Princeton ArtFinder Design Document

Team

Name: Princeton ArtFinder

Members: Rhea Braun (rheab@princeton.edu)
Alden Hunt (*Team Leader)* (achunt@princeton.edu)
Thomas Martinson (tmm2@princeton.edu)
Sadie Van Vranken (svranken@princeton.edu)

Overview

Princeton ArtFinder connects Princeton students and staff with information about the art they walk by every day. Using a mobile-first web app, users can easily locate and learn about the public art closest to them on campus. Data for ArtFinder is pulled from public and private APIs provided by the Art Museum.

Requirements & Target Audiences

Although students walk by dozens of public campus art pieces every day, few have any understanding of what each piece signifies or have any recollection of what pieces they've seen. In order to increase interaction with and understanding of public art pieces on Princeton's campus, Princeton ArtFinder is a simple and intuitive web app to identify the closest pieces of art to the user and provide information about that art.

This tool will be designed primarily for the use of students, faculty, and staff on campus, who walk by Princeton's art on a daily basis. With a geo-located, mobile-first design emphasis, ArtFinder is designed to be used outside, near the art it is designed to give information about. We also anticipate that this will be used by visitors to Princeton's campus, including tourists and prospective students. Although we are not designing primarily for these populations, our application will still be useful for them.

In order to engage students and faculty, we will prioritize the following functionality:

- **Ease of access and simplicity of use**: In order to be realistically used on campus, we will prioritize a simple interface with minimal, core functionality, so that the app can be quickly loaded from a phone to get immediate information.
- **Location-first information**: Both in our interface and search functionalities, we will assume limited knowledge about the art on campus and facilitate finding information about art primarily through where it is, rather than through the title or

the artist. This information will be integrated with existing area-based campus divisions, so that art can be searched by residential college and academic area.
- **Thematic groupings**: In order to facilitate classroom and educational use, we will allow users to group art into themes and develop guided tours to pair users with art pieces they are interested in. This will interface with existing campus art guided tours, such as the Women at Princeton and Princeton and Slavery tours.
- **Personal relevance:** ArtFinder will help connect students forge personal connections with art through the addition of affinity tags such as "East Asian Art" and "Women Artists," to guide students through the art on campus.
- **Multimedia learning**: We will interface with the art museum's existing collection of videos, images and sound files about the art on campus to provide an in-depth, media-rich learning experience for users.

Current Solutions for Learning about Campus Art

The Princeton Art Museum maintains Campus Art Princeton , a website that includes a list view of all 600 art pieces on campus, the ability to sort by campus neighborhood and display the results on a map of campus, and more information about each piece of art, including soundbites, videos and images about the pieces. This site provides much of the same raw information that ArtFinder will provide, but is not mobile or location-friendly, which means that it fails to provide students with a simple way to look up information about campus art. By building a more interactive solution, we will increase engagement with the campus' art collection.

Functionality

Functionality Tiers

Princeton ArtFinder will provide users with a few levels of functionality:

Basic Level

- ArtFinder's main page shows a map with drop points for pieces of local art, either centered around the user's current location (if location data enabled), or a central point on campus if not. More information about the art can be found by touching/clicking the drop point. The map can be panned/zoomed from its initial starting point.

Search

- Users can search for campus art by key metrics, such as artist, medium, area of campus, name of work, year, etc. When a user selects an object from the search, the map is centered to its location on campus.

Menu

- Users can choose from several guided tours, which provide an interface which guides users to several pieces of thematically related campus art.
- Location tracking and notification options can also be set from the menu.

Potential Use Cases

Use Case 1:

Katherine spends a lot of her time in the equad. Sitting on a bench while waiting for her Professor's office hours to start in 15 minutes, she opens ArtFinder on her iPhone. After agreeing to allow the site access to her location, she sees a basic splash page with a map displaying all the pieces of art within 50 meters of her current location. Interested by the dot closest to her, she taps on it, which displays a pop-up with the image of a sculpture, the name of the artist, and the sculpture's title. Tapping on the hyperlink associated with the title, Katherine is taken to a page which displays several images of the sculpture, a description, and two videos about the installation and creation of the sculpture. Happily busied, Katherine spends the next 15 minutes learning about the creation of the sculpture.

Use Case 2:

John, a philosophy professor, opens ArtFinder while on his way to lunch on Nassau street. When prompted to allow the website to access his location, he declines, being suspicious of government surveillance. John then sees a splash page with a map zoomed to display all campus art pieces. Wanting to know more about the art near him, John taps on the search bar at the top of the page and enters "Rocky College." The map then zooms to that location and a list of search hits returns, so John can explore both spatially and through individual records.

Use Case 3:

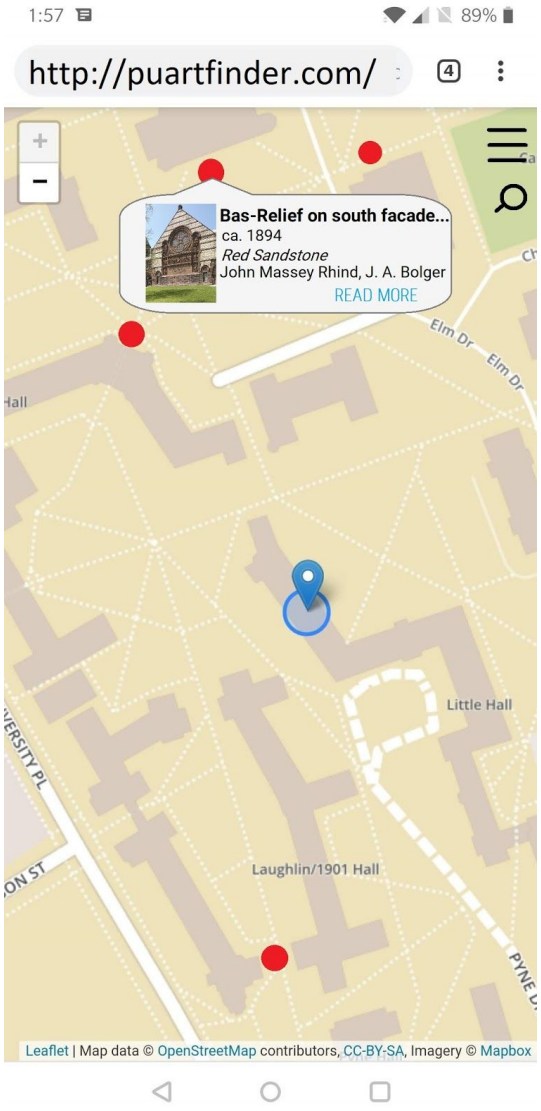
Emilia has been assigned to walk through the Princeton and Slavery art tour on campus for her Global History of Slavery class. She pulls ArtFinder, and after agreeing to share her location, sees the home page. She navigates to the Menu, then to "Guided Tours," and then clicks on the Princeton in Slavery guided tour. The app opens a new screen, which is split between an informational page displaying information first stop on the tour and a map showing where the first stop is located. She walks to that spot, reads the information displayed on the page, then clicks the next button to navigate to the next stop on the tour. A hyperlink is displayed at each stop that allows her to navigate to a separate page with more information about the piece before returning to the tour.

Design

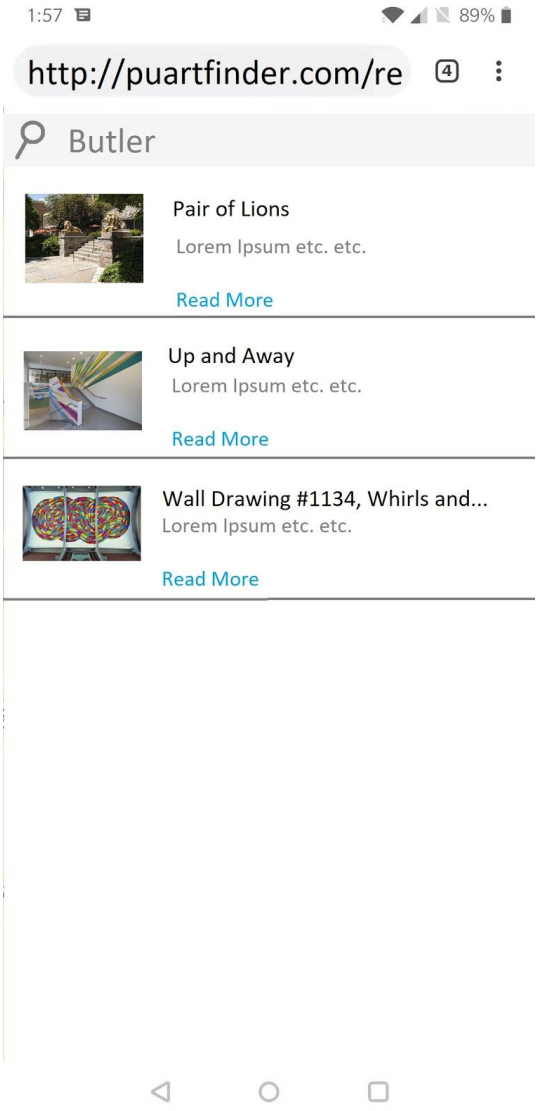
User Interface

The front end of ArtFinder will be built with the standard stack of HTML, CSS, and Javascript, enhanced with the use of Bootstrap for design and JQuery for Javascript, to ensure that the website's design works well on mobile. Throughout our development process, we will be optimizing the experience to work best on mobile devices, including smartphones and tablets.

For our mapping solution, we will be using Leaflet, an open-source, Javascript-based mapping system that will allow us to easily accomplish several of the core functions of ArtFinder, including location tracking, drop points for campus art installations, and standard scrolling/zooming functions.



Left: Basic Functionality



Right: Search Function

The server will be running Python programs using a Flask framework to implement the basic search and location functionalities. These core programs will grab object data from the database and prepare it for display on Leaflet in accordance with different use scenarios, interpreting location data and user interactions. The Flask server will be hosted on Heroku, where it will fall under the free use tier.

The database will store all relevant object data to search and guided tours. The database will first be created using the public APIs provided by the Art Museum using MySQL. Python helper functions will be created to quickly fetch object data as necessary for the server. Additional fields that will be added to those from the Art Museum's data will include coordinates, area of campus, and any "theme" connections for use in guided tours.

Timeline

Component Sequence

Different aspects of the project will have their own independent initial setup tasks, that will then merge to create a coherent product.

Database Order

1. Make Database using MySQL
2. Create helper function that uses API to add images/artworks to database
3. Create helper functions for database information retrieval

Server Order

1. Create main server
2. Create search functions

Front-end

1. Create barebones html index page
2. Incorporate Leaflet map and create Leaflet Javascript helper functions
3. Create menu interface
4. Create search interface
5. Javascript functions for menu and search

Weekly Timeline

3/10-3/16

- Initialize Database in MySQL and experiment with pulling data from Art Museum API.
- Begin basic front end frame with HTML/CSS for website. Experiment with integrating Leaflet map seamlessly into design.
- Construct basic Flask server to display homepage.

3/17-3/23 (Spring Break)

- Populate MySQL database with Art Museum public API data.
- Fully integrate Leaflet and website main page.
- Establish minimal interface between server and database.

3/24-3/30

- Clean up database information received from Art Museum. Add in campus art location data from newly available Art Museum API.
- Connect Leaflet with database location information via server.
- Begin to host server on Heroku and debug via the web.

3/31-4/6

- Root out lingering bugs between all three levels of ArtFinder. Especially, ensure that map is displaying the number of art pieces we want, in the right location.
- Add in Python logic which allows information to be displayed based on drop point selected, as well as individual webpages for each item generated via the database.
- Clean up website design and improve mobile usability. Also make sure mobile designs work well on desktop.
- Reach out to campus to find potential users/testers.

4/7-4/13 (Prototype due April 12)

- Meet with Art Museum to discuss any lingering core issues.
- Continue to clean UI in preparation for prototype delivery.
- Make sure drop points consistently link to correct webpages within Flask.
- Experiment with adding additional parameters to database to support future functions such as search, tours, etc.

4/14-4/20

- Experiment with search functions and their interaction with the database. Also, implement correct columns in database tables to handle tours and area searching.
- Prepare server framework necessary to handle search calls.
- Add a dummy search icon to the main page that will be modified to handle search in the future. Also, add page framework to handle tours.

4/21-4/27 (Alpha due April 26)

- Integrate a chosen search function fully with established database, server, and front-end system.
- Test and implement a few tours.
- Bug fix top issues with entire system.
- Distribute app to a few, core testers and invite user feedback.

4/28-5/4 (Beta due May 3)

- Fully implement all desired tours.
- Continue to bugfix major issues, especially those found through alpha testers.
- Optimize user interface based on tester feedback.

- Advertise app to broader campus and invite feedback.

5/5-5/11 (Demo on May 8)

- Bugfix all lingering issues with search and tours.
- Prepare media to demonstrate app.

Risks and Outcomes

We are confident that the core functionality of our app is a minimally achievable goal for the project, especially given the use of Leaflet, which is built for mapping apps such as ours. However, we know a few things will likely cause challenges along the way.

One known challenge is that while there are public APIs curated by the art museum that contain information on the artist, time period, title, description, etc. of every campus art piece, the geolocation data for pieces in the campus art collection is not currently available through a public API. We have talked with representatives from the Art Museum's IT team, who have said they will produce an API for us to use within a few weeks, but until that data is available we will not be able to test our system with every recorded piece in the campus art collection.

Another challenge we anticipate are the languages involved. Because we are pushing a mobile-first, responsive design, much of work will by necessity be done in front end languages such as Javascript, which our members have limited experience with. Time will be allotted to ensure we can work through problems that arise from integrating front end technologies.

Finally, we are aware of several smaller challenges. Notifications will be hard to implement, as we are developing a web app and not a native iOS or Android application. Searching will likely require integration with an external searching device that we may have trouble tuning. We will have to optimize our map-database interface well to avoid creating a buggy mess.