tigerride™ Final Report

Milestones

As is evident by the timeline on our design document, we were rather ambitious at the start of the project. Initially, we had planned to set up the database and perform basic ride filtering by the end of March. We wanted to begin hosting on Heroku by early April. We also planned on running Alpha tests by April 26 and Beta tests by May 3 in order to receive feedback from users to clean up bugs before our demo/submission.

However, things did not go quite as smoothly as we had hoped. We initially wished to work with SQLite as our database, but upon implementation we noticed that it deleted all of our user and ride data within a couple hours, or sometimes even minutes. Then we began exploring some less ephemeral options, and settled on Postgres… All in all, we did not have a properly working database set up until the beginning of May, which set us back about a month, and was the largest hurdle we faced in terms of making concrete progress.

Once the database was up and running properly, progress was made at a very quick rate. Some of our largest milestones included implementing the functionality of creating a basic ride and rendering the applicable results, creating options for Tigerbook-scraped and user-created profile pages, and developing an aesthetically appealing user interface. From there, we implemented features by order of importance, fully completing all but the user rating system and user profile pages (to see other students' information besides your own) by the May 8th demo day, and completing these final two tasks and cleaning up code by the May 12th submission deadline.

Design/Interface

Concept

For our overall design concept, we wanted to create something unique and aesthetically pleasing, and also very in line with Princeton. We decided not to utilize any front-end frameworks or libraries for two reasons. First, our website did not have any core interactive components that would have benefited greatly from utilizing a front-end library like Angular or React. In addition, we noticed that many past COS333

projects had rather similar appearances, most likely due to utilizing pre-existing web frameworks such as the aforementioned ones. However, we wanted to create something with a more unique style, and thus proceeded to layout with regular html and style with css and js. We utilized the basic Django template language which allowed each page to inherit headers and footers and fundamental css styling with the “extends” keyword. Furthermore, the language allowed us to directly access python variables, which was immensely helpful in displaying user and ride information. The current drawback of note is that some of the interface is not supported on Safari.

Knowing the importance of visual appeal, we hope our UI distinguishes our app from others and invites students to be more excited to utilize tigerride’s functionalities!

Landing Page

Our goal for the landing page was to create an interface that could give the first-time visitor a clear idea of what the app is used for. Since the app is intended to be used exclusively by Princeton students and faculty, we included an illustration symbolic of the university. Furthermore, to delineate the goal of the app in an engaging way (instead of simply stating something like “Find people to share rides with from campus to airports on any date & time that you desire,” which is both lengthy and uninteresting) we implemented a javascript function to spell out numerous location, date, and time combinations that people could potentially search up. This was implemented by utilizing the Typed.js Animating Library by Matt Boldt.

Buttons

We have three different button types, each with a unique purpose.

1. The buttons on the menu bar are simply comprised of text, with no bounding box or background color in order to maximize the white space (or in this case, pink space) in all pages of the website. This way, they have minimal presence that do not take away from the main purpose of each page.
2. The large rectangular buttons with diamond transitions that are found of the landing and home pages are given substantial size as they are generally the main features of their respective pages.
3. The small pink buttons darken and glow on hover in order to indicate that the user is about to make some sort of substantial decision, whether it be deleting a ride, joining a ride, or submitting information. Pressing these buttons always creates a change/update in the database.

Testing

Given our problems with our databases and migrations, we were unable to effectively use our sqlite3 database until the very end of the project for testing, which required us to push to Heroku every single time we wanted to test out a new change. Although this only takes about a minute or two each time, in the long run it ended up consuming a lot of time since we had built around 1000 times by the end. Though we realized how impractical this is, unfortunately none of us had the technical experience (nor did anyone on Google have the specific problems we had!) to be able to resolve our localhost issues until the very end of the project. Furthermore, another problem with pushing to Heroku to test was that we constantly had to make sure that other developers were not pushing simultaneously.

During the process of developing our app, we asked our undergraduate Princeton friends to act as the unbiased testers of the various functionalities. Through this, we were able to shape our interface to be more user intuitive, and to discern additional features which would be useful.

In terms of corner cases, most of this testing was done in pairs within our group. This was done in order to create rides from multiple users so that they could be matched up in various ways. The Django admin site through which we could view and edit database entries was enormously helpful to our testing, as was the ability to view heroku logs. Toward the end of the project, we also discovered the Heroku PostgreSQL shell, which allowed us to directly interface with the database using SQL commands.

Surprises/Unexpected Encounters

CAS Authentication

Despite submitting multiple requests to OIT to whitelist our website for CAS authentication, we did not receive approval until May, when one of our team members physically paid them a visit to follow up on our requests. There was also a point of confusion during which we thought that they had given us approval at the wrong url, leading us to interpret bugs incorrectly as resulting from a lack of permission. This delayed us another day, but thankfully Lance's tutorial made it very easy to implement CAS authentication once we received the final confirmation of whitelisting, and we

were able to have functional CAS authentication by the day of the demo. Jace's post on Piazza was extremely helpful in outlining the entire process.

Competition

It was disheartening to find out that there was another COS333 group working on a very similar project as us. We really wanted our app to be utilized by a large portion student body and have a lasting effect on the Princeton campus, and thought having two apps that do essentially the same task may divide up the user pool. If users are split between two apps, they are more likely to find fewer matches (potential matches could be using the other app). Furthermore, the other group finished their project before ours, and began promoting their app through social media and listservs early on. This made us a bit nervous, but actually turned out to be useful for us because we were able to figure how to make our app more appealing to students by implementing more useful features, and creating a more attractive user interface.

Falling Into a Rhythm

After the initial chaos and worry that came with a breaking database and migrations which would not apply, we were pleasantly surprised that as we approached a time crunch, we quickly fell into a rhythm and made significant progress. Moreover, we were able to enjoy ourselves while doing so. Approaching this app with no experience on most of our parts meant that we were fumbling and worried about a grade at the start; however, after finding our footing, our mentalities changed so that instead, we were actively looking forward to working on this app and any future ones we might build as well.

If there were more time…

If there were more time, we would increase the variety of origin/destination options we offer when creating a ride. We would do this by surveying the undergraduate population to find out which locations are popular, and perhaps add an "other" option for the less common places people travel from. We would also allow users to rank their preferred pick up and drop off locations on campus from a list of pre-approved places where Uber/Lyft can access. This would make it easier for students to coordinate a location on campus which is convenient for all parties involved.

We would also hope to utilize other APIs, for example those that would provide flight information regarding possible delays/cancellations, in order to notify other students in

the same ride group just in case a user's flight is inadvertently delayed and she/he has no way of getting in touch with her/his ride-share group.

Moreover, we would give the current students in a ride the option to accept or reject a different student who wishes to join their ride. This would have to be done carefully, as to not allow for unjust discrimination. However, we believe this would be a useful feature when a student has been heavily unreliable in the past (reflected in their user ratings), or in the case where timings barely align (ie. one student wishes to leave between 1-3pm, and another student is willing to leave between 3-5pm). In the latter scenario, a flight delay or other issue could mean a student waits for others for far longer than they wish. By allowing the accept/reject option, we hope to simply encourage accountability between students and practical groupings of students based on their flight times and willingness to wait.

We would create a feedback submission form accessible on our website. Currently, all of our emails are on the website as links. However, particularly if a user wants to submit an anonymous comment, a form would be easy and helpful.

We would also hope to utilize a phone-number verification system for our application. Currently the database automatically stores and saves any phone number or value inputted when a user is first asked to create a profile; however, it would be useful to have an authentication system first confirm that the phone number inputted is valid, then store it. This way, we can guarantee that riders are able communicate with each other via phone number, rather than through email, if they prefer.

Currently, when a user goes to edit their profile information, it is mandatory to update every field. However, we would like to change this so that they can select only the fields they wish to.

Lastly, we would utilize a cloud storage solution such as Amazon Web Services to store the user-inputted profile photos. Currently, because Heroku resets its dynos constantly and the photos are stored at a media/images url on our website, the profile images quickly disappear. For real use by the student population, this would have to be fixed.

Advice for next year's class…

Our advice for next year's class is to start early. No matter which goals you set, going into a project of this scale with little to no experience means there are obstacles you

can’t expect to know about yet without actually moving from planning to actually coding. Moreover, begin with a more reliable and scalable database such as Postgres from the start; migrating from SQLite to Postgres caused us many headaches and wasted so much valuable time. Learn to use Github correctly from the start, and write descriptive commit messages when it’s important to! Version control saved us on several occasions, and being able to find the specific commit and the code you changed at that point can be very helpful. However, it can cause significant headaches unless you properly work with branches and communicate well with your partners! Lastly, have fun with this project, and design your app with the intent for it to be used by the entire student population – aim big! Though we have a few functionalities to add and problems to fix still, we hope to implement all of the changes over the summer. We are currently speaking with students who are part of the Undergraduate Student Government and wish to promote COS333 apps, and hope tigerride will be used widely beginning next fall.