

Final Paper

Austin Gutierrez (1214766501)

Arizona State University

IFT 200

## Table of Contents

Summary -----	3
Main Page -----	4
Install Apache -----	5
Install MariaDB -----	6
Install PHP -----	7
Install PHP MyAdmin -----	8
Entity Relationship Diagram (ERD) -----	9
Table Creation in MariaDB -----	10
Forms -----	11
Queries/Reports -----	12
Website Outline -----	13
Failure Report -----	14
Conclusion -----	16

## Summary

Thank you for choosing Brew Brothers for your database needs. We are a Technology Company specializing in creating, maintaining, and improving relational database for everyone. Databases are essential and play an integral part of any business. We will not only establish the backbone to your database, but we will also develop a graphical user interface (GUI), in the form of a website, that will allow your employees to manage the data accordingly. This document outlines the steps taken, from start to finish, to create an example database using beer, spirits, and wine.

## Main Page

First things first, we must install the appropriate applications needed to lay the ground work of the database. For this project, we will be using a “LAMP” stack consisting of a Linux OS (Manjaro), Apache web server, PHP scripting language, PhpMyAdmin for data management, and MariaDB for the database. Once these programs are installed and configured for our needs, our next step is to outline the planned database.

To outline the database, we need to create an Entity Relationship Diagram (ERD) to establish the data, table, attributes, and relationship for our database. Once this diagram is created, it will allow us to organize the tables accordingly and ensure the proper primary and foreign keys are created. This will be used to help maintain the database and increase the efficiency as the database grows. After we have the outline, we will connect to MariaDB services and create the database and each table accordingly. After the tables are created, we can fill in the information from an already completed flat file.

Our next step in the process will be to create basic forms that allow user to input data into the appropriate table. This will be completed using PHP and HTML to provide an GUI. Each table will have its own form dedicated to inputting information into the database. For administration purposes, we will only allow information to be deleted through the admin account using phpMyAdmin. We will also use PHP and HTML to create the “Query” page. This page will allow user to query each one of the tables and display the results on the same page. More detailed queries can be developed later.

Once all these products are completed, they will be compiled into a website with the options to choose what they want to do. This GUI will allow for functionality of the database for the user with limited access based on their needs.

## Install Apache Web Server

Utilizing the pacman package manager, I typed the command into the terminal to install apache onto my computer. After Apache was installed, I had to enable httpd and verify that the server was running using the commands “sudo systemctl enable httpd”; “sudo systemctl restart httpd”, and “systemctl status httpd”. Below are the screenshots of the installation and verification that the server is running.

```
[austin@austin-pc ~]$ sudo pacman -S apache
[sudo] password for austin:
resolving dependencies...
looking for conflicting packages...

Packages (1) apache-2.4.35-1

Total Download Size: 1.51 MiB
Total Installed Size: 6.27 MiB

:: Proceed with installation? [Y/n] y
:: Retrieving packages...
  apache-2.4.35-1-x86_64             1541.2 KiB   760K/s 00:02 [#####] 100%
(1/1) checking keys in keyring [#####] 100%
(1/1) checking package integrity [#####] 100%
(1/1) loading package files [#####] 100%
(1/1) checking for file conflicts [#####] 100%
(1/1) checking available disk space [#####] 100%
:: Processing package changes...
(1/1) installing apache [#####] 100%
Optional dependencies for apache
 lua: for mod_lua module [installed]
 libxml2: for mod_proxy_html, mod_xml2enc modules [installed]
 curl: for mod_md module [installed]
 jansson: for mod_md module [installed]
 brotli: for mod_brotli module [installed]
 uwsgi: for mod_proxy_uwsgi module
 lynx: apachectl status
:: Running post-transaction hooks...
(1/3) Reloading system manager configuration...
(2/3) Creating temporary files...
(3/3) Arming ConditionNeedsUpdate...
[austin@austin-pc ~]$
```

```
(3/3) Arming ConditionNeedsUpdate...
[austin@austin-pc ~]$ sudo nano /etc/httpd/conf/httpd.conf
[austin@austin-pc ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[austin@austin-pc ~]$ sudo systemctl restart httpd
[austin@austin-pc ~]$ sudo systemctl status httpd
● httpd.service - Apache Web Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2018-10-24 12:45:49 MST; 13s ago
     Main PID: 1652 (httpd)
        Tasks: 82 (limit: 4915)
       Memory: 30.7M
      CGroup: /system.slice/httpd.service
              └─1652 /usr/bin/httpd -k start -DFOREGROUND
                └─1653 /usr/bin/httpd -k start -DFOREGROUND
                  └─1654 /usr/bin/httpd -k start -DFOREGROUND
                    └─1655 /usr/bin/httpd -k start -DFOREGROUND

Oct 24 12:45:49 austin-pc systemd[1]: Started Apache Web Server.
Oct 24 12:45:49 austin-pc httpd[1652]: AH00558: httpd: Could not reliably determine the server's fu
lines 1-14/14 (END)
```

## Installing MySQL/MariaDB

To install MySQL, I used the command “`sudo pacman -S mysql`”. During the installation, I was asked if I wanted to use mariadb or percona-server. I choose to use mariadb. Once mysql was installed, I had to run the `mysql_secure_installation` to set up the database and give me a admin account to manipulate data. After these steps were complete, I was able to log into my database to manipulate data.

```
[austin@austin-pc ~]$ sudo pacman -S mysql
:: There are 2 providers available for mysql:
:: Repository extra
   1) mariadb
:: Repository community
   2) percona-server

Enter a number (default=1): 1
resolving dependencies...
looking for conflicting packages...

Packages (4) jemalloc-1:5.1.0-1  libmariadbclient-10.1.36-1  mariadb-clients
               mariadb-10.1.36-1

Total Download Size:   32.57 MiB
Total Installed Size: 230.44 MiB

:: Proceed with installation? [Y/n] y
:: Retrieving packages...
  libmariadbclient-10.1.36-1-x86_64    4.4 MiB   844K/s  00:05 [#####]
  jemalloc-1:5.1.0-1-x86_64          300.4 KiB  392K/s  00:01 [#####]
  mariadb-clients-10.1.36-1-x86_64    1500.2 KiB 410K/s  00:04 [#####]
  mariadb-10.1.36-1-x86_64           26.4 MiB  2.13M/s  00:12 [#####]
(4/4) checking keys in keyring [#####]
(4/4) checking package integrity [#####]
(4/4) loading package files [#####]
(4/4) checking for file conflicts [#####]
(4/4) checking available disk space [#####]
:: Processing package changes...
(1/4) installing libmariadbclient [#####]
(2/4) installing jemalloc [#####]
Optional dependencies for jemalloc
  perl: for jeprof [installed]
(3/4) installing mariadb-clients [#####]
(4/4) installing mariadb [#####]
:: You need to initialize the MariaDB data directory prior to starting
the service. This can be done with mysql_install_db command, e.g.:
mysql_install_db --user=mysql --basedir=/usr --datadir=/var/lib/mysql
Optional dependencies for mariadb
  galera: for MariaDB cluster with Galera WSREP
  perl-db-d-mysql: for mysqlhotcopy, mysql_convert_table_format and mysql_s
:: Running post-transaction hooks...
(1/4) Reloading system manager configuration...
(2/4) Creating system user accounts...
(3/4) Creating temporary files...
(4/4) Arming ConditionNeedsUpdate...
[austin@austin-pc ~]$
```

```
[austin@austin-pc ~]$ sudo mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)
Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
... Success!

Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far
```

## Install PHP

To install PHP onto my system, I ran the command “sudo pacman -S php php-apache”. After the application was installed onto the system, I had to configure the Apache PHP module by editing the /etc/httpd/conf/httpd.conf file to include php7 modules, load the `mod_php7_module`, and include the `php7_module.conf` file. Once this was complete I had to restart the httpd service with the “sudo systemctl restart httpd” command (since I edited the configuration file).

```
[austin@austin-pc ~]$ sudo pacman -S php php-apache
resolving dependencies...
looking for conflicting packages...

Packages (3) libzip-1.5.1-1  php-7.2.11-2  php-apache-7.2.11-2

Total Download Size:    5.90 MiB
Total Installed Size:  32.69 MiB

:: Proceed with installation? [Y/n] y
:: Retrieving packages...
 libzip-1.5.1-1-x86_64                208.9 KiB   204K/s  00:01 [#####] 100%
 php-7.2.11-2-x86_64                  3.4 MiB    717K/s  00:05 [#####] 100%
 php-apache-7.2.11-2-x86_64           2.3 MiB    728K/s  00:03 [#####] 100%
(3/3) checking keys in keyring [#####] 100%
(3/3) checking package integrity [#####] 100%
(3/3) loading package files [#####] 100%
(3/3) checking for file conflicts [#####] 100%
(3/3) checking available disk space [#####] 100%
:: Processing package changes...
(1/3) installing libzip [#####] 100%
(2/3) installing php [#####] 100%
(3/3) installing php-apache [#####] 100%
:: Running post-transaction hooks...
(1/1) Arming ConditionNeedsUpdate...
[austin@austin-pc ~]$ sudo nano /etc/httpd/conf/httpd.conf
[austin@austin-pc ~]$ sudo nano /srv/http/test.php
[austin@austin-pc ~]$ sudo systemctl restart httpd
```

## Install phpMyAdmin

To install phpMyAdmin, I ran the command “`sudo pacman -S phpmyadmin`”. Once the application was installed, I had to edit the `/etc/php/php.ini` file to uncomment the extensions for `bz2.so` and `mysqli.so` (the `mysqli.so` will be needed when creating the forms later on with `php` and `html`). After this is complete, the next step is to edit the configuration file for phpMyAdmin located in `/etc/httpd/conf/extra/phpmyadmin.conf` to include the following lines (white text box in terminal image below). After this is added, we have to edit the Apache config file and add “`Include conf/extra/phpmyadmin.conf`” to it. Last step is to restart the `httpd` service and verify that it works.

```
[austin@austin-pc ~]$ sudo pacman -S phpmyadmin
resolving dependencies...
looking for conflicting packages...

Packages (1) phpmyadmin-4.8.3-1

Total Download Size:    5.98 MiB
Total Installed Size:  34.23 MiB

:: Proceed with installation? [Y/n] y
:: Retrieving packages...
  phpmyadmin-4.8.3-1-any             6.0 MiB   2.34M/s   00:03 [#####] 100%
(1/1) checking keys in keyring [#####] 100%
(1/1) checking package integrity [#####] 100%
(1/1) loading package files [#####] 100%
(1/1) checking for file conflicts [#####] 100%
(1/1) checking available disk space [#####] 100%
:: Processing package changes...
(1/1) installing phpmyadmin [#####] 100%
:: Running post-transaction hooks...
(1/1) Arming ConditionNeedsUpdate...

[austin@austin-pc ~]$ sudo nano /etc/php/php.ini
[austin@austin-pc ~]$ sudo nano /etc/httpd/conf/extra/phpmyadmin.conf
[austin@austin-pc ~]$ sudo nano /etc/httpd/conf/httpd.conf
[austin@austin-pc ~]$ sudo systemctl restart httpd
[austin@austin-pc ~]$ sudo systemctl status httpd
sudo: systemctl: command not found
[austin@austin-pc ~]$ sudo systemctl status httpd
• httpd.service - Apache Web Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2018-10-24 13:19:56 MST; 16s ago
   Process: 4650 ExecStop=/usr/bin/httpd -k graceful-stop (code=exited, status=0/SUCCESS)
   Main PID: 4653 (httpd)
   Tasks: 6 (limit: 4915)
   Memory: 9.8M
   CGroup: /system.slice/httpd.service
           └─4653 /usr/bin/httpd -k start -DFOREGROUND
             └─4654 /usr/bin/httpd -k start -DFOREGROUND
               └─4655 /usr/bin/httpd -k start -DFOREGROUND
                 └─4656 /usr/bin/httpd -k start -DFOREGROUND
                   └─4657 /usr/bin/httpd -k start -DFOREGROUND
                     └─4658 /usr/bin/httpd -k start -DFOREGROUND

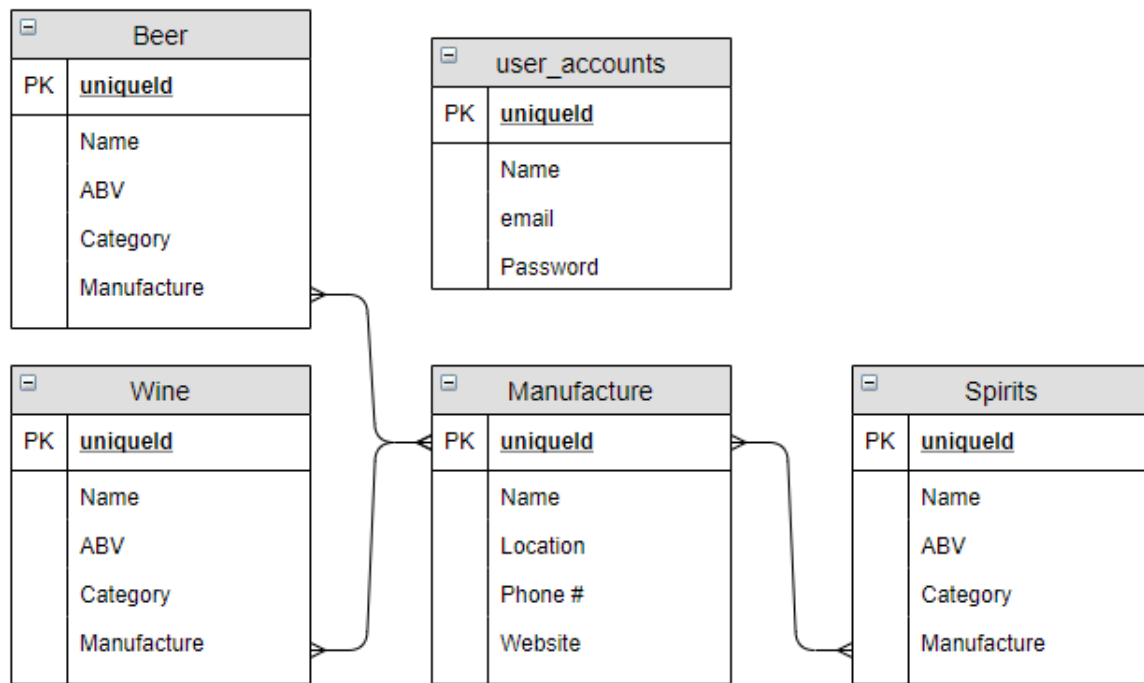
Oct 24 13:19:56 austin-pc systemd[1]: Started Apache Web Server.
Oct 24 13:19:56 austin-pc httpd[4653]: AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1 instead; this may cause incorrect
lines 1-17/17 (END)
```

```
Alias /phpmyadmin "/usr/share/webapps/phpMyAdmin"
<Directory "/usr/share/webapps/phpMyAdmin">
  DirectoryIndex index.php
  AllowOverride All
  Options FollowSymLinks
  Require all granted
</Directory>
```



## Entity Relationship Diagram

Once the applications for the LAMP stack were installed, it is time to build the Entity Relationship Diagram for the database. This diagram will identify each table that was created and their attributes. We will also identify the primary and foreign keys used to create the relationships in our database. This will also give us a visual representation of the database and workflow. For this example, all the items in the database will be linked together based off a category called Manufacture (brewing company).



## Table Creation in MariaDB

After the ERD was created and the outline of the database was complete, there was enough information available to create the tables into the database. To do this, I utilized the command line to connect to the database and create the table information. In order to do this, I had to connect to the database through the command line with the following command “mysql -u root -p -h localhost AdultBeverage”. This connected me to the database and allowed me to create the tables from the command line instead of using phpMyAdmin (alternate way)

```

Terminal - austin@austin-pc:~
File Edit View Terminal Tabs Help
| Tables_in_AdultBeverage |
+-----+
| Beer                    |
| Spirits                 |
| Wine                    |
+-----+
3 rows in set (0.00 sec)

MariaDB [AdultBeverage]> CREATE TABLE IF NOT EXISTS Manufacture(
-> Man_ID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
-> Name VARCHAR(100) NOT NULL,
-> Stree_Address TEXT, City TEXT, State TEXT, Country TEXT,
-> ZipCode TEXT, Phone_Num TEXT, Website TEXT);
Query OK, 0 rows affected (0.12 sec)

MariaDB [AdultBeverage]> show TABLES
-> ;
+-----+
| Tables_in_AdultBeverage |
+-----+
| Beer                    |
| Manufacture              |
| Spirits                 |
| Wine                    |
+-----+
4 rows in set (0.00 sec)

MariaDB [AdultBeverage]> describe TABLES
-> ;
ERROR 1146 (42S02): Table 'AdultBeverage.TABLES' doesn't exist
MariaDB [AdultBeverage]> describe Beer;
+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+
| Beer_ID    | int(11)       | NO   | PRI | NULL    | auto_increment |
| Name       | varchar(100)  | NO   |     | NULL    |                |
| ABV        | text          | YES  |     | NULL    |                |
| Category   | text          | YES  |     | NULL    |                |
| Manufacture | int(11)       | YES  |     | NULL    |                |
+-----+
5 rows in set (0.01 sec)

MariaDB [AdultBeverage]> quite
-> ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that c
orresponds to your MariaDB server version for the right syntax to use near 'quite
' at line 1
MariaDB [AdultBeverage]> quit
Bye
[austin@austin-pc ~]$

```

```

MariaDB [AdultBeverage]> CREATE TABLE IF NOT EXISTS Wine(
-> Wine_ID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
-> Name VARCHAR(100) NOT NULL,
-> ABV TEXT, Category TEXT,
-> Year TEXT,
-> Manufacture INT);
Query OK, 0 rows affected (0.11 sec)

MariaDB [AdultBeverage]> CREATE TABLE IF NOT EXISTS Spirits(
-> Spirit_ID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
-> Name VARCHAR(100) NOT NULL,
-> ABV TEXT, Category TEXT, Type TEXT, Age TEXT,
-> Manufacture INT);
Query OK, 0 rows affected (0.11 sec)

MariaDB [AdultBeverage]>

```

## PHP/HTML/MYSQL Forms

Below is an example script used to create each form and the result. This form is styled with HTML and utilizes PHP scripts to run MYSQL commands directly into the database. This allows for data to be inputted into the database with ease. The user doesn't need any knowledge of MySQL or programming to use these forms. These forms also display a query at the bottom of the form to allow the user to see if certain information is/is not in the database yet and provides the option to add the appropriate data that is missing.

```

<h1>Input a new brew <a href=" ../Home.php" style="float:right">Homepage</a>

<div id="contact-area">

    <form action="Beer_Process.php" method="post" name="Beer">

        <label for="Name">Name:</label>
        <input type="text" name="Name" placeholder="Amazing Drink" value=""
        <br>

        <label for="ABV">ABV:</label>
        <input type="text" name="ABV" placeholder="Alcohol content" value=""
        <br>

        <label for="Category">Category:</label>
        <input type="text" name="Category" placeholder="Stout, Blonde, Lager, La
        <br>

        <label for="Manufacture">Manufacture:</label>
        <input type="text" name="Manufacture" placeholder="Manufacture ID
        <br>

        <input type="submit" name="submit" value="Submit" class="submit-b

    </form>
<br><br><br><br>

</div>

<div>
    <table>
        <h2>Manufacture ID</h2>
        <caption>
            *If manufacture is not listed, please add information to data
            <a href=" ../AddMan-form.php" style="float:right">Add Manufact
        </caption>
        <?php
            $sqlMan = "SELECT DISTINCT Man_ID, Man_Name FROM Manufacture JOIN Beer ON Manufactur
            $resultMan=mysqli_query($db_connect, $sqlMan) or die("bad query");

            echo "<tr><th>ID</th><th>Name</th></tr>";
            while ($row = mysqli_fetch_assoc($resultMan)) {
                echo "<tr><td>{$row['Man_ID']}</td><td>{$row['Man_Name']}</td></tr>";
            }
        <?>
    </table>
        
```

### Input a new brew

[Homepage](#)

Name:

ABV:

Category:

Manufacture:

### Manufacture ID

\*If manufacture is not listed, please add information to database [Add Manufacture](#)

ID	Name
1	Breckenridge Brewery
2	Stone Brewing
3	Barrior Brewing Co.
4	Arizona Wilderness Brewing Co.
5	SanTan Brewing Company
6	Guinness Storehouse
7	Samuel Adams Brewery

```

//inserting data into table

$query=mysqli_query($db_connect, "INSERT INTO Beer
(Name, ABV, Category, Manufacture)
VALUES ('$name', '$ABV', '$Category', '$Manufacture');"
or die(mysqli_error($db_connect));

mysqli_close($db_connect);

//echo "Successfully Updated, Please return to forms";
header("Location: ../AddBeer-form.php?note=success");

```

## Queries / Reports

Below is the example script used to create the query/report page that allows user to query each table. The query searches two tables (the item and the manufacture) and combines the results to display the relevant information combined. Every time the user clicks on the respective query button, the information is displayed below. If new information is added to the database, it will display once the user clicks on the button again. These queries are completed using PHP to inject MySQL statements into the database and fetch the results. You can see the queries that were used below to get the appropriate information from each table

```

rm action="QueryForm.php" method="post">

<label>Beer</label>
<input type="submit" name="Beer" value="Query Beer"> <br>

<label>Spirits</label>
<input type="submit" name="Spirits" value="Query Spirits"> <br>

<label>Wine</label>
<input type="submit" name="Wine" value="Query Wine"> <br>

<label>Manufacture</label>
<input type="submit" name="Manufacture" value="Query Manufacture"><br>

<?php
if (isset($_POST['Beer'])) {
    $sql = "SELECT Beer.*, Manufacture.Man_Name, Manufacture.State, Manufacture.Country
    FROM Beer INNER JOIN Manufacture ON Beer.Manufacture=Manufacture.Man_ID;";
    $result = mysqli_query($db_connect, $sql) or die("bad query");

    echo "<table>";
    echo "<tr><th>ID</th><th>Name</th><th>ABV</th><th>Category</th>
    <th>Manufacture</th><th>State</th><th>Country</th></tr>";

    while($row=mysqli_fetch_assoc($result)) {
        echo "<tr><td>{$row['Beer_ID']}</td><td>{$row['Name']}</td><td>{$row['ABV']}</td>
        <td>{$row['Category']}</td><td>{$row['Man_Name']}</td><td>{$row['State']}</td>
        <td>{$row['Country']}</td></tr>";
    }
}

if (isset($_POST['Spirits'])) {
    $sql = "SELECT Spirits.*, Manufacture.Man_Name, Manufacture.State, Manufacture.Country
    FROM Spirits INNER JOIN Manufacture ON Spirits.Manufacture=Manufacture.Man_ID;";
    $result = mysqli_query($db_connect, $sql) or die("bad query");

    echo "<table>";
    echo "<tr><th>ID</th><th>Name</th><th>ABV</th><th>Type</th><th>Category</th><th>Age</th>
    <th>Manufacture</th><th>State</th><th>Country</th></tr>";

    while($row=mysqli_fetch_assoc($result)) {
        echo "<tr><td>{$row['Spirit_ID']}</td><td>{$row['Spirit_Name']}</td><td>{$row['ABV']}</td>
        <td>{$row['Type']}</td><td>{$row['Category']}</td><td>{$row['Age']}</td>
        <td>{$row['Man_Name']}</td><td>{$row['State']}</td><td>{$row['Country']}</td></tr>";
    }
}

if (isset($_POST['Wine'])) {
    $sql = "SELECT Wine.*, Manufacture.Man_Name, Manufacture.State, Manufacture.Country \
    FROM Wine INNER JOIN Manufacture ON Wine.Manufacture=Manufacture.Man_ID;";
    $result = mysqli_query($db_connect, $sql) or die("bad query");

    echo "<table>";
    echo "<tr><th>ID</th><th>Name</th><th>ABV</th><th>Category</th><th>Year</th><th>Manufactur";

    while($row=mysqli_fetch_assoc($result)) {
        echo "<tr><td>{$row['Wine_ID']}</td><td>{$row['Wine_Name']}</td><td>{$row['ABV']}</td>";
    }
}

```

Welcome to the Query!!
[Homepage](#)

Beer

Spirits

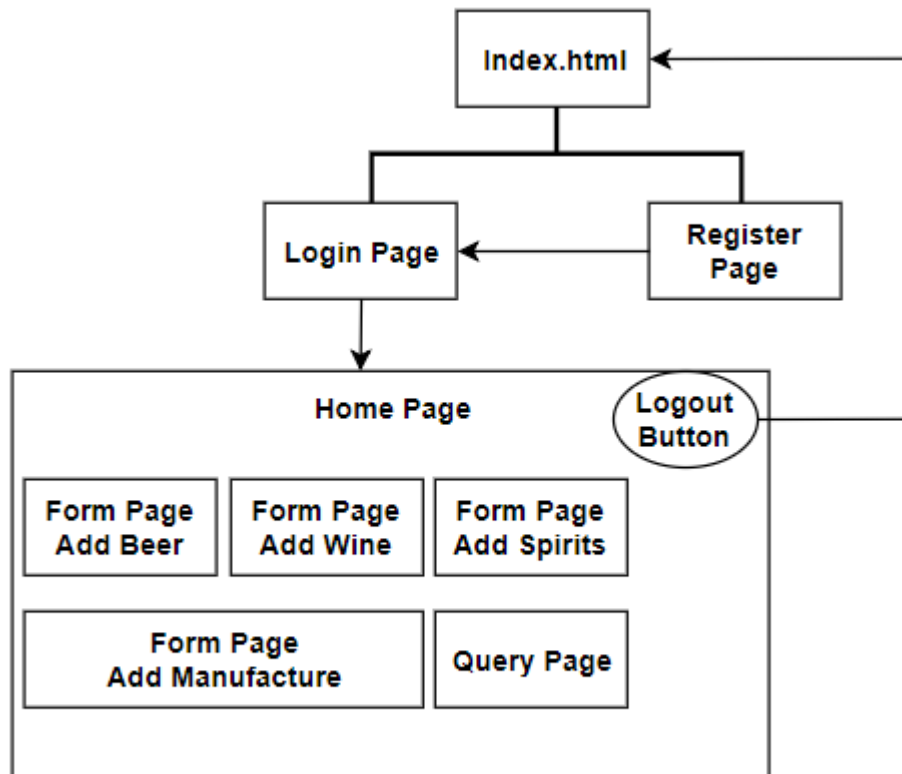
Wine

Manufacture

ID	Name	ABV	Category	Manufacture	State	Country
1	Breckenridge: Vanilla Porter	4.7	Porter	Breckenridge Brewery	Co	USA
2	Breckenridge: Oatmeal Stout	4.95	Stout	Breckenridge Brewery	Co	USA
3	Breckenridge: Christmas Ale	7.4	Seasonal	Breckenridge Brewery	Co	USA
4	Breckenridge: Twenty	7.8	Barrel Aged	Breckenridge Brewery	Co	USA
5	Stone Tangerine Express IPA	6.7	India Pale Ale	Stone Brewing	Ca	USA
6	Stone Ruination Double IPA	8.5	Double IPA	Stone Brewing	Ca	USA
7	Stone Ghost Hopper IPA	6.7	India Pale Ale	Stone Brewing	Ca	USA
8	Stone Scorpion Bowl IPA	7.5	India Pale Ale	Stone Brewing	Ca	USA
9	Banjo Blonde	4.7	Blonde	Banjo Brewing Co.	Az	USA
10	Citrazona	6.6	IPA	Banjo Brewing Co.	Az	USA
11	Copperhead Pale Ale	6.2	American Pale Ale	Banjo Brewing Co.	Az	USA
12	Mocha Java Stout	5.5	Coffee Stout	Banjo Brewing Co.	Az	USA
13	Nolan's Porter Ale	6	Porter	Banjo Brewing Co.	Az	USA
14	Abundantsea	10	Imperial Milk Stout	Arizona Wilderness Brewing Co.	Az	USA
15	Baboquivari	7.1	Belgian Blonde	Arizona Wilderness Brewing Co.	Az	USA

## Website Outline

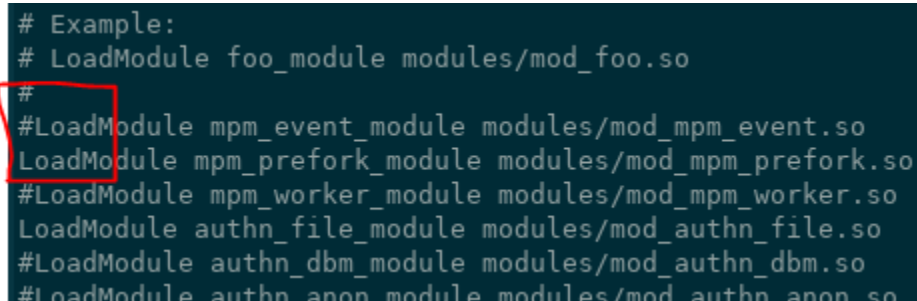
Once the pages are complete, we need a way to access the information and navigate between each form. To do this we will design a basic website layout that will allow individuals to access the appropriate forms. This will have an index page, login page (to verify users) register page for new users (this can be taken away and managed by the Admin if needed), and a Home page. The home page will be link, after the user has logged in, have all the available options for the user. These pages can be made available or unavailable based off the user's privileges. Right now, for testing purposes, the user will have access to all the pages listed below.



## Failure Report

I did face some issues when first working this project. Initially when installing Linux onto my system, everything was good until I booted back into windows. Once I did that the partitions with Linux installed onto it became corrupt and I had to redo the installation process. I found out that this was due to the UEFI boot loader and the grub boot loader needing to be updated after installing Linux. Once I completed the reinstallation and updating the bootloader everything worked fine.

Another issue faced during the installation of the LAMP stack was that I had moved through it too fast and didn't complete every step. This caused my Apache server to crash after installing php because I didn't comment out a specific line in the configuration files. This caused the wrong modules to be loaded and crashed the server. It was an easy fix that could have been avoided with attention to detail upon installation.



```
# Example:
# LoadModule foo_module modules/mod_foo.so
#
#LoadModule mpm_event_module modules/mod_mpm_event.so
LoadModule mpm_prefork_module modules/mod_mpm_prefork.so
#LoadModule mpm_worker_module modules/mod_mpm_worker.so
LoadModule authn_file_module modules/mod_authn_file.so
#LoadModule authn_dbm_module modules/mod_authn_dbm.so
#LoadModule authn_anon_module modules/mod_authn_anon.so
```

I ran into issues with my programming. I had syntax and grammatical errors that were hard for me to find. This was largely due to me typing too fast which resulted in misspelled words/commands. These errors caused the entire program to malfunction, or not display any information.

On the bottom of each form, I used a query command to display the information only relevant to the current table based off the manufacture. I ran into the issues of it displaying repeated information over and over. Upon investigation, I found out that I need to use the SELECT DISTINCT command instead of just SELECT for the query to keep this from happening.

```

    $sqlMan = "SELECT DISTINCT Man_ID, Man_Name FROM Manufacture
JOIN Beer ON Manufacture.Man_ID=Beer.Manufacture WHERE
Man_ID=Beer.Manufacture";
    $resultMan=mysqli_query($db_connect, $sqlMan) or die("bad
query");

    echo "<tr><th>ID</th><th>Name</th></tr>";
    while ($row = mysqli_fetch_assoc($resultMan)) {
        echo "<tr><td>{$row['Man_ID']}</td><td>{$row['Man_Name']}</
td></tr>";
    }

```

When conducting the query for the query page that joins multiple tables together in the results, I had an issue with the join statement that I was using. My JOIN statement was working but the while loop that I created to show the results were not working. I realized that this was because the table names were not unique. I used Name for both tables and it didn't know which it was supposed to display. When I changed the tables headers from Name to "Spirit\_Name" and "Man\_Name", the results displayed like they were supposed to.

```

if (isset($_POST['Spirits'])) {
    $sql = "SELECT Spirits.*, Manufacture.Man_Name, Manufacture.State, Manufacture.Country
FROM Spirits INNER JOIN Manufacture ON Spirits.Manufacture=Manufacture.Man_ID;";
    $result = mysqli_query($db_connect, $sql) or die("bad query");

    echo "<table>";
    echo "<tr><th>ID</th><th>Name</th><th>ABV</th><th>Type</th><th>Category</th>
<th>Age</th><th>Manufacture</th><th>State</th><th>Country</th></tr>";

    while($row=mysqli_fetch_assoc($result)) {
        echo "<tr><td>{$row['Spirit_ID']}</td><td>{$row['Spirit_Name']}</td>
<td>{$row['ABV']}</td><td>{$row['Type']}</td><td>{$row['Category']}</td>
<td>{$row['Age']}</td><td>{$row['Man_Name']}</td><td>{$row['State']}</td>
<td>{$row['Country']}</td></tr>";
    }
}

```

## Conclusion

I learned a lot during this process about mariadb, php, and apache. A large part of this was due to the fact that each application had to be installed and configured manually in the Arch Linux distro I was using. I am happy with my choice to use a LAMP stack because I have been able to find a lot of resources on the internet to help me resolve any issues I have had so far. I have learned through this process that it takes time to develop anything and sometimes you need to walk away and come back with a fresh set of eyes. Overall, I am happy with the outcome of my database and products for this presentation.



## References

Manjaro Linux installation steps retrieved from: <https://forum.manjaro.org/t/install-apache-mariadb-php-lamp-2016/1243/1>

MariaDB guide retrieved from: <https://mariadb.com/kb/en/library/a-mariadb-primer/>

PHP7 tutorial from tutorialspoint.com retrieved from:

[https://www.tutorialspoint.com/php7/php7\\_scalar\\_type\\_declarations.htm](https://www.tutorialspoint.com/php7/php7_scalar_type_declarations.htm)

HTML/PHP/MYSQL form tutorial retrieved from:

<https://www.youtube.com/watch?v=yhsVNdNF9sU>

HTML/PHP/MYSQL tutorials retrieved from: <https://www.w3schools.com/default.asp>

How to create login system YouTube.com retrieved from:

<https://www.youtube.com/watch?v=LC9GaXkdxF8&index=4&t=0s&list=PLgJYeu29xjqMAyRRw--jmYC6wBKHb6vpb>

Display query results in html table Youtube.com retrieved from:

<https://www.youtube.com/watch?v=pc0otVM80Sk&index=3&t=0s&list=PLgJYeu29xjqMAyRRw--jmYC6wBKHb6vpb>