EECS 448 – Team 14

Daniel Johnson

10/21/2020

Software Architecture of our Prototype

The prototype of our project fits very well into the 3-tier software architecture. The web application that we made has a presentation tier which shows the map, route, directions etc. We then have a logic layer to get info from the presentation tier and organize it in sensible ways. Lastly, that information is fed into the data tier which in our case is the two API’s that we used, Google Maps API and Google Directions API. We chose this software architecture methodology because feeding direct input from the user to an API can result in errors and it made it easier to define specific roles for team members to fit info.

We allow users to click on KU buildings to add them to their route, but we also give them a search box that they can include anything in. So, allowing for n number of places in the route requires us to formalize their route with a structure before we hand it to the API. The Google Directions API requires an origin, a destination, and allows for a number of waypoints. So, route.js provides a structure that takes the user’s input and divides it into origin, destination, and waypoints. This is why the logic tier of the architecture is required.

The 3-tier architecture also allowed us to split the workload better. Like we have for previous projects, we split our team into two sub-teams. The best way we thought of doing this was to create a UI team and a Directions team. This fits in well with the 3 tiers of our architecture because we are not the ones writing the data tier because it is a public API. The UI team deals with user interactions which is the presentation tier, and the Directions team deals with taking information from the UI team and formatting/feeding it to the data tier. This is why the 3-tier software architecture helped us format our team and create even workloads.