IMY 220 Project 2020

Photo sharing website

The project must be completed by the end of Thursday **5 November**.

We recommend that you test it on the server throughout the semester.

Contents

Objective

Create a photo sharing website that provides an interface for sharing and saving photos. You must use the technologies and techniques as specified in the project spec section to write your own code.

Your website must have a unique visual design that fits with the theme of the website (i.e. a photo sharing website). You are not allowed to simply use another website's design or create a "knockoff" of another website when designing the visual theme of your site. Note that you also don't have to use the terminology used in this document to describe functionality, such as "post", "friend", "album", etc. You may name these however you want based on the theme of your website.

The purpose of the project is for you to demonstrate the skills that you have learned in IMY 220 - not anything else. Therefore, you will not get marks for proving your proficiency with any other skills OR for using code that is not your own (e.g. from other libraries or templates).

Submission Instructions

- ***Late projects will not be accepted.***
- You have to upload your files to the **imy.up.ac.za server** as well as **ClickUP**.
- You can only FTP from campus. No excuses will be accepted for having difficulty uploading if you started uploading too late.
- *IMPORTANT: Please do not move your files around to different directories (in order to fix the directory structure) after you have finished writing your code.*

imy.up.ac.za server

Uploads to the server will be **blocked at exactly <u>10:30</u> on Friday 6 November**.

- Please make sure that you start uploading long before the deadline.
- Upload and test your project on the **imy.up.ac.za** server throughout the semester. This is very important as your local server might ignore bugs in your project that the server would not.
- You can only upload to the server from campus.

ClickUP

The submission link on ClickUP will close at **exactly <u>10:30</u> on Friday 6 November**.

- Do not wait until 10:30 to upload because your submission might take longer than usual to upload.
- Instructions:
    - Place all your project files, along with your exported MySQL database (in the form of an SQL text file) in a folder named with your Position_Surname.
    - Compress the FOLDER using a zip archive format.
    - Upload the compressed folder to ClickUP to the relevant project link.

IMPORTANT: Lecturers will not be responding to project related e-mails 24 hours before the deadline. No project related queries will be answered 24 hours before the deadline.

Project demo

You will have to demo your project from the **imy.up.ac.za** server in order to get marks for it.

- You will have to demo using either Google Chrome or Mozilla Firefox.
- When you demo your project, you will be required to show how you have implemented all the instructions by demonstrating the functionality on the website itself (not in the database).
- Demo sessions will be made available on ClickUP closer to the deadline.
- Demo sessions will take place on **Monday 9 November** and **Tuesday 10 November**.
- ***You will receive no marks if you do not demo your project.***

You will lose marks in the following conditions

- If your system does not have all the required functionality.
- If you use any other technology or technique that is not part of the course.
- If you use code that is not your own, i.e. if you use code generators or templates, you will not get marks for it.

*You will get **no marks** in the following conditions*

- If you plagiarise.
- If you do not upload to imy.up.ac.za
- If you do not upload to ClickUP.
- If you do not demo.

Main pages

All pages must clearly display the name of the website creatively and have a unique visual design.

Splash page

- When an unregistered user or a user that is **not logged in** visits the website, they must only be allowed to see a “splash page”.
- A splash page is the webpage that the user sees first before being given the option to continue to the main content of the site.
- Splash pages are used to inform new users about the services provided by the website and to promote it. The splash page is thus where you would give a tagline and/or an explanation of the purpose of your website for new visitors.
- The splash page is the only page that does not have the same consistent layout as the rest of the webpages.
- The splash page must contain at least the name of the website as well as a log in and registration form.

- *BONUS: catch new users’ attention and show off your design skills by creating a well-designed landing page that explains the goal of your website. Divide this page into well-designed sections and use the page scroll to trigger visual effects when the user scrolls down. How you make use of page scroll is up to you.*

Home page

- When a user is **logged in**, the home page must contain the **local activity feed** for that user in reverse chronological order.
- A user should then be able to switch between their local activity feed (that displays the activity for all that user's friends) and a global activity feed that contains the activity for all users. Also display this in reverse chronological order.

Profile page

- A user should be able to go to their profile page when they are logged in.
- The profile must have a profile image.
- *BONUS: allow users to add an image using drag-and-drop.*
- A profile must display appropriate personal information about a user, e.g. birthday, work, contact info, relationship, etc.
  Note: Since all this information will be publicly visible, no private information must be shown.
- Each user's list of friends, followers, and who they are following should appear on their profile as well but should only be visible if the user who is logged in is friends with the user whose profile they are viewing. (These details should also be visible if a user is viewing their own profile.)
- A user should be able to edit appropriate information in a logical manner.
- When you visit the profile page of another user, it must clearly indicate if you are connected to them. There must be two types of connections: followers and friends
  - Any user can follow any other user (without requiring confirmation from the other user). If one user follows another, the posts of the user being followed will appear on the follower's newsfeed.
  - Any user can send any other user a friend request, which must be confirmed before they become friends. Both users' posts will then appear on each other's newsfeeds and the users can send each other messages. They are also able to view each other's list of followers and friends on their profiles.
- There must be some logical way to connect with a user, i.e. to follow them or send them a friend request.
- When you visit the profile page of another user you are friends with there must also be an option to send messages to the user.
- All the user's posts and albums must appear on their profile page. You must come up with a sensible and visually appealing way to display this. Keep in mind that there might be many posts and many album with many images inside each, which means you must come up with an intuitive and visually appealing way to provide access to each.

Activity feeds and posts

The website should show two activity feeds: a local activity feed and a global one. Activity in this case refers to creating new posts or new albums and to adding new items to a album.

The local activity feed consists of all the activity for the current user as well as for all their friends, while the global activity feed consists of all the activity for all users on the website. The activity on the activity feed must contain the image and accompanying text and tags for each post.

Activity must be displayed in an aesthetically appealing way. Simply listing the pieces of activity in Bootstrap panels underneath each other is not sufficient, find inspiration from other websites/applications that use newsfeeds, activity feeds, etc.

Posts

A post in this context refers to an image with a text description, and one or more hashtags. The following applies with regards to creating and editing posts:

- Any user can create a post.
- A post must contain an image and a text description.
- *BONUS: allow users to add images using drag-and-drop.*
- *BONUS: allow users to add more than one image to a post and provide an easy and intuitive way to view multiple images in the newsfeed or post page.*
- The post's information must be displayed in the newsfeed activity as well as on a post's page, which is functionally similar to a profile page, but shows the post, description, hashtags, all comments, and which album a post belongs to (the last of which is not shown in the newsfeed).
- A post's description must also be able to contain hashtags. You can decide whether users should be able to add them to the text description or in a separate input.
  Note: A hashtag is an interactive piece of text that is created by prefixing a predefined symbol to it, e.g. #lunchtime. When a user clicks on a hashtag, it must be the same as performing a search for that hashtag, i.e. it must display all other posts that have the same hashtag as part of their description.
- The user who made a post should also be able to update post's description or hashtags, but not the image(s).
- Other users should be able to add comments to a post.
  - These should be displayed in a logical way, for example, if there are 50 comments on a post, only a few of them should be displayed on the newsfeed and the rest should be accessible by going to the post's page.
- Other users should be able to report a post (users cannot report their own posts). A user who reports a post should be able to select a reason for doing so from a list of predefined reasons (only admins are able to create new reasons for reporting posts). You must come up with a sensible interface for accomplishing this.
- The creator of a post should also be able to delete their posts. This should automatically delete all comments that were posted for the post being deleted.
- *BONUS: when a post being deleted is the only post inside an album, the album should automatically be deleted as well.*

Albums

The following applies with regards to creating and managing albums:

- Any user can create albums, which are collections of photos. There should be a sensible and intuitive way for users to create albums and add specific photos to albums.
- An album can have a name and description which can be changed by the user who created the album.
- An album can also have hashtags, which should function similar to those of posts.
- *BONUS: When adding hashtags to an album, use the list of hashtags contained by the photos inside an album and provide them as recommended hashtags to add. Hashtags that appear more frequently should appear first in the recommendation list and the list of recommended hashtags should not contain duplicates of hashtags. You must come up with an aesthetically pleasing and intuitive way to provide this functionality.*
- Users should be able to add any of their friends to an album, in which case those friends can also add images to the album but cannot change the name, description, or hashtags.
- Albums can also be made private in which case only the user's friends are able to view the album.
- *BONUS: create functionality for recommending albums to a user based on the hashtags of posts they have previously added to their albums. Hashtags that appear more often in their albums should be given preference when recommending albums. You must come up with a sensible and aesthetically pleasing way to integrate this into the rest of the website.*

Reported posts

- The functionality regarding reported posts are only accessible by admins.
- Apart from the normal functionality, admins can also view all reported posts, which should be displayed in a logical manner with the reason they were reported.
- An admin can then decide whether to delete the post or whether to remove the report on the post
  - When a post is deleted this way, the same should apply as when a user deletes their own post.
  - When a report is removed from a post that has been hidden due to being reported too many times (see below) the post becomes visible in the newsfeed again.
- If a post has been reported more than five times, hide the post in a newsfeed and display an appropriate message informing users that post has been reported and provide functionality that shows it if they want to view it anyway. You must come up with a sensible way to do this.
- *BONUS: come up with a way to detect users who are abusing the "report post" functionality. To obtain these marks, you will have to come up with a sensible system for assessing the situation and a user-friendly way for admins to deal with such users.*

User Roles

The unregistered user

An unregistered user can do the following:

- Register as a new user on the splash page. The splash page does not offer any functionality other than allowing the user to register or login.
  Note: An unregistered user cannot interact with the website in any other way.

The registered user

A registered user can do the following:

- Log in and log out.
- See all activity from themselves and all their friends (local activity feed) as well as from all users (global activity feed).
- Search for posts or users and view other users' profile pages.
- Edit all appropriate fields on their own profile.
- Add new posts.
- Manage (edit/delete) their own posts (this is discussed further under Posts).
- Delete their own profile. Deleting their profile removes their user account from the database.
  Note: This should automatically delete all posts that were added by that user as well as all comments associated with those posts.

User interaction

A registered user can do the following:

- Connect with other users by following them or sending a friend request to connect.
- View the complete user profiles of users that they are friends.
  Note: When a user is not friends with another user, they can only see that user's name, profile picture, and their albums.
- Unfriend other users.
- Privately send messages to other users.
  Note: When a user sends a private message to another user, that user must get a notification when they log into the system and the notification must go away when the user opens the message.

The admin (user account)

Some users should be able to log in as admin. The administrator has all the powers of a registered user.

Additionally, they can do the following:

- Manage (edit/delete) all activity.
- Manage (edit/delete) all posts.
- Manage (edit/delete) all user accounts.
- Add new reasons for reporting posts.

Search

Add search functionality that allows a user to search for the following:

- Other users by name, username, or email address.
- Posts by description
- Hashtags
- *BONUS: The search functionality must be able to find something even if the search term is a little incomplete or spelled incorrectly; i.e. fuzzy string searching: the technique of finding strings that match a pattern approximately (rather than exactly).*

  Note: this does not refer to simply adding wildcards to your searches. A fuzzy search should be able to find words that have letters missing, swopped around, etc.

Search results must be interactive, i.e. where appropriate they must return links, for example, to user profile pages, post pages, etc.

Usability

- Take care to design a website that is intuitive and easy to use. It is your own responsibility to come up with usable and intuitive interactions for the website's functionality.
- For example:
  - There must not appear a log out button if a user is not logged in, or a log in button must not be visible if a user is already logged in.
  - When a user creates an account, they must automatically be logged in with that account.
  - All form fields must have working labels, i.e. when you click on the label, focus is given to its accompanying input.
  - Navigation must stay consistent throughout pages.
  - Text that is interactive must look interactive (buttons, links, hashtags).
  - Clicking on the website logo/name must take the user to the home page.
- *Tip: Test the usability of your site by asking other people (especially non-IT students) to interact with it to find out how easy it is for them to navigate.*
- You will be penalized for sloppy design that is detrimental to the user experience, such as debugging alerts (or any unnecessary alerts).

Visual design

- Your website must be easy on the eye. Take care to design a website that measures up to professional standards.
- Use colour schemes that match, e.g. browse the colour lovers website (http://www.colourlovers.com/) or use Adobe Color (https://color.adobe.com/) or Coolors (https://coolors.co/) to find colour palettes and patterns or search online for colour scheme creators.
- You must redesign all form elements to fit with your website's visual theme, in other words, you are not allowed to have any default CSS/Bootstrap input boxes, buttons, etc.

- Use fonts that look good and fit the overall design of the site. You must embed fonts for your website.
  Note: Fonts must be legible. Do not use more than three fonts for the entire website. Do not use stylized fonts for body text. Do not use standard Windows fonts. Embed fonts with CSS.

Technologies and Techniques

You may only use the following technologies and techniques:

Technologies:

- Valid HTML5 everywhere (no exceptions). Make sure that when the HTML is returned from the server and viewed on the client (view source) that the HTML is correct, i.e., ONE doctype, ONE head, ONE body.
- CSS3 for styling, embedding of fonts, etc. Do not use inline or embedded style sheets.
- Bootstrap CSS framework for layouts etc.
  You **have to** use additional CSS to style elements according to your website's theme. You will lose marks if there are any default Bootstrap styling is visible on your website.
- jQuery for layout design, element manipulation, and AJAX usage.
- JSON to transfer your data back and forth between client and server (for example, during AJAX calls).
- AJAX for server calls. Use AJAX only when it is appropriate and would positively affect user experience, such as performing an action that requires database interaction in a case where you do not want to refresh the page.
- At least one instance of a Promise for asynchronous behaviour.
- Use SQL and MySQL database.

Techniques:

- Responsive web design with at least two breakpoints.
  Note: Your website must display correctly on any resolution.
- Show instances of functional JavaScript.
- Error free PHP. Sensible coding, you must only connect to your database in one place. Keep your code DRY (**d**on't **r**epeat **y**ourself).
- Input validation. Extend the default HTML5 input validation where appropriate (for example, to only let users join if they are above a certain age).
- Session management.

Favicon

- Design and embed a favicon that fits the theme of your website in all your pages.

Database

- There must be **LOGICAL** test data in the database for demo purposes.
- There must be at least the following:
  - 10 posts in the database.
  - 5 albums.
  - 10 registered users.
- Make sure that image sizes are small enough to stay within the total submission size limit of 50MB.
- Export your complete database to include in your ClickUP submission.

Directory structure

- Structure all your files into a logical directory structure.
  Tip: Start off by figuring out your directory structure instead of moving your files around after you've coded your website, since your references will change if you move files around.
- Name your splash page index.php/index.html.
- Make use of correct file naming rules.
- Put images, fonts, scripts, css files, etc. in their own folders.
- Use relative file paths for references.
- Make sure the slashes in your references are not backslashes.

Completing the project

You have until 5 November to complete the project.

You have to submit on imy.up.ac.za and ClickUP.

You will have until 10:30 on Friday 6 November (the end of the practical session) to submit the project because it can only be submitted from campus (on imy.up.ac.za) and you have to test and make sure that both your submissions are correct.

To avoid running into problems on 6 November, it is recommended that you test your project on the server throughout the semester. Do **not** use this time to work on the project.

Make sure to retest the entire site after submitting to server to ensure that what you will be demonstrating for marks is fully functional.

FTP – things to look out for

- You will have to put your project on a production server for demo purposes.
- The server will work similar to your local server, but there might be a few differences that you need to be aware of.
- The server has PHP v7 installed on it.
- The server displays all PHP errors and warnings, so it is essential to display all errors and warnings in your PHP code or local server settings while you develop.
- The server does not have file writing permissions switched on by default, so you have to change the file permissions in Filezilla for when you want the user to upload images etc.
- The server is less lenient when it comes to bugs in your code. E.g. on your local server images might still display if the file extension case in your code differ from your image file extensions, but on the server it would not display! OR back slashes in file references might work on a local server, but it has to be forward slashes in for it to work on a server.
- THUS: Your project would run smoothly on the server if your code is bug free
- AND it will not take you long to FTP if you test your project on the server throughout the semester.
- The only reason it might take long is if you upload and you realize that there are bugs in your code and you have to start debugging.
- A local server (that you used on your laptop) is not as strict as a production server when it comes to bugs. The only way to ensure that your code will work on the production server is to test it throughout the semester so that you can debug as you go.

Mark breakdown

Project Progress (Deliverable 1) 15%

Project Progress (Deliverable 2) 15%

Final Project (Deliverable 2) 70%