        echo $msg;
```

Home Page:

```
    <style>
     *,header, body, .container,footer, #second, .service, .paragraph, .ourService, .title,
.container .box, .container .box .boxContent
     {

    <?php

    if (isset($_COOKIE['fontSize'])) {
        echo "* { font-size: " . htmlentities($_COOKIE['fontSize']) . "; }";
    }
    else "\t\tfont-size: large;";
    if (isset($_COOKIE['fontColor'])) {
        echo "* { color: #" . htmlentities($_COOKIE['fontColor']) . "; }";
    }
    else "\t\tcolor: #333;";
    if (isset($_COOKIE['backColor'])) {
        echo "body { background: #" . htmlentities($_COOKIE['backColor']) . "; }";
    }
    else "\t\tbackground: #fff;";

  ?> }
</style>

    ?>
```

Weather 1 Page:

```
<select name="city" id="city" required>
            <option value="">Select a city</option>
            <?php
            // array of cities
            $cities = ['New York', 'Dubai', 'Dammam', 'Tokyo', 'Cairo','London','Washington','Rome',
               'Riyadh','Muscat','Kuwait City','Florence','Fuji','Paris','Doha','Rabat','Baghdad','Berlin','Moscow','Manama','Ottawa'];
            sort($cities);
            $currentLetter = '';

            foreach ($cities as $city) {
               $firstLetter = strtoupper(substr($city, 0, 1));

               // Check if the current letter is different from the previous letter
               if ($currentLetter != $firstLetter) {
                  // Add a dividing line with the alphabet letter
                  echo "<option disabled style='background: #B8CFD4; padding-left:2px; font-size:17px; color:#000;'> $firstLetter </option>";
                  $currentLetter = $firstLetter;
               }
               echo "<option value='$city' name='city' >$city</option>";
            }
            ?>
         </select>
```

Weather 2 Page:

```
<!DOCTYPE html>
<html>
<head>
    <title>Weather Forecast</title>
    <link rel="stylesheet" type="text/css" href="css/style5.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.4.0/css/bootstrap.min.css">
</head>
<body>
    <div class="container">
    <h2><?php echo $_POST['city']; ?> - Weather Forecast</h2>
        <!-- PHP -->
        <?php
        if (isset($_POST['city'])) {
            $city = $_POST['city'];
            $api_key = '4fd753e86220f1947a861721bea1ccce';
            $api_url = "https://api.openweathermap.org/data/2.5/forecast?q={$city}&appid={$api_key}&units=metric";

            $json_data = file_get_contents($api_url);
            $weather_data = json_decode($json_data, true);


          if ($weather_data['cod'] === '200') {
                $forecast_data = $weather_data['list'];

                $days = array();

                foreach ($forecast_data as $forecast) {
                    $date = date('l', $forecast['dt']);

                    if (!isset($days[$date])) {
                        $days[$date] = $forecast;
                    }
                }
```

```
        echo '<div class="card-gallery">';

        foreach ($days as $day => $forecast) {
            $weather_description = $forecast['weather'][0]['description'];
            $emoji_icon = '';


            if (strpos($weather_description, 'cloud') !== false) {
                $emoji_icon = '☁️';
            } elseif (strpos($weather_description, 'rain') !== false) {
                $emoji_icon = '🌧️';
            } elseif (strpos($weather_description, 'clear') !== false) {
                $emoji_icon = '☀️';
            } elseif (strpos($weather_description, 'snow') !== false) {
                $emoji_icon = '❄️';
            } else {
                $emoji_icon = '❓';
            }

$temperature = $forecast['main']['temp'];
            $min_temperature = $forecast['main']['temp_min'];
            $max_temperature = $forecast['main']['temp_max'];
            $humidity = $forecast['main']['humidity'];
            $wind_speed = $forecast['wind']['speed'];

            echo '<div class="card" onclick="handleCardClick(this)">';
            echo '<div class="card-header">' . $day . '</div>';
            echo '<div class="weather-icon">' . $emoji_icon . '<div class="weather-temp">';
            echo'<span class="weather-te">' . $min_temperature . '°C<br>';
            echo'<span class="weather-te">' . $max_temperature . '°C<br>';
            echo '</div>';
            echo '</div><br>';
```

```
            echo '<div class="weather-description">' . $weather_description . '</div>'.'<br>';
            echo '<div class="weather-info">';
            echo'<span class="weather-label">Temperature:</span>' . $temperature . '°C<br>';
            echo'<span class="weather-label">Humidity:</span> ' . $humidity . '%<br>';
            echo'<span class="weather-label">Wind Speed:</span> ' . $wind_speed . ' m/s<br>';
            echo '</div>';
            echo '</div>';
          }
           echo '</div>';
        } else {
          echo '<div class="card">Error retrieving weather data. Please try again.</div>';
        }
    }
    ?>
        <div class="container-btn">
        <button name="submit" class="btn"><a href="weather1.php">Back</a></button>
        </div>
</div>
```

Logout code:

```
<?php
    session_start();
    require_once "config.php";

    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    if (isset($_POST['Submit'])) {
        Try {
            $_SESSION = array();
            session_destroy();
            header("Website: login.php");
            exit();
        } catch (PDOException $e) {
            $error = "error:" . $e->getMessage();
            echo "An error occurred: $error";
        }
    } elseif (isset($_POST['submit1'])) {
        header("Location: home.php");
        exit();
    }
?>
```