Team: git push --force

Overview:

For our web application, we are creating a website for the UMass Outing Club. One of the functions of the website to manage a database of club members and upcoming trips. It will also provide a way to track which members have signed liability waivers and completed the necessary paperwork. This will save club administration a lot of effort as they had been doing this manually before. The website will provide a convenient platform for club members to sign up for trips, and for administrators to manage the club. There have been no changes to the general idea of the website since Project 1.

Team Members:

- Susan Pan
- Caroline Yu
- Kyle Vedder
- Jeffrey Shao
- Nonu Bajaj
- Stefan Kussmaul

Github Repository:

https://github.com/desporous/CS326Project

Design Overview:

- **Models:**
  - User Model: A club member and user of the website, who can sign up for trips and leave comments
  - Trip Model: A trip hosted by the club, which users can sign up for
  - Comment Model: A comment left by a user on a particular trip. Used to build comment sections
  - Notification Model: A message the user receives based on a certain action or update they should be aware of (e.g. someone replies to a comment of theirs)
- **Views:**
  - Index: Basic landing page, some website info and an image slideshow
  - Dashboard: List of posted trips that are upcoming. User can click to get more information.
  - Profiles: profiles of users including profile data, profile pic, and link to their waiver
  - Trip Info: In-depth information about a hosted trip. Includes a comment section.
  - Admin Management: Information about users that are signed up.

Problems:

One of our main challenges was debugging template issues, because there's no way to print from the template. There are a lot of small oddities in templates, so it can be frustrating when they don't render the way you'd expect them too. We were also fairly ambitious with our User and Trip models, and implemented our own custom permissions in the User model (e.g. can_comment, can_join_trip). Getting the UI to look exactly how we wanted (and using correct Bootstrap style) took a long time.

Supporting image uploading for user profile images was also tricky because there was not much documentation on the subject.

Successes:

We have a solid foundation for implementing sessions and multiple users. This includes comments, notifications, and sophisticated models that support a lot of functionality including permissions and levels of administration. Our template, url routing, and view code is solid, and uses good practices. We also improved team communication. Overall, our website has a consistent and visually appealing design, and will be able to support many cool functionalities once we implement user authentication.

Individual Write-Ups:

Stefan:
I tried to help the team understand a lot of the concepts (such as templates, views, models, etc.) I designed the models and a good amount of the site architecture, as well as non-interactive implementations of comments and notifications. This has been a great technical challenge, and I've had a lot of fun deciding on architecture for the more advanced functionality. The site will become much more exciting once we make it interactive in part 3.

Nonu:
I worked on the template, navbar, profile page, waiver page, index page, styles.css, and underlying code for most (if not all) of the templates.

For the template: I updated the template to include CDNs for Flat UI, included the navbar, and made it Bootstrap-ready.

For the navbar, I integrated it into the template, added a check to see if the user is logged in (show user stuff vs sign in stuff), and highlighted the current page.

For the profile page, I unified the style, changed some of the fields to better reflect our model, and added a check to see if the user can join trips (reflected in the "Pay Membership Fee" to-do).

For the waiver page, I unified the style, updated the contents to reflect the actual UMOC liability form, and polled the user data to automatically fill in all fields.

For the index page, I unified the style and added in the slideshow from our Project 1's index.

For styles.css, I made sensible classes that can (and should) be used site-wide. I like having one style sheet (rather than one for each page) because this helps unify our website design.

Other: I spent a great deal of time unifying the design for the pages I worked on. I tried to copy the appearance of the trip page, but it wasn't written in a Bootstrap compatible way, so I made my own version (that can be easily implemented site-wide) and proceeded to use that instead. I also cleaned up a bunch of code and/or rewrote code in a more predictable and user-friendly way.

Jeffrey:
I worked on part of the dashboard page, though Caroline did the initial set up, the initial waiver page and the admin trip planner page. I also fixed some of the styling on some pages to match the trip_info page so that the website pages would match each other.

Kyle:
I worked on adding the initial base template and nav template. I also developed the profile template and fixed the nav template to accommodate the user name.

Caroline:
My main contributions to the team this week were creating a writing the urls in the umoc folder (in total only like 6-7 lines of code) and 4 of the views.py functions (TripInfoView, UserInfoView, TripListView, and index), index.html with a sample of counts of trips (just to get project started), listing all the trips on the dashboard (django side only: url mapping, view functions which was a fraction of the dashboard template task), displaying trip info (django side: url mapping, view functions and displays with html/css I wrote from project 1: Mock UI), and creating profiles for specific users (django side only: url mapping, view functions which was a fraction of the profile template task). I also did some setups for the html templates only displaying count of trips in each of them, just to get the project started. It was

completely changed afterwards according to the initial mock ui and functionality each member made for their pages from project 1. I just wanted to show the members how to grab model attributes in the templates and how these variables are passed from the views.py. In total I've contributed or made the following files below:

- *Urls.py*
- *views.py-specifically on functions: UserInfoView function, TripInfoView function, TripListView function, index function*
- *Dashboard.html*
- *Trip_info.html*
- *Trip_info.css*
- *Index.html*
- *Comment.css*

Susan:

I set-up the initial website skeleton for Django and worked on the admin management page. I also tried to give suggestions here and there. As I worked on the admin management page, I noticed that our original goal of being able to easily change user level between user, leader, and admin can be easily done directly in the /admin page instead of making a whole new button for every person in the admin management page. Thus, the admin management page will be more of a user listing page for the admin to see at any time. However, if it is just a list of all users, I also feel like this page has lost its meaning because this can also be viewed in the /admin page. I changed the set up view of how trips and users are listed in the /admin page.