My goal was to look at methods to experiment with ways to create efficient updates for emergencies on the campus map. Ideally, I had the idea of a person witnessing an emergency/accident and using an app to report it. Then an automated process would add the polygon/point representing the emergency and its details on the map. This is quite far beyond the scope of my knowledge and the scope of this class project, but I thought it would be helpful to share my initial thought on the matter.

Keeping a realistic perspective, I attempted to solve the question of how to make quick updates to alert people of emergencies. More importantly, I attempted to think of an easier way to do it than that of present methods. At the time, the tools GeoJSON, GitHub (as well as GitHub Gist and Raw Git), and JSFiddle were what I used to experiment with adding points and polygons to the map (with the attributes of the accident/emergency). Despite efforts to better comprehend, I feel that my knowledge and understanding of these tools is subpar. It would perhaps be more ideal for a person with a familiarity of more computer science related topics to partake in this endeavor. Yet, I can at least explain (in perhaps more layman’s terms) one way of doing this.

* Using GeoJSON.io, search and zoom to your area of interest, and using the tab on the right of the map, create the point/polygon representing the emergency (this can be done by editing each figures attribute table)
  + Save the map to “anonymous github” (click link that appears)
* After being taken to GitHub Gist, click the “Raw” button to get the raw code for the object(s) you created in GeoJSON.io.
* Copy the URL from the web browser that has the raw code and navigate to Rawgit.com
* Paste the URL in the primary text entry box. This will generate two new URLs below. We want the URL on the left under “Dev/testing”
* Copy this final URL and navigate to JSFiddle
* Within JSFiddle there are 4 quadrants: top left is HTML code, the top right is CSS code, the bottom left is JavaScript (what we are concerned with in this demo), and the bottom right is the output of your work
* **Note: An access token to Mapbox’s API as well as a slippy map from Leaflet.js was already given in the JavaScript quadrant**
* Paste this final URL in line 28 (in this demo specifically). This will load in the campus-map-overlay GeoJSON file from the data folder
* In the output window on the lower right, you will essentially have a simulation of what the campus map will look like with the updated features.

The first minor challenge to note is the standard of symbol use for specific emergencies. Since we will have people doing the updating, there is room for miscommunication. For example, if there is a car accident, one user might use a symbol titled “road block” while another user might use a symbol titled “car”. A set of standard criteria must be created to be followed. Additionally, the symbols must be created with a mindfulness toward User Centered Design. The symbols used must have an inherent nature as to be easily interpreted by the user.

The main challenge, however, of this process is essentially the crux of this project which is to find an easier way to do this. This process is a bit cumbersome and requires some knowledge of these code-based web sites. A method for doing this that doesn’t involve working with any code would be beneficial so that people required to do this task don’t have to learn all of these extraneous factors.