Mapster

Vijay Velagapudi, Laura Desmond-Black, Audrey Leung
Info 290TA | Fall 2014

Introduction

Mapster is a web application that allows users to view various events on a map and experience these events in a narrative format. Our long-term vision for Mapster is for users to be able to upload their own images and media, annotate them with events of their choosing on various maps, such as historical maps, Open Street Map, or Google Maps, save this information, and share this annotated map with other users.

There could be several use cases for this application. For example, in an educational context, teachers or students could map historical events on a map to more visually understand the chronology of events that took place as well as the location in which they took place. Individuals could potentially map past or future events to see the geographic plot of the map as well as provide their users a narrative format to share the travel locations in addition to text and supporting images for a more rich photo sharing application.

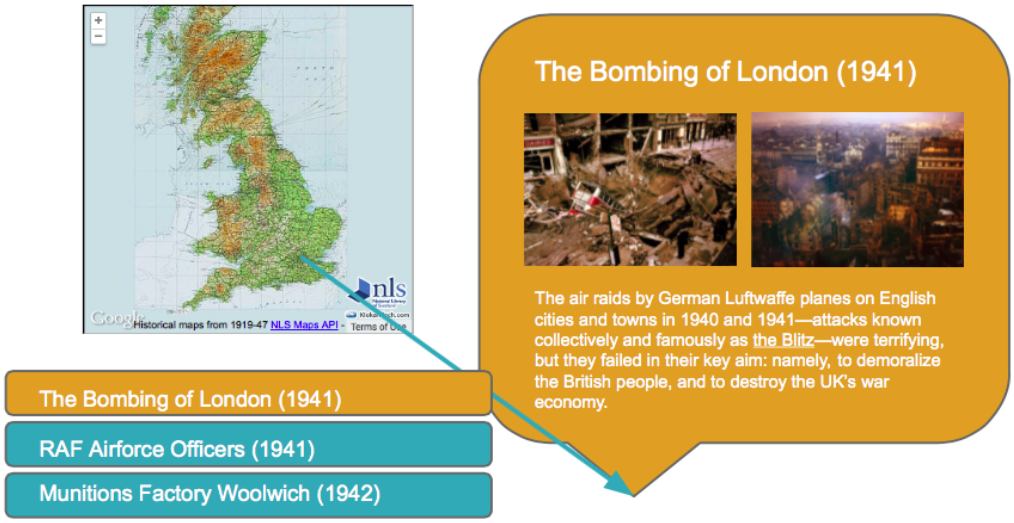
We originally did not start out with this idea and did a fair amount of exploration and pivoting before deciding on this project. Initially, we were aiming to create website to visualize potential careers based on current skills. However, upon further exploration of the public O*NET data where we were obtaining job skills information, we found that the data was not relevant enough for our purposes. Thus, we continued to research other sources of data and open APIs that that could potentially create an interesting mashup application. While searching, we discovered the National Library of Scotland Historic Maps API that provided historical maps of Great Britain for free non-commercial use. This spawned our current idea of mapping events on locations and allowing users to view them in context later on.

Mapster is currently functional with a database, web framework, and user interface. The back-end consists of a non-relational MongoDB database currently hosted on MongoLab. The web framework was created using Flask, and the front-end was created using HTML/CSS, jQuery, Bootstrap, PagePiling, and Leaflet. We designed the web application so that each component, such as the images, markers, and event annotations can potentially be

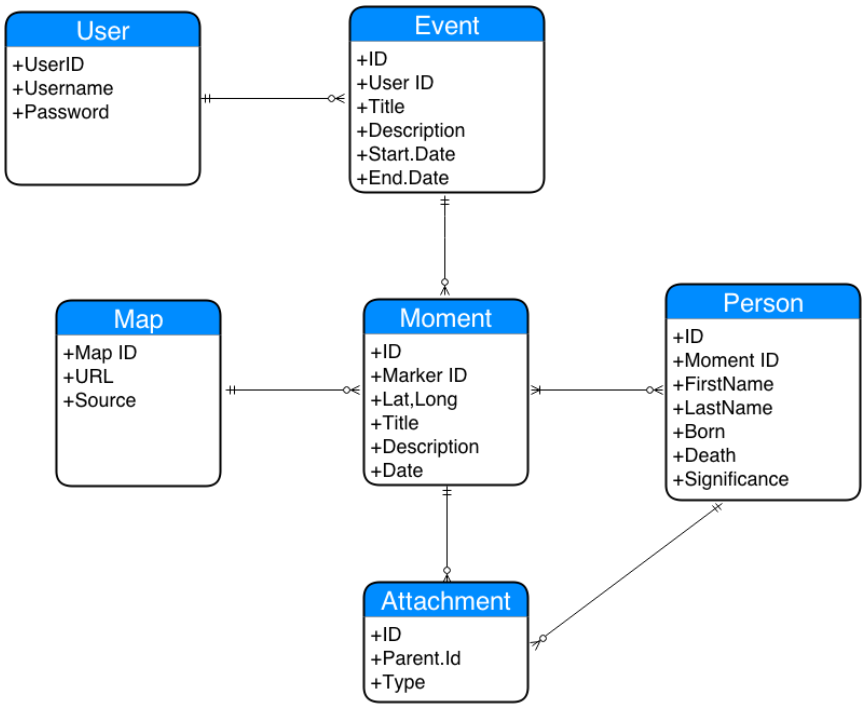
customized by the user, providing that the supporting user interface is also built. Therefore, in the long run, it would be fairly straight forward to build on the work that we have done this semester and allow users to make very customizable annotations using Mapster.

Description of Solution

We were originally inspired by mapping historical events from the NLS historical maps API, but we realized that users do not have to be restricted to the small subset of historical events that are relevant to this particular maps API, and that we could allow the users to annotate on any event and potentially allow them to use a map of their choosing.



Above is a snapshot of how we originally envisioned being able to view events that were annotated on this historical map. This screenshot shows the historical map of Great Britain provided by NLS and to the side of the map would be an accordion list of events showing just the event titles. Upon clicking on an event title, a popup would appear with a supporting description and images of the event along with a map marker on the location. Not pictured above is that users would also be able to toggle “next” and “back” to view the next or last event on the map.



Above is the Entity-Relationship Diagram that we initially created to organize the relational database. Each moment is related to the map, the event, related people, and other attachments such as images or document files. The intended design of this database schema was to make it extensible by making it easier to incorporate any future changes and by trying to compartmentalize many of the changes so that it would be easier to make changes to the application in the future.

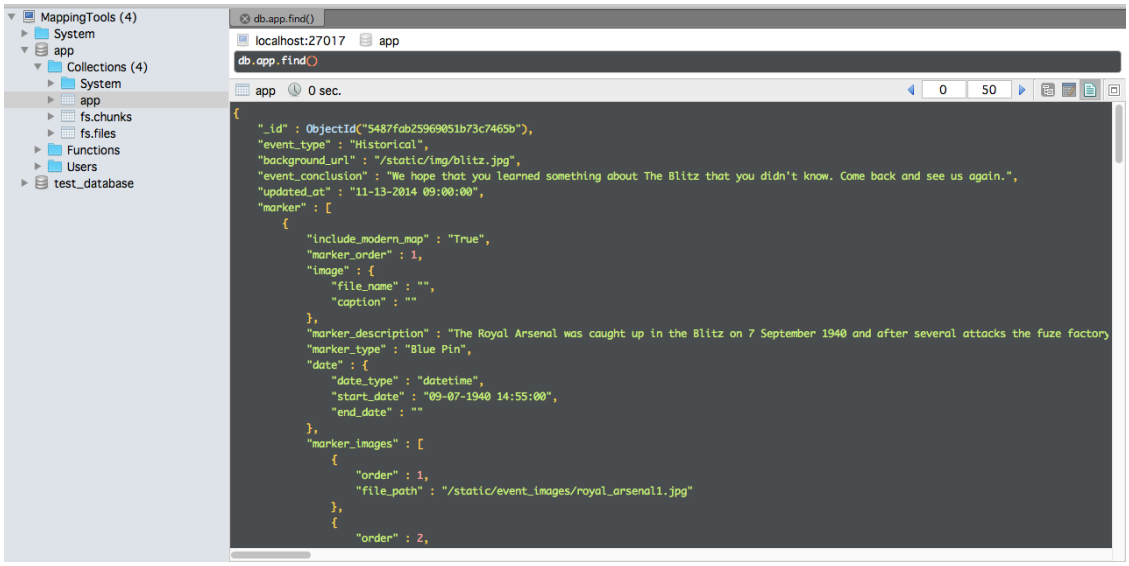
Below is the technology stack that we originally planned to use to create Mapster. Since this was our first time using many of these technologies, we decided to start with using Flask, which is based on a language that we were already familiar with -- Python. We initially planned to organize and store the information in a relational database using SQLite, given the simplicity of the platform and integration with Python. We hoped to learn AngularJS and OpenMaps to implement the viewing of events on the map and the navigation through the different events. We planned for Vijay to work on the backend database and server-side as he has had the most experience with those in the past, and Laura and Audrey to work on the frontend as they were more interested in learning about design and user experience.

We planned the website to have the following flow: The first time a user visits the site, he/she would login or create his/her username and password. Then, he/she would be prompted to create an event with multiple moments that would be annotated on that map. The user would annotate descriptions of the event and moments within that event. Additionally, the user could add attachments such as images or documents as well as add descriptions of people related to the event. Users would also have the option to easily embed this map on an external website such as a blog or educational website to share with others.

Description of Results

During the initial stage of work on the project, we decided to make some modifications to our approach after speaking with the professor. Firstly, and most importantly, we decided to start development from the presentation layer rather than by designing the UI elements to allow the user creation of the maps. The reason we made this adjustment was that we wanted to ensure that we had the most compelling aspect of the program built before we started working on building the interface, which would be a more straight forward exercise. Secondly, we altered our technology stack slightly to take advantage of some of the benefits of certain platforms. The diagram below shows the technology stack that we eventually ended up using:

We moved from using SQLite to MongoDB since we were doing a lot of front end handling of JSON data. For this reason it made sense to use MongoDB which uses a similar key-value pair syntax that would enable better interoperability between the backend and the frontend. During this process, we had to modify our data model a little bit to account for the shift from using a relational database. Below is a snippet of our our MongoDB collection.



We also switched from using OpenLayers to Leaflet for mapping due to the fact that we used Leaflet in class on some homework assignments and were already familiar with the framework. Originally, we wanted to use AngularJS to provide a more interactive user interface but we decided against using it because we were already using a lot of new technologies that we were not familiar with and we did not want to introduce additional complexity. We eventually settled on using a JQuery plugin called PagePiling which allowed

us to replicate some of the interactive features that we wanted from AngularJS such as the scrolling effect to navigate to new events. Finally, during the course of the semester, we realized that some technologies were much harder for any of us to use than we originally imagined. For example, using css for layout was much more difficult when going beyond basic elements. This meant that while we planned to work on separate areas that focused on our strengths and interests, we often had to work on multiple areas of the application in order make progress. Due to our lack of experience, we often had trouble scoping the different features that we wanted to implement and this made it difficult to set realistic targets for each project milestone. Despite not achieving all of the different features that we set out to do, we felt like we learned a lot through what we did actually implement. Furthermore, we learned a lot of important skills such as starting small and adding complexity over time. We believe that we would approach this project differently were we to start the semester over again and we think that that is a good indication of what we have learned over the semester.

Technical Details and Using the Application

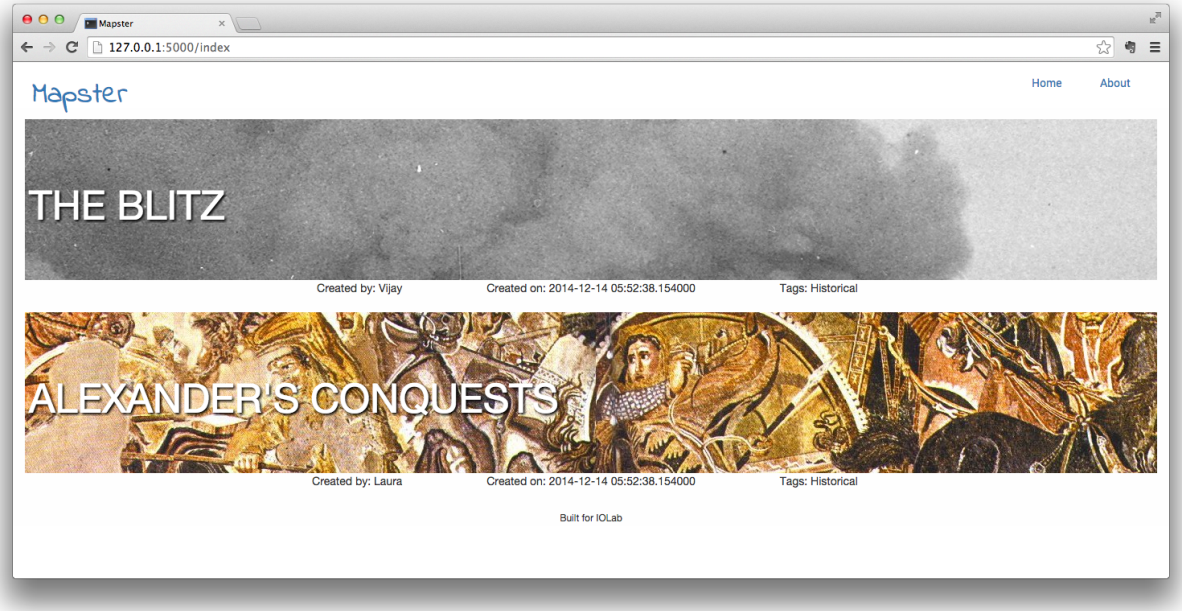
Deployment

The application is currently deployed on Heroku and running at the following url: https://powerful-dusk-7554.herokuapp.com/. The database is hosted by MongoLab which is also an add on Heroku. We originally developed the application by running a local copy of MongoDB and having the seed data programmatically inserted dynamically. However, deploying to Heroku, we had an easier time using a cloud MongoDB instance than to create our own database instance. Our code lives on our Github private repository and is located at this url: https://github.com/vijayv/mappingtools.

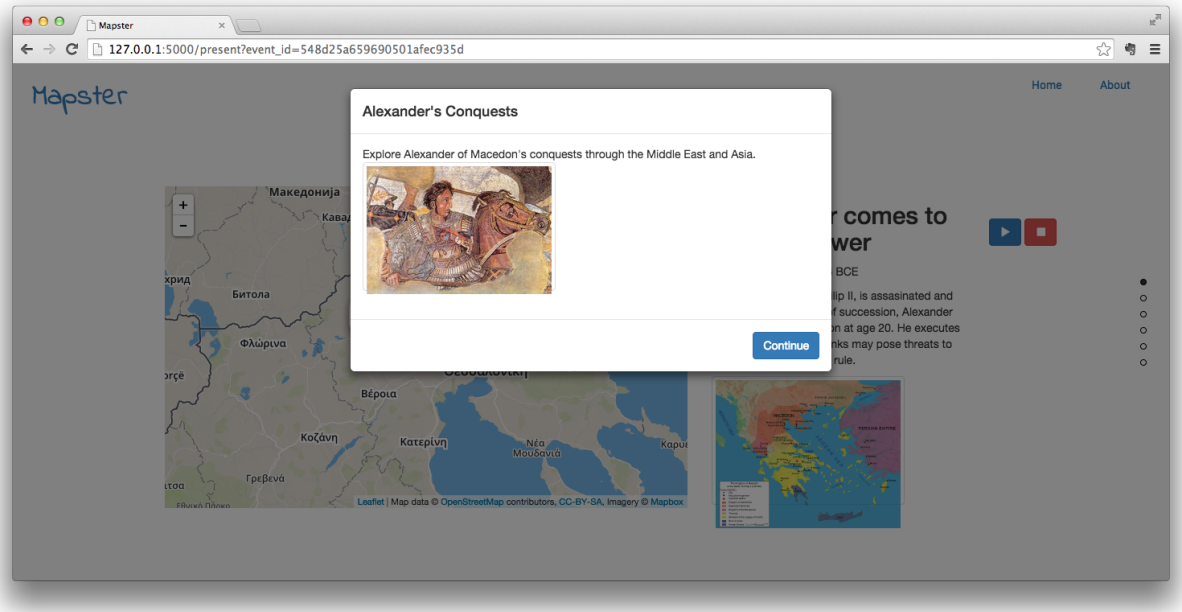
How to Use the Application

A user of the application would take the following steps to use the application in it's current form:

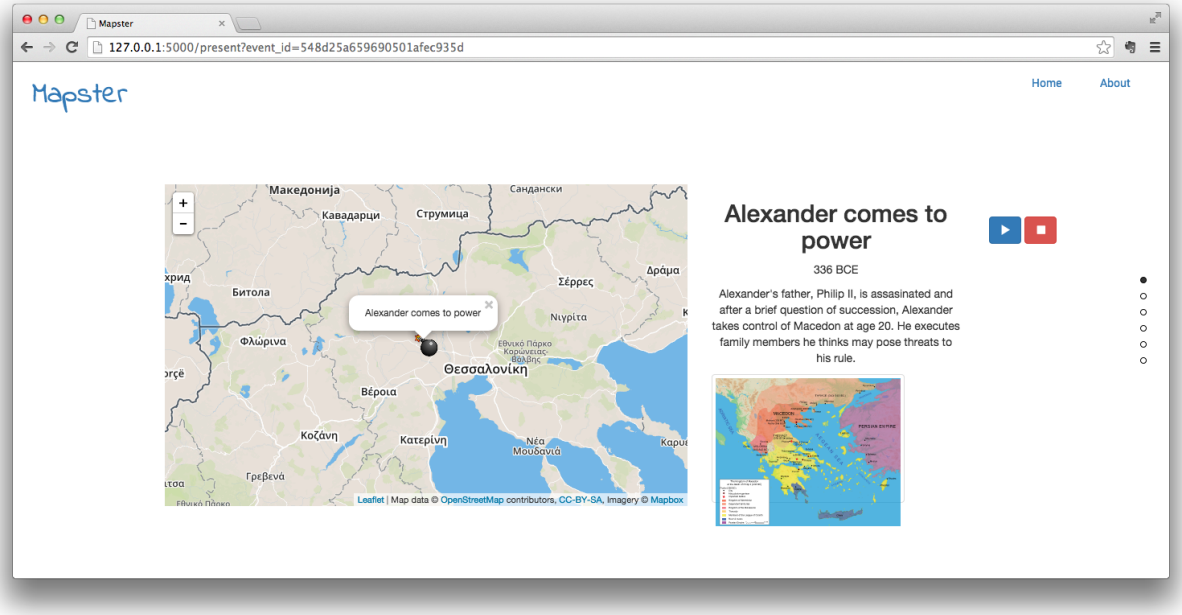
1. Enter the URL into a browser to view the homepage which lists all of the mapped experiences available for viewing:



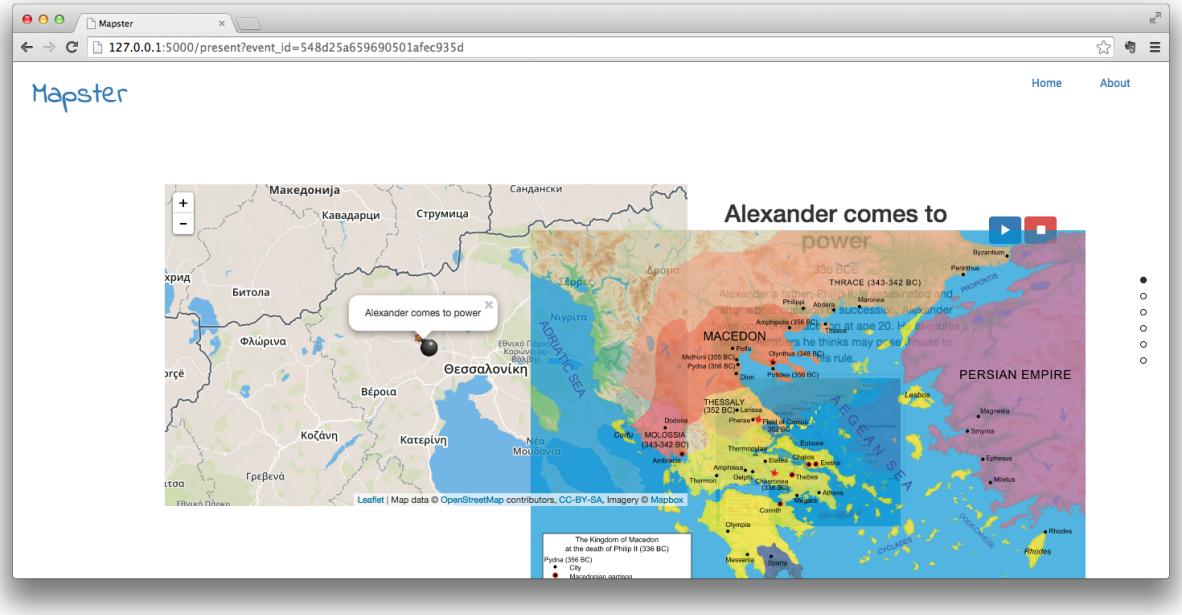
2. Once the users selects an experience to view the application will move to the introduction page for that mapped experience:



3. After clicking “continue” the user can begin to navigate the events of the mapped experience. A set of circle icons on the right side of the page shows the user how many events are in the mapped experience:

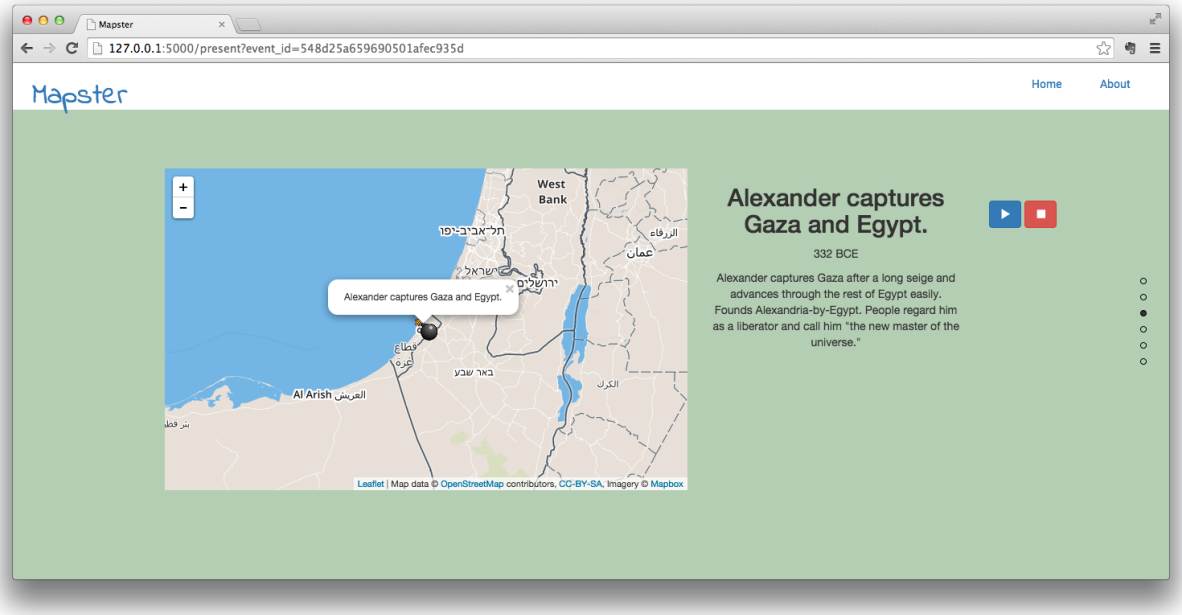


The user can hover over an image in the event description to see a larger version of the image:

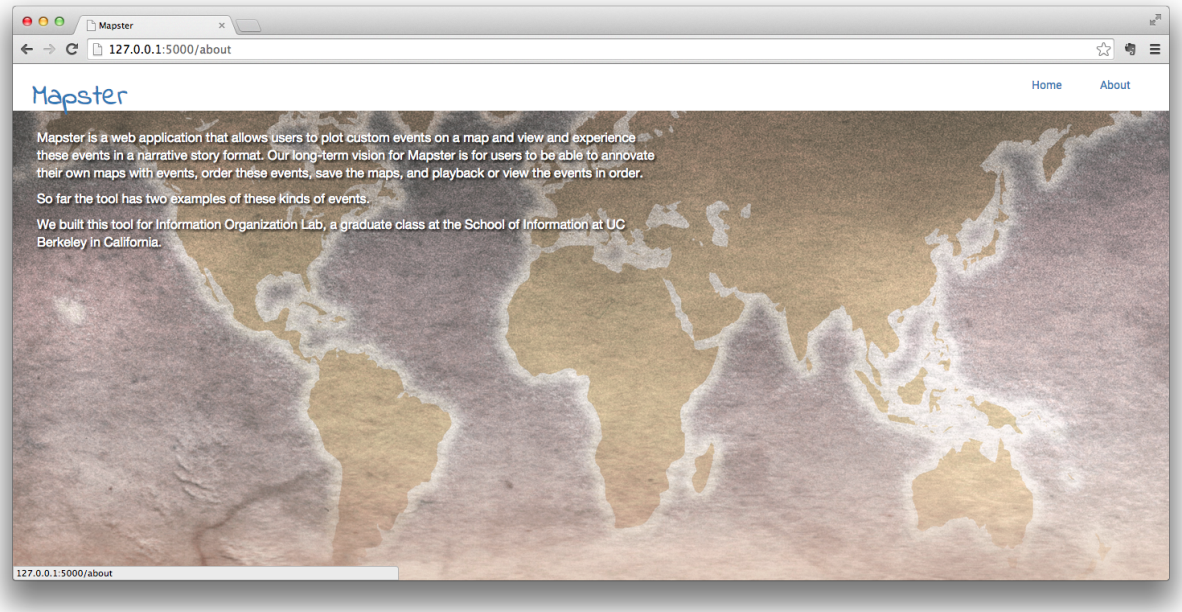


4. A user can move through the events in the mapped experience using the forward button, scrolling down (or up), or by selecting a circle icon from the right edge of the page. The play button will reveal the next marker after 10 seconds. Using the icons allows the user to move immediately to any other event within the mapped experience.

Additionally, the user is able to interact with each event map without affecting the other maps on the page:



5. A user can go back to the homepage anytime by clicking on the “Home” link at the top right of the site or by clicking the “Mapster” logo at the top left. To find out more about the project a user can also access an “About” page with the link at the top right:



Running Application Locally

The application can be run locally by cloning the repository and following the directions in readme.md:

Getting Setup

1. git clone git@github.com:vijayv/mappingtools.git
2. Highly recommend using virtualenv
3. If using virtualenv, activate it
4. Install from requirements file: pip install -r requirements.txt
5. Start flask server: python app.py
6. Navigate to http://127.0.0.1:5000

* The application requires internet access to be able to plot the maps and to speak to the database.

Running the database locally

1. cd to app directory
2. mkdir ./db
3. Start mongodb: mongod --dbpath ./db
4. Change URL string to use localhost in app.py and data.py
5. seed data: python data.py
6. run: python app.py

Testing

Most of our development was done using Google Chrome, however, we tested the application in other browsers. We did not encounter significant differences between Firefox and Chrome.

	Google Chrome	Safari	Mozilla Firefox	Opera	Internet Explorer
Home page	✓	✓	✓	✓	unable to test
About page	✓	✓	✓	✓	unable to test
Map - Blitz	✓	x	✓	✓	unable to test
Map - Conquests	✓	x	✓	✓	unable to test
Map - Play Button	✓	Button has no highlight	Button has no highlight	✓	unable to test
Map - Stop Button	✓	Button has no highlight	Button has no highlight	✓	unable to test

Known Bugs/Need Additional Development

Bug	Additional Description
Some dates are stored as strings in the database	This is because we did not want to have a hard requirement for the formatting of dates so that we could allows users to input but 387 BC as well as 9-1-2012. However, this will not work with user generated maps.
Play button does not automatically stop after it reaches the final event. Instead, it will try to keep moving the slides even after the end has been reached.	
Images are currently stored locally on the file system. This is not scalable and should be moved in the long run.	
img elements overlapping with the mapbox element on zoom display the map attribution tag on top.	