PHP Developer Assignment

The source code is organized in 9 different files:

login.html

It contains an html form prompting the user to log in. Once the form is submitted, the data is sent to the “login_check.php” file using the post method.

login_check.php

In this file, a connection with database gets established and a query is made, using the user’s email in order to find out the user’s type (admin/employee).

If the type is equal to admin, then after the admin has logged in, he is taken to the user management page where a clickable table of all existing users, is available. In order to create the table, a select query was used, which returned the user’s first name, last name, email and password as different columns of the html table. Each row of the table is clickable and each row represents a different user. When the admin clicks on a row, a modal update-form is displayed, pre-populated with the user’s info (except password fields). The form is pre-populated by making an Ajax request. The Ajax request is received by the “fetch_users.php” file that will be described later. Once the update form is submitted, the data is sent to the “update_user_action.php” file. On top of the table with the users’ info, there is a create button to create a new user. When the admin clicks on the button, a modal create-form is displayed, providing the necessary fields (first name, last name etc.) in order to create a new user. The data is sent to the “create_user_action.php” file. In order to avoid having duplicate email entries, an Ajax call is made to the “check_email.php” file that alerts a warning message if email already exists in db.

If the type of the user is equal to employee, then after the user has logged in, he sees a table of all past applications for vacation. On top of the table there is a submit request button, that opens a modal form to create a new application. Once the form is submitted, the data is sent to the “vac_request.php” file.

create_user_action.php

Establishes connection with database. Receives the data from the html form and inserts them into the db.

check_email.php

Establishes connection with db. Receives the Ajax request. A select query is made, in order to find out if the received email address already exists on db.

fetch_users.php

Establishes connection with db. Receives the Ajax request. A select query is made, in order to find all info about the user with the given email.

update_user_action.php

Establishes connection with database. Receives the data from the html form and updates accordingly the db.

vac_action.php

Establishes connection with database. Receives the data from the html form and inserts them into the db. A query is made to find the admin's email from db and using session variables, a query is made to find the employee's (that requested vacation) email. Then an email is sent from the user to the admin using the php mail() function. The body of the email is an html document containing a form. Inside the form are 4 hidden input fields, defining the userID, the date that the application was submitted, and the first day as well as the day of the vacation. There are also two submit inputs with the same html name but different values. The one is the Approve link and the other one is the Reject link. Once one of the two submit's is clicked the data is sent to the "reply.php" file.

reply.php

Establishes connection with database. Receives the data from the html form and updates the application status on the db. In a same manner as before sends a email from the admin to the user, notifying him of admin's decision. Window gets automatically closed after running the php code.

p_style.css

Style of the html elements.

<u>Assumptions made:</u>

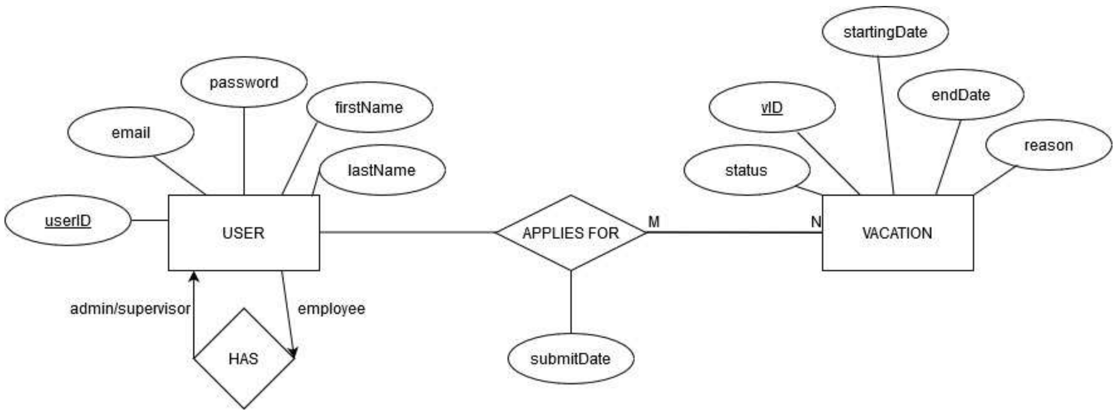
1. Only one admin can exist at any time. (*Should have included form control***)

2. Employees are allowed to ask for vacation on the same dates.

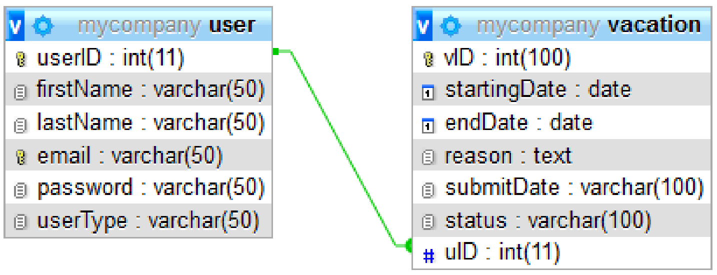
3. The same employee is allowed to submit multiple requests, even if previous requests are

   still on "pending" mode.

*** Controls that should have been included as well: password safety control, login control.*



ER Model



mycompany user
userID : int(11)
firstName : varchar(50)
lastName : varchar(50)
email : varchar(50)
password : varchar(50)
userType : varchar(50)

mycompany vacation
vID : int(100)
startingDate : date
endDate : date
reason : text
submitDate : varchar(100)
status : varchar(100)
uID : int(11)

Relational Model