## Part 1: Web Framework

-Which web framework did you choose for your project and why

We decided to use Ruby on Rails as our web framework for multiple reasons. We chose Rails because of of it’s large library of gems and the ability to have a lot of open source options out there that we can build our code on top of, using gems such as omniauth if we felt the need to add in OAuth authentication. That might be an option in the future to allow users to store their favorite parking spots, maybe even rate which ones have the best value. We felt that using Rails would be the best choice to allow us to expand our features such as with this. We also felt that using Rails as our framework would allow us to scale to meet more demand in the future if we wanted to be able to expand this into multiple cities. We also liked and have used ActiveRecord a small amount so that was another benefit of using Rails as our framework. Both of us have also worked with Python a fair amount so using Rails as a framework would make us step out of our comfort zone and into something that would require more work, by using Ruby and other additional languages/types used with Rails such as Haml.

-How does it fit in with your project goals and architecture

We wanted to get the experience of developing an app with Ruby on Rails as it is a very useful language and framework to use currently. It aligns well with our project goals and architecture because it allows us to use all the externals that we need to such as JavaScript and JQuery through a gem, along with being well integrated already with ActiveRecord which uses Object Relational Mapping.

-What has your experience been so far with the framework

It has been very interesting to learn all the new features of Rails but it is also a steep learning curve and although there is a lot of documentation available it is a lot to learn. Since a lot of our application depends on JavaScript interactions, because our app relies heavily on Google Maps JavaScript API, a challenge so far has been implementing that. We also face a lot of design decisions in that we have to weigh the benefits of using certain external languages along with our framework such as choosing erb or Haml, or using coffeescript instead of plain JavaScript. However, this does make the project more interesting because it requires us to learn the advantages and disadvantages of other languages, gems, etc.

## Part 2: Information Architecture

-What are some key user personas of your application and what are the information needs of these users interacting with your system

We envision three key user personas for our application- web, mobile, and accessing through an API. The web and mobile user’s needs would be very similar save for a few differences and capabilities resulting to differences in design. We see the web user as needing access to a map of Berkeley that stores all the parking information of areas within Berkeley that they plan to drive to. In terms of given user input, they would want to search by street, area, parking rate, max time, and then be shown a map corresponding to their input. The web user would also want more of an interactive experience, rather than having to input text every time they fire up the website. They would also be able to use the map itself and hover over certain streets to display parking information of that street, to provide a more interactive way to interact with the app. However, the mobile user would not be able to “hover” over a map so that feature may not be able to be implemented on mobile, or converted into a click feature if that is possible to do. The map itself would take up the majority of the screen as that is the key component of the app, which is designed to be simple in order to give users unobstructed access to the information. The third user would be that of a developer looking to utilize the information we provide, the street data. They would be able to use our existing code to get current street parking information and use that however they use, instead of using the currently very outdated parking information site. As a side note, we also believe designing it with APIs in mind will force us to use convention over configuration which will result in us writing better code.

-What entities or objects are you storing in the application. How do these map to your user's information needs

There are two objects that we are storing in our application, a Street and a Coordinate. We needed to store them as two separate objects because we have a has\_many relationship from Street to Coordinate. These two objects will be combined to create Polylines within the Google Maps JS API, and then “drawn” onto the map on the page, which can then be interacted with either through hovering or user text input. We plan on making the user text input as user friendly as possible by creating dropdown choices for some of the more ambiguous choices such as parking rate, max time, and area. We will also be looking into auto completion for the street field but we are not sure yet if that is possible. This maps to our user’s information needs because on the front it seems direct and simple, which is what we want the users experience to be since they may be in a rush or on the go with their phones looking to find the nearest parking spot to go to.

-How do these entities interact with each other - Which model did you choose(Relational, non-relational and Hybrid)

We chose a relational database because our information architecture is simple in that we only have two objects that we want to use at the moment. In the future we may look at giving users the option to save their favorite parking spots but for now we are sticking to just storing street data. Using a relational database suits our needs best because it also works well with Rails (we are using ActiveRecord)

-Describe how this information architecture will scale up

Our main challenge with scaling up our architecture is how we design our database and how we are going to manage to “scrape” the parking data from various city websites in the future. For this project, we are planning on manually entering in the coordinates from Berkeley City’s website but that is not scalable for the future so we will need to find ways to do that. We could write JavaScript functions to grab the HTML DOM elements in the future that could grab the necessary parking information from a website. Another need will be how we design the Google Map so that it will be fast enough to load for a user when our data starts to increase. We talked about offsetting this by only loading data for certain areas that the user is currently looking in, or areas that they search for and thus reduce the amount of time it takes to initially load the page.