ISE 311 Network Programming Project

Documentation Form

Prepared by:

Fahrettin Baştürk

Berk Utkan Cinek

Nuri Can Özer

Professor:

Eren Aykın

ABSTRACT

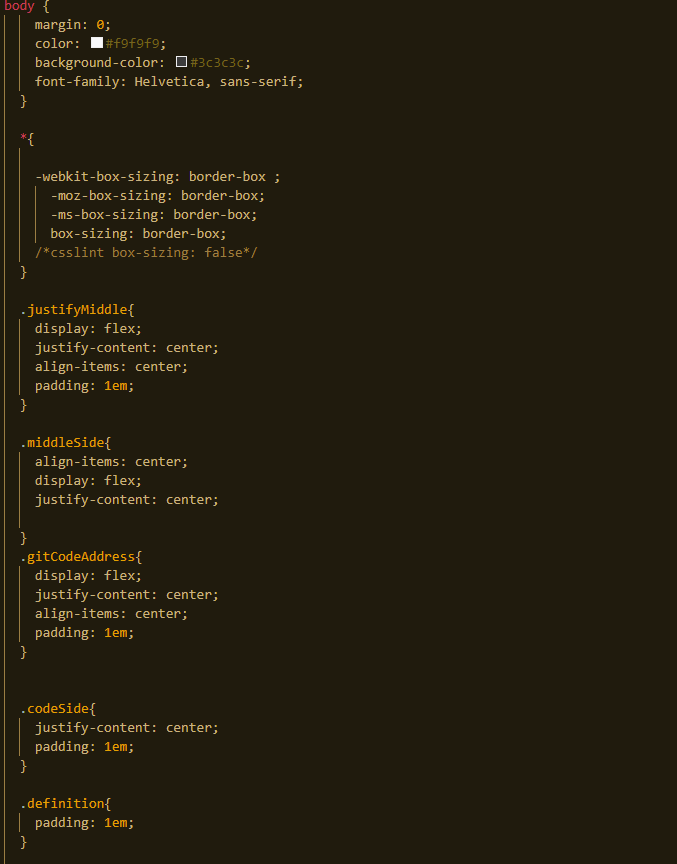
This auxiliary usage documentation, which we wrote considering the minimum requirements in line with the creation of the project, will be transferred respectively to the style of the project, creating data with php, transferring data to the database, deleting data and manipulating and updating the existing data. In these sections, we will focus on the most important issues and show the working status of the project. We will examine the codes in the document that we have made and the points mentioned above, in the php, css, javascript topics.

1. CSS part of the code:



As you can see from the figure we have started off by making a form named HomeworkForm

and making some labels to take input from the user



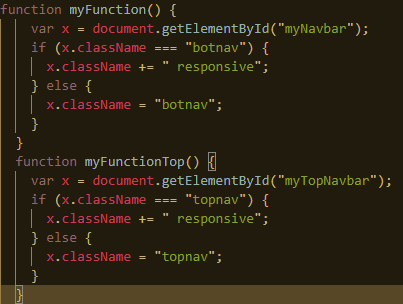
which is then followed by a long piece of code for general design and look of the website



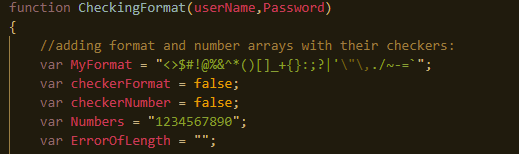
next we have some unordered lists to store data as you can see in the figure above but also there are special code

1. Javascript part of the code:

Moving on to the Javascript part of the code in the file JavascriptCode.js we have 2 simple function for detecting special cases for myNavbar and myTopNavbar which are are for navigating to the top and bottom of the page



Moving onto the next section which is in the form.js we can see a standart start to a function named checkingformat which accepts userName and Password values along with some variable declarations

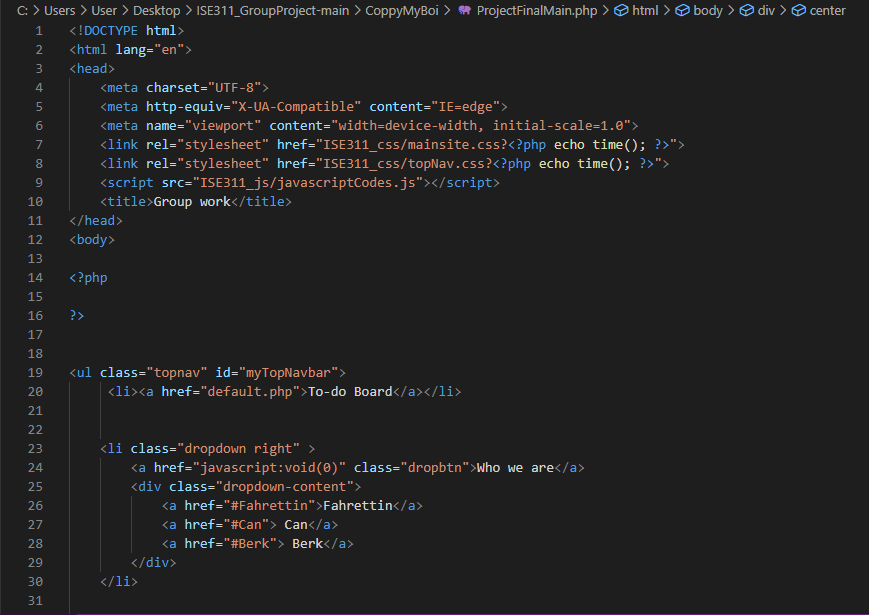


1. PHP Part of the Code:

Here we start PHP scripting with the HTML written inside the PHP file. In general, as seen in the first figure, we first linked the CSS and JavaScript files. When connecting php code with time(); We also added the function. The main purpose of this is because we want to show the Unix timezone as well.

Here we start PHP scripting with the HTML written inside the PHP file. In general, as seen in the first figure, we first linked the CSS and JavaScript files. When connecting php code with time(); We also added the function. The main purpose of this is because we want to show the Unix timezone as well.

We continued with PHP codes inside the body tag. First of all, we started the "To-Do list" menu on the navigation part with two list structures and the "Who Are We" structure on the right part. We made CSS see it by giving class and id names.



Session\_start() used to log on and off on a request. Since Fahrettin already has a server here, he entered his own server's information after session\_start()

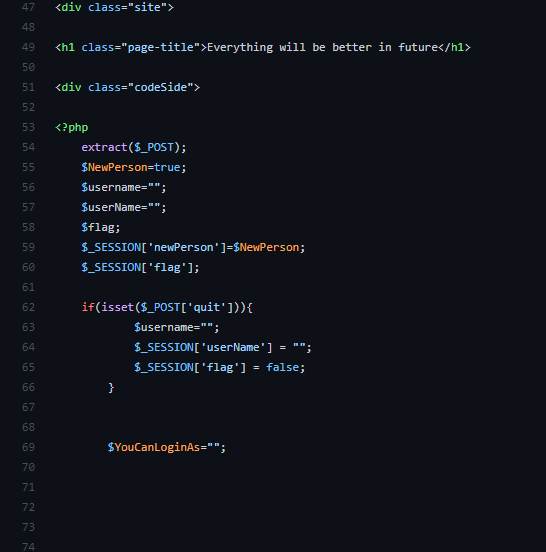


Here are the HTML and CSS tags for the To-Do Board and on the right to reach the names of the creators of this site.

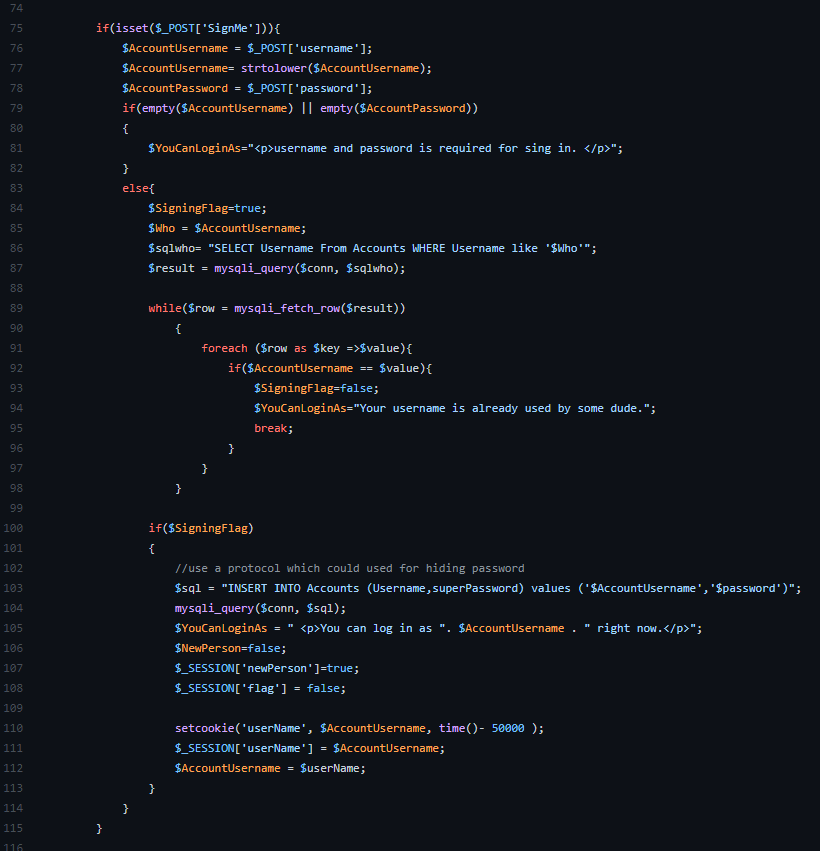


After an h1 header, extract() function is used to convert all names from POST to variables. We will provide the new person username and password with the Session() function. The purpose of doing this is to keep user information securely on the server during sessions.

After an h1 header, extract() function is used to convert all names from POST to variables. We will provide the new person username and password with the Session() function. The purpose of doing this is to keep user information securely on the server during sessions. Then, with the isset() function, we query whether the 'quit' variable is defined with the POST method.

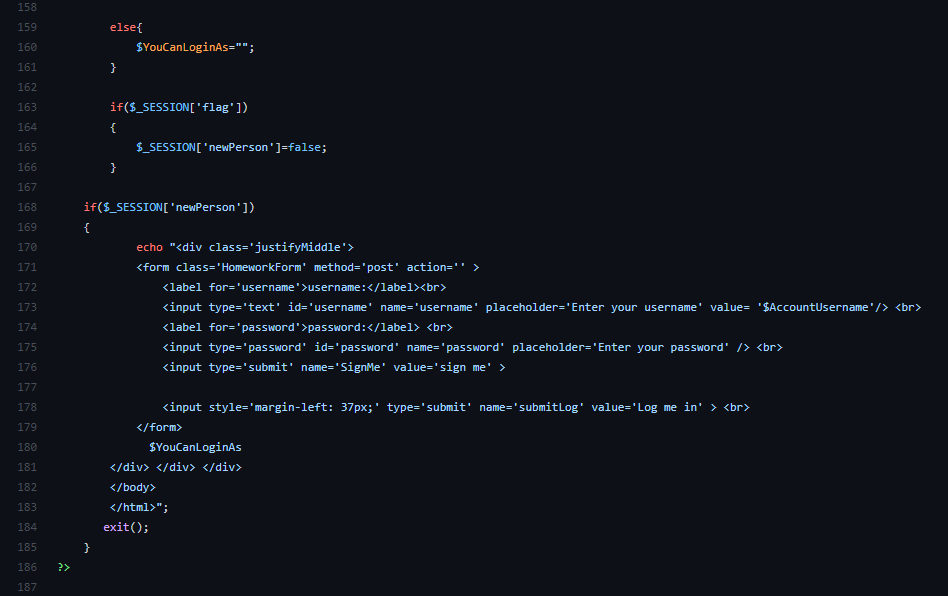


When we come here, a big if block welcomes us. The purpose of this if block is for user's sign operations. First, it checks whether the user has written password and username. If it is, the sign flag becomes true and it creates a query with the mysqli\_query() function. At the same time, with the mysqli\_fetch\_row() function, we checked whether the username is used by another user in the database by bringing each row from the database and returning it as an index. When we come here, a big if block welcomes us. The purpose of this if block is for user's sign operations. First, it checks whether the user has written password and username. If it is, the sign flag becomes true and it creates a query with the mysqli\_query() function. At the same time, with the mysqli\_fetch\_row() function, we checked whether the username is used by another user in the database by bringing each row from the database and returning it as an index. In the last string in this section, we have added the SigningFlag, if true, with INSERT and query to the database. We also set the information as cookies with the setcookie() function. We also used SESSION.

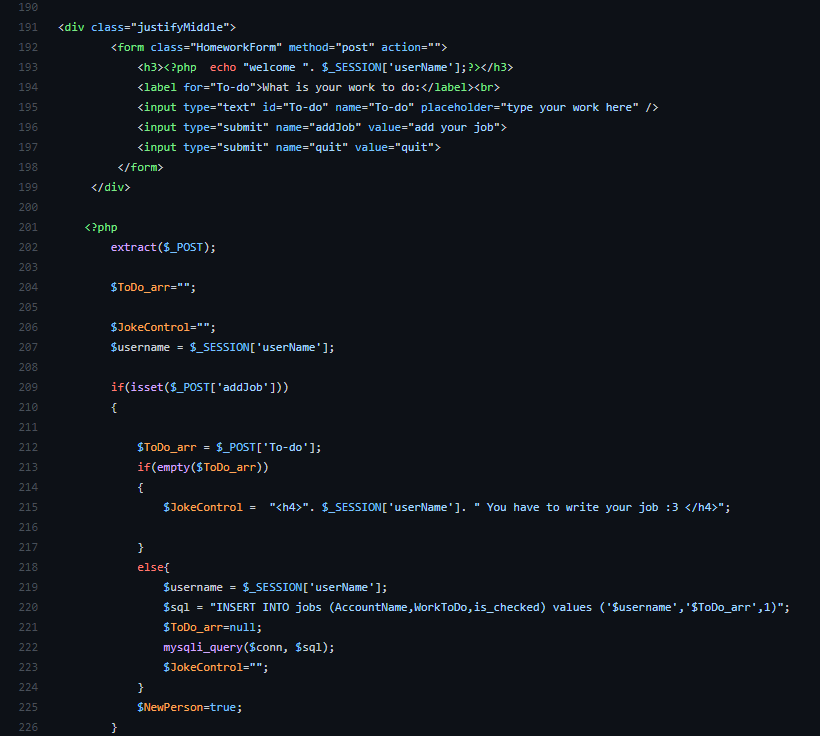


In fact, almost the same situations as in the previous part continue with the elseif part in this part. In this section, the login process is performed. First, we query the elseif section with the isset() function to see if the submitlog is defined. Then we gave a warning message that username and password should not be empty with if. Then we query the accountname that the user wrote, after finding the accountname with the same name with mysqli\_fetch\_row(), we checked whether the password is the same. With SubmitFlag, we removed the newperson status if it was correct, and gave a warning message if it was entered incorrectly.

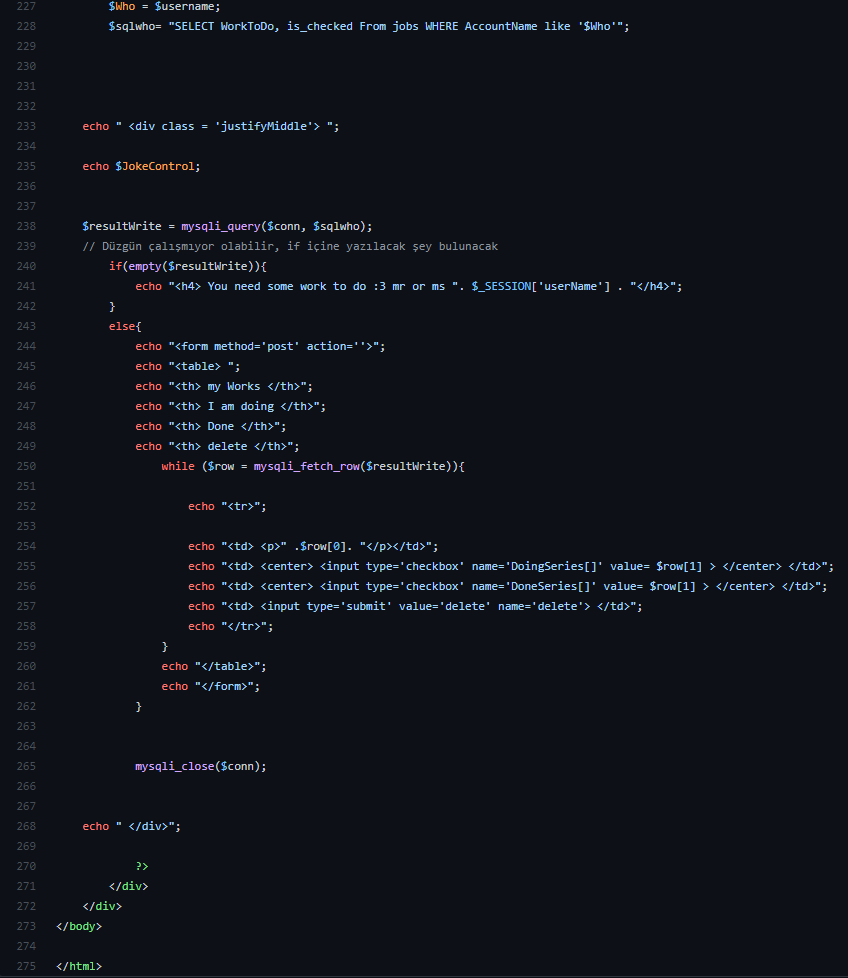


In the else part, the whole program goes back to the user login, sign in part. All of the labels here are written inside the echo

This part is where the To-Do functions start after the entries made. We made the desired To-Do form with options such as addJob, quit. For the "you need to write a job" part, which is a small but necessary warning, we checked the To-Do arr with the empty() function (whether it is empty). If it's not empty, we INSERT jobs with jobs.



In the last part, if a task has been written, the table form showing the "doing", "done", "delete" parts of the written tasks will be met. When the processes end, the connection is terminated with mysqli\_close().



**How It Is Working?**

We showed you how to register, log in with the created registration, and then add a task, respectively.

Graphical user interface, text, website

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated