Anonymous Posting Website

A mini project for the course "Technologies for Web and Social Media"

made by Tonko Bossen

Name: Tonko Bossen

Student mail: tbosse20@student.aau.dk

Student number: 20203031

This mini-project is a forum page for people to write about their thoughts. They consist only of text, to keep a simple mind aspect only. Each post is listed with a date and a place for a user name, but as the page is for anonymous users, it says all of them are anonymous. It is interactive with a contact form for visitors to get in contact. The website is made in HTML and CSS and uses JavaScript and Angular to implement client sites. jQuery is used for adaptations. PHP is in charge of the server site, with XAMPP working as server side. MySQL is used to manage the database system.

Text examples from: https://www.lipsum.com/

Github repository to client- and server files: https://github.com/tbosse20/webDevMiniProject

Setup the server and client site:

1. Download the following repository.

2. Download and install XAMPP (https://www.apachefriends.org/download.html).

3. Move repository folder "AnonymousPostingWebsite" into the folder "C:/xampp/htdocs".

4. Launch XAMPP and press "Start" under "Action" next to "Apache" and "MySQL".

5. Go to the url "localhost/CreateDatabase.php" to create the correct database ("Admin" next to "MySQL" allows securing the correct setup).

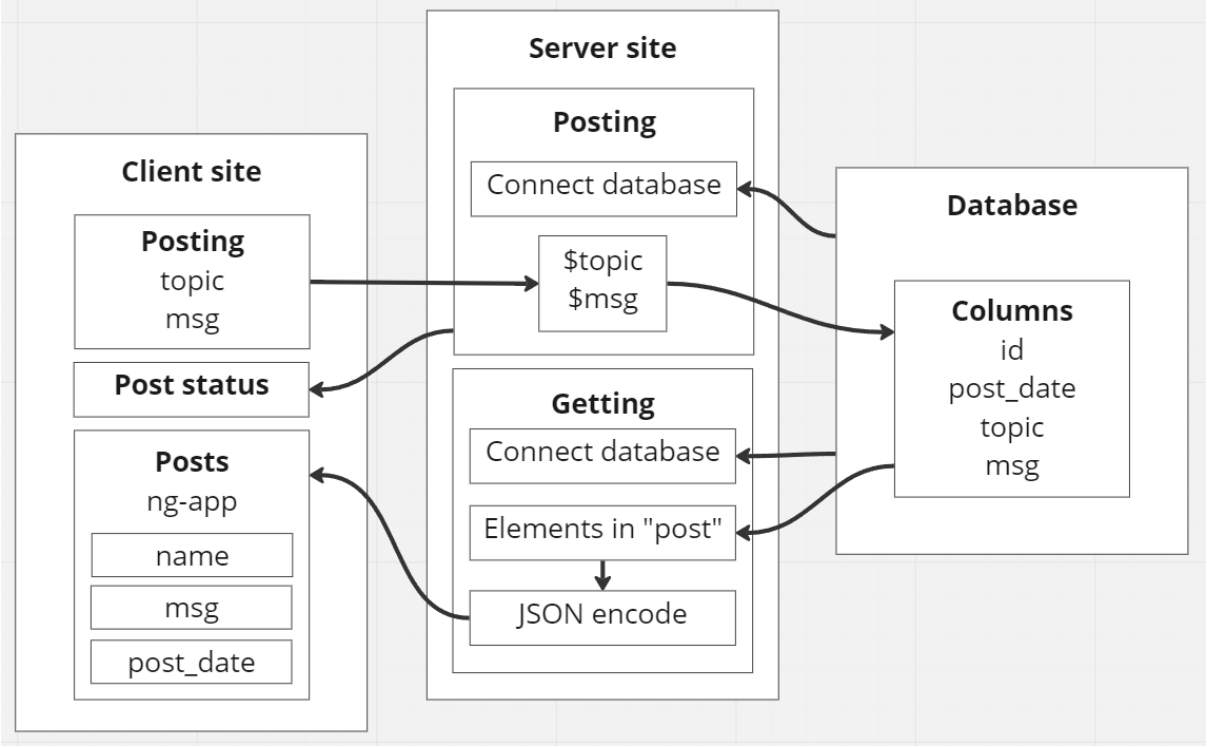
6. Go to the url "localhost/forum.php" and start posting!

Over all structure

The website is separated in the following list:

- Client site (“forum.php”)
    - Creating and posting a new text
    - Displaying previous posts by other people
    - Styling the page (“forum.css”,
- Server site
    - Creating the database table with correct columns (“CreateDatabase.php”)
    - Posting incoming texts to MySQL database (“postServer.php”)
    - Getting previous posted texts (“getData.php”)

Visualisation of the pipeline from client site, server site and to the database.



MySQL and JSON server version

The site is implemented using MySQL to meet the requirements, but is also implemented using a JSON file. It is possible to implement the JSON file into MySQL, but would not utilise the tables and columns that MySQL is good for. Therefore the version implemented for MySQL does not have the possibilities of comment. To enable the JSON version and comments, there are a few elements in the script that have to be changed. The reason for using a JSON file, is due to the proper way of doing it, without any requirements. The files for the adaption are also in the repository. This makes it possible to make a comment system too, and to remove the MySQL server. This makes it less costful, and easier to maintain. The report will mainly focus on the MySQL version, to explain the database. The elements in the script that are not discussed probably are used for the JSON version. The comment section in the “forum.php” file implemented as HTML and Angular has to be uncommented and two strings in the “forum.js” file, has to be changed. This is commented on in the script.

XAMPP MySQL database

The server site needs a server and a database to function. For this the program XAMPP is being used. A link to the download page is in the repositories readme file. XAMPP is a software that handles a server in Apache to execute PHP code and a database in MySQL. The website is connected by launching “localhost” in a web browser. This connects to the files in the folder “htdocs” inside of the XAMPP folder, where it is installed.
The MySQL database is separated into databases, tables, columns, and values. Each column is a different category for the table to contain value in. This program has the columns “id” being the given id of the post. “Post_date”, which is the date the post was submitted. “Topic” is the given topic or title of the user. They can’t write any text they would like here. “Msg” has the value of the message or text the post contains. Each of these columns is collected in the table “posts” in the database “forum_posts”. The setup of this table and columns is automatically made using the “CreateDatabase.php” script and is included in the “ConnectDatabase.php” script. The tables are however only constructed when they are missing.

Posting using <forms> and PHP POST and GET

The website is built using HTML syntax. With the necessary “head” and “body” tags and linking of different files such as JavaScript and CSS. The file is a PHP-file, to enable forms, connection to server and posting. The posting section is made using a “form”-tag. The form is a post method, and has the incoming text, and the submit button as sub-elements. As the site is anonymous, there are no requirements for a login with any user names. The post could get an adoption with a title or a custom username for each post, but to keep the design simple, it has been chosen only to have a text. The “form” element in HTML is used for sending or saving information on a website. The “action” attribute goes to the given url adresse, which in this case is the PHP file for sending the text information to the MySQL server, depending on the “method” attribute, which is set to “POST”. Each element inside of the form is an “input”-, “textarea”-, or “button” tag. Textarea and button are as said, but input can be all kinds of input types such as a button, text, email, password, number, slider, etc. The attribute “name” in each element is used to post the value inside of each input into the PHP file in action. When the submit button is being pressed, the values are posted to the file “postServer.php”. When the submit button is pressed the AJAX code in the “forum.js” script is launched. This will pack the data from the form using their id and send the data to the server client script. This is the script that can be changed to “postJSON.php”, to enable comments.

The “postServer.php” file is used for connecting the MySQL database, packing the values posted in the form, sending them to the database, and proceeding back to the forum page. Firstly the connection of the database has to be established. This is done by giving the four values “host”, “user”, “password”, and “database” to the *mysqli_connect* function. Host is the domain, which is in this case “localhost”. User is which user with which connection restriction there is prohibited. This program uses “root”. Password is the given password, which is by default empty. Finally the desired database to connect to, this is using “forum_posts”. After a successful connection has been established, the script will pack the values posted in the form. Id is being set to zero every time, to. The date the post is being published is made in the file automatically by using the UTC (Coordinated Universal Time) value. As this is done fixed into a string, the value shown later is not in local time. The message is received by using *$_POST[“<name>”]*, where *<name>* is the desired element from the form to handle. To post the value in the textarea to the name “msg” is being used, as it matches. To send these values to the database, a command has to be packed, here named

$sql. This tells MySQL what to do, with what, and where. Here it is inserting the given values into the given columns, shown as *"INSERT INTO <table> (<columnsNames>) VALUES (<values>)"*. The connection to the database and the sql command is launched and the submission is complete. The variable $status is holding the status of success/error for later. To finish off, the established connection to the database is disconnected and the system continues back to the forum page url. The url is updated with the string "?status=$status" with the $status variable. This would allow the forum page to return the result of the post submission to the user. The value could be handled using the PHP method *$_GET["<name>"]* and print this on the forum page using *echo* in PHP. Instead AJAX is collecting the print out from the variable "response". This is handed into the HTML element with the id of "status".

SELECT from MySQL INTO Angular

To show the content in the database on the client site, PHP is used once more along with Angular. Angular is a JavaScript framework to help make adaptable webpages. It works by having packed data, such as a JSON or XML file. This data is handled using a controller as shown in the file "forum.js". This makes the HTML page reusable as the content can be modified depending on the JSON or XML file. In this project, a JSON file is being made from the content of the database. This makes it easy to use the "ng-repeat" command in Angular. This command will repeat the HTML elements with the listen values in an array. Each element in the JSON file consists of each row in the database. The desired value is collected for each element in the post.

To convert the content of the database into a JSON it is firstly collected from the database. This is done by accessing the connection to the database, as when posted earlier. MySQL is getting the data with the command *"SELECT * FROM posts ORDER BY post_date DESC"*. The star symbol "*" means that all elements are being selected from the table "posts", and ordered by the "post_date", but descending to get the latest post first. The selected rows are then looped into an array and encoded into JSON. The JSON file is fetched in the app controller in the "forum.js" script. Here it is handled and made into an application that the "forum.php" page can connect to using "ng-app" and "ng-controller".