**Final Project Part 2: Ji-Rex**

**Alex Pan, Jihea Khil, Jiho Chang, Ji Young Lim**

**Website Layout**

The entire website will have the same header and footer that will include: Organization Logo, Login, Members (if member logged in), Search, and icons that link to Social Media Websites. Moreover, the website will have six main categories that will appear in a “menu bar” all across the different pages: About Us, Projects, Gallery, Donations, Contact, and Blog. All of these categories will have subcategories that will appear as a drop down menu when the user hovers over each category. A line will appear under the category that the user clicks or hovers over to indicate where they are on the website. The user will not have an option to click on the main categories, and the user can only be able to click on the subcategories. For example, if the user wants to see the “Projects” category, the user will have to click on “past,” “current,” or “future” to see the actual contents. When the user clicks on “past” and reaches the bottom on the content, there will be links that will guide the user to the “current” page or the “future” page (Details on the subcategories will be in the paragraph below). We could have had all of the subcategories of the main category in one page. For example, for the “Projects” category, we could have had the “past,” “current,” and “future” all on one page. However, after careful consideration, we decided to separate all the subcategories into separate pages. It would be easier for the webmaster (admin) to edit each category separately, since he or she will be editing only one at a time. Moreover, if there is a problem after editing the page, only one section will have a problem instead of the entire main category. It will also make it easier for the users to navigate through each page, without having to scroll up and down too much. Overall, for the design purposes and the reasons above, we will have each subcategory into separate pages, and the users cannot click on the main categories, but use it to hover over to click on the subcategories.

By clicking the "Members" top-level link, the menu bar will change to different categories: Profile, Files, Admin. Each category will have different subcategories. Profile will have View Profile and Edit Profile. Files will have Upload Files and Download Files. Admin will have Manage Members, Pending Requests, and Change Contents. The overall layout will be the same as the main website, but Members (and admins) will have more pages that they can access. Since these pages are irrelevant to ordinary users, we decided to display them only to the members when they are logged in. By having a separate menu bar, it also distinguishes which pages are allowed to members-only.

**Search**

There will be a search bar on the top all throughout the website. This will allow the users to find the information that they need in a fast and easy way. Once the user inputs a word, we will create a search functionality that searches through tags, captions, and contents on the website. The results will be displayed on a different page, and the users will be directed to where they want to be when they click on the link of their choice.

**Login**

There will be a login system for the users. When the user clicks on “Login” there popup on the screen using Lightbox via Javascript. Once the user and password validates, the Login on the top will change to Log Out. This will also create a new button and give more access to those members with authorities. For those without login information, they can register by clicking on the link below the user/password box. If they forgot their username they click to send an e-mail with their username or if they lost their password they can chose to send an email to reset on this page.

**Member**

When the member logs in, they would be able to access the Members page located on the top of the page next to the login. The members’ page is divided up into three main categories: Profile, File, and Admin.

**Profile**

When the members hover over Profile, they can view their information on their name, user ID and password, when they joined the organization, and membership status. They can also see the documents that they uploaded. Another subcategory of Profile is “edit profile.” The user can edit all their information on this page.

**File**

File has two subcategories: view file and upload file. All the members with access to this page will can view the files. Those who have are granted access by the admin to upload files can upload the files.

Both the pages will look like a forum style. The most recent posts will be on the top, and the users can download the information from the database from “view the files” page. For the “upload file” page, the member will be asked to provide all the information including appropriate names before successfully uploading the document.

**Admin**

Admin category will have three subcategories: Change Membership, Approve Requests, and Change Contents. These are the three main functionalities that will be performed on Admin page. Change membership subcategory will display a table of all the members and their information. There will be a checkbox at the end of each row, so that the admin can select multiple members and change their level/permission or delete their info (expel membership). Approve requests will display a table of all the members who requested a change in their level. It will show their current level and the level that they desire. There will be a checkbox and an “Approve” button next to each of them so that the admin can approve multiple of them or one by one. Change contents will have sub-menus of Donations, Projects, Gallery, and Blog. By clicking different pages, the admin will be directly to another page with different forms so that the admin can easily change information of those pages. Different pages will display different forms depending on what kind of information it contains.

All six categories in the “menu bar” will have subcategories.

**About Us**

About Us category will have four categories: Mission&Purpose, History, Executive Board, Chapter Presidents. Users who visit this website for the first time will want to know more about the organization. This main category serves this purpose. The users will be able to click on the “Mission&Purpose” and know what ideology or belief the organization is based on. This will state the mission statements and purpose of the organization. The users will also click on “History” to find more about the organization’s history. There will be a timeline on when the organization was first founded, establishment of new chapters, and the highlights for each project. On the Executive Board page, there will be pictures and descriptions of each executive board member. Similarly, the Chapter Presidents will also have pictures and descriptions of each individual. The reason for these subcategories is for the users to know the leaders who are leading this organization. All of these subcategories explain what the organization is “about.” We believe that it will be easy for the users to find these four subcategories in the “About Us” section.

These four pages will consist primarily of text and pictures. The users can access all the other sites by clicking on the dropdown menu for “About Us” on the “Menu bar” All of the contents on this website will be similar to what is on the [www.silkroadinc.org](http://www.silkroadinc.org) website.

**Projects**

Projects category will divide up into Past, Current, and Future. We placed these three into the “Projects” category since each of them will explain in detail about the “Past” projects, “Current” projects, and “Future” Projects. Since SilkRoad is a non-profit organization that has annual projects that represent their organization, we believed that this should be one of the main categories. For the online users who are interested in knowing more about what the organization does and have done, they can find out specific information about the projects through the “Projects” category. We divided the subcategories into three parts because there are more than six past projects so far, and with proper descriptions and pictures the page will get too long if we combine the current and future projects into one page. It is easier and clearer for the users to see the specific projects in at one sight by clicking on “past,” “current” or “future” projects. We want the users to easily navigate through the website and get the feeling that our organization is professional.

For the specific categories of the website, past project page will contain pictures and descriptions of the past school building projects and shelter building for countries that faced natural disasters. Since there is only one big project every year, it would not be too much for users to scroll down to read the information. For “Current” project subcategory, the users will also be able to see pictures of the new site and location as a moving picture using javascript. Underneath the picture will be a short description on the specific details about the project. For more details, such as specific finances, the users can download files for how much money the organization needs to build the finish the project. For the “Future” project subcategory, we will have a list of all the upcoming projects.

**Gallery**

The gallery will have region, chapter, and activities as the subcategories. By clicking on “Region”, the users can choose East Asia, South Asia, South East Asian, and America. They will be able to see the gallery by the region of their choice. By using AJAX, further options will appear next to the last dropdown menu bar, until there are no more options to choose from. Since SilkRoad is an organization that has chapters in different regions and projects across the world, it would be easier for the users to navigate based on region. As for the pictures for each chapter, the users can click on the “Chapters” subcategory and choose the chapter of their choice to see the entire gallery. Users who want to participate in certain chapters could see pictures from those at their school or close to where they live. For those who are interested in SilkRoad activities, they can click on “Activities.” The activities will divide into fundraisers and projects.

There will be a map on the “Region” page that the users can click on. If they click on America, they will see pictures that were taken in America. If they click on China, they will see pictures that were taken in China. There will be a sort menu above the pictures, and the users can choose which continent, country, and date the pictures were taken (Users can choose a time frame: December 2010 to January 2011). For the “Chapter” gallery page, there will be a picture of the chapter (like an album in Project 3) that will link to the specific gallery for each chapter. There will be a sort menu that the users can use. They can use the drop down menu for the different chapters, and choose a time frame. Similarly, for “Activities,” there will be two pictures that the user can click: fundraisers and projects. Depending on what the user clicked, they can see all the pictures from fundraisers and projects. Using the drop down sort menu and time frame (giving a range for the dates), the user can limit the pictures they want to see.

**Donate**

The Donate category will divide into two subcategories: “Help whom?” and “Donation.” These two categories fit into these two topics. The “Help whom?” is a game designed for those who don’t know what project they want to fund. Although they can donate to the organization, and let the organization to decide on where they need to use the money, this game is to help the users to decide and actively participate on who to help and actually see who they are helping and how they are being helped. This game will eventually lead the user to a specific donate page. For users who have a sense of where to donate, they can click on the “Donation” page. The user can use to donate to the organization, fund an entrepreneur, and fund school materials.

The client doesn’t want to make the “Donate” category too complicated. They said the more complicated it is, the less people are likely to donate. SilkRoad wants three different categories. For those who don’t know who to donate to, we will create a short ‘donation game’ to see who the user is more willing to donate to. There will be short AJAX/PHP generated questions that the users will answer until a specific clause is reached. At the end of the game, there will be a donation that the user may be interested in. The user can use donate an amount of his or her choice from that page. “Donation” page will have three pictures that the user can click on. One will be for the organization, second one will be for a microfinance project to fund an entrepreneur, last will be to fund for school materials. There will be pictures that depict each of these categories. Once the user clicks on this, the user can put in the amount of his or her choice. The user will be able to see how much people have donated so far on the website.

**Contact**

The contact page will have subcategories of Facebook, Twitter, and Contact. For those who miss the little icon on the top left corner, they can click on “Contact” and reach the organization through Facebook or Twitter. There will also be a Contact form that will ask for a proper e-mail, name, and content for questions and comments. All of these three categories are relevant in that it allows the users to contact the organization through social media and e-mail.

The Facebook and Twitter subsections will actually link to the SilkRoad Facebook and Twitter page. Hopefully this would allow the users to like or follow the organization’s page. The contact form will have a simple form with e-mail, name, and content. We will use AJAX to valid the forms and return error messages if needed.

**Blog**

Blog category will have subcategories of each chapter. Since only Chapter Presidents (Permission = 2 🡪 we have to add new permission level in the database section) can write blog posts, it will be most reasonable to divide the blog page by chapter presidents. If Chapter Presidents are logged in, there will be a “write” button on the very top right corner of every blog page, so that they can easily write new posts. Even if they are on different subcategories, the “write” button will be still on the top so that they can automatically write on that specific subcategory (chapter).

On each page of Blog category, all the blog posts written in that subcategory will be displayed, from the most recent to the oldest. There will be different page numbers both on the top and bottom of the page so that users can easily navigate through the blog posts (Example: 1 . 2 . 3 . … >> last). Default display will be 10 blog posts per page, but we will also have a dropdown bar to allow the user to select how many blog posts they want to view per page (1, 5, 10, 25, 50). Blog page will also have a search input box on the top to let the users search through blog posts only. By clicking advanced search button, they will be able to search by Author, Title, Contents, Date, or other attributes of the blog post.

Each blog post will have “Go to Top” link at the end so that the user can go to the top of the page easily (to search or navigate through different pages). There will be comment section on the bottom of each page also. It will be only visible when the user click “View Comment” link, because it will be too cluttered if there are a lot of comments for each blog. Writing comment section will be only visible to logged in members. If not logged in, it will say “Please log in (with a link) to write a comment.” If logged in, it will automatically show their name and first 4 letters of their username, followed by leftover number of asterisks. (Example: Firstname Lastname (abcd\*\*\*)). There will be a text area to write their comments and submit button to submit. After the comment is submitted, it will be displayed on the bottom of the blog post. If the logged-in user writes a comment, it will have “edit” and “delete” buttons next to that comment. Admin and Chapter President will have these buttons for any comment.

On the right side of every blog page, there will be a separate box for Tags. When writing a blog post, people can add tags or choose from the list of tags already used before. Example: Piazza. Tag box will show tags in different font sizes depending on the popularity of the tags. (Top 1-5 will be large, top 6-10 will be medium, leftover will be small). The tag box will be always on the right side, even if the user scrolls up and down the blog page, so that the user can easily navigate to different pages.

Photos

Albums

SubRegion

Region

Regions\_IN

Photos\_IN

SubRegions\_IN

LikedPost

Tags

Tagged

IsComment

Posts

Comments

Users

Donations

Causes

Projects

Files

**Database Design (Relational Database Schema):**

The design contains 18 tables: Albums, Photos, Photos\_IN, SubRegions\_IN, SubRegions, Regions, Regions\_IN, Users, Post, LikePost, IsComment, Comment, Tagged, Tags, Cause, Donations, Files, Projects

A well-designed database system gives us the tools to add more functionality to the SilkRoad website. First of all, by having an internal database backend system, The SilkRoad will behave as a dynamic website that is tailored to individual users. It will also allow us to implement search features that can query across any data content we have already stored in the database. Furthermore, the users would be able to actively interact by changing and uploading documents and files into the database. Only one admin will have access to deleting the files and granting certain members access to secure data. Our database will contain information regarding users, photos and albums, blog posts and blog comments, donations made and by whom. Below is a table translation of the ER diagrams. For each table, a description of the attributes, keys, and constraints are given.

**Albums:**

1. Album\_ID - Int (Primary Key and Autoincremented)

2. Title - Varchar(1000)

3. Created - Date

4. Last\_Mod - Date

The Album\_ID is the unique id that will serve as the primary key to this table. The Title field is the title of the album, the Created field is the date the album was created, and Last\_Mod field is the date the album was last modified.

**Photos:**

1. Photo\_ID - Int (Primary Key and Autoincremented)

2. Name - varchar

2. Caption - Varchar

3. Url - Varchar

4. Date\_Taken - Date

The Photo\_ID is the unique id that will serve as the primary key to this table. The Caption field is the caption associated with the current picture, the Url field is the url of where the picture is, and the Date\_Taken field is the date of when the photo was taken.

**Photos\_IN:**

1. Album\_ID - Int (Foreign Key from Albums Table)

2. Photo\_ID - Int (Foreign Key from Photos Table)

Entries in this table show the relationship between Photos and Albums. If there is a entry that links album #1 to photo #3 for example, then this shows that photo #3 is in album #1. The Album\_ID will match a single entry in the Albums table and the Photos\_ID will match a single entry in the Photos table.

**SubRegions\_IN:**

1. SubRegion\_ID – Int (Foreign Key from SubRegions Table)

2. Photo\_ID - Int (Foreign Key from Photos Table)

Entries in this table show the relationship between a photo and which subregion the photo is in. There is a constraint here that says that a Photo must belong to at least one subregion.

**SubRegions:**

1. SubRegion\_ID – Int (primary key)

2. Name – VarChar

Entries in this table represent a SubRegion. Each photo is associated with a region. Regions are further classified into subregions. The name field just names the region, like Korea subregion inside of the Asia region.

**Regions\_IN:**

1. SubRegion\_ID – Int (Foreign Key from SubRegions Table)

2. Region\_ID – Int (Foreign Key from Regions Table)

Entries in this table represent relationships between SubRegions and Regions. There is a special constraint here that says that each SubRegion belongs to exactly one Region.

**Regions:**

**1.** Region\_ID – Int (primary key)

**2.** Name – VarChar

Entries in this table represent a Region. Each photo is associated with a region. Regions are further classified into subregions. The name field just names the region, like Korea subregion inside of the Asia region.

**Users:**

1.User\_ID - int (primary key)

2. Username - Varchar

3. Hashpwd - Char(64)

4. Name - Varchar

4. Permission - Int

Entries in this table represent users who can log into the website. The Username field will be the primary key, the Hashpwd field is the hashed version of the user's password (hashing will be done as discussed in class), the Name field is the name of the user, and the Permission field will be a integer that signifies how much permission the user has over the whole website (Member = 1, ChapterPresident = 2, Admin = 3).

**Posts:**

1.       Post\_ID - Int (Primary Key and Autoincremented)

2.       User\_ID - Int (Foreign Key from Users)

3.       Post – Varchar

4.       Url – Varchar

5.       Date – DateTime

6. NumLiked - Int

Entries in this table represent a blog post. A blog post will keep a record of when and who posted the entry. Furthermore, it will have the text content stored in the Post column and the user could also decide to add a picture, which will be stored in the url. The Date field is crucial to the blog because we will need to order by this field to keep the comments and blog posts in the correct order. NumLiked field will keep track of number of likes. Every time a logged in member likes/unlikes a blog posts, it will automatically update this field. Aggregation function could have been used, but since it will require the database to calculate the total number of likes in LikePost table, it will slow down the page too much. Instead, we will keep a separate variable.

**LikePost:**

1.       Post\_ID - Int (Foreign Key from Posts Table)

2.       User\_ID - Int (Foreign Key from Users Table)

Entries in this table create a relationship between posts and users who likes it. A post is the blog post and every time a user likes it, it will add their User\_ID to this table. The post will look through this table to determine if there will be a “like” button or “unlike” button, depending on whether or not this person has already liked this post.

**IsComment:**

1.       Post\_ID - Int (Foreign Key from Posts Table)

2.       Comment\_ID - Int (Foreign Key from Comments Table)

Entries in this table create a relationship between posts and its comments. A post is the first blog post and a comment is any other texts appended to that initial post

**Comments:**

1.       Comment\_ID - Int (Partial Key)

2.       User\_ID – int (Foreign Key from Users)

3.       Post – Varchar

4.       Date – DateTime

Entries in this table represent a comment. A comment is just any text that other user may post onto an initial post or respond to another comment. The Date field is crucial to the blog because we will need to order by this field to keep the comments and blog posts in the correct order.

**Tagged:**

1.       Post\_ID - Int (Foreign Key from Posts Table)

2.       Tag\_ID - Int (Foreign Key from Tags Table)

Entries in this table create a relationship between posts and its tags. A post is the blog post and a tag is a tag associated with that blog post. There can be multiple tags to one blog post. Multiple blog posts can have same tag as well.

**Tags:**

1.       Tag\_ID - Int (Partial Key)

2.       Tag\_name – Varchar

Entries in this table represent a tag. A tag is just any text that the author may have created or used when they wrote each blog post.

**Causes:**

1.       Cause\_ID - Int (Primary Key and Autoincremented)

2.    Name - Varchar

3.       Description - Varchar

4.    GoalAmount - Double

5.    CurrentAmount - Double

6.    StartDate - Date

7.    EndDate - Date

8.    CoverUrl - Varchar

Entries in this table will represent different causes of the donation that are asking for donations. Each Cause will have a unique ID number to represent it. It will also have Name and Description field to describe itself. GoalAmount and CurrentAmount fields have double-type value to indicate the goal and current amount of donations this cause has collected. StartDate and EndDate are also important to indicate the time range of this cause. Cover\_URL field will have url/filename of the cover photo it will use.

**Donations:**

1.    User\_ID - Int (Foreign Key from Users)

2.       Cause\_ID - Int (Foreign Key from Causes Table)

3.       Amount - Double

4.       Date - Date

Entries in this table will represent a donation. A donation will consist of an amount donated from a specific donor. Donor will be indicated by using User\_ID (from users) or -1 if not logged-in). The donation to a specific cause will be indicated using the cause\_id. Amount field will be used to indicate how much money was donated. Date field will be used for the date donation was made.

**Files:**

1. File\_ID - Int (Primary Key and Autoincremented)

2. Name - Varchar(1000)

3. Uploaded - Date

4. Caption - Varchar

5. User\_ID - int (foreign key from Users)

The File\_ID is the unique id that will serve as the primary key to this table. The Name field is the name of the file, the Uploaded field is the date the file was uploaded. User\_ID is the foreign key from the Users table, so that only the user who uploaded this file AND the admins can edit/delete this file.

**Projects:**

1. Project\_ID - Int (Primary Key and Autoincremented)

2. Name: Varchar

3. Caption - Varchar

4. Cover\_Url - Varchar

5. PCF – int

6. Url - Varchar

The Project\_ID is the unique id that will serve as the primary key to this table. The Name field is the name of the project. The Caption field is the caption associated with the current project, the Cover\_Url field is the url of the cover image of this project. PCF field describes if this project is past (1), current (2), or future (3) project.

**PHP INTERACTIVITY**

-Database Functions: multiquery, single query, retrieve results array

-Ajax Photo Search: needs php function to auto generate tables given an array of photos

-Checking if user has certain permission before generating the html

-Sanitize Inputs into Search

-FormHandling for all functionalities like sort, select specific albums, etc

-Add/Edit Photos/Albums

**PHP interactivity:**

SilkRoad (Home):

Although most of our interactivity for home page will be in jQuery, we are hoping to implement the slideshow/gallery of pages using AJAX. When the gallery arrow is clicked, data is sent to PHP. It uses the data to call on the relevant image and corresponding text for the sent data, and then returning it to the AJAX call. The jQuery will then output the data onto the home page. The jQuery will give a nice presentation, but the actual information to be rendered will be on a separate PHP page generating the contents of the gallery (including embedded links).

SilkRoad Projects:

Similar to the home page, projects page will have an interactive timeline on the top of the page that will produce images/relevant data using an AJAX call. A separate PHP page will handle the request and return the relevant information which will be rendered/transitioned into the content area of the timeline using jQuery animations.

SilkRoad Mission:

This is another jQuery heavy interactive page using AJAX. Similar to the home page and history page, the PHP will be used to pull relevant data send by the jQuery.

Donation Game (Original Participate menu):

It will be an interactive survey game where members can use to determine specific causes they might want to donate to. It is a series of AJAX PHP generated questions that will eventually lead to a specific cause to donate to. Each Question will have linked choices, which will be used to make AJAX calls using GET and return more content until a specific cause is reached. Clicking on the cause link will redirect the member to the campaign page for that cause.

Blog:

**Initial Display:**

The blog will display entries sorted by timestamp (LIFO). Entries have to be queried from the Database using PHP’s MySQL functions. The entries will only be a summary of the complete blog post, thus it will only have the title (linked to the full blog entry), tags, author, and 2-3 lines of the blog. Clicking the title link will redirect the reader to the full blog. After clicking on a link, the data will be passed using GET. The information passed will be used to query via PHP MySQL and a table-formatted blog entry will be rendered using PHP’s print function.

**Blog Pagination**

Originally, the number of blogs per page will number 10, but the reader can change the number of blogs displayed per page using a dropdown box. The page will be divided according to the timestamp (LIFO), and there is one page for every 10 blog posts. The number of blogs and content for the page will be rendered depending on blogs per page and the page number selected. It will use AJAX to pull out specific blog posts corresponding to the reader’s actions. The PHP part will use MySQL queries and return the data based on the page and number of blogs per page. The jQuery for AJAX calls will then render the Blogs accordingly.

**Search functionality:**

Users will be able to search through the blogs by tags, and authors and the possible results will be produced by typeahead (javaScript). The query information will be passed using GET. It will then refresh the page and use PHP MySQL to query the relevant blog posts information and then iterate+print the output in the blog content page using structured table format.

**Blog Submission:**

Chapter Presidents (members with Permission attribute greater than 1) can submit blog posts. PHP will be heavily used on this page for form checking/validation. After the entries are validated, AJAX will be used to attempt to INSERT the data into the database using TRANSACTION methods we mentioned in class. The result will be message for successful submission or a detailed error.

**Wordle function with tags:**

On the side, there will be a section of links of tags of different font sizes. The font sizes represent the popularity of the tags (bigger=better). When the page initially loads, the tag link Wordle will be produced from PHP MySQL querying for the 10 most popular tags and their counts. Then the result will be outputted in a randomized fashion in the section (having their corresponding size, but may be in different order for the sake of design). Clicking on individual tag link will produce a list of Blogs (sorted by time stamp LIFO) in the blog content area. The action will be done by using PHP MySQL query. The output will be structured using tables.

**Top 10 Poplar Blog posts**

When the page initially loads, a list will be rendered from PHP MySQL querying for the 10 most popular posts based on the number of likes each receives (numbered from 1~10). Each blog will be presented by their title, which is linked. Clicking on a link will redirect the user to the full blog.

**Comments:**

Members can make comments after  successfully logging in. PHP will handle the form validation and similar to making a blog post, AJAX will be used to render the comments or error to database INSERT query.  The PHP MySQL will involve steps for transition so that if the process does fail, a corresponding error message will help the commenter identify the issue. Clicking on a commenter’s name will redirect the page to that member’s profile.

Contact:

Contact page will involve form entries that will send an email using the server domain. Similar to other pages, it will use AJAX to valid the forms, returning error messages if applicable. After validation, it will use the PHP mail function to send the email and a message will be displayed to show success.

Donation:

**Initial Display:**

The blog will display Donations  sorted by timestamp (LIFO). Entries have to be queried from the Database using PHP’s MySQL functions. The entries will only be a summary of the complete blog post, thus it will only have the cause name (linked to full explanation of the cause), goal amount, current amount, beginning/end dates, and thumbnail of the campaign’s cover-photo. The data will be produced from PHP MySQL queries to the Causes and Donations Tables. PHP graphics may also be used to render the percentile status bar for the progress of the fundraiser. Clicking on the cause name will lead the viewer to a full detailed cause/campaign. The process will involve information from the link passed via GET method.

**Cause Pagination**

Originally, the number of causes per page will number 10, but the reader can change the number of blogs displayed per page using a dropdown box. The page will be divided according to the timestamp (LIFO), and there is one page for every 10 blog causes. Users can click on different pages to navigate through the later/earlier causes. It will use AJAX to pull out specific blog posts corresponding to the reader’s actions. The PHP part will use MySQL queries and return the data based on the page and number of causes per page. The jQuery for AJAX calls will then render the causes accordingly.

**Cause Submission:**

Admins (members with Permission attribute with the value of 3) can submit Causes/campaigns. PHP will be heavily used on this page for form checking/validation. After the entries are validated, AJAX will be used to attempt to INSERT the data into the database using TRANSACTION methods we mentioned in class. The result will print a message for successful submission or a detailed error.

**Cause Donation:**

We will have a SilkRoad merchant account/email address for the donation. Given the merchant account, we will use paypal’s simple JavaScript interface to transfer donations. The JavaScript will be used along with PHP to keep track of individual donation. We will use JavaScript to listen for the IPN, which is the callback produced by PayPal once the actual PayPal transaction went through, and output corresponding success/failure messages. If the transaction was successful, AJAX and PHP will be used to access the MySQL database (transaction), which will be updated with the new donation information.

Gallery:

The page will be divided into many different subcategories. After clicking each macro category, there will be subsequent dropdown menu with subcategories. Each category will have subsequent dropdown menu link/s. Each time a change is made (another option on 1 of the dropdown list is selected), a different subsequent dropdown list will appear next to it and the gallery in the content page will change correspondingly. The galleries are paginated (similar to blog post and donation). The Gallery will only display their cover photo, name and date, until the viewer clicks the gallery. After a specific gallery is clicked, the page will render the pictures in the gallery. Changing the dropdown list will refresh the content produced to the gallery list. This page will rely heavily on PHP AJAX to render the content (including the subsequent dropdown list rendering) after user makes changes to the dropdown sorting option or click on a specific gallery.

Search:

**Search functionality:**

Users will be able to search through everything. The search will rely on AJAX, PHP’s MySQL queries to return the information and then produce the output in the search content page using structured table format. Additionally, each result’s linked title, relevant thumbnail (if available), and a short description of each search result will be displayed in order to give the user more information. Clicking on the link will redirect the page to the relevant content.

Members:

After logging in, members can view profile and edit profiles.

**Edit Profile:**

A form will be provided for the portion of the profile that the members can edit, including password. Similar to other forms, the PHP will be used to validate the entries. PHP MySQL’s transaction will be used to INSERT the data from each field. AJAX call will be used to achieve this without refreshing the page. The result will produce a message of success or corresponding errors. This will include a dropdown menu bar listing different permission levels (members, Chapter presidents, or admin), and “request” submit button so that they can request permission changes to the admin.

**View Profile:**

Using PHP MySQL queries, the page will pull up the user’s current profile.

**Files:**

Members will be able to upload and download files through the server. Files will be validated using PHP, but almost any type of files could be uploaded. By clicking the name of the file, members will be able to download the files to their computer. By retrieving the user\_id for each file using MySQL queries, the member who uploaded that file (and the admin) will have a button available to delete that file.

Login:

**Login:**

The login portion will initially utilize a Lightbox via JavaScript. Then the username and password fields will be validated using AJAX via PHP page through MySQL queries. The username will be searched and if it’s found, a hashed password entry will be compared to the database. A success/detailed error message will be produced after validation/database quer. After logging in, the member will retain their login information until they press log out or close out of the browser. The information will be kept track using session variables.

**Reset Password:**

After choosing to reset the password, the member will be prompted for their username and email address. The form will be validated using AJAX PHP MySQL and a message of success/error will be printed. After a successful validation and the username/email exist and match each other, an email will be sent with a new randomly generated hashed password.

**Find Username:**

After entering a valid email (form validation through AJAX PHP MySQL), the server will send a PHP email (with the username) using mail function.

**Register:**

It will contain forms for member registration so that a user can access members page (Permission = 1) and comment on blog posts (Permission = 2). The entries will be validated using AJAX PHP. PHP will attempt to insert DATA using its MySQL transaction functions. If the input is invalid, then an error message will be produced. If the registration is successful, a “pending request” message will be sent. Before the admin confirms the membership, the member will have 0 for permission.

Admin:

**Manage Members:**

The admin can delete members and change memberships. The interface will be an unchecked checklist with member’s name beside each checkboxes. The roster checklist will be printed from query result using PHP MySQL functions. Admins will have an additional “admin” label next to their name/names. A delete button and a “make admin” button will be available for submitting the form. We will use the PHP MySQL DELETE query function for DELETE and UPDATE function to change a member’s permission to “admin”. SQL transactions will be used and thus error messages will reproduce in case of database error. The result will be passed onto the page using POST and the page will refresh with a new roster.

**Pending Requests:**

There will be a list of visitors who are trying to be members or a list of member who are trying to be Chapter Presidents or Admins. Each entry on the list will have their corresponding checkbox and also corresponding name and email address. The list will be printed from query result using PHP MySQL functions. A confirm submission key will change the value of the selected members from 0 to 1, 2, or 3 depending on what level they requested.  We will access the database using UPDATE via PHP MySQL. The submission will be queried using POST and corresponding success/error message will be produced after the page refreshes and completely loads.

**Change Content (content management system):**

There will be two main areas: a section for dropdown list and the content alteration area. After logging in, the admin will see a dropdown list representing every page. Similar to the gallery page, as the admin there will be more/less dropdown list that will be rendered representing more subcategories.  At each level, if there are contents that can be edited/deleted, it will be produced items from a checklist. If the items can be edited or deleted, then the corresponding button will follow the list of items. Selecting a specific item on the dropdown list will involve AJAX, which in turn will utilize PHP code. The PHP code will rely on MySQL queries, which will be used to pull relevant data to create the dropdown list and items in the checklist, and the jQuery AJAX response call will render relevant information along with edit and delete button correlating to the objects (whether it’s album name, album description).

**Edit:**

        After clicking the edit submission button, a list of items that were checked before the submission button will be rendered and additionally, relevant forms and the form submission button will be rendered using AJAX PHP. PHP will be used to validate form in the backend for each item. The PHP MySQL transactions will be used to UPDATE the information. In the situation where the validation fails or the transaction/queries contained error, relevant information will be produced using jQuery’s callback function. The messages representing success and error will be rendered using a popup box. If there is an error, closing the popup warning box will maintain the page, and if the process was successful, it will refresh the page.

**Delete:**

        After clicking the delete submission button, PHP MySQL transactions will be used to DELETE the information. In the situation where the database transaction/queries contained error, relevant information will be produced using jQuery’s callback function. The messages representing success and error will be rendered using a popup box. If there is an error, closing the popup warning box will maintain the page, and if the process was successful, it will refresh the page.