

Beer Me KC

Team 4
Increment 2



1. Introduction

The craft beer boom currently has an unequivocally upward trend today as more and more people are drawn away from the traditional American lager. As with any cultural experience, the craft beer hobby attracts a wide variety of personalities. Some people simply want to go out for a beer with a friend. More fastidious beer aficionados are only sated by pouring themselves into an obsession; a relentless pursuit to try them all. Enter BeerMeKC, a mobile web application that will enable beer consumers to locate the best craft beer for them, with an early primary focus on the greater Kansas City area.

2. Project Goal and Objectives

2.1 - Primary Objective

The primary goal of this app is to provide a mechanism for beer consumers to locate craft beers in their geographical region. The tool will house information on local breweries and taprooms - their current supply of beers, seasonal beers, and more. Users will have the ability to create and maintain an individual profile. They will be able to keep a wishlist of beers they would like to try, while also tracking the beers they have already tried. Untappd[1] already provides this feature, but we plan to focus on local breweries. One of the main initiatives of this app is to provide Kansas City beer fans a unique web app where they can interact with their friends.

2.2 - Specific Objective

Kansas City beer drinkers will be able to enhance their lives with this app. Locals generally consult Facebook groups, friends, and Yelp reviews, as examples, to learn about the beer in the region. Because of this, social media will be important in this app. Concerning social media, the end user should be able to do the following:

- Tweet an experience at a local brewery
- Share an experience on Facebook
- Search for specific types of beer in the area
 - By beer category
 - By brewery
 - By popularity
- Write a review for a brewery
- Maintain a unique profile
 - Wish list of beers
 - Beers they have tried
 - Favorite breweries/beers
 - Experiences

Brewers and brewery employees will also be able to contribute to the app, by doing the following:

- Tweet and write Facebook posts advertising a current special

- Maintain their specific lineup of available beers
- Post a profile featuring a brewery employee
- Potentially share information on recipes for specific beers

Besides the above, all user types will also be able to interact with one another using message boards specific to the app. Message board forum categories will include General Discussion, Local, US Craft, Import, and more.

2.3 - Significance


Kansas City craft beer drinkers have many ways to learn about beer, interact with friends discussing beer, and find a location with a beer they might enjoy. However, all of these sources of information are separate and there is no consolidated mashup providing everything in one place. That is why our project is important and will be used.

3. Project Background and Related Work

Many solutions exist that provide some functions that our web app will have in the form of web apps and APIs. BMKC is an app that hopes to fulfill as many functions that will be relevant to a beer consumer as possible. First of all, our primary resource of data will be BreweryDB[1]. This API is very robust and well defined. So as to have more rapid access in the future the the data and more freedom with how we will use it, we will store it in a MongoDB. Virtually all of our early data will be provided by BreweryDB.

BreweryDB appears to be the most comprehensive API storing information on specific beers. However, there are still listings that are partially incomplete or missing. BMKC will help to fill in those gaps. Here is an example - a beer from Kansas City's Big Rip Brewing [2], called 237 Milk Stout. On the BreweryDB website, the beer information is listed as follows:

237 Milk Stout
Big Rip Brewing Company
Edit Beer



No Label

Name: 237 Milk Stout
Brewed By: Big Rip Brewing Company
Style: Sweet or Cream Stout

No description available

ABV	IBU	SRM	OG
5.5%	N/A	N/A	N/A

The listing is missing important information such as IBU (International Bittering Units) and OG (Original Gravity). Breweries do not always list the OG, but IBU's are usually listed on beer menus. A user could update the listing, a brewery could validate the update, and our database will be updated. Next time the request is made for the beer, it will be retrieved from MongoDB with updated parameters.

Social media requires a mention because it drives everything when it comes to online relationships and information share. Facebook[3] and Twitter[4] integration will both be important components to our project.

Finally, Google Maps API [5] will be used to help users locate breweries. This API provides the ability to find locations, calculate a route to a destination, and filter location results. All of these functions will be taken advantage of by BMKC.

4. Proposed System

4.1 - Requirements Specification

Functional Requirements:

- Distinguish between three types of users; brewer, brewery employee, and user
- User Authentication
- Detect the user's location
- Plot on a map all craft beer locations
- Locate a place to drink given a user's favorite beer type
- Locate breweries within a specified distance
- Allow users to provide a proposed update to a listing (see section 3)

Nonfunctional Requirements:

- Scalable; we must be able to expand to other cities easily
- The user must be using an Android device or web browser
- The UI must be easy to use
 - We will use a minimalistic design giving the user a few options on each page

4.2 - Framework Specification

4.2.1: Assumptions

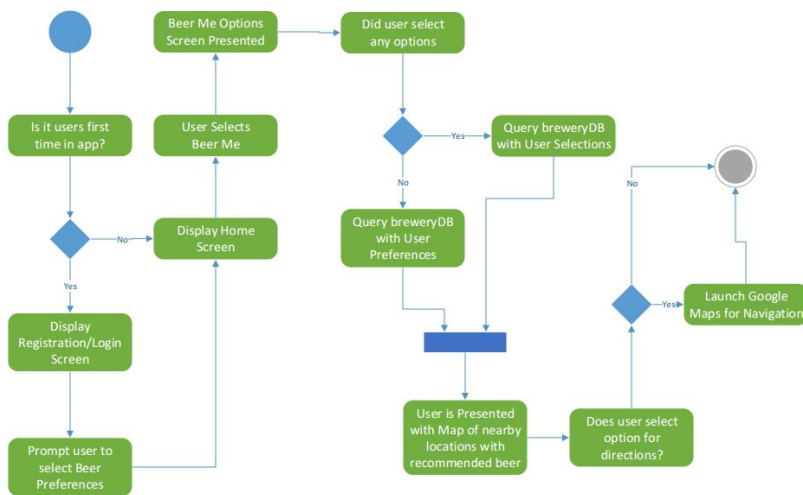
BMKC's dependencies, assumptions, and constraints can and will change as the project evolves. As these changes occur, the Project Plan will receive a new version ID in order to track evolution of the project as it is being implemented. The assumptions at the time of the first Project Plan release (1.0) are the following:

- Where the BeerDB data is present, it is assumed it is up to date. During the course of the project, some listings may be manually updated in order to demonstrate functionality

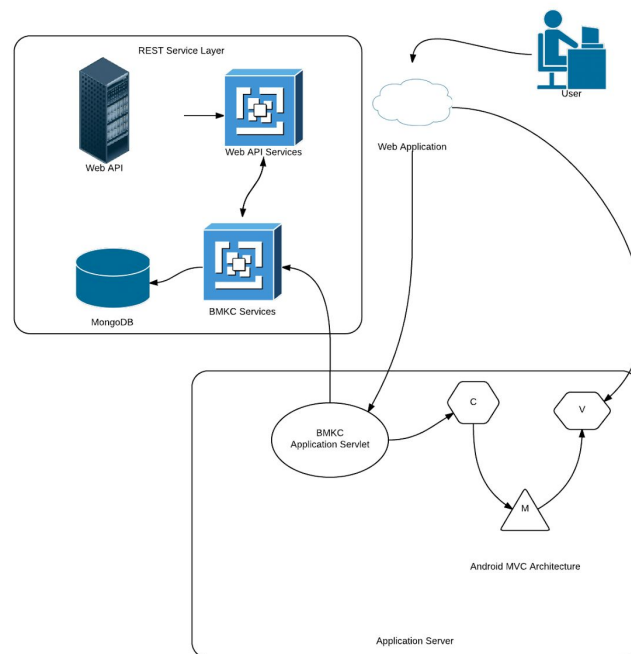
and robustness of our app, but it is out of scope of this project to update all information on Kansas City breweries.

- Other APIs work as advertised - Google Map API will provide accurate directions and search results, for example.
- Iteration planning will involve hard deadlines. If a feature is not implemented within seven days of iteration completion, the team will discuss possible re-allocation of responsibilities
- The final system developed will work as intended and users will have enough experience with web applications to know how to use it.

4.2.2: UML



4.2.3: System Architecture Diagram



4.3 - System Specification

Existing Services

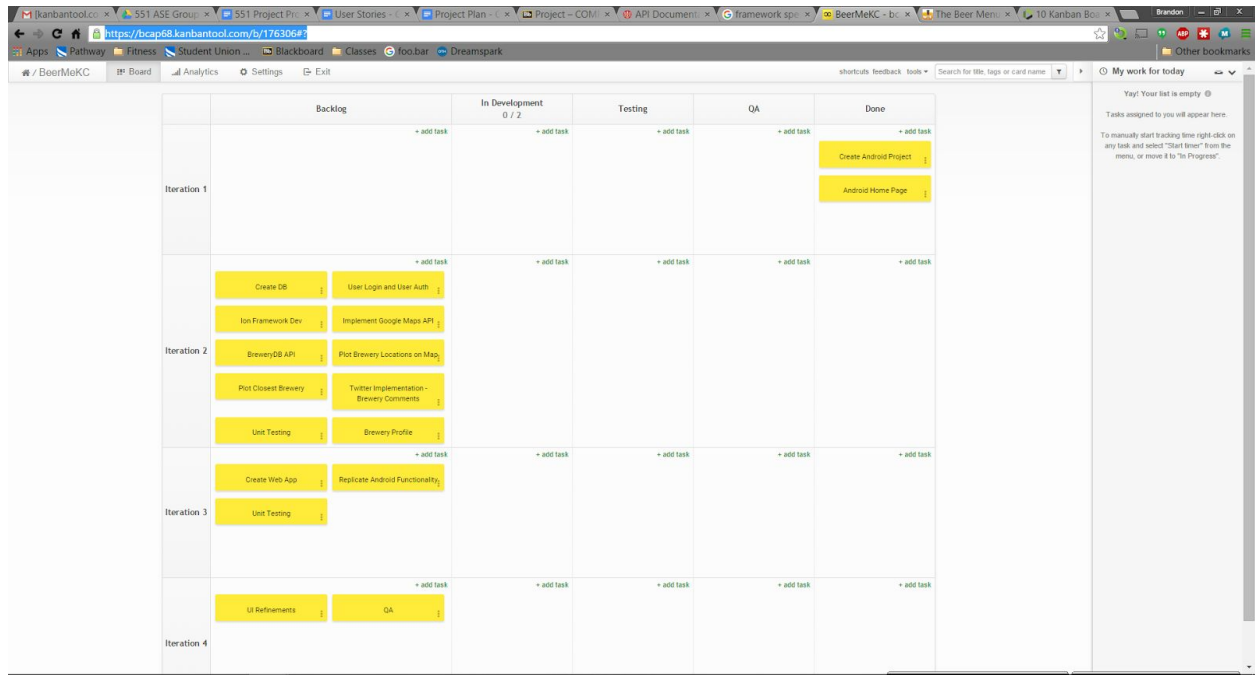
API	URL	Description
Facebook	https://developers.facebook.com/products/sharing	Share something on Facebook
Twitter	https://dev.twitter.com/rest/public	Tweet something on Twitter
BreweryDB	http://www.brewerydb.com/developers/docs	Find information on beer and breweries
Google Maps	https://developers.google.com/maps/?hl=en	Find locations and directions
Mongodb	https://www.mongodb.org/	Data Storage

New Services to be Built

- BreweryDB API Consumption Services
- BreweryDB Data Conversion to MongoDB
- MongoDB Consumption

5. Project Plan

Pictured below is the project plan using Kanban.



A basic approach to our plan is to complete our Android Application in Iteration 2, but realistically it may spill over into Iteration 3. In Iteration 3, we would like to implement everything from the Android application into the web application. Iteration 4 will be primarily for fine tuning our application and quality assurance.

Current Work Delegation:

- Brandon Andrews
 - Android Development
 - Front End/UI
- Todd Brees
 - Web Application Development
 - Database Administration
- Jordan Larson
 - Web Application Development
 - API Service Implementation
 - Database Administration
- Matthew Velasquez
 - Android Development
 - Front End/UI

Our current work delegation divides the project in half and forms subgroups. Ideally, this will allow us to communicate more effectively and concentrate on fewer technologies. Again, these delegations are speculated and members may be moved around given time constraints or expertise.

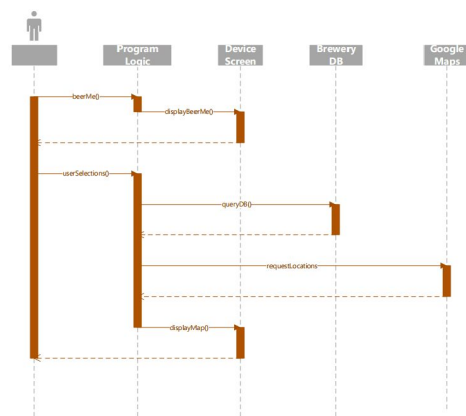
6. First Increment Report

6.1 Summary of Diagrams

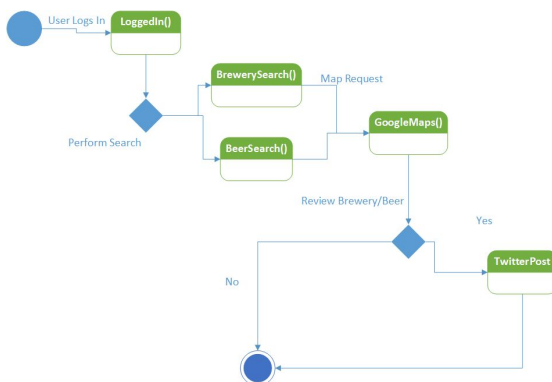
Class Diagram



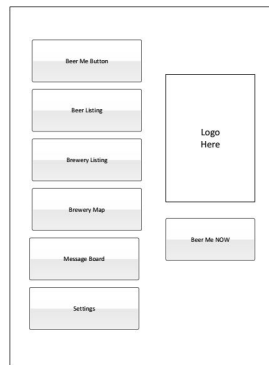
Sequence Diagram



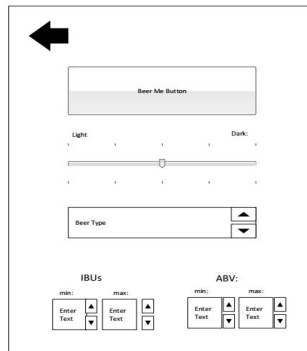
State Diagram



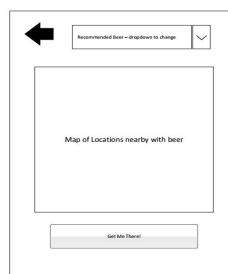
6.2 Initial wireframes



Pictured above is the wireframe for the main page of our Android Application.



The wireframe above allows the user to find a beer based on the following attributes: light or dark, type, International Bitterness Units (IBUs), and Alcohol by Volume % (ABV).



The wireframe above shows the screen where the user is able to locate breweries near them. There is a drop-down that allows the results to be filtered. The results are plotted on a map and the user can click the button at the bottom to open Google Maps and begin navigation.

7. Second Iteration

7.1 Introduction

Working in the Agile methodology, we focused on the primary framework of our application and the objective of representing map data. After some initial work in android studio, we transitioned to brackets and the ionic framework. Once we began to work in this context, our focus was on the following functionality:

- Initial (one-time) login
- Application Structure
- Map functionality
- Integration of MongoDB API through nodejs server
- Establishment of services with breweryDB in AngularJS

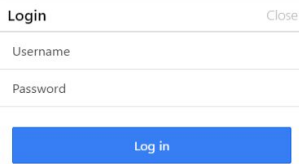
This core functionality will give us the base framework as we proceed to adding the recommendation logic and user data moving into the next increment.

7.2 Objectives/Features/APIs Used

As we approached iteration 2, we wanted to take learnings from the tutorials and translate them into our application's functionality. Looking at use cases, the immediate goal was to create a login experience for the user, and then be able to provide the user with location-based data. Based on these priorities, we spent more time working on the base layout, login functionality, map execution, and collection of data. While working with ionic, we experienced challenges bringing all of these components together, but ultimately were able to complete separate modules that were delivering the desired outcomes, integration is now a key focus moving forward.

7.2.1 Login Functionality

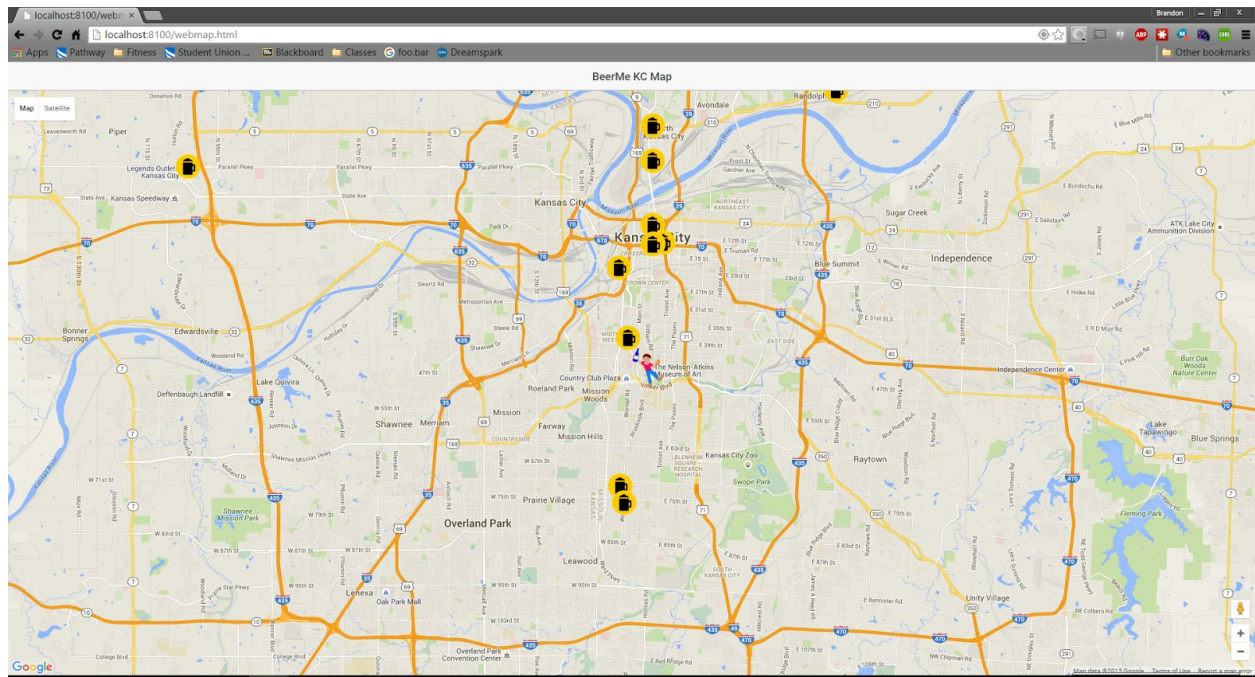
The login page was initially created during iteration 2. This constituted the addition of the login.html page seen below as well as additions to the controllers.js file. These additional functions handle the data being passed from the login page as well as triggers to close the form once the user navigates away. For the purposes of testing a simulated login delay was added but this will be replaced once the true login system has been implemented. Adding the OAuth functionality to the login page proved to be difficult as configuring ionic to pass the proper credentials to complete authentication is an ongoing item. In addition to this, methods were explored to add the ability to use a Facebook or Google+ sign-in as a means of gaining access to the application. Registration functionality will also be included as users will be stored in the mongo db.



Login	Close
Username	
Password	
Log in	

7.2.2 Map Functionality

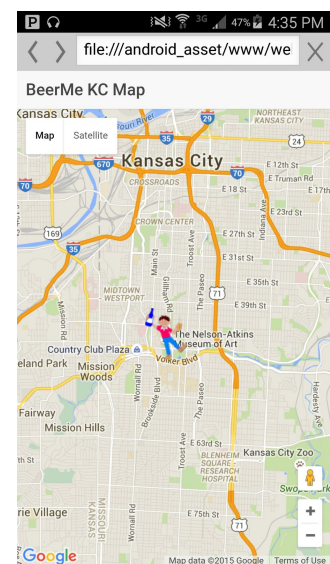
Implementing the Google Maps REST API has proven to be a challenge for cross-platform development between the web and Android. The map would not render on Android, but it would on the web and vice versa. Fortunately with AngularJS, we were able to modularize an Android map and a web map, then render one or the other based on what platform was running. This was deemed inefficient and to be against the primary principle of the Ionic Framework, cross-platform development.

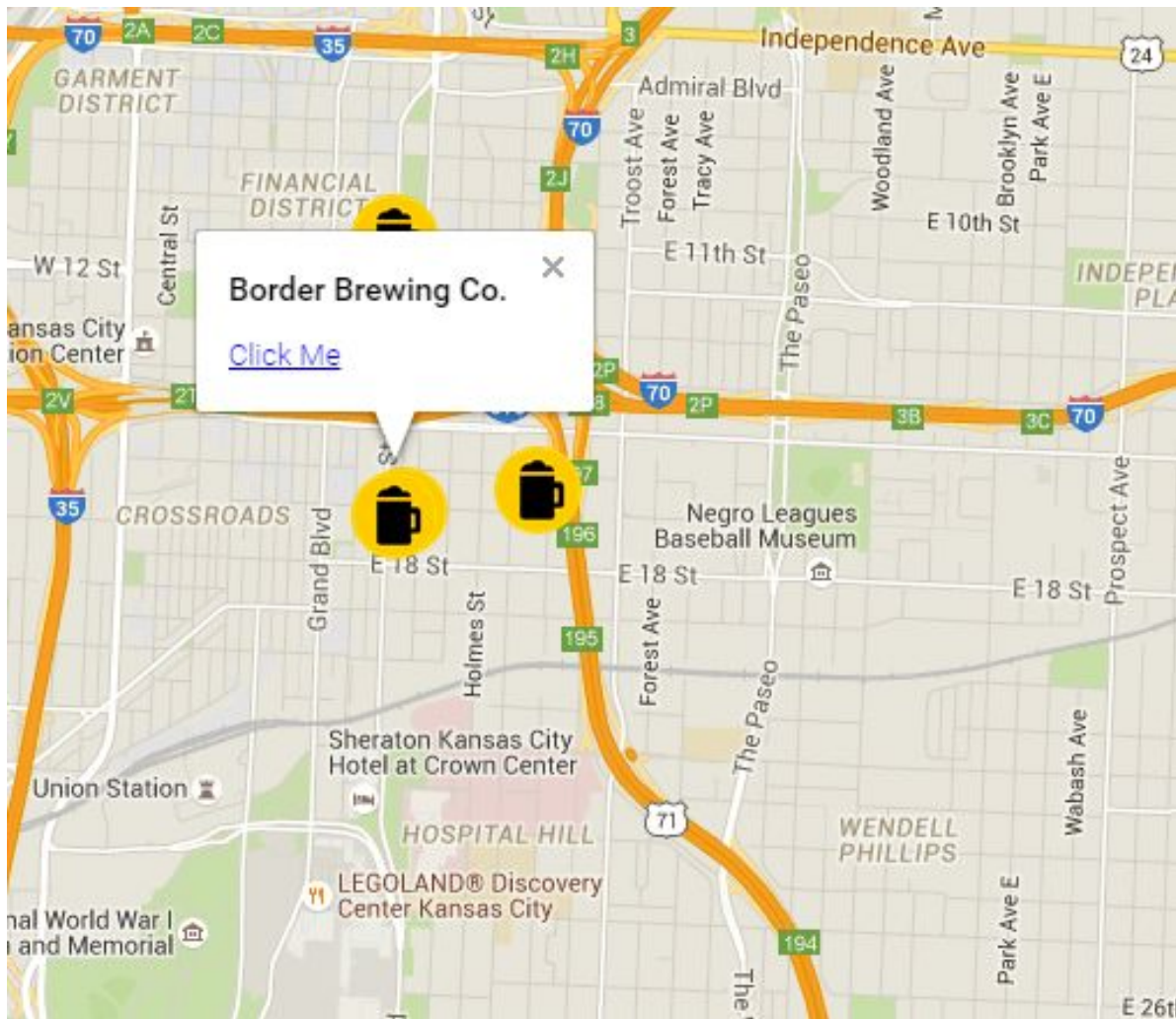


ngCordova possesses a plugin for an In App Browser:

<http://ngcordova.com/docs/plugins/inAppBrowser/>

This plugin provides a web view of the map and solves our cross-platform development problem. As seen below, the map centers on the user's position and places a marker (represented by a cartoon character holding a beer). Additionally, we used the factory design pattern to make a request to the BreweryDB REST API and place markers for all of breweries within Kansas City.





Each marker has an information window which populates the name of the brewery as well as a link to get driving directions from the user's current location to the location of the brewery.

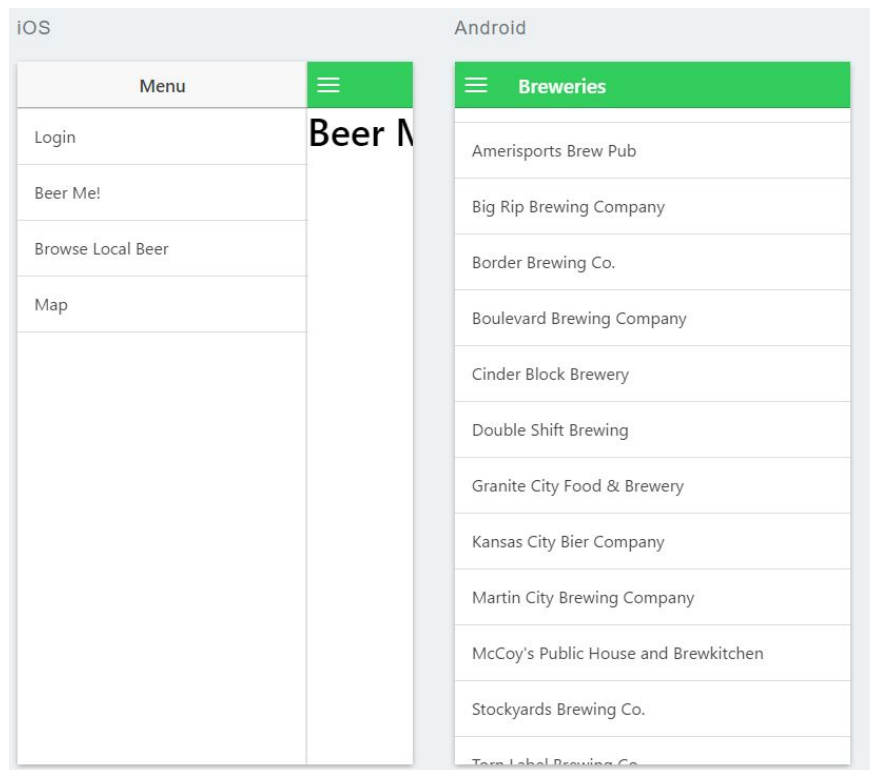
The link is dynamically created based on the brewery's latitude and longitude retrieved from the REST API:

```
"<a href='https://www.google.com/maps/dir/Current+Location/' + record.latitude + "," +  
record.longitude + "'>" + "Click Me"</a>"
```

With the mapping and location of the breweries, we turned to the BreweryDB API to assist with our beer information.

7.2.3 Kansas City Beer Information

Using the BreweryDB API, we developed a view that shows a list of the breweries in Kansas City locale. The list appears as follows on the app:



Kansas City Area Breweries

The list of breweries is sorted alphabetically, and if a user selects a brewery, brewery metadata is populated into another view. At a later date, we will have a list of beers displayed for every brewery a user selects.



Individual Brewery Info Page

The individual brewery info page lists the brewery's name, a description of the brewery, the logo, and later it will include a list of the beers offered as well as the ability to select a specific beer to view specific information for that beer. All data will initially be obtained from BreweryDB, but users will have the option to contribute more information through our app if a listing is incomplete.

7.4 Design of Services/Architecture

7.4.1 - NodeJS, ExpressJS, and a new overall architecture

We understood the value of writing an overall API for our webapp, so we created a server to serve that purpose. The server is Node and Express based and will ultimately be the location where we fetch any data from. AngularJS business logic will be utilized to further refine result sets of data so that we do not spend too much time writing multiple REST calls. After all, in order to retrieve Google Maps data, for example, We need to make a REST service call to our NodeJS API server, which then reaches out to Google Maps.

```
93 // Create endpoint handlers for /users
94 router.route('/api/users')
95   .post(userController.postUsers)
96   .get(userController.getUsers);
97
98 router.route('/api/users/:id')
99   .get(userController.getUser)
100   .delete(userController.removeUser)
101   .put(userController.updateUser);
102
103 router.route('/api/beers')
104   .get(beerController.getBeers)
105   .post(beerController.postBeer);
```

The end result is an API with easy-to-consume routes for future development. In iteration 3, we will use a combination of these routes to retrieve all beers and then filter results. The filtering will occur in our AngularJS project.

We have begun using factories for REST requests so they can easily be reused.

7.5 Testing / Implementation /Deployment

Unfortunately, we were unable to move beyond unit testing in our current increment as we ran into issues coordinating the various services which really came into play as we began to integrate into a single application.

7.6 Project Management

To manage the project, we are using the Kanban tool. Additionally, we have been very relaxed in designating work. Going forward however, we are going to have daily SCRUM meetings via Google Hangouts. This is to enforce more accountability and to increase productivity.

<http://bcap68.kanbantool.com/b/176306>

7.6.1 Work Completed

Person	Description	Time Invested (hours)
Brandon	Google Maps/BreweryDB Mashup	35+
Jordan	NodeJS/ExpressJS API Server Implementation	30
	MongoLab REST Implementation Using APIServer	15
	BreweryDB Data Retrieval and Initial AngularJS Views	30
Matt	Created Login and Registration Views	5
	Researched Implementation of Robust Login System	20
Todd	Miscellaneous Administrative Work	10

Some tasks were completed by group members that ultimately did not provide any impactful value to the project and have not been listed in the table above. For example, three of the team members worked on trying to get authentication/authorization working to an extent that allows integration of Passport API and OAuth 2.0 authentication. However, that was an overly ambitious goal and has now been backlogged.

7.6.2 Future Work

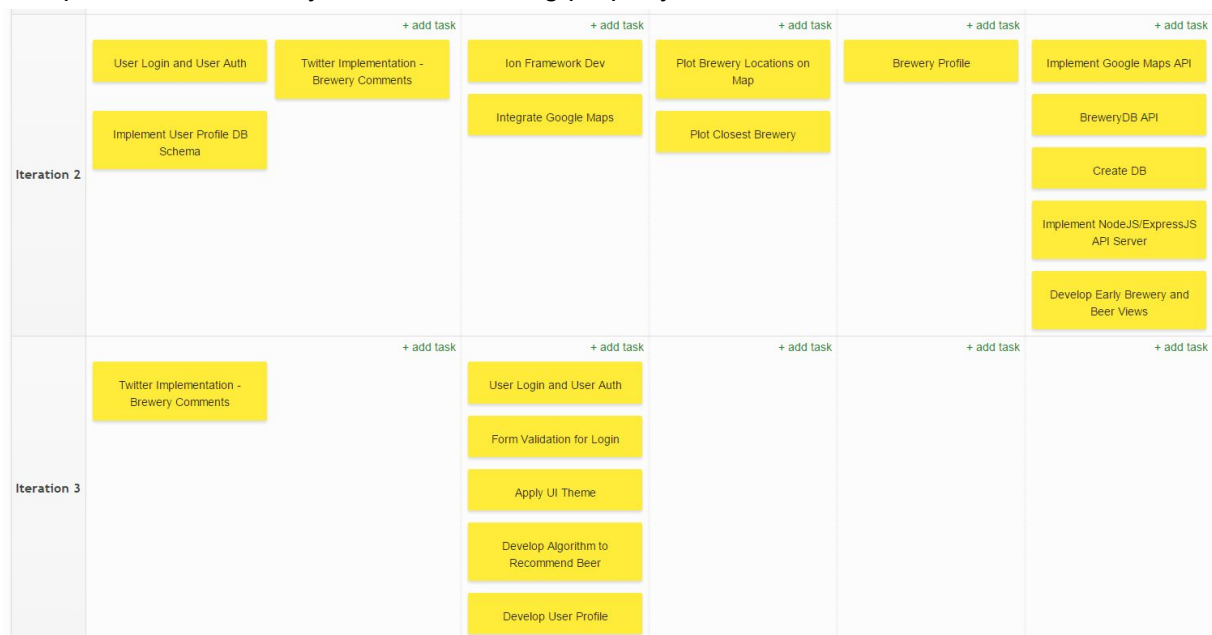
The end result of increment two is a server API with easy-to-consume routes for future development. We have begun working on implementation of the front end and we do have some plans established for what the business logic will be like in determining a beer for a user based on their interests.

In iteration 3, we will use a combination of these routes to retrieve all beers and then filter results. The filtering will occur in our AngularJS project.

Person	Description	Time Invested (estimated hours)
Brandon	Login/User Account Control/Form Validation	20
Jordan	Finish Full Implementation of Node/Express API Server	20
	Develop a Working Algorithm for Finding an Optimal Beer Based on User Preferences	15
Matt	UI Enhancements	20
Todd	Custom CSS and Database Updates	20

Issues/Concerns

One of the issues that we are having is integrating the Google Maps and BreweryDB mashup to the project. We started the project with the Ionic “sidemenu” template. This template utilizes URL routers, and for a reason unbeknown to us, the Google Maps REST API refuses to render a map albeit the BreweryDB data is loading properly.



Shown above is our Kanban overview for iteration 2 and 3.

7.7. Deployment

Project Management Link:

<http://bcap68.kanbantool.com/b/176306>

Since the majority of our work was on individual modules, we were unable to complete a fully realized deployment in iteration 2. As we have moved into iteration 3, the unification of our application has been a key focus as we bring the parts together into a working unit.

<https://github.com/brandrews722/BeerMeKC-Version2>

9. Bibliography

<http://www.joshmorony.com/category/ionic-tutorials/>

<http://ngcordova.com/docs/plugins/>

<http://www.bigripbrewing.com/>

<http://www.facebook.com/>

<http://www.brewerydb.com/developers/docs>

<https://developers.google.com/maps/?hl=en>

<https://kanbantool.com>