Whew! This project was definitely a lot of trial and error for me as I navigated the unfamiliar territory of Shiny. I knew from the start that my project choice would come along with a lot of learning, and boy was I right!

Inspired by the campaign finance data that we used in class, I downloaded the data from the FEC website to make a Shiny application focused on the Senate race for Blunt’s open seat. Up until late Tuesday night (yes, the day before the deadline), I was using the actual data from class to build, but I started to feel insecure about how old it was, so I went searching for newer stuff on the FEC website. I used data from [this specific lin](https://www.fec.gov/data/elections/senate/MO/2022/)k, using the top table on the site, which took away a lot of the work I was previously doing to filter, summarize, and account for refunds. However, this choice also meant that I was redoing a bunch of stuff last minute.

The new data I chose was basically a table that I had been making via R coding, directly downloadable from the FEC, but with a few more fields like whether the candidate is going for an open seat or challenging, and how much cash the candidate has on hand at the end of the period. The data is for candidates running in 2022, and the most recent reporting seems to come from the end of March, per the “coverage\_end\_date” column.

As a whole, I found this project quite challenging, since I came in knowing nothing about Shiny. However, I anticipated it would be a struggle at times when I was deciding on this project, and I chose to do go with it anyway. In the end, I feel like I’ve learned a lot about Shiny while exercising my “learning-how-to-learn” muscles.

As you can probably see by going through my history of commits on GitHub, there were some aspects that I repeatedly got wrong before I figured it out. Reactivity was probably the hardest thing to wrap my head around, then various aspects of building the graphs.

In the end, I mostly achieved what I was picturing for my app, the only exception being that I struggled to make the web app that I was building live on the rsconnect servers. I followed instructions online, but I kept getting errors that said my app.R was not returning a Shiny object. I looked through the Stack Overflow advice for others with this error, but the good folks on there were only able to offer solutions after looking at the asker’s specific code, and I struggled to diagnose what was wrong with my code using that. My hypothesis is that something in how I manipulate the data at the top of the page might be incompatible with Shiny, and since my whole app is built on that, it just seemed a bit out of my depth to figure it out and connect to the server. Thus, my app is viewable by taking the files from GitHub and loading it through my rNotebook or app.R file, rather than me figuring out how to set up a URL.

Despite my server struggles and the many other issues I grinded through to get my final project, I’m pretty happy with where I landed and the Shiny knowledge I’ve accumulated along the way.