Parkday – Technical Take Home

The objective of this exercise is to write code to improve upon an existing web tool.

The final product can be an email summarizing your thoughts and a Github link to your code, or you can consolidate it all into the Github project (just use the readme to explain your approach).

We ask that you turn back the finished product within a week, during which you can check in with Adam as much as you need. We encourage you to check in as often as you feel appropriate with progress reports, questions, feedback requests, etc.

OVERVIEW

One of the key tasks at Parkday is good communication with our food vendors. It is vital we keep them up-to-date with our data and they let us know if they have any issues or potential delays. One of the ways we do this is by emailing vendors a summary of the number of meals we currently have reserved for them. This is currently a static email sent out the afternoon before service. The issue here is that the numbers can change at any time so the email they receive may not have accurate information.

Your task is to build a dynamic web page that will open as a result of a click from an in-email button and display an up-to-date count of the current meals for a service in a presentable and easily readable format. The meals should also have a breakdown of their ingredients to confirm with the vendor. It will also contain a way for the vendor to alert us if the orders are on time or late.

With this task we are testing your code-writing abilities as well as seeing how you can work to a simple brief and your ability to work with us. We want you to have the same level of autonomy and collaboration so feel free to reach out as often as possible and tackle things in the way you think feels best.

EXERCISE

The web page will be opened by the vendor clicking a button on an html email. The web app will need to be able to read in a variable service_id from the link. The actual service_id used here is irrelevant but it should be able to be a variable given input and should be obscured from the user. You will likely want to read it in from the URL but any way you choose is fine.

You will be provided with a CSV dataset of ‘meal_opt_ins’.

All of the opt-ins have the creation date of 2023-03-21 but have different hour marks. You should not count opt ins that have a timestamp after the current time. So if you open the page at 2pm it should have a different count than if you open the page at 7pm, for example. Feel free to change the created_date and/or service_date to the current/future date if it makes it easier for you, it’s not important.

There is no specific way the page should look, while your code, communication and understanding are far more important than your design skills - try to treat this as a production ready page that we would share with outside vendors.

The way you display the meal counts and their ingredients is entirely up to you, as is the way for the vendor to contact us - can be through a data form upload, email, slack or any and all of the above!

The task shouldn’t take too long but do let us know if time is a factor over the next week or so. We want you to do your best so take your time and feel free to start the work only when you feel able to put some time towards it.

Optional but suggested steps:

- Start the week with an estimate of how much time you think this will take you, feel free to break it out into as much or as little detail on the subtasks as you see fit, and include this projected vs. actual time spent in your initial and final reports.

- Check-in at least once (can be progress, challenges, questions, etc.) – so Adam can review and follow up and give any direction if you need it.

- Think about the process and the flow of information. Ask questions around what information is being submitted to start, and what information you need to receive back upon submission.

- Consolidate these thoughts and communicate with Adam. If you need further clarity or have different ideas for a flow than suggested above, suggest those. Either way is 100% OK, it just requires clear communication so we can evaluate accordingly.

The best deliverables include the following characteristics:

Code

- Clear, concise, readable code
- Comments where necessary
- Debugged and functional out the box
- Don't reinvent the wheel - find & use libraries as needed

Communication:

- Be able to give time estimates and progress reports (daily quick emails on what was completed, any challenges)

- Be able to ask for help if you need it