Project Title: Learning Path Creator

Completion date:

Table of Contents

Contents

Introduction

Briefly describe the purpose and functionality of the application.
The core function of the application is to create learning paths, which are basically a “map” for certain educational topics which the user chooses.Other users follow this learning path to build knowledge piece by piece how the author of the learning path intended for said topic.How this is done is by giving the ability to each registered user to create a learning path and share it with the rest of the people on the website.Each learning path contains a title (mandatory) a description, when it was created,the language,category (also mandatory), author and a comments section aswell as the likes.Once a user clicks the learning paths he will see all of the sections (also specified by creator) that will lead to the outside sources (through clicking a link the creator provides) which the learner uses to master the topic.These sections have their own information similar to how the big learning path does.User’s that didn’t create the learning path don’t have access to change anything in it, just view it, however they do have an option to copy which leads to a new learning path with all of the previous information being put under them as if they were the one to create it to begin with.All users also have the option to click ‘copy url ’ in order to share the learning path.

Setup and Installation

There is no setup or installation necessary just go to the website and that's it (unless you would like to create/upvote/copy learning paths in which you would also have to make an account!)

Application Architecture

No frameworks or libraries were used for this project but we did however build our own mini content management system. The app is divided into a public directory and a private one.The public directory includes all of the files the user should be able to see. This is where we store the images , the images uploaded by users,the javascript,css and even the php files which contain content that should be seen by the user.The private section is where the CMS is stored which includes the CMS.php folder to create instances for everything that belongs in the CMS namespace and folder (through constructors we call every time we want to call one of these classes),a Database.php which is basically PDO with the added method of runSQL which makes it easier to run sql queries since its build to handle both normal queries and the ones with binded parameters in it.Every other class in the CMS namespace just has the instatiation of the database object and the sql queries that are used later (and some that aren't used just yet.We didn't delete these queries since we do plan on expanding this project in our free time and some of these queries we know we are going to use but we just didn't in the project yet because we ran out of time).The Validate.php file contains all the validation logic for the user with static functions in it,and the bootstrap.php file is what connects the public and src folders. It has an autoloader, which helps load by itself (as the name suggests) all of the classes that are used by each page individually, the database connection, and the cms variable.This class is used in more or less every php file in the public folder, whether that be directly or through it being incorporated into the header.php.In the public folder we also have the components folder which is just the header and the footer, both of whom are included many times in almost every page

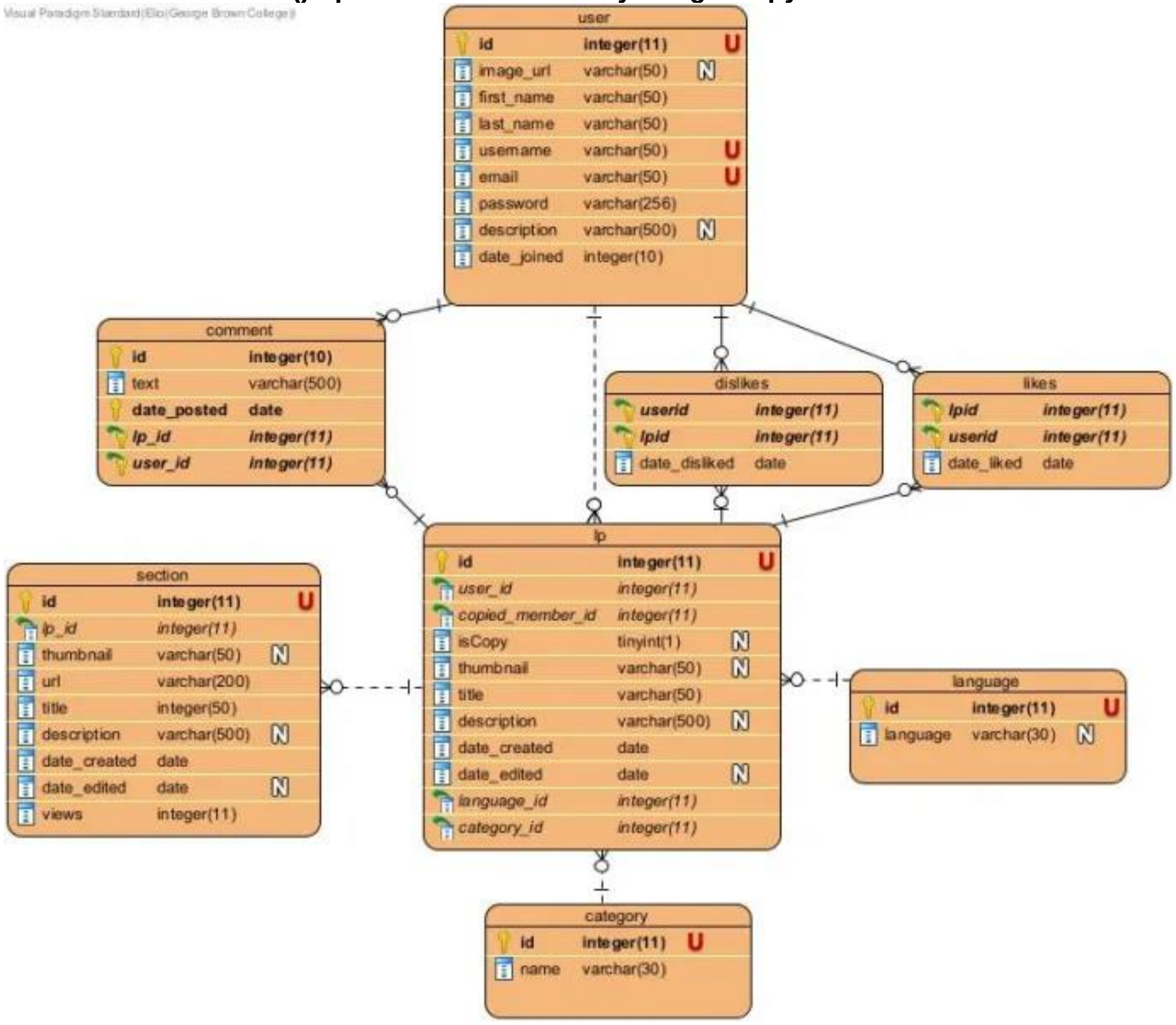
User Authentication and Authorization

As for user authentication, this is a very important topic so we decided to do it with both javascript (to show the user instantly what he needs to do and what he has gotten right) and php to truly verify the information in the backend, as stated previously validation.php has a lot of static functions we declared that help with this validation, and its pretty basic stuff like not a number (for fname and last name) not less than 3 (for username)and regexes (for emails and passwords although password has way more restrictions).These functions are called and passed the parameters specified by the user to do the check and then they shoot back a response in a variable that is shown below the form.Together with this we have also used javascript to enforce the same rules but through javascript we dynamically change what is shown to the user in every input box, and show a green box if he has gotten It correctly. As for authorization in every page it's a mix of url parameters, user id stored in session, and id retrieved from the database for individual row that helps enforce stuff such as editing copying upvoting and so on is only done by authorized users.

Database Interaction

The actual execution of the SQL query is done through a $this->db->runSQL($sql, [$id]) call, where $this->db suggests that the method is part of a class instance and runSQL is

a method to execute SQL queries. The second parameter [$id] is an array of values to bind to the parameters in the query.
->fetch() is called on the result of the SQL execution. This suggests that the query is expected to return a single row of data. The fetched data will include various fields from the specified tables.
The result of the fetch() operation is returned by the getCopy method.



Database Queries

Queries used for the comments:-

1. $sql ="SELECT u.username, u.image_url, c.text, c.date_posted
   FROM comment c
   JOIN users u ON u.id = c.user_id
   JOIN lp l ON l.id = c.lp_id
   WHERE l.id = :id";
2. $sql ="SELECT u.username, u.image_url, c.text, c.date_posted
   FROM comment c
   JOIN users u ON u.id = c.user_id

```
        JOIN lp l ON l.id = c.lp_id
        WHERE l.id = :id";
3. $sql ="SELECT u.username, u.image_url, c.text, c.date_posted
        FROM comment c
        JOIN users u ON u.id = c.user_id
        JOIN lp l ON l.id = c.lp_id";
        return $this->db->runSQL($sql)->fetchAll();
    }
4. $sql = "INSERT INTO comment (text,date_posted,lp_id,user_id)
            VALUES (:comment,NOW(),:lp_id,:user_id)";
          $this->db->runSQL($sql,$comment);
          return true;
    }
5. $sql = "SELECT u.username, u.image_url, c.text, c.date_posted,c.user_id,c.lp_id,c.id
        FROM comment c
        JOIN user u ON u.id = c.user_id
        WHERE c.lp_id = :lp_id";
        return $this->db->runSQL($sql,[':lp_id'=>$lp_id])->fetchAll();
    }
6. $sql = "DELETE FROM comment WHERE id = :id";
```

Queries Used in dislikes:-

```
1. $sql = "SELECT COUNT(*)
              FROM dislikes WHERE lp_id = :id AND user_id = :user_id";
2. $sql = "INSERT INTO dislikes (lp_id,user_id,date_disliked)
             VALUES (:lp_id,:user_id,NOW())";
3.     $sql = "DELETE FROM dislikes WHERE lp_id = :lp_id AND user_id = :user_id";
4. $sql = "SELECT COUNT(*)
              FROM dislikes WHERE lp_id = :id";
```

Queries used in likes:-

```
1. $sql = "SELECT COUNT(*)
             FROM LIKES WHERE lp_id = :id AND user_id = :user_id";
2. $sql = "INSERT INTO likes (lp_id,user_id,date_liked)
               VALUES (:lp_id,:user_id,NOW())";
3.     $sql = "DELETE FROM likes WHERE lp_id = :lp_id AND user_id = :user_id";
4. $sql = "SELECT COUNT(*)
              FROM LIKES WHERE lp_id = :id";
```

Queries Used in Lp:-

1.

Server-side scripting (PHP)

Describe how server-side scripting is handled in the application and highlight any particular PHP functionalities used.

Responsive User Interface

The responsiveness is done using media queries of several screen sizes.In most cases it is done simply by changing the font size,padding sizes (for example on the filtering area) and going from grid or flex with display row to flex column (since this stretches the layout vertically instead of horizontally, which helps show the website better on phones), there weren't any cases in which we hid parts of the display shown on larger screens in smaller ones, since we were able to fit everything else

Error Handling and Validation

Error handling is done mostly on the CMS part of the code for each sql query retrieval through try and catch blocks, but if an error does occur it is still shown on the frontend by checking if what the query results in is true or false.Other error handling is also used in parts such as when we try to upload an image and a few other cases.Validation for the stuff such as uploading learning path sections etc, is mostly done in the public folder for each file individually since a lot of specifications change for each one.

Documentation and Code Quality

We have made sure that every logic is commented. Moreover indentation and readability of the code is also taken in consideration and used optimized design.

Project-Specific Functionality (Innovative Feature)

1. **The comment feature for each learning path**. Each user has the ability to express their opinion and what they think of the learning path.Each user also has the ability to delete their own comment whenever they feel the need to.The comment shows the profile picture of the user commenting, the date and time the comment was made, the name of the commenter and the comment itself. All you have to do to access these comments is click the comments link in each one of the lp's.If there are no comments, be the first!
2. **More in-depth filtering.**Learning paths cannot only be filtered by searching parts and keywords but they can also be filtered by using the category and language you want the learning path to be.Lets say you search the keyword woodworking, but you don't understand English, there might be dozens of English learning paths on woodworking, well all you have to do is flick the language filter on the language of your choosing and boom!.Let's say you want to learn something but don't exactly know what. Well you still have a clue of what it wants to be in, you want to learn maths and sciences, well just click so on the category filter and some of the most exciting maths and sciences lp's will pop up.
3. **Like/Dislike ratio**.Each learning path doesn't just have the ability to like and dislike but you can also see the ratio of the like to dislike, which can indeed go to the negatives if an lp is really hated… try to not make one of those.
4. **Date created and edited for lps and sections.** Lets say you want to learn something that has changed recently.Plenty of times when searching for a learning path like this you might come across a lot of old content without even realizing so.Well we fixed this issue by adding the date the lp was added but ALSO by changing the time it was created, so if the author constantly updates the content in the lp you will be able to see that it is being actively maintained, and that it is or isn't up to date with what you are looking for.You will always be able to see when the lp was created

5. Views.Truth is some learning resources are better than others, some are just more popular.We have made sure to keep track of how many times each section is clicked in order to give you a better idea of what attracts people more.However make sure you don't skip on the unpopular sections, we are sure there are some hidden gems here and there.
6. You will see going through our application more sprinkles of additional functionality such as the validation of the user registration with javascript and more, however these are just the key ones that we felt the need to emphasise on, since they were also the most major ones.

Additional Notes

The application is pretty easy to use so we don't think there will be the need for additional notes

Team Member Information and Contributions

Provide contact information for all group members, including names, student IDs, email addresses, URL to presentation video and URL project folder on member's GBLearn account.

All URLs must open in a new window.

The database and innovative features were completely group efforts and not worked on by just one individual, all of us put just as much effort into those two categories.

Full Name: Elio Fezollari Student ID: 101410182 GBC email: elio.fezollari@georgebrown.ca	
Video URL	https://www.youtube.com/watch?v=B1WB-KSUpk0
GBLearn URL	F3410182.gblearn.com
Tasks completed by member	Please provide a list of the tasks completed by this member.
	Project management, Layout of the code
	User Authentication and Authorization
	Voting System
	Sharing and Cloning of the learning path,

Full Name: Aum Hemang Zaveri Student ID: 101413047 GBC email:AumHemang.Zaveri@georgebrown.ca	
Video URL	https://www.youtube.com/watch?v=3tyfeNyCV_Y
GBLearn URL	F3413047.gblearn.com
Tasks completed by member	Please provide a list of the tasks completed by this member.
	Quality Insurance and organization of code
	Search learning path based on title and description
	User Profile
	Image upload

Full Name: Mia Truong Student ID: 101446598 GBC email: Mia.Truong@georgebrown.ca	
Video URL	https://youtube.com/shorts/nC6ZiETGgZw
GBLearn URL	f3446598.gblearn.com
Tasks completed by member	Please provide a list of the tasks completed by this member.
	Prepare the project for final submission and submission
	Control of the structure and usebility of functions
	Learning Path Creation