

CLOCKWORK

Cullin Lam, Andry Lora, Son Nguyen



JANUARY 28, 2016

VERSION 1.0

Table of Contents

Project Goals.....	3
Project Features.....	3
Alpha	3
<i>Event Feed</i>	3
<i>Event Detail</i>	4
<i>Event Add</i>	4
<i>Sign-In Account System</i>	5
Beta	6
<i>Event Timer</i>	6
<i>Event Feed Infinite Scrolling</i>	6
<i>User Profile View</i>	7
Final Release.....	7
Software Components.....	8
Meteor.js.....	8
Angular.js	8
Ionic.....	8
Additional Components	8
<i>Meteor: accounts-password package</i>	8
<i>Ionic: infinite-scrolling</i>	8
<i>Components to Develop</i>	9
Developer Resources.....	9
Possible Issues	9
User Demographic	9
Project Time Line	9
Statement of Acceptability	10

Project Goals

ClockWork is a hybrid mobile application that seeks to stream line the planning it takes to hang out with your friends. ClockWork provides its users the ability to quickly publish and join events. It removes the hassle of having to call, text, or message your friends to find who is available to hang out.

Project Features

Alpha

The features required for Alpha release are as follows:

- A. Event Feed (Public Access)
- B. Event Detail (Public Access)
- C. Event Add
- D. Sign in Account System

Event Feed

The Event Feed view shall serve as the main page of the ClockWork Application. Here, users will be able to view published events. In addition, users will be able to navigate to additional views, such as the Event Detail and Event Add views.

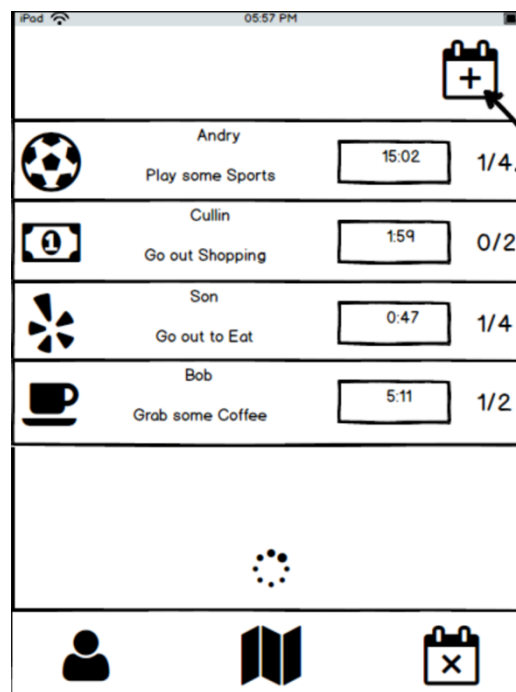


Figure A. Event Feed View

Event Detail

The Event Detail view shall provide users with more information about published events on the Event Feed view. An expanded description of the event shall be displayed. Also a list of users attending shall appear. Lastly the attend button shall be placed here so that those interested may notify the event owner that they will join.

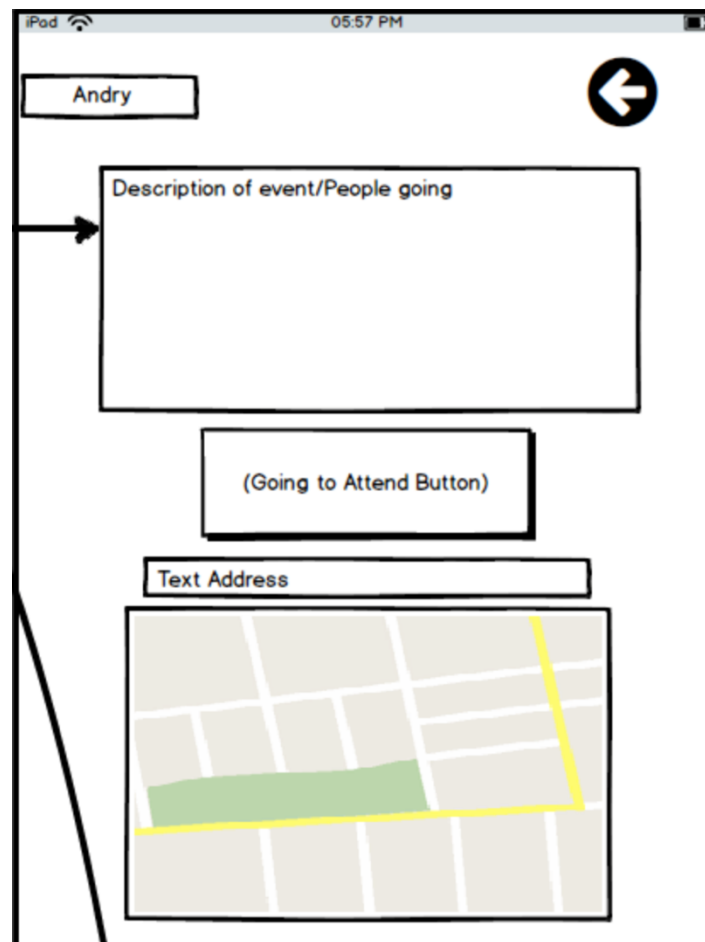


Figure B. Event Detail View

Event Add

The Event Add view shall display to the user a form that will be used to create a new event. A drop down selection for event type i.e. food, game, movie, sport, etc. will constitute one of the fields. In addition, the user will be required to provide a description of the event. We have decided to allow the address field to be optional, as it will allow flexibility for interested users to agree on an appropriate location for the event. Lastly event expiration time is a required field. This field sets the duration of how long the event will remain active on the event feed. After the allotted time the event will be closed to new attendees, and will be removed from the

feed. The event owner and the attendees will receive a notification that the event is now closed and the event members should begin communicating on event specifics or begin their event.

iPad 05:57 PM

Pick Event Type ▼

- Out to eat
- Play Some sports
- Go out Shopping
- Get some Coffee

Write Description here

Enter Address Here


Insert Time limit

CONFIRM EVENT AND POST

Figure C. Event Add View

Sign-In Account System

This application shall have a user account system that stores user information such as login email and encrypted password. In addition, profile information shall also be stored. Users will be required to login in order to use this application. A login in view shall be displayed if the user is not signed in. From this view the user may also sign up for an account.



Log in to your account

[Forgot your password?](#)

LOG IN

[Create an account](#)

Create your account

Get up and running with Ionic in no time.

By signing up you agree to the Ionic.io [Terms of Service](#).

SIGN UP

Already have an account? [Log in](#).

Figure D. Login View

img src: <https://apps.ionic.io/>

Figure E. Sign Up View

Beta

The features required for Beta release are as follows:

- I. Event Timer
- II. Event Feed Infinite Scrolling
- III. User Profile View
- IV. Ability to Add friends

Event Timer

We hope to implement a live timer on the Event Feed View for each event. Similar to an auction this timer will count down the time remaining for users to join an event before it is closed. Server code shall check active events and close any expired events (See Figure A).

Event Feed Infinite Scrolling

Once the number of active events becomes quite large, it will be impractical to force the client to load all the events in the event feed view. Instead we shall implement infinite scrolling and have the client fetch additional events as the user scrolls down.

User Profile View

We hope to implement a public user profile view for each user, so that other users may be able to easily identify and recognize each other. Users will be able to share a short bio or description about themselves.

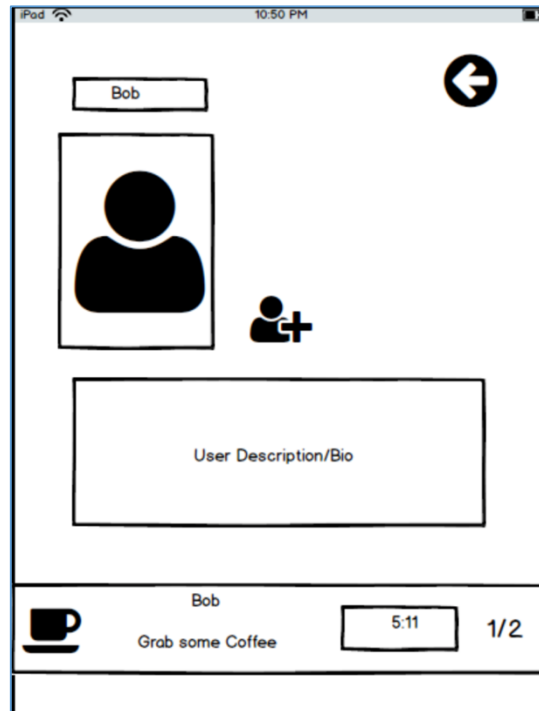


Figure F. User Profile View with add friend button

Add Friends

We hope to implement an add friend feature. Adding friends will allow the user to see their friends' events on the Event Feed view. This feature will necessitate a notification system to approve friend requests. The alpha build will allow any user to see all the Events on the Event Feed.

Final Release

Additional features required for a full final release are as follows:

- I. Ability to leave event
- II. Symbols
- III. Event Sorting
- IV. Event Feed Search Bar
- V. Friend List View
- VI. Flaky User Rating
- VII. In app Messaging
- VIII. OAuth login

These features will most likely not be implemented during the course of this semester project.

Software Components

Meteor.js

Meteor is the framework we will be using to develop this application. In relation to the MEAN Stack, Meteor includes or supports the same components such as Node.js, Mongo DB, and Angular.js. The key difference between Meteor and MEAN is the use of an integrated approach to accessing the database rather than building and using a RESTful API layer. Meteor includes its own API for specifying execution of client or server code. Both Node.js and Mongo DB are integrated into the Meteor environment, therefore launching Meteor kicks off both together. Lastly Meteor also includes Apache Cordova, a tool which allows the development of native mobile applications using HTML, CSS, and JavaScript.

Angular.js

Angular.js is the framework we will be using for developing our application's frontend which entails views, view controllers, and data models. Although Angular 2.0 is currently in beta release, we shall be developing in the latest release of Angular 1. We anticipate using angular packages in our application, and thus using Angular 1.0 will ensure that these packages will be compatible with our application code.

Ionic

Ionic is a frontend framework that extends Angular with mobile UI components. The components found in the Ionic framework will allow us to create an appealing and functional user interface quickly.

Additional Components

Meteor: accounts-password package

The accounts-password package is a Meteor package that provides to us a Mongo collection for our users. The Mongo collection stores usernames, emails, encrypted passwords, and profile information. In addition, it provides methods for logging users in and out of the application, password recovery, and reactive state variables that our application can use to identify the user.

Ionic: infinite-scrolling

Ionic provides a directive called ion-infinite-scroll that allows us to specify actions to take when the user reaches the bottom of a view by scrolling. For example, ion-infinite-scroll has an attribute called on-infinite which we can use to call a function from our view controller to load more data in our Event Feed view.

Components to Develop

Thanks to the Meteor framework and the other available components, we anticipate the majority of our development time spent creating MVC's using Angular. In order to achieve our project goals, we will be responsible for creating the application views stated in the Project Features section. In addition, we will also be responsible for building appropriate database schemas for storing anticipated data. The Event Timer feature will also have to be developed. Our solution is to store the event expiration date time in our Mongo collection which the client will use to display a timer until the expiration time. This method will avoid the client having to read and write constantly to our backend Mongo collection.

Developer Resources

The following resources have been identified in helping us develop this application:

- ❖ <http://www.angular-meteor.com/tutorials/socially/angular1/bootstrapping>
- ❖ <https://www.meteor.com/tutorials/blaze/creating-an-app>
- ❖ <http://docs.meteor.com/#/full/>
- ❖ <http://ionicframework.com/docs/>
- ❖ <https://www.codeschool.com/courses/shaping-up-with-angular-js>

Possible Issues

Security

User Testing – i.e. cross platform testing

User Demographic

Busy People young

Project Time Line

A tentative schedule is shown below:

DATE	ITEM	Responsibility
[Feb-02-2016]	Set Up Sub Directories in Git	[Cullin/Andry/Son]
[Feb-10-2016]	Web Template of UI	[Cullin/Andry/Son]
[Feb-12-2016]	Research Unlimited Scrolling	[Cullin]
[Feb-14-2016]	Research Image Upload	[Andry]
[Feb-15-2016]	Implement Sign In	[Son]
[Feb-16-2016]	Implement Event List	[Cullin]
[Feb-20-2016]	Alpha Testing	[Cullin/Andry/Son]
[Mar-11-2016]	Implement Event Detail	[Cullin/Andry/Son]
[Mar-12-2016]	Implement User Profile	[Andry]
[Mar-14-2016]	Timer Function	[Cullin/Andry/Son]
[Mar-15-2016]	Google Map API Location of Event	[Cullin/Andry/Son]
[Mar-20-2016]	Test Cross Platform	[Cullin/Andry/Son]
[Mar-21-2016]	Beta Testing	[Cullin/Andry/Son]
[Apr-10-2016]	Completion	[Cullin/Andry/Son]

Statement of Acceptability

In terms of this course, this project shall be considered acceptable if all Alpha and Beta features are implemented. Once development has reached Beta, events found on the Event Feed shall no longer be public to all users, instead users may only see events published by their friends.

If time permits, we would like to implement features listed for the Final release version. Any additional features that require the creation of new views will call for sensible view path routing, and manipulations to navigation bars and tabs.