George Zalokostas

gzalos6@gmail.com

Application portal  
PHP PROJECT

Documentation

**Contents**

[Setting up the environment 2](#_Toc78134214)

[index.php 3](#_Toc78134215)

[admin.php 3](#_Toc78134216)

[create\_user.php 4](#_Toc78134217)

[delete.php 4](#_Toc78134218)

[update.php 5](#_Toc78134219)

[user.php 5](#_Toc78134220)

[create\_application.php 6](#_Toc78134221)

[validation.php 7](#_Toc78134222)

[functions.php 7](#_Toc78134223)

# Setting up the environment

* This project was made using the Atom Editor found [here](https://atom.io/)
* Also in order to use a web server, XAMPP was used for the Apache server. It can be found [here](https://www.apachefriends.org/index.html)
* PHP Version : 7.4.21
* Lastly, in order to use the email function in our localhost project, we have to set the SMTP settings. I used mine through gmail:

1. SMTP username: Your Gmail address.
2. SMTP password: Your Gmail password.
3. SMTP server address: smtp.gmail.com.
4. Gmail SMTP port (TLS): 587.
5. SMTP port (SSL): 465.
6. SMTP TLS/SSL required: yes.

On XAMPP, you have to edit the php.ini files with our custom settings in order to send the email successfully.

# index.php

This is our starting directory file. Here we can see a form for our email and password. One can enter with his credentials and be redirected to admin.php or user.php based on his status. There is a global variable $\_SESSION[‘email’] which works as our username, and a $\_SESSION["logintype"] of either ‘admin’ or ‘user’. These credentials can let us browse admin.php or user.php without any issues such as changing the URL from user to admin and enter a page which we shouldn’t access normally.

Index.php is connected to the database in the dbconnect.php file through include() and users can enter the portal.

# admin.php

As an admin, there is a banner on top of the page showing us our email, and two buttons called Logout and Create User.

* Logout redirects us to index.php
* Create user redirectes us to create\_user.php

On the body, we can see a table with the column names:

* Id : This is the id of the user/admin in the database
* First: The first name of the user/admin
* Last: The last name of the user/admin
* Type: The type which is either admin or user
* Actions: There are two buttons called Update and Delete that can modify the row.

# create\_user.php

This is the page that the admin can create a new user. There is a form with the name , email , password and the type of user in a dropdown.

In order to submit the form, any inputs shouldn’t be left blank, the email should contain @ and . in the string ( it’s validated through the built-in function filter\_var($email, FILTER\_VALIDATE\_EMAIL)) and lastly, the password and re-enter password fields should match. Otherwise the page is refreshed, the fields are blank and a message prompts the admin to re-enter the correct values.

# delete.php

This is a php only file that you can’t access through the portal alone, where it executes a query that DELETES FROM users WHERE id=$id, so each delete button in admin.php can delete the row’s user. The page is refreshed and the table show the new data.

# update.php

This file is similar to create\_user.php , only this time the fields have values.

Upon pressing update on a user, a query runs that SELECTS \* from that user and each field’s value is replaced on that form. The admin can modify the name , the email, the password and the type of user (making him admin if he wants to) .

Again If any fields are left blank or the passwords don’t match, the page is refreshed prompting the admin to enter acceptable values.

# user.php

On this file, there is a header that shows the user’s email , and two buttons:

* Logout: Redirects us to index.php.
* Submit request: Redirects us to create\_application.php

On the body, there is a table with the date that a vacation form was submitted, another column called dates requested that shows the dates the current user asks to get his vacation, the number of days requested and the status that’s pending by default.

The table is displayed in **descending order** of the Date submitted column.

# create\_application.php

This is the page that the user can ask for his vacation dates. There are two input forms of type “Date” that the user can choose From and To, and also a “Reason” text area, describing his reason of submitting the request.

In case of leaving the text area blank, of the current date being **less equal** to today’s date, or the “Date To” field is chronologically less than “Date From”, they are considered wrong input and the page gets refreshed printing an error message.

In case of correct input, the values are submitted to the database’s table “applications”, and also an email is sent to the supervisor following the body:

*“ Dear supervisor, employee {user} requested for some time off, starting on*

*{vacation\_start} and ending on {vacation\_end}, stating the reason:*

*{reason}*

*Click on one of the below links to approve or reject the application:*

*{approve\_link} - {reject\_link} ”*

The supervisor can then choose one of the two links, e.g

**Accept:** http://localhost:8080/project/validation.php?action=accept&uniqid=60fd9477971e4  
**Decline:** http://localhost:8080/project/validation.php?action=decline&uniqid=60fd8132971e4

# validation.php

This script only page gets the action and the id from the URL

For example:

$action = $\_GET['action']; // $action value = accepted

$uniqid = $\_GET['uniqid']; // $uniqid value = fd8132971e4

A query runs that fetches the row with the current uniqid value, and updating the status of pending to either accepted or rejected. Lastly, another email is sent to the user, notifying him that his request is either accepted or rejected on the X submission date.

# functions.php

This script only page contains four functions:

* dateDiffInDays
* createGuid
* sendEmailToSupervisor
* sendEmailToUser

**dateDiffInDays** : Takes as two dates as arguments (e.g dateFrom, dateTo), and calculating their difference using a built-in function called strtotime. It subtracts date2-date1 , and their difference is rounded. If it’s a positive value, it means that the dateTo is chronologically ahead of dateFrom. Otherwise it’s the same day, or dateTo is days behind dateFrom. We can easily input the dates the users chooses, and find if they are valid ( not the same day, and dateTo > dateFrom).

**createguid:** it creates a has string using random numbers of different ceiling and floor values, e.g:

mt\_rand(0, 65535)

mt\_rand(16384, 20479)

mt\_rand(32768, 49151)

thus creating a “pseudo-hash” string that we can use as the id of the user’s vacation request.

**sendEmailToSupervisor:** creating the subject, the header, and the body strings that get passed in the mail() function. It also has two variables called $accept and $decline that hold the URL that the supervisor can click. e.g:

$accept = 'http://localhost:8080/project/validation.php?action=accept&uniqid= fd8132971e4;

$decline = 'http://localhost:8080/project/validation.php?action=accept&uniqid= fd8132971e4;

**sendEmailToUser**: Again it has the strings that gets passed as arguments to the function mail() and an email is sent to the user notifying him that the supervisor has made his choice.