CS Capstone Design

Technical Demo Grading Sheet (100 pts)

TEAM: CartoCosmos

Overview: The main purpose of the "Technical Demos" is to very clearly communicate the extent to which the team has identified key challenges in the project, and has proven solutions to those challenges. Grading is based on how complete/accurate the list of challenges is, , and how convincingly and completely the given demos cover the given challenges.

This template is fleshed out by the team, approved by CS mentor, and brought to demo as a grading sheet.

Risky technical challenges

Based on our requirements acquisition work and current understanding of the problem and envisioned solution, the following are the key technical challenges that we will need to overcome in implementing our solution:

C1: Switch to non-Earth projections for other planetary bodies. All map-viewing applications only support Earth's settings, such as its radius and projections. We first need to figure out a way to support non-Earth projections in Leaflet. It will then be challenging to make this solution modular so that it can be used in many mapping applications. In order to prove a good solution, we would have to show three maps of Mars demonstrating that Mar's projections can be used and shown correctly in Leaflet. For this challenge, we will also show the scale bar changing depending on the projection.

C2: Create a GUI that is usable by a variety of users. Because the planetary science community is very large, the number of users and their knowledge varies drastically. We need to create a GUI that wraps the Leaflet map that any user will understand how to use. In order to prove a good solution, we will create storyboards that show the different layout options.

C3: Users need to be able to swap between different layers. We will need to be able to query GeoServer to get the information we need to create our layers. GeoServer only works with Earth codes, so we will need to tweak our query to receive the correct data back. For a demo, a good solution will be to show a layer switcher with all of Mars' layers USGS has. We will be able to switch between these layers in the demo.

C4: Create and display overlays. Again, we will need to query GeoServer for overlays such as the surface feature names and gridlines that show the bodies' quadrants. Making sure that these overlays are pulled, created, and displayed on the map correctly are challenging because GeoServer only accepts Earth's codes and we need to verify that Leaflet can handle the overlays. In order to prove a good solution, we will need to show the surface feature names and gridlines shown displaying correctly on a Mars map.

C5: Add Latitude and Longitude transformations. Leaflet has no native functionality to change latitude and longitude settings. When a user moves their mouse, they need to be able to view the correct coordinates. However, Leaflet has no native mouse position control, causing us to use a third-party plugin. This plugin may not work with our latitude/longitude settings switcher and show incorrect coordinates. In order to prove a good solution, we will need to show the lat/lon of the mouse position and include any transformation functions needed to swap between the different settings.

C6: Use Leaflet formatting standards. Because we want to submit our projection package as an official plugin to Leaflet, we need to follow along with their coding standards. This includes documentation and creating new classes. For a good solution, we will need to show that all of our code follows their format.

C7: Create a Leaflet map in a Jupyter Notebook. An environmental requirement for this project is to use Jupyter Notebook to interact with a Leaflet map. We need to research how to create a Leaflet map in a Jupyter Notebook and use all of our added functionalities such as the lat/lon switcher. In order to prove a good solution, we will need to show a Leaflet map of Mars loading in a notebook.

Challenges covered by demos:

In this section, we outline the demonstrations we have prepared, and exactly which of the challenge(s) each one of them proves a solution to.

Demonstration 1: Leaflet Map with a projection switcher, layer switcher, and lat/lon settings switcher.

Challenges addressed: C1, C3, C4, and C5.

Flight Plan: Step by step overview of demo

1. Show layers of Map loading and added to the layer switcher for the cylindrical projection.
2. Swap to north-polar and south-polar stereographic projections and show the layers for those projections. As we swap projections, make note of the scale bar changing.
3. Swap the lat/lon settings and move the mouse around to show that it is changing the coordinates.
4. Load the surface feature names overlay for Mars onto the map.

Evaluation:

- ✔ Convincingly demo'd each of listed challenges?
- ✔ Other evaluative comments:

Demonstration 2: Storyboards with different GUI layouts

Challenges addressed: C2

Flight Plan: Step by step overview of demo

1. Show the OpenLayers GUI layout.
2. Show the different GUI layouts we created.

Evaluation:

- ✔ Convincingly demo'd each of listedAQ23 challenges?

✔ Other evaluative comments:

Demonstration 3: Leaflet format

Challenges addressed: C6

Flight Plan: Step by step overview of demo

1. Show code snippets of a "Hello World" example following Leaflet's standards.
2. Show snippets of the documentation output.

Evaluation:

✔ Convincingly demo'd each of listed challenges?

✔ Other evaluative comments:

Demonstration 4: Leaflet map loaded from Jupyter Notebook

Challenges addressed: C7

Flight Plan: Step by step overview of demo

3. Show the Jupyter Notebook and the code inside of the cells.
4. Run through the notebook to create a Leaflet map of Mars showing a base layer.

Evaluation:

✔ Convincingly demo'd each of listed challenges?

✔ Other evaluative comments:

Other challenges recognized by not addressed by demo:

All were addressed