**Database Application Programming**

**Project Status and Design Report**

|  |  |
| --- | --- |
| **Topic:** | *Topic 5: Database Constraints and Table Joins* |
| **Date:** | *11/6/20* |
| **Revision:** | *6.0* |
|  | |  |  |  | | --- | --- | --- | | **Task** | **Hours Worked** | **Hours Remaining** | | *Project Report* | *0.5* | *0* | | *Update view posts page* | *1.5* | *0* | | *Add ability to add comments to posts* | *2* | *0* | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
| **GIT URL:** | *https://github.com/MrDuise/Blog-Project.git* |
| **Hosting URL:** | https://cstblogapp126.azurewebsites.net |

**Supporting Design Documentation**

**Install Instructions:**

Installed MAMP, setup a database called ‘registration’ with a table called ‘users’ to store the registration information. Table includes the columns ID, USER\_NAME, LAST\_NAME, Email, Password, and role. Role is changed after registration and by default is empty. ID is auto incremented. You also need a table called user\_posts. This table is connected to the users table using MySQL Workbench. The user\_posts table has the columns ID, Blog\_Title, Blog\_Entry, and Users\_ID. You also need a table called ‘user\_comments’. This table has the columns ID, and User\_Comments. In order to run this code, you need MAMP and mySQL workbench installed, the server ports set to the MAMP default, and the web server being used to be set to Apache. The files need to be in the document root C:\MAMP\htdocs. The database can be cloned using the dll file included in the github repo.

In order to host and deploy application, an Azure account is needed. The following is the instructions that I followed to host my application after creating my azure account.

The following are a few guidelines for building and uploading your PHP Project:

1. Make sure you have built your PHP Project with the right version to match the version of PHP you are using:
2. The version of PHP can be set in your Applications Configuration section in Azure.
3. You can upload and configure your PHP Project as follows:
4. Create a default PHP application in Azure:
5. Click the '+ Create a resource' icon from the Azure Portal and search for PHP.
6. Make sure that the Publisher is set to Microsoft. Select Web App + MySQL from the list. Fill out the form
7. Open your application from your Dashboard.
8. Deploy your Application:
9. Zip up your PHP project into a file named [appname].zip
10. Under the Development Tools section click the Advanced Tools icon, select the Go link, and select the Tools->Zip Push Deploy menu
11. Delete the Azure created default files from the application (if they exist)
12. Drag and drop your zip file onto the page
13. Deploy your Database in Azure:
14. Under the Settings section click the MySQL In App icon and click the Manage icon to open phpMyAdmin.
15. Make a new database, making sure the title is the same as the one you are going to import. Also make sure that you have the right server selected as by defaut the Azure will not be selected. You can check which server is being used by clicking the drop down window above the list of databases.
16. Import your Database DLL
17. Under the Development Tools section click the Console icon.
18. In the Console enter 'type D:\home\data\mysql\MYSQLCONNSTR\_localdb.txt' to get your MySQL Connection Properties.
19. Update your MySQL Database Connection properties in your application (note your hostname will need to be formatted as hostname:port).
20. Test your application using your Azure Domain name:
21. Go to https://[app name].azurewebsites.net.

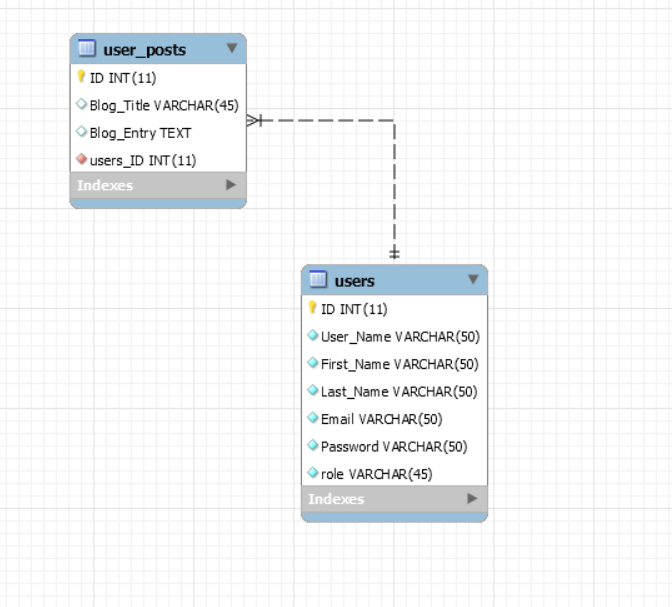
**General Technical Approach:**

*Version 6 adds the ability for the user to add comments to a post. The comments are stored in a separate table called user\_comments and linked to both the users table as well as the user\_posts table using the primary key ID of both tables*

**Key Technical Design Decisions:**

*The view posts page has been changed so that only the Title of the post can be viewed on the screen. In order to view the individual post, you must click the view post button.*

**ER Diagram:**

**

**DDL Scripts:**

[*https://github.com/MrDuise/Blog-Project.git*](https://github.com/MrDuise/Blog-Project.git)

**Sitemap Diagram:**

*Diagram

Description automatically generated*

**User Interface Diagrams:**

A picture containing shirt, bag

Description automatically generatedA picture containing shirt, bag

Description automatically generated

**Other Documentation:**

All other functionality is the same as in version 5.0

|  |  |
| --- | --- |
| **Topic:** | *Topic 5: Database Constraints and Table Joins* |
| **Date:** | *11/6/20* |
| **Revision:** | *5.0* |
|  | |  |  |  | | --- | --- | --- | | **Task** | **Hours Worked** | **Hours Remaining** | | *Project Report* | *1* | *0* | | *Update site-map* | *1* | *0* | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
| **GIT URL:** | *https://github.com/MrDuise/Blog-Project.git* |
| **Hosting URL:** | https://cstblogapp126.azurewebsites.net |

**Supporting Design Documentation**

**Install Instructions:**

Installed MAMP, setup a database called ‘registration’ with a table called ‘users’ to store the registration information. Table includes the columns ID, USER\_NAME, LAST\_NAME, Email, Password, and role. Role is changed after registration and by default is empty. ID is auto incremented. You also need a table called user\_posts. This table is connected to the users table using MySQL Workbench. The user\_posts table has the columns ID, Blog\_Title, Blog\_Entry, and Users\_ID. In order to run this code, you need MAMP and mySQL workbench installed, the server ports set to the MAMP default, and the web server being used to be set to Apache. The files need to be in the document root C:\MAMP\htdocs.

In order to host and deploy application, an Azure account is needed. The following is the instructions that I followed to host my application after creating my azure account.

The following are a few guidelines for building and uploading your PHP Project:

1. Make sure you have built your PHP Project with the right version to match the version of PHP you are using:
2. The version of PHP can be set in your Applications Configuration section in Azure.
3. You can upload and configure your PHP Project as follows:
4. Create a default PHP application in Azure:
5. Click the '+ Create a resource' icon from the Azure Portal and search for PHP.
6. Make sure that the Publisher is set to Microsoft. Select Web App + MySQL from the list. Fill out the form
7. Open your application from your Dashboard.
8. Deploy your Application:
9. Zip up your PHP project into a file named [appname].zip
10. Under the Development Tools section click the Advanced Tools icon, select the Go link, and select the Tools->Zip Push Deploy menu
11. Delete the Azure created default files from the application (if they exist)
12. Drag and drop your zip file onto the page
13. Deploy your Database in Azure:
14. Under the Settings section click the MySQL In App icon and click the Manage icon to open phpMyAdmin.
15. Make a new database, making sure the title is the same as the one you are going to import. Also make sure that you have the right server selected as by defaut the Azure will not be selected. You can check which server is being used by clicking the drop down window above the list of databases.
16. Import your Database DLL
17. Under the Development Tools section click the Console icon.
18. In the Console enter 'type D:\home\data\mysql\MYSQLCONNSTR\_localdb.txt' to get your MySQL Connection Properties.
19. Update your MySQL Database Connection properties in your application (note your hostname will need to be formatted as hostname:port).
20. Test your application using your Azure Domain name:
21. Go to https://[app name].azurewebsites.net.

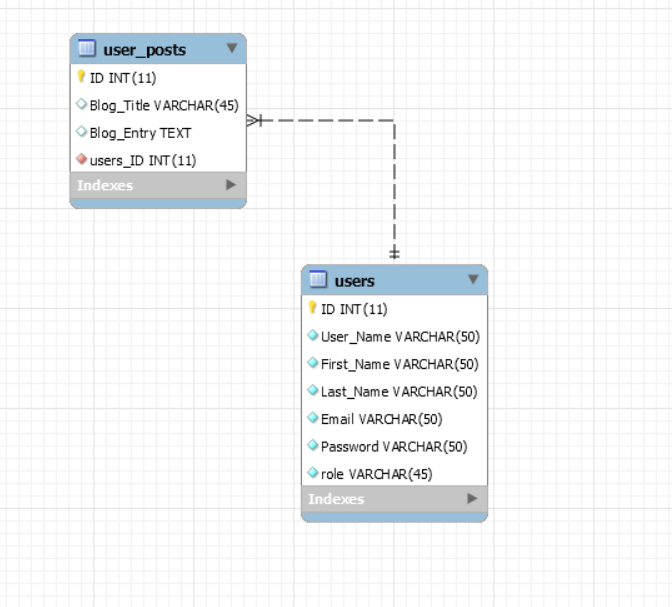
**General Technical Approach:**

*Nothing new has been added, all functionality is the same as in v4.0*

**Key Technical Design Decisions:**

*Nothing new has been added, all functionality is the same as in v4.0*

**ER Diagram:**

**

**DDL Scripts:**

[*https://github.com/MrDuise/Blog-Project.git*](https://github.com/MrDuise/Blog-Project.git)

**Sitemap Diagram:**

*Diagram

Description automatically generated*

**User Interface Diagrams:**

*No new interfaces designs have been made*

**Other Documentation:**

*Nothing new has been added, all functionality is the same as in v4.0*

|  |  |
| --- | --- |
| **Topic:** | *Topic 4: PHP Arrays and HTML Tables* |
| **Date:** | *10/29/20* |
| **Revision:** | *4.0* |
|  | |  |  |  | | --- | --- | --- | | **Task** | **Hours Worked** | **Hours Remaining** | | *Make admin page* | *2* | *0* | | *Make user posts page only display that users posts* | *1* | *0* | | *Project Report* | *4* | *0* | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
| **GIT URL:** | *https://github.com/MrDuise/Blog-Project.git* |
| **Hosting URL:** | https://cstblogapp126.azurewebsites.net |

**Supporting Design Documentation**

**Install Instructions:**

Installed MAMP, setup a database called ‘registration’ with a table called ‘users’ to store the registration information. Table includes the columns ID, USER\_NAME, LAST\_NAME, Email, Password, and role. Role is changed after registration and by default is empty. ID is auto incremented. You also need a table called user\_posts. This table is connected to the users table using MySQL Workbench. The user\_posts table has the columns ID, Blog\_Title, Blog\_Entry, and Users\_ID. In order to run this code, you need MAMP and mySQL workbench installed, the server ports set to the MAMP default, and the web server being used to be set to Apache. The files need to be in the document root C:\MAMP\htdocs.

In order to host and deploy application, an Azure account is needed. The following is the instructions that I followed to host my application after creating my azure account.

The following are a few guidelines for building and uploading your PHP Project:

1. Make sure you have built your PHP Project with the right version to match the version of PHP you are using:
2. The version of PHP can be set in your Applications Configuration section in Azure.
3. You can upload and configure your PHP Project as follows:
4. Create a default PHP application in Azure:
5. Click the '+ Create a resource' icon from the Azure Portal and search for PHP.
6. Make sure that the Publisher is set to Microsoft. Select Web App + MySQL from the list. Fill out the form
7. Open your application from your Dashboard.
8. Deploy your Application:
9. Zip up your PHP project into a file named [appname].zip
10. Under the Development Tools section click the Advanced Tools icon, select the Go link, and select the Tools->Zip Push Deploy menu
11. Delete the Azure created default files from the application (if they exist)
12. Drag and drop your zip file onto the page
13. Deploy your Database in Azure:
14. Under the Settings section click the MySQL In App icon and click the Manage icon to open phpMyAdmin.
15. Make a new database, making sure the title is the same as the one you are going to import. Also make sure that you have the right server selected as by defaut the Azure will not be selected. You can check which server is being used by clicking the drop down window above the list of databases.
16. Import your Database DLL
17. Under the Development Tools section click the Console icon.
18. In the Console enter 'type D:\home\data\mysql\MYSQLCONNSTR\_localdb.txt' to get your MySQL Connection Properties.
19. Update your MySQL Database Connection properties in your application (note your hostname will need to be formatted as hostname:port).
20. Test your application using your Azure Domain name:
21. Go to https://[app name].azurewebsites.net.

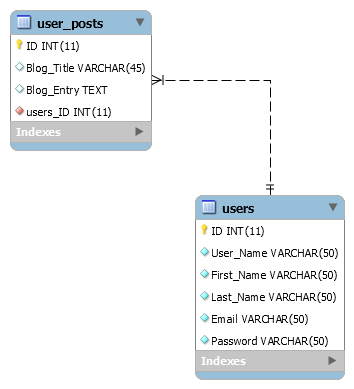
**General Technical Approach:**

The only technical things that have changed are that now each user can only view their own posts that they have written. They cannot see everyone’s. This will be changed at a later date. There is also an admin page that allows for deleting and editing of the posts as well as seeing all the names of the users. Only someone with the role of “admin” has the ability to see the admin page.

**Key Technical Design Decisions:**

No new technical designs were implemented. Only one new page was added which is the admin page.

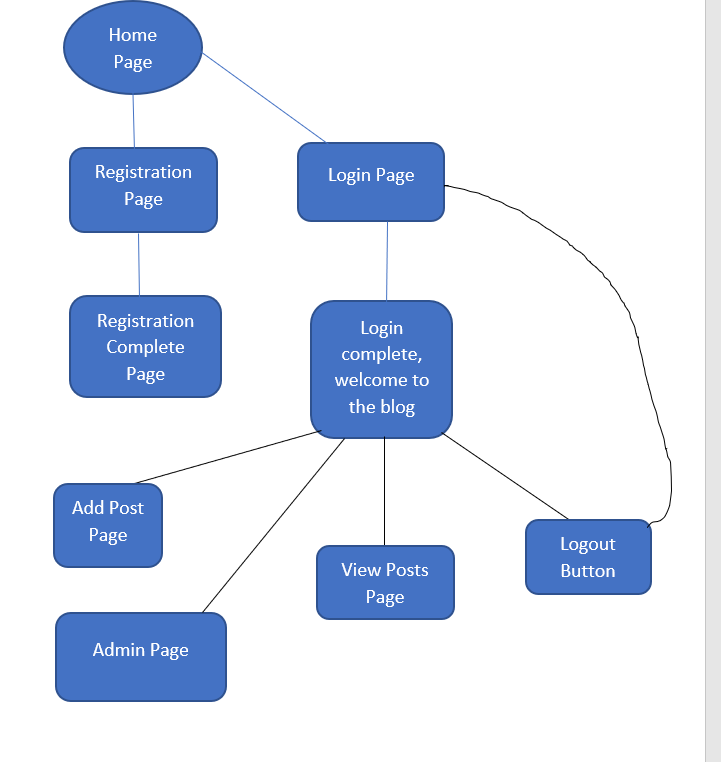
**ER Diagram:**



**DDL Scripts:**

*This should contain a link to Bitbucket where the DDL script can be downloaded.*

**Sitemap Diagram:**

**

**User Interface Diagrams:**

No new designs were implemented.

**Other Documentation:**

No new designs were implemented.

|  |  |
| --- | --- |
| **Topic:** | *Topic 3: PHP Functions* |
| **Date:** | *10/13/20* |
| **Revision:** | *3.0* |
|  | |  |  |  | | --- | --- | --- | | **Task** | **Hours Worked** | **Hours Remaining** | | *Make logout page* | *1.5* | *0* | | *Make Add blog post page* | *1.5* | *0* | | *Make blog view page* | *2* | *0* | | *Make main welcome page for when user logins in* | *2.5* | *0* | | *Make functions page* | *0.5* | *0* | | *Edit CSS* | *4* | *0* | | *Edit main home page, login page, register page, and data handler pages* | *3* | *0* | | *Make footer page* | *2* | *0* | | *Project report* | *3* | *0* | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
| **GIT URL:** | *https://github.com/MrDuise/Blog-Project.git* |
| **Hosting URL:** | *N/A* |

**Supporting Design Documentation**

**Install Instructions:**

Installed MAMP, setup a database called ‘registration’ with a table called ‘users’ to store the registration information. Table includes the columns ID, USER\_NAME, LAST\_NAME, Email, and Password. ID is auto incremented. You also need a table called user\_posts. This table is connected to the users table using MySQL Workbench. The user\_posts table has the columns ID, Blog\_Title, Blog\_Entry, and Users\_ID. In order to run this code, you need MAMP and mySQL workbench installed, the server ports set to the MAMP default, and the web server being used to be set to Apache. The files need to be in the document root C:\MAMP\htdocs

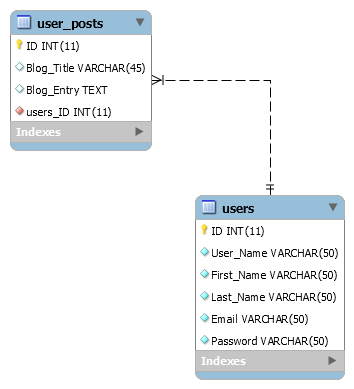
**General Technical Approach:**

For version 3.0 of my project, which is a part of the milestone 3 requirement, I expanded on a lot on both the CSS and functionality of the pages. I made each file a php file but added in html code when needed. This allowed me to add footer files to my pages. All form entry uses the ‘post’ method now instead of the ‘get’ method, for security reasons. Main-welcome page will not allow a user to view page if they are not logged in, if a user attempts to view said page with the back button after logging out, they will be asked to login back in. The form to add a post checks for certain key words that are not allowed, and will block the post from being added to the database if they are detected.

**Key Technical Design Decisions:**

As stated above, I made each file a php file, allowing me to add external files when needed. Multiple users with the same username are not allowed to register. When user logs in, the username that they registered with is on display in the nav bar. When user clicks log out, they are greeted with the login page. The icon right next to the title in the tab has been changed from the default.

**ER Diagram:**

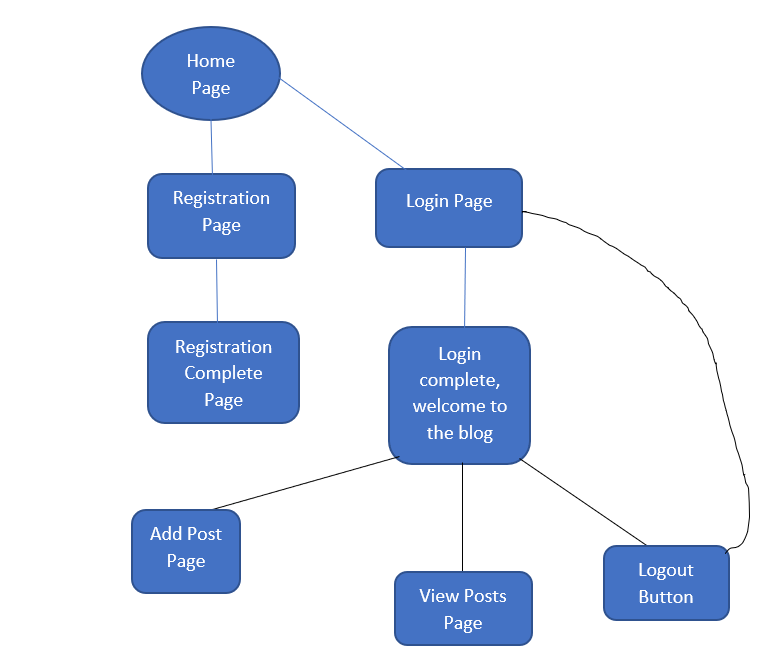


***The tables are linked so that one user can have multiple posts***

**DDL Scripts:**

*N/A*

**Sitemap Diagram:**

**

**User Interface Diagrams:**

Diagram

Description automatically generated

Main Page After Login

Diagram, schematic

Description automatically generated

Add Post page

**Other Documentation:**

*I had no other drawings or documentation*

|  |  |
| --- | --- |
| **Topic:** | *Topic 2: PHP Control Constructs, Operators, and Variables* |
| **Date:** | *09/30/20* |
| **Revision:** | *2.0* |
|  | |  |  |  | | --- | --- | --- | | **Task** | **Hours Worked** | **Hours Remaining** | | *Updated Registration form with required data fields* | *1* | *0* | | *Updated Welcome Page* | *1* | *0* | | *Created login page with connection to Database with required data fields* | *1* | *0* | | *Adding CSS to pages* | *2.5* | *0* | | *Finish documentation* | *2* | *0* | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
| **GIT URL:** | *https://github.com/MrDuise/Blog-Project.git* |
| **Hosting URL:** | *N/A* |

**Supporting Design Documentation**

**Install Instructions:**

Installed MAMP, setup a database called ‘registration’ with a table called ‘users’ to store the registration information. Table includes the columns ID, USER\_NAME, LAST\_NAME, Email, and Password. ID is auto incremented. In order to run this code, you need MAMP installed, the server ports set to the MAMP default, and the web server being used to be set to Apache. The files need to be in the document root C:\MAMP\htdocs

**General Technical Approach:**

For version 2.0 of my project, which is a part of the milestone 2 requirement, I expanded on a lot on both the CSS and functionality of the pages. Each page now has a navbar at the top, allowing access to each of the main pages. The pages are the welcome page, the registration page, and the login in page. This expands the usability of the blog in so that the user does not have to use the back button to go back to an earlier page. The registration complete page also has a navbar at the top allowing the user to either register a new user or login to their account.

**Key Technical Design Decisions:**

Version 2.0 does not allow blank entry forms in either the registration or login pages. All data forms must be entered in order to either create a new user or login. It does allow an unlimited number of attempts to login in though as well as multiple users to be inserted with the same username and password. This will be changed in the next revision.

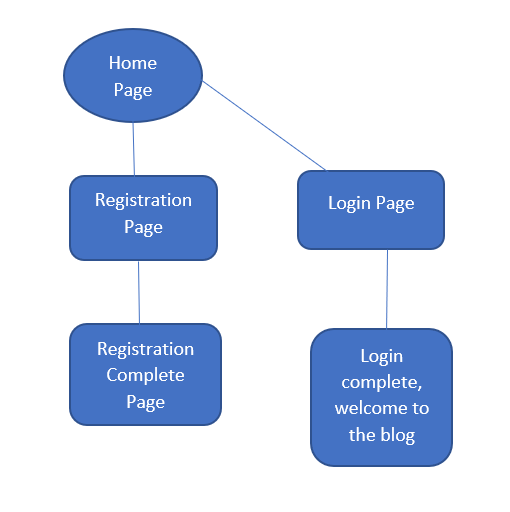
**ER Diagram:**

*N/A*

**DDL Scripts:**

*N/A*

**Sitemap Diagram:**

**

**User Interface Diagrams:**

**Diagram

Description automatically generatedA close up of a piece of paper

Description automatically generated**

Registration complete page

Welcome page

**Diagram

Description automatically generated**

Registration Page

**Text

Description automatically generated**

Login page

**Other Documentation:**

N/A

**Database Application Programming**

**Project Status and Design Report**

|  |  |
| --- | --- |
| **Topic:** | *Topic 1: Introduction to PHP and MySQL* |
| **Date:** | *09/18/20* |
| **Revision:** | *1.0* |
|  | |  |  |  | | --- | --- | --- | | **Task** | **Hours Worked** | **Hours Remaining** | | *Created HTML Registration Form* | *1* | *0* | | *Created Welcome Page* | *1* | *0* | | *Created Data Capture page with connection to Database* | *1* | *0* | | *Adding CSS to pages* | *2.5* | *0* | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
| **GIT URL:** | *https://github.com/MrDuise/Blog-Project.git* |
| **Hosting URL:** | *N/A* |

**Supporting Design Documentation**

**Install Instructions:**

Installed MAMP, setup a database called registration with a table called users to store the registration information. Table includes the columns ID, USER\_NAME, LAST\_NAME, Email, and Password. ID is auto incremented. In order to run this code you need MAMP installed, the server ports set to the MAMP default, and the web server being used to be set to Apache. The files need to be in the document root C:\MAMP\htdocs

**General Technical Approach:**

For version 1.0 of my project, which is a part of the milestone 1 requirement, my project is very simple. I have a welcome page, registration page, and some text that gets displayed after the data has been entered into the form. Text for each page is centered into the center of the screen as I felt it looked better than being off to the side. It focuses the user’s attention. The welcome page displays text welcoming the user to the page with a button below that says register. Clicking on this button opens the registration page, which has a php form action attached to it allowing the data the user enters to be inserted into the database. This page also displays a message when the user has completed registration.

**Key Technical Design Decisions:**

This is a basic attempt. Code does not include any security or error checking. Database entries will be made even if no data in entered into the form.

**ER Diagram:**

*N/A*

**DDL Scripts:**

*N/A*

**Sitemap Diagram:**

*A close up of text on a white background

Description automatically generated*

Login page and Blog home page have not been started yet. This diagram was designed after the coding process had already begun. Currently the only pages are the welcome page, register page, and registration complete page.

**User Interface Diagrams:**

This is a first attempt, basic page. Nothings was planned out ahead of time. I just wanted to make the registration page. Cover/welcome page was added after register page was already complete.

**Other Documentation:**

N/A