Sprint 1 Retrospective

Sublettr

11 OCTOBER 2017

Team 9: Michael Vieck, Joel Van Auken, Jeremy Lehman, Josh Newlin

What Went Well

We had everyone working on things mostly throughout the sprint with no one really having any downtime. Almost all services are propped up now. When almost all parts of the project (frontend, backend, etc.) were on github and ceased to be empty projects, all group members worked sufficiently to try to meet the goals before the sprint's end. We had adequate meetings and great communication within our slack channels. Group members were quick to respond to and fix problems dealing with their facet of the project. When other members had issues with their work, others stepped up and helped. We feel as though we are adequately prepared and able to take what we've learned from this sprint and apply that knowledge to future sprints.

What Did Not Go Well

We went into this project with the assumption that the backend was going to be up and running week 1 or 2, this was not the case as we found out later on during our sprint. There was some time not spent on certain facets of the project during the earlier half of the sprint, mostly not setting up services soon enough. Also, integration was done last minute because the backend was not up yet for the front end to hook up to it, which ended up breaking several times before the front end was connected fully. There was also a major disconnect between the front end team and the backend team on how our data models should have been organized, with fields existing on the front end but not the backend, and vice versa.

Starting the planning process of this sprint without having adequate time to test certain features' feasibility caused some headaches. This was primarily dealing with the scraping services not going as planned because of safeguards put in place by the websites we planned to scrape. We had two primary target websites for scraping, Facebook and BoilerApartments. The first site, Facebook, was unable to be scraped since you are not allowed to access the Facebook Group Pages Graph API if you are not an admin of the group. Additionally, if you attempt to scrape Facebook's website, they will eventually detect you are repeatedly requesting their web pages and will blacklist your IP as part of their DDOS prevention. In terms of BoilerApartments, their website uses dynamically built pages from JavaScript to display data. This was an unforeseen complexity,

which added much more time to the process of creating a scraping script. In addendum to our other difficulties during this sprint, it made it hard for this task to be completed since we were struggling at the same time to put together core functionality.

The front end libraries we used for some of the main core components of our application did not have all of the customizability we would have liked it to have. Google's material design library for Angular did not allow much freedom to change importable components, such as cards, drop down boxes, and side navigation. A lot of the css had to be done manually, which slowed all of us down and reduced our productivity. Instead of focusing on the feature itself, a lot of the frontend work went into small design fixes and minute details. Before any front end code was done though, we created a mockup of the front end. This was good at first, but the design left out many key details of our user stories like a profile page and user navigation.

We had intentions to run our server in a docker component to minimize environment variables across devices. However, when attempting to configure docker for each member, we realized that docker was going to add too much overhead for not enough gain. Some members spent a significant amount of time trying to initialize docker before deciding to scrap the idea completely. This resulted in lost time and effort for something that did not pan out and was ultimately deleted.

Areas to Improve

We should try not to underestimate the hours required to do certain stories. One of the largest mistakes made during this sprint was aiming too high and planning the sprint with too many stories. Sometimes you really do bite off more than you can chew and this sprint was a perfect example of that.

In the future, we really should sit down and try to see if a user story is really several stories masked as one and think of any future stories that may come up as we are working out the project. We could also try to jump on the earliest opportunity to set future services up. Once the service is propped up, the group members were more likely to work on them.

It would also be to our benefit to have better coordination between the backend and frontend team to prevent duplicate work and enable good team work. As touched on earlier, there was a disconnect between the two teams on our data models so a possible fix for this would be to create a document that clearly displays how the front end is modeling its data and what the backend expects for its API endpoints. This should clear up any confusion.