Module 01

# Set Up Local Development Environment

Joshua McCann

# Intro

The following information is an examination of the process of creating a Web App to work with MySQL in Microsoft Azure. The information is broken up into the step by step process, screen shots, and a reflective paragraph of the process.

# Step-By-Step Installation

1. Create the web app in Azure
   1. Log into Microsoft Azure at portal.azure.com
   2. Click on the + sign to install a new tool
   3. Search for Web App + MySQL
      1. Enter in the app name (I chose the name dbjmccanwebdev)
      2. Enter in the resource group (I chose new resource group named AdvMobWebDev)
      3. Select the App Service plan and Database name (I used ClearDB for the provider)
      4. Because I did not have a database ready to go, I created a new database from the app name. Azure then created both the app and the database together.
   4. The next step was to create a FTP user name to login with
      1. Click on Deployment Credentials
      2. Enter the deployment username (I chose jmccannWebDevAdmin)
      3. Enter the password (I will not be including the password)
2. Set-up FileZilla to work with the FTP protocols
   1. The next step is to setup the FTP protocols to allow FileZilla to connect for uploading purposes
      1. This is as simple as copying the correct information over from Azure into FileZilla
         1. Host url and user can be copied directly from the AppService information
         2. The password created before needs to be input
         3. Logon Type, Encryption, and Protocol should be left the same
   2. Once everything is created, connect to the FTP using the saved Site Manager
3. Set-up the database
   1. First, to modify the data, phpMyAdmin needed to be installed on the web app in Azure
      1. Go down to Development Tools and click on Advanced Tools
      2. Click on Go in the next field
      3. Find and install phpMyAdmin using
         1. In the new window, click on “Site extensions”
         2. Click on “Gallery”
         3. Click on “phpMyAdmin” and click on “install”
         4. Click on Accept on the notification
         5. Restart the Site
   2. The database was completely clear when installed on with no data. For this test data needed to be uploaded into the system.
      1. Go to generatedata.com
      2. Setup some data types to include in the upload
      3. Copy over the SQL Query to manually input a new table into the phpMyAdmin
4. Upload HTML and PHP files
   1. Navigate to the wwwroot folder in FileZilla and upload the PHP and HTML
      1. I uploaded a HTML base file using jquery mobile I had created previously. Before uploading, I updated much of the previous code into the mobile app and created a base template for use.
      2. I uploaded the PHP files included with the class, modified and updated the code to use the CSS files for uniformity with the html file and updated the PHP code to work with the database on my Azure Web App
5. Set-up Adobe DreamWeaver to work with the FTP protocols
   1. Because the database code does not work unless installed on the server, the database code could not be tested appropriately until after it was uploaded. In order to streamline this process, Adobe DreamWeaver was connected to web app server using the same FTP protocols as FileZilla
      1. This is as simple as copying the correct information over from Azure into FileZilla
         1. Host url and user can be copied directly from the AppService information
         2. The password created before needs to be input
         3. Logon Type, Encryption, and Protocol should be left the same
   2. Once connected, the html and php files could be modified and synchronized on the server

# Screen Captures

# 

Figure - Initial Settings Azure

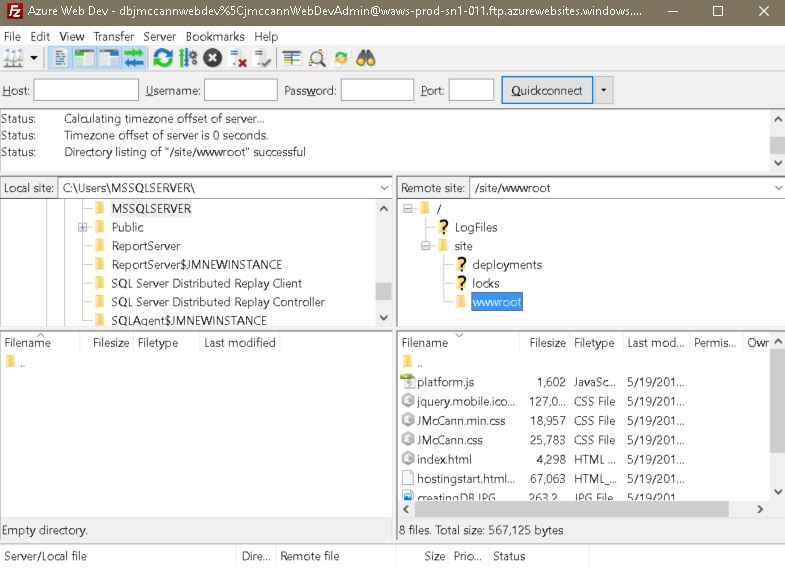


Figure - Uploading HTML and PHP Files

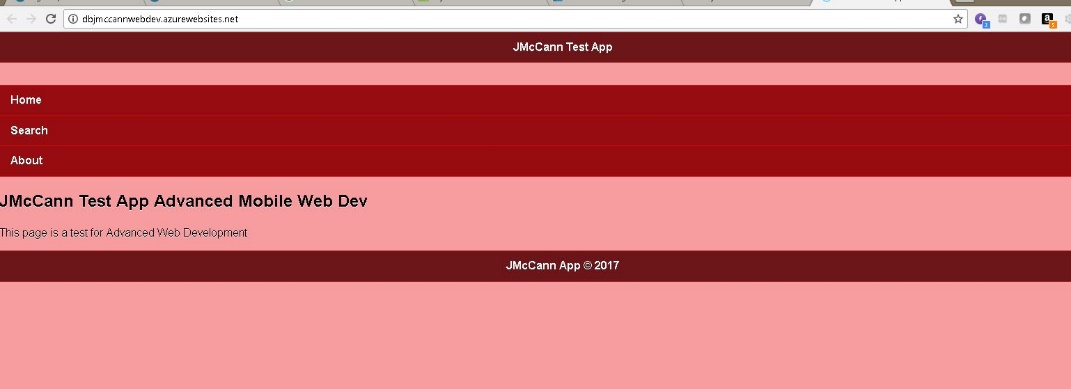


Figure - Displayed HTML

# Reflection

This was actually a very easy process for me since I had to create an Azure database in a different class. I ended up modifying the PHP code in the end to change how the information was collected. The changes I made included wildcard into the search, allowing the search to be more robust, and including the email as a display item. I also changed the limit so multiple query results could be displayed.

The updated code can be found below:

$sql= "SELECT id, fname, lname, email

FROM testtable

WHERE fname LIKE CONCAT('%','$fname','%') AND lname LIKE CONCAT('%','$lname','%') LIMIT 50";

$result = mysqli\_query($conn, $sql);

if (mysqli\_num\_rows($result) > 0) {

// output data of each row

while($row = mysqli\_fetch\_assoc($result)) {

echo "ID: " . $row["id"]. "<br>";

echo "First Name: " . $row["fname"]. "<br>";

echo "Last Name: " . $row["lname"]. "<br>";

echo "Email: " . $row['email'] . "<br><hr>";

}

}