Documentation + Update Guide for cornellbadminton.com

Prepared by Daniel Zhan

Introduction

I built most of this website over the span of two days using a template (a lot of HTML, CSS, and JS files were pre-written to define stylistic layouts) from HTML5 UP. I additionally wrote a few JS files to help update and display players in the team, alumni, and board pages. Admittedly the website's current functionality could probably have been achieved using an online website preparation service (like Wix) that requires no programming background to put together, but it would be a shame for a computer scientist to have to resort to such tools. Additionally, the website's codebase probably does not conform to website standards (I just put files and folders wherever I felt like as long as the end result looked alright) or Git standards (everything is on the `main` branch); I felt that the simplicity of the website allows us to cut these corners without too much hassle.

The rest of this document briefly summarizes what to do with Git, organizes the pages of the website into sections, and then details potential future work for those who are more ambitious with regards to UI/UX design than I.

Git

Git is a version control tool to help programmers work together simultaneously and remotely on coding projects. It's ubiquitously present in software nowadays and any competent programmer should be familiar with some type of version control tool, Git being the most popular. Download Git here. Going to this repository and copying the link then allows you to run `git clone <link>` in your command terminal (on Windows), which "clones" (downloads) the folder to whichever directory you had open when you ran that command. Entering `cd <folder name>` then moves you into the repository folder (`cd` stands for "change directory").

For our purposes, Git allows us to "pull" (download an updated version of) code from the online repository (a version of the code stored online, and in our case also the code that is displayed on the website). You must be in the repository folder for Git to know where you're pulling from (the folder has a hidden file with information on where to download code from). Run `git pull` to pull code (make sure to do this every time before you work on the website).

As an analog to pulling, pushing sends your updated code to the online repository. After you're done working and saved your changes on your machine, push the code with the following:

`git add -A` (adds all files for updating)

`git commit -m “<commit messages>”` (stage the changes with a commit message, detailing your updates

`git push` (pushes the changes to the online repository).

Index

This is the “main” page of the website, and the page that displays if someone were to search up our club and click on the link. This page isn't very complicated in terms of code or interdependencies - most of the text on the page is found in the file `index.html` and can be modified by just changing the text in the file and saving (and then pushing). The free play schedule should be updated whenever it changes (at least once a semester), and please do change the contact email to whoever on board wants to be the point of contact. The announcement section should be changed whenever you feel there is a major announcement that merits being displayed here, along with a date.

Team

The Team section and subsections are the most complicated (but still not very complicated) sections of the website. Clicking on either “Team” or “Current Team” directs you to the current team page, whereas clicking on “Alumni” or “Board” directs you to the alumni and board pages respectively. You may notice that the players' information is not directly stored on the html files corresponding to the website pages - this is because JS scripts import this content from the JSON files that hold onto this information (the JSON files are named so that figuring which JSON database file corresponds with which page is obvious). Updating these JSON files and pushing then changes the players who are viewable on the corresponding pages.

Note that in order for an image for each player to appear, you have to put a PNG file of the player named in the format "<first name>_<last name>.png" into the images folder. The JS scripts automatically resize the image for you, but you may need to make the image square or else risk distortion on the page. Just copy the format of the current JSON files when entering the information for new players and you shouldn't have any issues.

Awards

This page is pretty straightforward - the captions and images take on the form of a feed. Just copy the format of the previous awards when displaying new awards and there shouldn't be any problems.

Media

This page is straightforward as well. It's filled with Instagram embed links that display the post on this page. You can get the embed link of a post by clicking the three dots next to a post on the instagram page.

Future Work

As mentioned before, the page isn't very stylistically complicated at the moment, but having direct control of the code that determines how the page displays affords you a lot of flexibility over what you want the page to show. Here are a few suggestions for future work:

- Fix the board JSON files to not require extraneous information such as "fun fact". This will require working with the board view JS script as well.
- Change the media page from one that depends on Instagram embed links to a wall of images stylized however you want. (The way VSCO displays images is pretty cool.)
- The header and footer of the entire website is separate code in each page, which is a horrendous example of repeated code. (Every time you would want to change the header picture, or add/remove a tab on the navigation bar, or even just change the copyright year on the bottom, you need to do this for all of the html files.) Figuring out a way to store the header and footer in a singular separate file fixes this problem.