

David Gabay

Project 7

Image uploader using all languages and tools

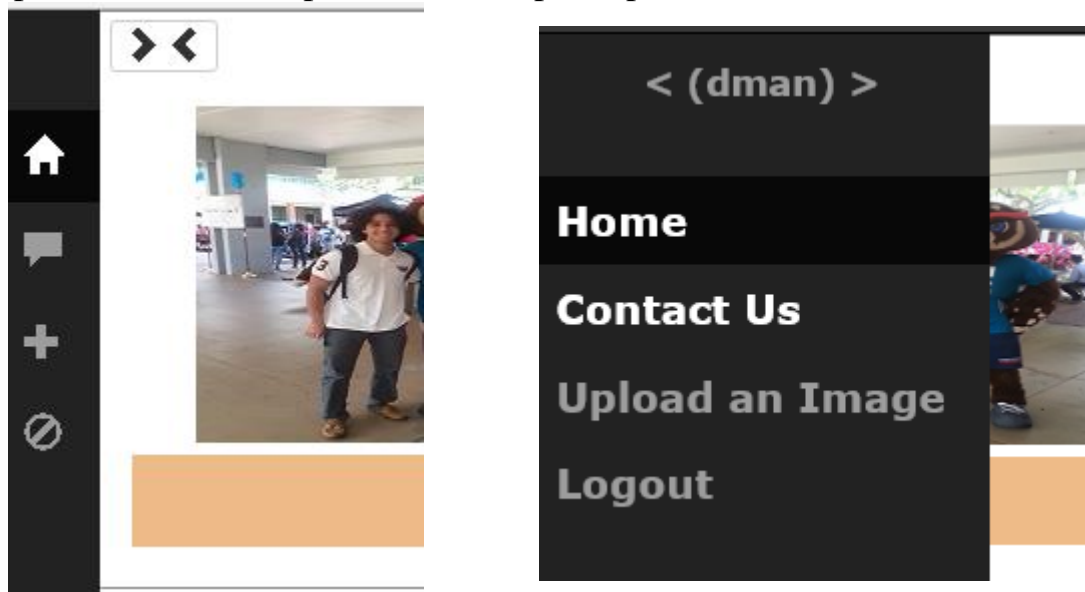
This project was most difficult combining all my skills and knowledge to create a full webpage. I used a lot of the basicuploader example to guide me through the concepts and managed to get about 95% or more of the required functionality to work and I only had trouble with the image filters section and saving the image to my desktop folder labeled users. However, the images saved to the remote folder called users so this was ok. Overall the project pushed my skills to levels they have never reached and I was able to create a website that was very modern and useful in today's scenario. In this project, I also followed a few of the examples given in Nixon's book to guide me through some of the features.

At first I focused on getting the bulk of the functionality to work this meant having poor visuals and good functionality. To start this I used some of the examples from Nixon's book, so first I made sure I was able to connect to MySQL database and create a table called 'WALL' which contained all the required fields including the name of the img, the user, etc. Later on, I was asked to alter the table so it would contain another column that would hold the filter of the image that was chosen. So, I had lots of trouble in this section because I was using the starter basicuploader code and it was saying that there were too little parameters or too much parameters in the code. However, after checking the code the parameters were fine and for hours I could not debug the issue. This was highly annoying, so I kept browsing the internet to see if I could find any clues why this may be happening eventually I came to the php original documentation for the prepare and binding of the parameters and I was not aware of the first parameter being ssssss means that the parameters being entered were all strings.

```
11
12 function SavePostToDB($db, $user, $filter, $title, $text, $time, $file_name)
13 {
14     /* Prepared statement, stage 1: prepare query */
15     if (!$stmt = $db->prepare("INSERT INTO WALL(USER_USERNAME, STATUS_TITLE, STATUS_
16     {
17         echo "Prepare failed: (" . $db->errno . ") " . $db->error;
18     }
19
20     /* Prepared statement, stage 2: bind parameters*/
21     if (!$stmt->bind_param('ssssss', $user, $title, $text, $time, $file_name, $f
22     {
23         echo "Binding parameters failed: (" . $stmt->errno . ") " . $stmt->error;
24     }
25
26     /* Prepared statement, stage 3: execute*/
27     if (!$stmt->execute())
```

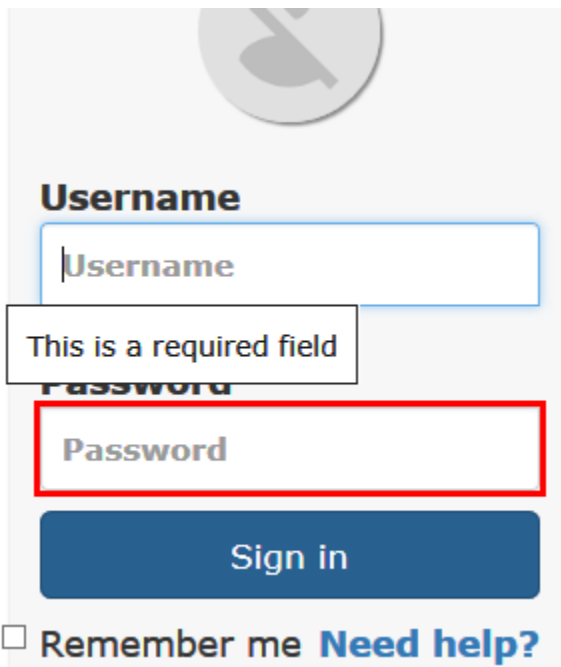
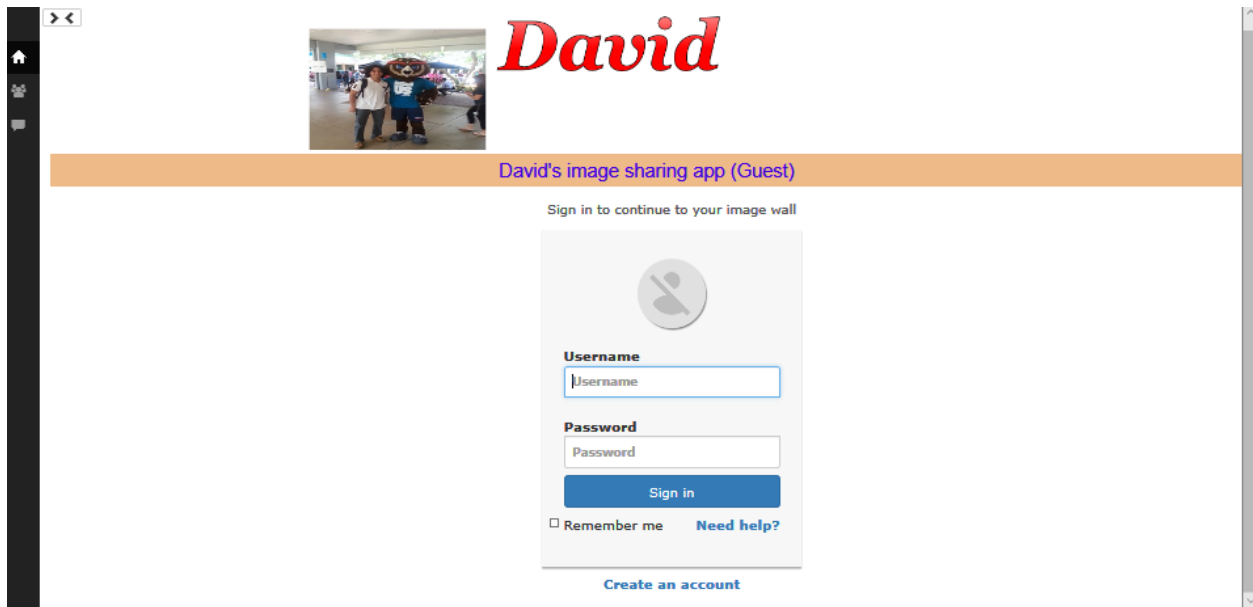
Thus I kept leaving the extra s out since I had no clue what it stood for. So this was a learning experience that will not occur again. One troubleshooting method I used for this was I performed the insertion to the database in a method that used a straight forward grab parameter and insert them into an insertion string, then query the string. This method worked so I was sure it had something to do with the prepare and bind_param functions which I was uneducated on.

The next part I started working on was getting a more mobile friendly website I did this using the bootstrap modules. And I started with the header file instead of the ancient look it had before from Nixon's examples I looked into getting a more modern header which would allow the user to navigate throughout the website. I found an example from bootsnipp a bootstrap user community that post different examples of bootstrap components.



So, I adapted this example which I went ahead and changed the glyphs used from bootstrap. From this side of page navigation bar which changed whether you are signed in or not, you can navigate to any page you need including the home, upload image, sign-out, and sign in page. It also has a button that when it is clicked the navigation bar expands and allows the user to be able to read what each button in the header does. I also had issues with implementation of this header since I needed to import the right JavaScript, and CSS to make the header function and look correctly. I didn't know that I had to import a file with all the fonts and glyphs required for the headers.

Next, was the sign in page, where I again started from the Nixon's book to get an idea of how I would implement the sign in feature. From there I looked to






























the internet to try and find some examples on how the sign in page should look and how to implement it with my current project. On Bootsnipp I found an example that worked fine and looked visually pleasing. So, I used this template within the index.php file and basically if there was a user already signed in the div containing the sign in section will not be displayed. If, however the user didn't sign in yet it displayed the sign in

page, where the user can enter there sign in information then click submit and be directed to my authorization php file. When the user enters the correct credentials to sign in he is taken to the auth.php file where if the entered information is correct he is automatically redirected to a success page and 3 seconds later redirected again to the home page. My redirect page is very bland as of now and it would definitely be something I improve on given I had more time. Another interesting addition to my sign in page is the Create an account link on the bottom this is used as a link to the sign up page.

The sign in page also has some error handling features such as the one seen here that is you cannot sign in unless all fields have been completed.

A key feature to simplifying the entire project was using the concept showed in Nixon's book about using a single header.php file in almost all of the displayed screens this saved me the work of creating a navigation bar on every page in my project. This however required me to make an adaptive header/ navigation bar, because showing the same features on the navigational bar would not be acceptable since a user who has yet to log in or create an account should not have a view of a link to see all photos on the wall or even a link to add an image.

On the sign up page I am currently using the Nixon book example however I plan on re designing the entire page to look more appealing and make room for more error handling. But it does function well and serves its purpose by allowing the user to enter their username and password and hit submit.

 		userid	password
<input type="checkbox"/>	 Edit  Copy  Delete	bsmith	32aa0c466818e1ccba25b8793db98c94
<input type="checkbox"/>	 Edit  Copy  Delete	pjones	53eb1f29c1f8a132441a4fad1d6f667d
<input type="checkbox"/>	 Edit  Copy  Delete	dman	dfa1f555d38f5ccf38e1aeafcf2c07fb
<input type="checkbox"/>	 Edit  Copy  Delete	oren123	638c6810dd38432a6ffadfce235c4ab
<input type="checkbox"/>	 Edit  Copy  Delete	oren1234	920688bd88d6a1ba0e177f17bdf6dddd
<input type="checkbox"/>	 Edit  Copy  Delete	game	8b10b276e58bdcc69f746215cc31c49a
<input type="checkbox"/>	 Edit  Copy  Delete	example	920688bd88d6a1ba0e177f17bdf6dddd
 Check All / Uncheck All With selected:  Change  Delete  Export			

In this example you can see that the usernames are saved and the passwords are hashed for security purposes. Specifically the password for the username example is "password".

One feature I did add to the sign up page is whenever the user types their desired username, if the username is already taken I display a red x saying that username is already taken. And vice versa when the username is available I display a green check mark saying that username is available. I implemented this feature using AJAX which makes an http request to checkuser.php with the value of the username entered and in checkuser.php that name is compared to all the names in the database, and returns the results in the form of the green check or red x. this is also a great form of error handling since now there can't be two users with the same username, however if the user really wants he can still submit the signup with

the same username despite the message. This is a problem I would have to fix in the future, given more time.

David's image sharing app (Guest)

Please enter your details to sign up

Username

example

✓ This username is available

Password

••••••••

Sign up

In this example you can see that when I type the same username to sign-up the second time an error statement is displayed.

David's image sharing app (Guest)

Please enter your details to sign up

Username

example

X This username is taken

Password

••••••••

Sign up

Next, was the form where you entered the details of the image you are going to upload. This page has a field where you enter the name, title, comments upload a photo, and apply a filter. The name, title, and comments section all have box to enter text which are all a required field for submitting the photo to the wall. To upload the photo you hit the browse button where you can choose a photo and it is displayed on this screen once chosen. Last is the filter section with three options greyscale, Nostalgia, and original, these are radio buttons thus one of the three must be selected once you choose to have a filter and click the button. Here you can see an example of how a filter looks before you upload the image.

< Bootsniip >

Home


About Us

Contact Us

Name

Title

Text



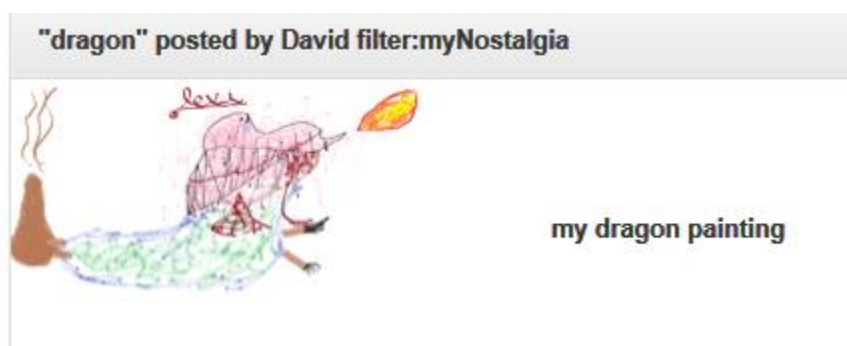
Filter Photo

My Nostalgia ☒
Grayscale ☐
Revert to Original ☐

The image wall is a list that displays all the images uploaded to the wall and that are contained in the database in phpmyadmin, which also contains the filter that they use.

			USER_USERNAME	STATUS_TEXT	STATUS_TITLE	IMAGE_NAME	TIME_STAMP	IMG_FILT
<input type="checkbox"/>	Edit	Copy Delete	David	my dragon painting	dragon	1491950535.jpg	1491950535	myNostalgia
<input type="checkbox"/>	Edit	Copy Delete	David	example	cool	1491950667.jpg	1491950667	grayscale
<input type="checkbox"/>	Edit	Copy Delete	David	example	cool	1491950693.jpg	1491950693	myNostalgia
<input type="checkbox"/>	Edit	Copy Delete	David	example	halo	1491950924.jpg	1491950924	myNostalgia

Here you can see how the dragon painting image below looks on the database.



I can also work on making the wall more appealing and display more information including the time they uploaded the image. I did however have an issue which I still can't debug and that is the following error:

```
Notice: Undefined variable: server_root in  
/home/dgabay2015/public_html/p7/BasicImageUploader/php/functions.php on line 51
```

This error is due to the `server_root` variable being undefined however I have included the file that contains that variable and still the error occurs, this error also seems to be the reason why the image never gets saved to the desktop folder anymore as opposed to before implementing the filter option.