PorkBun Report

Khatanbuuvei Bold and Justin Chang
porkbun.herokuapp.com

Important Choices and Decisions

When making choices on how we should develop PorkBun, our main consideration was how to make a site that is easy and intuitive to use, yet also interactive and free enough to give users enough choices. We found this to be challenging at times, because giving users too much choice could actually make the site annoying and confusing to use.

1. Interface

One of the biggest choices we made early on was to make much of the site look and feel like what most Princeton undergraduates are familiar with: TigerMenus. By doing this, not only would users have a more comfortable time transitioning to our website, but we would also hopefully distinguish ourselves from past projects that either did not have TigerMenus (created in 2017) to model their interface off of or chose to create a whole new interface for the dining hall experience. It was pretty clear to us that students were used to the TigerMenus interface - a daily menu with columns representing the dining halls - so we saw no practical reason to confuse users with a new interface.

This proved to be a good decision not only on the user side, but also for development, as we were able to adapt a lot of open-source code already written. After confirming with Alex Xu (creator of TigerMenus) that it was okay to look at and adapt the code in his Github repository, we were able to adapt much of his scraping scripts, though not all of this process was straight copy-and-pasting because we had to convert the Python scripts to Javascript.

2. Technologies Used

Another decision we made early on was to stick to Javascript in essentially all that we did. This meant that we used the Express web framework, Node runtime environment, and a combination of EJS and jQuery for front end manipulation. We initially made this

decision to limit the number of programming languages and frameworks we would need to learn, especially with the time constraint involved, but eventually found it to be a good choice for other reasons as well. Specifically, it was extremely easy to find support on the Internet for problems that we had, because we found that all the tools mentioned above are very popular among web developers. Node also worked seamlessly with Heroku, so deploying was fortunately not an issue.

Beside the language that we wrote in, another key technological choice was what kind of database to use. Though we had experience using SQL through one of the assignments, we decided to use MongoDB, because we preferred its dynamic and flexible nature as opposed to the strict structure of SQL.

To host our database, we chose to use MongoDB's own service, MongoDB Atlas, because it had easy-to-read documentation and seemed like the natural choice to freely deploy our database. There were some initial concerns about this option, one of them being that there is a 512MB storage cap for the free tier. However, after a bit of testing, we figured that this was more than enough space for our purposes, as we could easily implement a way to delete outdated dishes and if the number of users ever got to the point where 500MB was not enough, Princeton's $23 billion endowment could surely support a paid tier. Another concern was uncertainty about the speed of accessing data stored in MongoDB's cloud platform. Because most of our app's interactions with the database must occur in real-time (autocomplete, preference updating), it was essential that accessing and updating data was quick. Fortunately, we were pleased to find that this was the case, as none of the features we tried to implement were limited by the speed of data accesses.

3. Intended Devices

Perhaps our biggest mistake and most unwise decision was to build PorkBun using our personal laptops as our test machines. While this made testing incredibly easy and efficient, this meant that we spent much less time testing our site on the devices that many of our potential users would use to access our system: smartphones. While it is true that Princeton students use laptops to visit TigerMenus, many students love the convenience of being able to check TigerMenus on their phones and quickly glance at the day's menus. As such, by primarily focusing on the desktop experience while playing around with the user experience of our website, we lost a ton of valuable time to test using smartphones, mainly relying on Bootstrap to dynamically resize our desktop experience for a handheld phone.

Milestones and Timeline

In general, we accomplished all but two of the features that we outlined before we started to work on PorkBun. The two features we chose to forego were: 1) email notifications and 2) an option to turn off preferences in the weekly menus page so that users could view a fresh set of weekly menus and search for items manually. If that second feature sounds confusing, that's exactly one of the reasons we chose to remove this feature from our priorities. As we were working on PorkBun, we realized more and more that not only were these features unnecessary to our core service, but if done rashly and without nailing our main features and interface down first, they could detract from the intuitive and user-friendly experience that we were going for. Thus, quite early on in our process, we chose to prioritize other features, leaving these two ideas at the back of our minds just in case we had extra time at the end of the semester (we didn't).

More challenging than actually implementing the individual features of the website, however, was following the timeline that we set before we started to work on this website. Particularly, we found that many of the beginning steps of our project, which we thought would be relatively easy compared to the later steps, actually took more time than the later steps. For example, adding elements to the database, a step that we thought would be relatively quick, took a lot of testing through trial-and-error, while features like autocomplete, which seemed difficult to implement, turned out to be much simpler than expected. This meant that though the timeline that we set up was very helpful in figuring out the next steps at a given stage, it ended up becoming a pretty fluid agenda that needed to account for extra time in a given week spent on thoroughly testing and researching how to implement certain functionalities.

Improvements and Future Considerations

As of now, PorkBun has all the features we deemed necessary for a user base to grow. If we were to have more time on this project, however, we have multiple areas that we would focus on, using the experience that we've gained so far to really elevate the role that PorkBun can have on users' lives.

1. Mobile Support

As mentioned before, one of the biggest shortcoming of PorkBun is that we didn't prioritize mobile users when testing our website. Thus, when it comes to the mobile experience, our website may have areas that feel annoying or cumbersome to use. In

the future, we could use more advanced features of Bootstrap to create an interface that is more dynamic and mobile-friendly than what we currently have. This would be very important in retaining users that typically don't like to spend too much time on their phones, but rather check it a few times a day for some important information.

2. Other Stretch Goals

In addition, as mentioned in the Milestones sections, there were two features we set aside in order to focus on the main purpose and functionalities of PorkBun: email notifications and an option to turn preferences off and search the menus. However, these ideas are not necessarily bad ones. In fact, they provide users an even greater ability to make use of the dining halls on campus and if implemented correctly, they can be integrated seamlessly into the site, just like the other functions we have implemented already. Additional time would allow us to think not only about how to implement these features technically, but also in a way that does not clutter any page and seems natural to the user experience.

Advice for the Future

If future COS 333 groups were to ask us for advice or insights from our experience with PorkBun, we would focus on three things:

1. Frequent Meetings

Be prepared and willing to meet frequently with your group. Even as a two member group, we found ourselves meeting quite regularly to discuss the changes we made and changes we were working on. This became incredibly important as the individual features and pages we were working on were merged and brought together, as it was increasingly important for us to know not only how to read and edit the code we each wrote, but also how to make changes and add to each other's code. Of course, this doesn't always need to take place in physical meetings (one might resort to reading and analyzing code separately or asking through emails/messages), but we found this to be the most effective and time-efficient way of getting caught up in what the other person was doing. Additionally, we realized that although a group project is a great way to get to make new friends, teaming up with someone you know has a lot of advantages, the most obvious one being ease of communication, especially in the earlier stages of the project.

2. Effective Division of Work and Standardized Practices

It's important to find a way to divide the group's work in a way that is effective for your individual group. Many groups with members who have experience in either frontend or backend will divide their work along those lines, but that's not the only way to split the work. Because neither member of our group had much expertise with either frontend or backend, we decided that rather than splitting the work along those lines (which we figured would require constant communication about what each of us needed from the other), we would start by working on two relatively separate areas of the website: the menus/dishes and the user's preferences. Both areas required programming for the full stack, as both involve interactions with the database and user interface, but by doing so, we were able to reduce the communication needed to implement basic features and resolve nearly everything at our regular meetings (mentioned above).

That being said, no matter how you divide the work among your group, we recommend that before anyone starts working, everyone is on the same page on coding standards and practices. From important things like names of files and the organization of your repository, to seemingly insignificant things like the number of spaces to use for indents or names of local variables, having a quick discussion on these expectations can not only make it more pleasant to look at your groupmates' code, but also make it easier and quicker to read and understand.

3. The Idea Itself

It's easy to think of this project as something that will only last a semester, but it is extremely important to pick an idea that you are interested in, or that you could at least see yourself using. We don't say this because we think it's important for COS 333 alumni to continue working on their projects for the rest of their lives, but because it makes this process so much easier if you are interested in your idea. Many of the ideas that we bounced around before deciding on PorkBun would have made this project much harder to complete because we frankly weren't that interested in them. It's human nature to enjoy working on something that we see as fun and enjoyable, so we highly recommend that you don't narrow down your list simply asking "what ideas can I picture myself investing some time into every week to complete," but rather, "what applications (once finished) will I be excited to spread around or at least use myself?"