Hamza Qureshi
05-430 Programming Usable Interfaces
Hong; Xharaj
23 April 2018

Assignment 7: Implementing a Tool Project

Part 1

For this assignment, I sought to create a tool that would allow people in Pittsburgh to learn more about their surroundings. One of Pittsburgh's most plentiful yet underused resources is its massive parks system. To help encourage a higher visitorship to Pittsburgh's parks, I created One Park a Day. One Park a Day is designed to present users with a new park to visit each day of the week. A user who visits the site each day for a week would thus be able to learn about seven different parks that they can visit.

This tool presents several pieces of important information about its selection of parks. On the first page, the tool presents the name of a park, the neighborhood it is located in, two positive things about the park, one negative thing about the park, and an interactive map of the park on the right. On the second page, an image of the park and a description is presented. On these two pages, the content changes at midnight each day thanks to a series of manipulations of the Date( ); object in Javascript. On the third page, an interactive StoryMap.js presentation allows users to read about the history of all the parks in the system. The last page presents some information about One Park a Day and its goals.

This tool engages both new and old residents of Pittsburgh as well as visitors. The target user is a permanent or temporary Pittsburgher who enjoys the outdoors and is interested in learning more about the city's parks. The tool engages users by providing different content each day of the week and by providing information about the parks that is not very widely disseminated.

Park 2

- **Today's Park - Page 1** - by loading the home page, the current date and time is presented as well as the name and neighborhood of today's park, two positive attributes and one negative attributes relating to the park, and a corresponding map of the park.
    - **Google Maps** - by clicking and dragging on the map, the user is able to look at the surrounding area. By scrolling or clicking the "+" and "-" buttons, the zoom level of the map changes. By clicking and dragging the orange person icon onto a highlighted roadway on the map, the Google Street View mode is activated, and can be exited by clicking the "x" in the upper right corner.
    - **Refresh** - by clicking the logo in the upper left corner or the "Home" button in the upper-right corner, the home page refreshes.
- **Today's Park - Page 2** - by pressing the DOWN arrow key, the next page appears, which presents the name of the park, an image of the park, and a short description

adapted from the information on Page 1. By pressing the UP arrow key, page 1 reappears.

- **One Park a Day StoryMap - Page 3** - by pressing the DOWN arrow key, the next page appears with a full-page interactive StoryMap. By pressing the UP arrow key, page 2 reappears.
    - **StoryMap** - Clicking on the “Start Exploring” button or the right arrow loads the first slide in the StoryMap. Clicking on either arrow buttons loads either the previous or the next slide. Clicking and dragging on the map allows users to look at surroundings on the map. Alternatively, clicking on any of the icons on the title slide immediately transitions to its respective slide.
    - ***Note: In order to scroll to the previous or next page, first click on the header.
- **About One Park a Day - Page 4** - by pressing the DOWN arrow key, the next page appears with a description of the website and informational images. Pressing the UP arrow key returns to page 3.

Part 3

- FullPage.js
    - I chose to use FullPage.js because in my prototype, I refrained from creating scrollable pages and wanted to use a library that would allow me to maintain that aesthetic. FullPage.js also has animated transitions between pages built in which makes it easier to create.
    - I used FullPage.js to build the structure of my HTML file. I was then able to treat each section as a different canvas.
    - Using FullPage.js helps my website run more smoothly and removes the need for a bulky navigation bar or multiple pages.
- Google Maps Javascript API
    - I chose to use the Google Maps Javascript API so that the site could display an interactive map that allows users to locate the park and its surroundings immediately.
    - I used it on the first page and used Javascript to change the center of the map each day. I also used Javascript to style the map to give it a silver color palette.
    - Using the Google Maps Javascript API provides users with a sense of place when given a new park, making it easier for them to eventually reach the park.
- StoryMap by Knight Labs
    - I chose to use StoryMap because I wanted a highly interactive way to present historical information. StoryMap is anchored by a map, which harkens back to my use of the Google Maps Javascript API.
    - I used it on the third page to present the history of each of the parks I used. Knight Labs hosts the entire element on its own website, so including it simply required an iframe code.

- Including the StoryMap component allowed for a greater opportunity to use interactivity to learn more about the oft-known histories of each of the parks. Each park has a unique backstory, making StoryMap the ideal medium to present them with.

Part 4

I made three major changes between my InVision prototype and the final tool implementation. First, I created a new section on the home page to present information about the website instead of a separate page to keep it less clunky. I also lessened the scope of the website for the purpose of the assignment to only focus on Pittsburgh to make sure all of its components could be implemented. Finally, I removed the carousel of images on the main page and instead created a new section with FullPage.js in order to make the main page less bulky.

Part 5

My biggest difficulty in implementing One Park a Day came when attempting to present a different park each day. Initially, I thought to use p5.js because of its extensive date and time options, but it proved to be difficult to manage syntactically. Instead, I switched to using the Date( ); object in Javascript and a series of if-statements to allow the implementation to work the way I wanted. This method also assures ease of scalability, so I could easily add more parks if I chose to.