DBMS Project Final Report

Akhil Gudivada

April 15th 2018

Introduction

A database project was selected with the ideology that it would both be a useful application with future development, and put into practice the theory and tools discussed in the course. For this project, a citation management application was designed. The idea of this application was that the app needed to be easy to use, functional, resourceful, yet have reliance on a database which relates to the course. The title of the application is *Reference Hub*. The general idea of the *Reference Hub* application, was to host in online so that the restrictions to use this could be limited. By using an online hosting environment which will be discussed in detail in its respective section, the user has more flexibility as to how and where the service can be used. For example, if the user is working on a paper or project, finds a relevant resource that they want to save, they can use the application on a mobile device, a personal computer, or even a public system granted internet access is provided. Taking these precautions allows the user to not be confined to a specific device or space to use the service. Similarly, a key feature of the application design was to restrict the user minimally, and make the resource feel like a tool as opposed to a task which efficiently stores information for the user project.

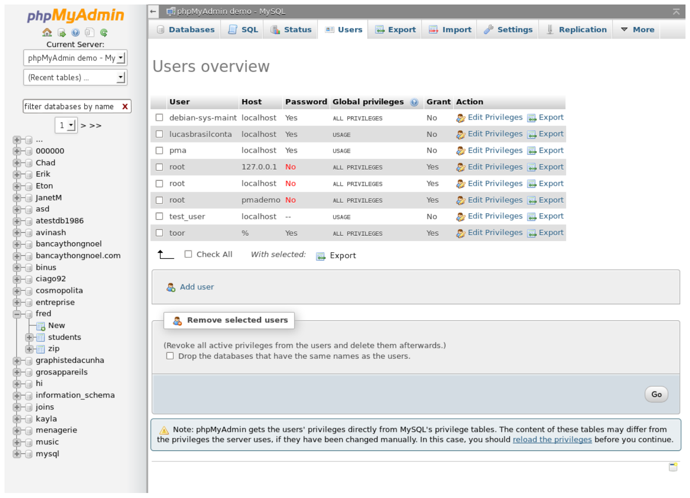
Application Requirements

As briefly discussed in the introduction, the project requirements revolve around making a fully-functional web application which can be easily used by users who may or may not have a strong technical aptitude. The first requirement in the application was to create some kind of security system so that user data could be protected between the users, and also so that each user can use the application which would provide them a personalized set of references. This log-in feature should allow different users to log-in the system in different locations, and use the product concurrently without any issues. The next requirement was to be able to have the application up and running on an external server which hosted the app on the internet. The next requirement of the application was to have the application efficiently use a database to insert, display, and modify the data associated with *Reference Hub*. The next requirement dealt with the front-end of the application. A user-friendly interface needs to be implemented in order for the user to feel comfortable using the application. Although this requirement is more subjective than the others, it can be verified as long as there are no bugs or any abnormalities while the user is dealing with the UI. The last major requirement was to have dynamic data be updated concurrently throughout the program. For example, when a user updates a record by modifying a reference or adding one, the system should be able to accommodate this change without depending on the other users in the system.

Application Components

Database

One major component in this application is the storage of data used. In this project, a MySQL database was used. The database was convenient to use as it already was provided by the hosting service (discussed in detail below). Although not a requirement to use a database associated with the hosting service, the process of establishing, maintaining and modifying connections in the PHP pages runs much smoother when this is the case. To manage the database, a service called *phpMyAdmin* was used which allowed creation, modification, and deletion of databases and tables along with potential connections with the finished webpages. It also allows the administrator to watch the databases and ensure that they are running properly for the application. Moreover, this service allows testing of the connections by letting the admin pass dummy data from the front- end of the application. The details of this will be discussed in the testing section. The first database was created in order to meet the first re-



Users overview

	User	Host	Password	Global privileges	Grant	Action
☐	debian-sys-maint	localhost	Yes	ALL PRIVILEGES	No	Edit Privileges Export
☐	lucasbrasilconta	localhost	Yes	USAGE	No	Edit Privileges Export
☐	pma	localhost	Yes	USAGE	No	Edit Privileges Export
☐	root	127.0.0.1	No	ALL PRIVILEGES	Yes	Edit Privileges Export
☐	root	localhost	No	ALL PRIVILEGES	Yes	Edit Privileges Export
☐	root	pmademo	No	ALL PRIVILEGES	Yes	Edit Privileges Export
☐	test_user	localhost	--	USAGE	No	Edit Privileges Export
☐	toor	%	Yes	ALL PRIVILEGES	Yes	Edit Privileges Export

☐ Check All *With selected:* Export

Add user

Remove selected users

(Revoke all active privileges from the users and delete them afterwards.)
☐ Drop the databases that have the same names as the users.

Go

Note: phpMyAdmin gets the users' privileges directly from MySQL's privilege tables. The content of these tables may differ from the privileges the server uses, if they have been changed manually. In this case, you should reload the privileges before you continue.

Figure 1: A visual representation of the database management tool

quirement. Setting it up allowed users to concurrently use the system and have access to only their respective data. This initial database contained a user's first name, last name, email address and password. By setting up the email address as a primary key, this ensured no two users could share the same email address, and with a SQL command in the PHP pages, this could be controlled by sending an error message when a duplicate email address was in the process of being created. The attributes were set up as variable characters with the default limit, and the HTML page was left to ensure the user enters in an email address in the valid format. In addition to holding the information necessary to ensure that users had proper credentials while using the service, this database also keeps track of the user's name in order to provide a greeting to them when they log-in successfully by displaying a message tailored specifically for them. When a user logs in to the application, they are prompted for their user-name and password, but not their name. Upon successful entry into the application, another SQL script retrieves their name given that it matches the tuple containing their entered user-name(email) and password.

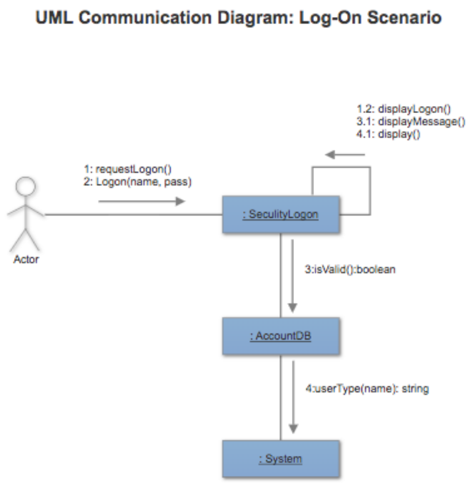


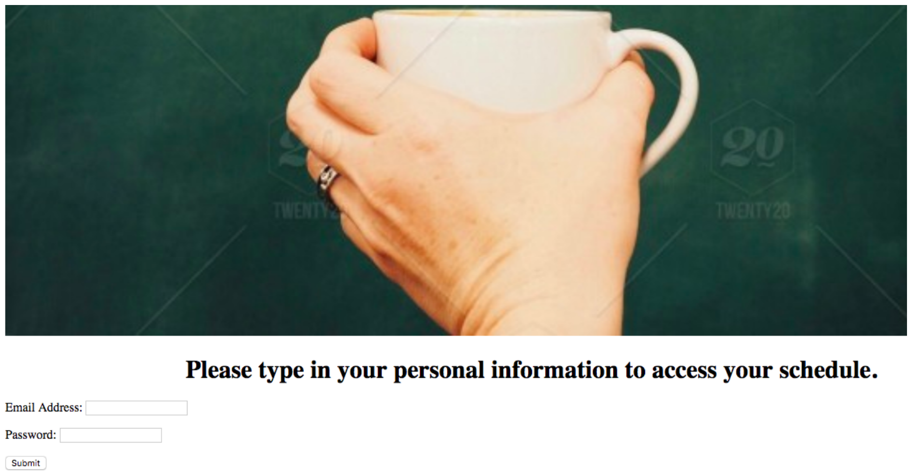
Figure 2: A Diagram outlining the log-in system

The next database was set up to hold the user entered information for their references. This was essentially the main database component of the system as it had the user entered information. The database was also set up with the main requirement of keeping it user friendly and very flexible by allowing user error such as not entering in the correct data types. The Reference-Id was set as the auto-incremented primary key as no two references should share this. The additional columns for information such as reference type, author, publication year, etc. was added in. This was the main table which was used in the testing process to ensure data was being stored in the respective columns as the respective data types.

Front-End Design

The front-end to the application was written with a combination of HTML5, PHP, and CSS3 languages. Although the back-end, connections, and functionality of a system generally have higher priority in projects, the front-end is the only portion the user deals with and therefore the only exposure the user gets with the application. Therefore, it is important to create a model that is both user friendly, efficient, and easy to look at.

The welcome page of the application was built in order to appeal to the user and draw them into the application. It provides the application name, a large clear image, and a brief description of the application. The images for this page, as well as any other pages in the application were acquired from a source which allows commercial use of royalty free images (pexels.com). Adding these images were a big part of appeal to the application. As mentioned before, as the user only deals with this portion of the application, in order to fully appreciate the complexity of the app, it helps provide that confirmation to the user. The welcome page also contains a link to the next section which takes the user to the log-in system. Here, the user has the option of either logging in the system as an existing user, or signing up as a new user. If the user selects to register, they are taken to a registration page which intakes their first name, last name, email, and password. With methods described in the previous subsection, if the user enters an invalid combination (duplicate email) then they are prompted to try with a different email. Once the user enters all valid input, they are taken back to the page which allows them to sign in as an existing user. On this page, a similar PHP script allows them to sign in, at which point a SQL script embedded within the PHP page, checks for a match based on the email and password the user gave. If there is no match, the user is denied access and prompted to either try again, or create a new account. Given that the user enters a valid combination, at this point they are taken to the welcome page upon clicking *submit* via an on-click button method in the PHP script. In addition to the HTML pages providing application functionality, CSS helped format the pages. Along with color, layout, text appearance, and font, CSS made the pages look much more user-friendly. Background color was added to make the visual appeal of the application rise and look less plain. The colors were varied between web pages. Another unexpected challenge was finding proper colors to assign the text so that they would be visible in contrast to the background color. Some online resources could have been used to automatically format the HTML pages with an existing CSS to fit, but often times, the layout would be much different, so a custom CSS had to be coded to match the specifications of the web pages.


Please type in your personal information to access your schedule.

Email Address:

Password:

Figure 3: A HTML page prompting the user to sign in to the application

At this point, if the user enters in matching information, the SQL script mentioned in the *Database* section, retrieves both the first and last name of the user and displays it on the next welcome page by displaying *Welcome, first name last name!*.

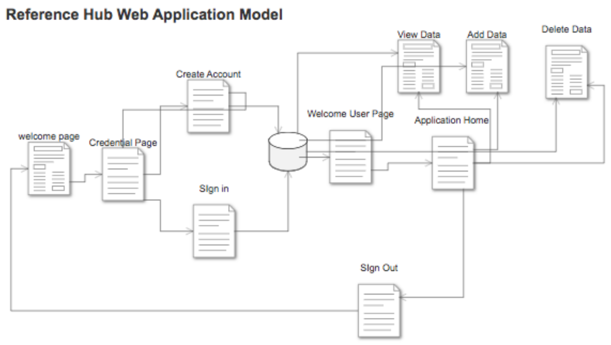
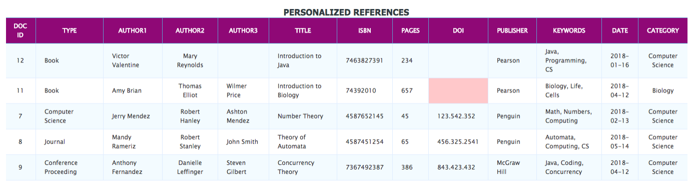


Figure 4: A Diagram outlining the system model

This is a similar feature used by many web applications to connect to the user and give them a more personalized feel when using the application. After this point, the user is able to click the *proceed* button which uses another on-click button method in the PHP script leading the user to the application home page. Here the user is able to access the three main functions for the product. They are able to start a new project by adding in references, add references to an existing project, delete an existing reference, or view their current reference list. Both addition and deletion pages have embedded SQL scripts that follow the format of inserting/ deleting all user entered variables. This is done by assigning a variable to each user input within the PHP web-page and storing this information in a form within the page. Each variable in PHP is denoted by a $ symbol. For example when the user entered reference type, the variable would be denoted as $RefType. The specific details can be found within the PHP files which will be submitted. Storing the user input in variables allows an embedded SQL script to be run which can then take these variables (holding user-entered information) and place this information into the correct database table and column using the *INSERT* command. Similarly, deletion can be performed the same way. The more difficult process is to display the view. In order to make the view appear in a HTML page as shown below, a loop with new threads needed to be formed in the HTMP page. This loop checks to see if there is an entry in the corresponding database column, and if there is, creates a thread which forms a cell and displays values into the cell. The cells along with the title, color, and size were all coded into the embedded HTML page.


PERSONALIZED REFERENCES

DOC ID	TYPE	AUTHOR1	AUTHOR2	AUTHOR3	TITLE	ISBN	PAGES	DOI	PUBLISHER	KEYWORDS	DATE	CATEGORY
12	Book	Victor Valentine	Mary Reynolds		Introduction to Java	7463827391	234		Pearson	Java, Programming, CS	2018-01-16	Computer Science
11	Book	Amy Brian	Thomas Elliot	Wilmer Price	Introduction to Biology	74392010	657		Pearson	Biology, Life, Cells	2018-04-12	Biology
7	Computer Science	Jerry Mendez	Robert Hanley	Ashton Mendez	Number Theory	4587652145	45	123.542.352	Penguin	Math, Numbers, Computing	2018-02-13	Computer Science
8	Journal	Mandy Rameriz	Robert Stanley	John Smith	Theory of Automata	4587451254	65	456.325.2541	Penguin	Automata, Computing, CS	2018-05-14	Computer Science
9	Conference Proceeding	Anthony Fernandez	Danielle Leffinger	Steven Gilbert	Concurrency Theory	7367492387	386	843.423.432	McGraw Hill	Java, Coding, Concurrency	2018-04-12	Computer Science

Figure 5: Displaying database entries as a view on a HTML page

This information can be found in the *display-all*.php file.

Hosting

Although the web application could be hosted locally on one system, as mentioned in the requirements section, this application was hosted online to provide ease of access to the user. x10hosting.com was used as a service which allows file hosting under that domain name free of charge for a limited amount of space. This service also provided the MySQL server along with the management tool of phpMyAdmin. The hosted application also allowed for testing with different browsers and devices. The application was verified to be functional on mobile devices as well. Furthermore, the online service allows others to test out the application and in real scenarios, a potential bug or error could be discovered by the sample of users who test out the application. Since this was web hosted, no engine was required to run the hosting service such as *Apache*. All of the software is indirectly provided by the hosting service provider. The files were uploaded into a web directory, and as long as all files are in the same directory, the functionality remains the same as if they were on a localhost network. These files can also directly be modified within the hosting environment. They can also be organized into different folders and are accessible through the root commands. The error logs for each of the SQL files are also displayed and will be submitted as an error log file. The connection files which were coded in PHP to ensure database connections are located in this directory as well. The connections were established by creating a function within a PHP page that is provided the credentials of the database such as the name, server, port, username, and password. The full details of this information can be found in the *connections*.php file.

Testing

Since the hosting was provided through a third party source, the need for testing was reduced. First, the database connections were established. Once established, the connections remained fairly stable, so data was entered on to the HTML forms and the database was checked on the database management tool *phpMyAdmin*. Once the data was ensured to be storing in the correct locations, all functions were then tested. The create, update, delete operations were all extensively tested for various types of data. User concurrency was tested by having various users log-in to the system at the same time and update their project references. This process was thoroughly repeated until there were no inconsistencies in the data. Testing the application proved that all the established requirements were satiated. The application provides user-friendly layout which allows the user lead-way by entering varying data types. Different projects can be added on, and the information gets stored in the database regardless of type. The security requirements were met by testing the application with different users. First, the application could not be accessed without a user registering their account and logging in. This ensures that the application can be properly monitored and fully administered with admin and user roles. In addition, each user was only able to view their respective projects, checking off another established requirement. The final requirement was met, but was rather subjective as a user-friendly environment. Although no formal test to meet this requirement exists, the fonts were all visible and the images as well as content were positioned in their respective places. No bugs or error messages were generated. All the requirements that were established for this project were met.

Future Implementation

Like any application, there remains a lot of work that can be done to improve the performance and functionality of this application. First, when deletion is taking place, the user should be prompted to confirm if they would like to delete a reference for sure, as they may have accidentally clicked the button. This will help ensure that no important data is lost from the user. In addition, a service which prints the references for the users in IEEE citation can be implemented. This feature could take all the input data and transform it into an official citation. Another potential feature could be that a library of books could be imported into the database along with images, so that the users can get a visual representation of the references that they are adding. Although this would be a larger update, it can be incrementally updated.

Conclusion

Overall, this project helped enhance my skills in web-application development. It also re-enforced the theoretical aspects of databases that were discussed in the lecture such as SQL queries, indexing, concurrency control, and even database connection features. Many challenges were faced throughout the project, mainly dealing with establishing the proper connections between the front and back-end of the system. However, once the connections were established, they remained up throughout the remainder of the project. The application was also incrementally updated by making minor changes one after another before coming out with the displayed application. There is a lot more room to expand the application, and I believe there is practical use for this application. I plan to develop it further for myself if nothing else. I was able to learn a great deal of information as the project helped bridge my theoretical knowledge into practical, real-world implementation.