Technical Design of Tailgating Project

EHRD 477

Jeffrey Blackard

2018

**Hosting:**

AWS EC2 t2 micro instance

Software Used:

Ubuntu (18.04.1)

Apache2 (2.4.29)

PHP7 (7.2.10-0)

MySQL DB (5.7.24-0) extension: MYSQLI

PhpMyAdmin

The website is hosted on a EC2 t2 micro instance on Amazon Web Services under the free tier. The instance is running Ubuntu 18.04.1. Software installed is Apache2, PHP7, MySQL Database and PhpMyAdmin. The one instance should be enough computing power for this proof of concept but it would need more power in the webserver to keep up with the demands in a production environment. There will be thought of implementing a second instance as well as a load balancer for testing during the presentation phase. The webserver has a public IP which has been linked to my personal domain through the course of this project which is maintained on Google Domains. Code is currently managed on GitHub (<https://github.com/concon2015/ehrd477tailgate>) to control versions as well as serve as a backup to roll back unsuccessful changes.

**Database:**

The database for the website is one database (registration) that has two tables (availablespots and users). The table “users” contains a field for the user ID, username, email, hashed password and the registered spot. The “availablespots” table has the list of spots that can be registered and uses the spots available query to cross check with the users table to find the available spots.

Database: registration

Table: users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Username | Email | Password(hashed) | Spot |
| 1 | Test | [test@test.com](mailto:test@test.com) | 16d7a4fca7442dda3ad93c9a726597e4 | A1 |

Table Creation:

CREATE TABLE `users` (

`id` int(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`username` varchar(100) NOT NULL,

`email` varchar(100) NOT NULL,

`password` varchar(100) NOT NULL,

`spot` varchar(50) NOT NULL,

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

Table:

|  |
| --- |
| Spots |
| A1 |
| A2 |
| A3 |
| A4 |

Table Creation:

CREATE TABLE `availablespots` (

`aspots` varchar(100) NOT NULL,

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

Queries:

Spots Available: (This query pulls all non-registered spots)

SELECT \*

FROM availablespots

WHERE aspots NOT IN (SELECT spot FROM users)

Selected Spot: (This query pulls the users spot for use on login page)

SELECT spot FROM `users` WHERE `username` = 'test'

Insert: (This query inserts session data from registration page to a new column in users)

INSERT INTO users (username, email, password, spot)

VALUES('$username', '$email', '$password' ,'$spot')";

Double Register: (This query doesn’t allow two users who loaded the website at the same time to register for the same spot)

SELECT 'spot' FROM `users` WHERE `spot` = '{$spot}’

**Web Pages:**

**index.php**



This serves as the main landing page for a logged in user. It starts a php session that allows data to be stored across web pages in the same session. It starts by checking if a username is set in session variable $\_SESSION[‘username’] and if that is not set it redirects to login.php. It also prints a welcome message, username, spot number and email for successfully logged on users. Lastly it includes a hyperlink to logout that will set $\_SESSION[‘logout’] = 1 which will unset the username variable previously mentioned and go back to the login screen.

**registration.php**



This page functions as the input for registration from the user. This begins by including server.php for database connection information as well as registration processing. The html on this page gives fields for each user input (username, email, password, password verification, and spot). The select field runs the available spots query and lists each option in a dropdown menu. Lastly there is a hyperlink for already registered users that will redirect to login.php.

**server.php**



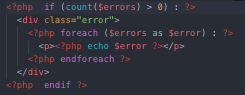
This page is the bulk of the website. At the start it begins the php session to allow variables to be carried to other pages. Following that there are declarations to each variable and session variables to be used as the target of the registration page. The variable $db contain the database connection function with the appropriate credentials. The code under “//registers the user” receives the data (protected from MOST techniques SQL injection by mysqli\_real\_escape\_string function) and validates that every form has input and that the passwords match. The page uses an md5 hash on the password to make sure that plain text passwords are not stored on the local server or easily decryptable. The page submits the data using the insert query, sets the session variables (username, spot, success) and redirects to index.php. The code below “//login user” functions very similar to the registration code in this document but handles the input on the login.php page and check credentials and sets the session variables.

**login.php**



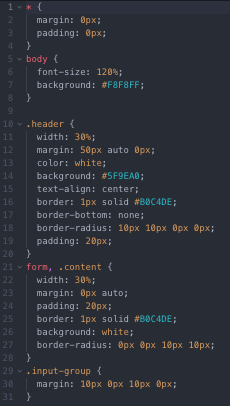
The login.php page has two input fields (username and password) and sends data via a post function to server.php to function as described above. There is also a hyperlink that says “Not yet a member?” that links to register.php. It also gets the design of the page from style.css.

**errors.php**



This code is included in login.php, server.php and registration.php and allows error handling and to check and make sure the form is filled out correctly as well as the database connections are successfully made.

**style.css**



This page allows the style of the website to be set and carry over to the multiple display pages. The code here also controls the width of the page as well as the hover feature of the images on the registration site.