Wager App

Product Guide

Written by: Tyler Kaye, William Chance, Richard Bush, Michael Swart

Product Guide for Wager:

Purpose:

Wager is a social gambling application based around the fact that people want to make friendly wagers on a variety of different things but there isn't a formal way to handle them. Wager gives our users this functionality and much more. Using wager, users can open bets to the general public or browse through bets based on who they are friends with or even their geolocation.

1. Logging In / Signing Up:
    a. If you want to Log In: please type in your email and password associated with your account
    b. If you would like to Sign Up please click the sign up button:
        i. Fill out the form elements and select and crop a photo of yourself
            1. Make sure that your Venmo username is correct otherwise you will not be paid
        ii. On submit the application will let you know if any of your form elements are not ok (username must be unique, password cannot be password, and elements must be a certain length to make sure that they are real)

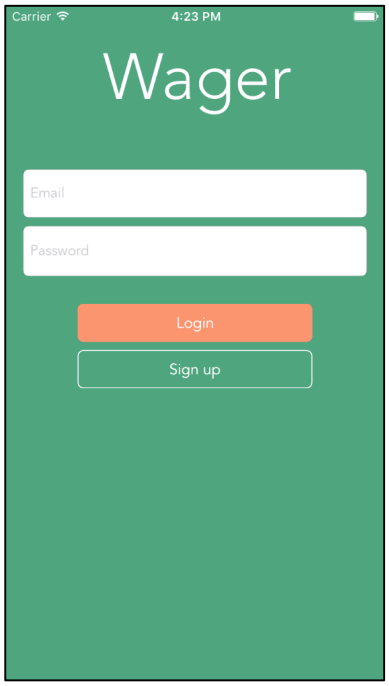


Figure 1: Sign In Page

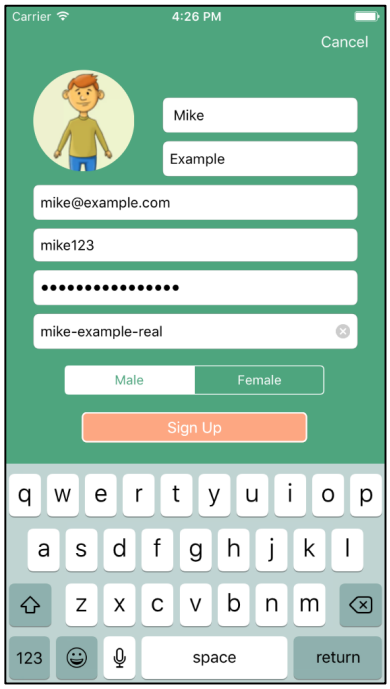


Figure 2: Sign Up Form

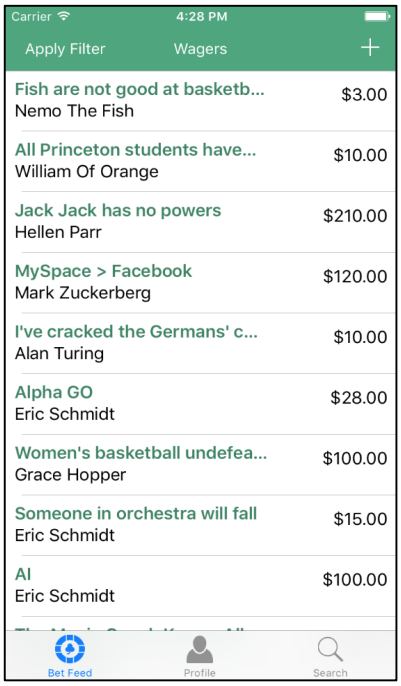


Figure 3: Wager List Page

2. Viewing Profile and Editing Profile:
    a. In order to view your profile, click on the profile icon in the Tab Bar on the bottom
    b. Your profile contains your information, betting PNL (profit / loss) and rating
    c. Additionally, it contains all of your bets.
    d. If you are just signing up, then you will have no current bets, but that will soon change!
    e. To edit your profile, click the edit button in the top right corner of the screen
    f. To sign out, click the sign out button in the upper left corner of the screen

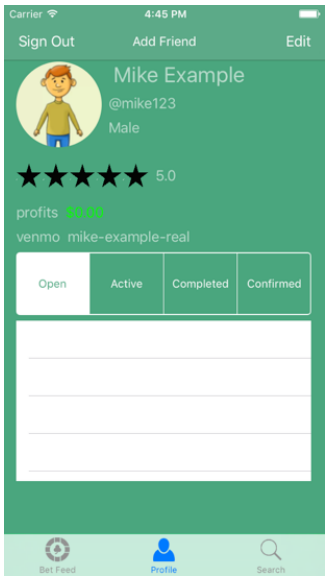


Figure 4: Profile Page

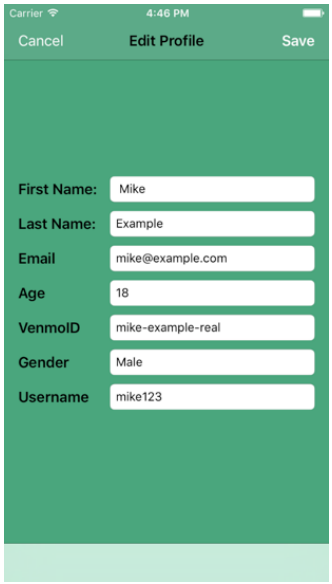
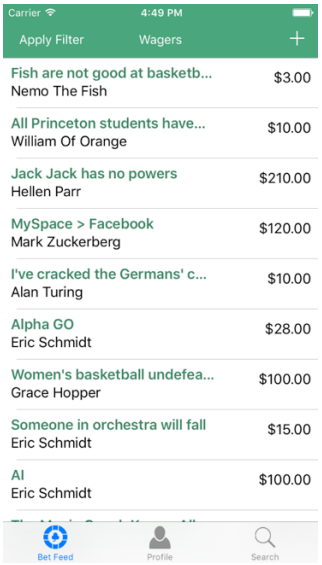
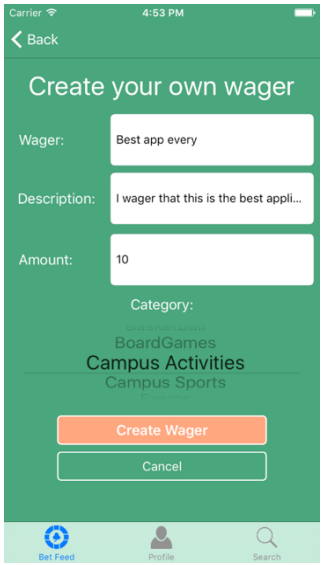


Figure 5: Edit Profile Page

3. Viewing Bets, Creating Bets, and Applying Filters:
    a. To go to the bet feed, click on the Wager icon at the bottom of the screen
    b. To Add a new bet, click the plus sign in the upper-right hand corner and fill out the form
        i. Wager Name: title of the wager, try to make it concise
        ii. Wager Description: longer description of the wager. Place details here
        iii. Wager Amount: how much the wager is for
        iv. Category: what type of bet is it? Select a category from the options or select "Other"
    c. To set the filters for the page go back to the Bet List page and flick the filters button
        i. There are several filters that one can use to dynamically change the bets in the view
        ii. Here are the filters available:
            1. Friends: view only your friends' wagers or all possible bets
            2. Wager Status: select wagers of a specific type as described below
            3. Wager Category: select only wagers of a specific category
            4. Near You: view only wagers within a specific radius of you
                a. Select this option and a picker will pop up for you to select a radius
        iii. Hit the "Set Filter" button to set the filters and go back to the Bet List View
        iv. Hit the "Set Default Filters" to set the default filters t
            1. These will automatically populate then Bet List View on opening the application





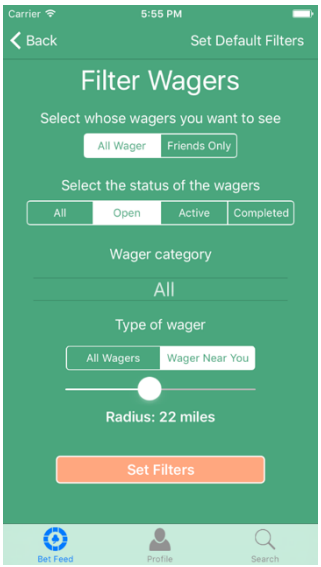
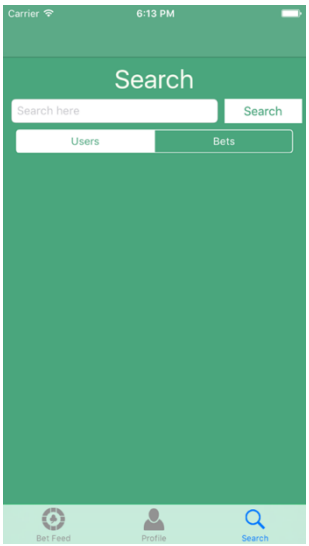
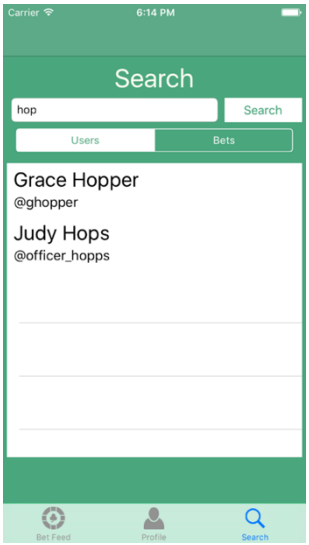


Figure 6: Bet List View Page Figure 7: Add New Wager Figure 8: Filter Selection Page

4. Bet Lifecycle: Bets are organized into 4 categories:
    a. Open: a user has posed the bet, but no one else has taken the other side yet
    b. Active: wagers in which both sides have been taken
    c. Completed: the result has been determined and is awaiting payment
    d. Confirmed: the wager has been fully completed and payment was successful

5. Use the Search Functionality:
    a. Click on the search button to go to the search page
    b. Use the toggle to search for either bets or individual profiles
    c. Things you can search for in a profile: First Name, Last Name, Username
    d. Things you can search for in a bet: Bet Name, Bet Description
    e. The search uses regular expressions and will find a field given any part of it
    f. Can click on any entry to be brought to the respective bet page or profile page





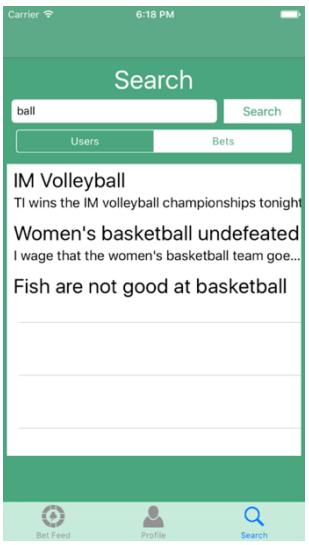


Figure 9: The Search Page Figure 10: Search for Users Figure 11: Searching for Wagers

6. Look At Other Wagers and Go To Other Profiles
    a. In the search, profile, or bet page you can click on a wager to be brought to its page
    b. The Wager Page:
        i. On the wager page we can view the wager and the people participating
        ii. Click on the pictures of those involved in the wager to be brought to their page
        iii. The button on the bottom and description below will be explained later in this manual

Figure 12: Individual Wager Page

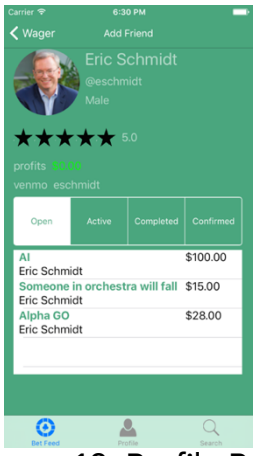


Figure 13: Profile Page

7. Accepting Another Wager and Going Until Completion:
    a. Set the filters to display all "Open" or "Active" Wagers then click "Set Filters"
    b. Click on a Wager in the Wager List View and Examine the amount / description
    c. Click on "Take Bet" to accept the Wager → your profile image will dynamically appear
    d. Once the Wager is done, click on "Complete Bet" to say who won → Click who won
    e. If you lose → click "Pay Up" and select payment type
        i. If you select "Venmo" it will bring you to the application with the fields filled out
    f. If you win → can let the application know whether the loser paid up

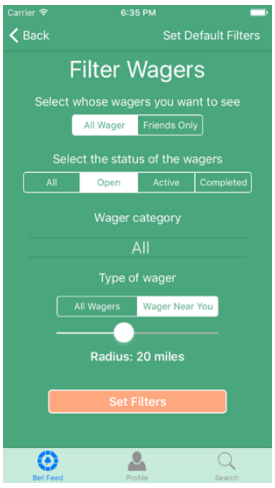


Figure 14: Step A

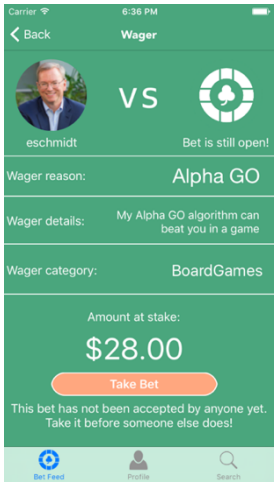


Figure 15: Step B

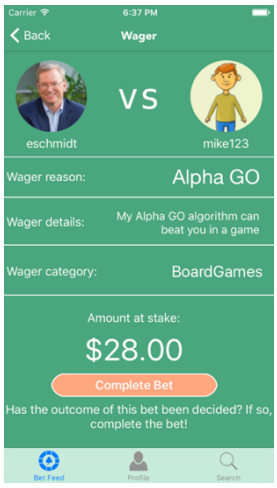


Figure 16: Step C

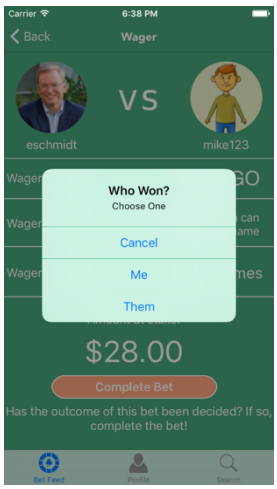


Figure 17: Step D

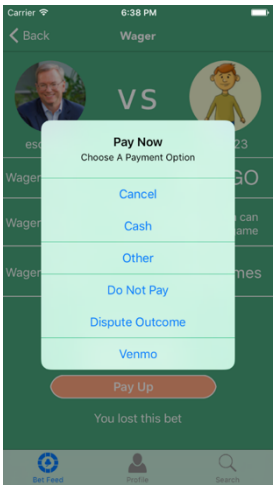


Figure 18: Step E

Developer Guide:

Summary:

- Built in IOS with Swift 3.
- Uses Firebase for storing data and authentication
- Utilizes GeoFire for location-based querying within firebase
- Built with Cocoa Pods (must use .xcworkspace file to build)

Basic View Controller Architecture:

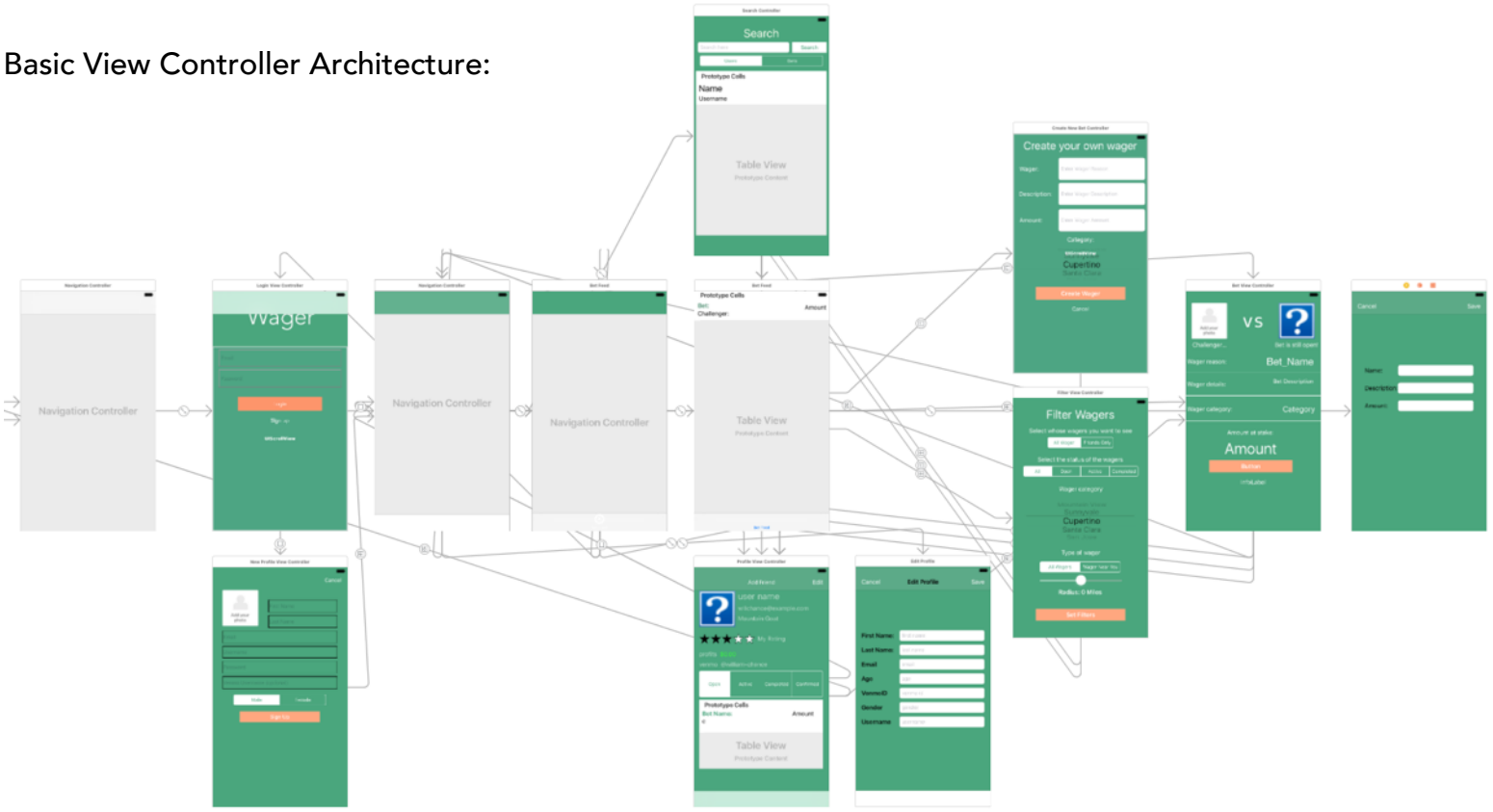


Figure 19: Basic View Controller Architecture

All View Controllers:

1. Login ViewController:
    a. This controller contains the login screen. Furthermore, the page contains a state-listener for the Firebase Authentication.
2. New Profile ViewController:
    a. This is the sign-up controller. Each of the fields has an Error Handler to ensure that the input is correct and valid. Additionally, this file contains the code for image upload and storage into Firebase. Finally, when the “Sign Up” button is clicked, the error handlers makes sure that the inputs are valid before creating a Profile object and putting it in the Firebase Database. Then the page segues to the Bet List TableViewController.

3. Bet List TableViewController:
    a. This is the main page of the application. Thus this section will be broken up into several sections based on functionality.
        i. Basic TableView Controller:
            1. The delegate methods for the TableView are handled in this file. A list of Bet items are held and the TableView populates itself using this list.
        ii. reloadRows()
            1. This function contains the code to populate the items[] list of Bet Items. First the method clears items[] so that there is nothing in the list. Then, if the filter is a geo-location based query, a GeoFire reference is made to the "bet_locations" child of the database. GeoFire.query() queries the database for all bets within a specific radius of the current location that is set elsewhere.
            2. Then, each entry is run through the checkFriends() and checkBet().
        iii. checkFriends()
            1. This method is called when the self.friendsOnly variable is activated. It queries the "Friends" section of the database to make sure that each bet is a child of the current user's section of the "Friends" database.
        iv. checkBets()
            1. This method makes sure that only bets of the proper type (Open, Active, …) are appended to the list.
        v. Default Filters ;
            1. In the restOfViewDidLoad() function, the user defaults on the IOS device is utilized. If user defaults are present, then these will be loaded each time the application opens up. If they are not present, then they default values for the application are at the top of the file.
        vi. Segues:
            1. This ViewController has a button and segue to go to (a) each individual bet and (b) the filter page.
4. Filter ViewController:
    a. This ViewController controls the filters page in which the user can select which filters to apply to the Bet List TableView. This page can only be reached from the list view and it can only segue to it as well. There are two buttons on the page. "Apply Filters" segues to the Bet List TableView Controller and in the process sets the values for the variables in the controller that control the filters. The "Set Default Filters" button first sets the NSUserDeafults in the IOS device to the currently selected filters (a dictionary is stored). Then this button acts identically to the "Apply Filters" button by segueing to the list controller and setting the filter values.

5. Bet ViewController:
    a. This controller is where and individual bet is shown. Furthermore, this controller contains most of the logic for the bet cycle and enables the user to progress along this lifecycle by using AlertViews. Additionally, the Vemmo integration is in this page. This step is executed by creating a dynamic URL that will open up the Venmo application if it is on the phone and the detail of the payment and username of the recipient are already filled out.
6. Create New Bet ViewController:
    a. This controller is where a new bet is created. The code contains several error handling mechanisms to ensure that only valid data can be used for the creation of a Bet Item. If the data in the form is valid, then a Bet Item Object is created and put into Firebase. Finally, GeoFire is used to tag the bet at its current location.
7. Edit Bet ViewController:
    a. This controller enables a user to edit a current bet (but it can only be reached if the current user is the bet's challenger and the bet is still open).
8. Profile ViewController:
    a. This controller shows the profile view. If this controller has been passed a profile object in the prepare() method of a previous controller, then it shows that profile, otherwise it uses the current user's profile which is being stored in App Delegate. Finally, this controller also has a TableView and the delegate methods are placed in this file.
9. Profile Editor ViewController:
    a. This controller contains the code necessary for a user to change his/her current profile. It will only update the database if the value of a field has changed.
10. Search Controller:
    a. This controller contains the search functionality of the application. A user can either search by Profile or Bet Item depending on the value of the toggle switch. On successful search, a table view will populate with the possible results. Each on of these TableView Cells is clickable and will navigate the user to the respective Bet/Profile page.

Other Important Files:

1. App Delegate:
    a. This file contains the code that gets the users current location and stores it in the file which is accessible to every ViewController. Additionally, this file stores the Profile object for the current user.
2. Main.Storyboard:
    a. This file contains the layout for the entire application (Figure 19)

Database Structure and Fields:

- Detailed below in Figure 20

User	
UserID	String
Email	String

Profile	
UserID	String
FirstName	String
LastName	String
Email	String
PNL	Float
Bets	[BetID]
Picture	Image
Age	Int
Rating	Float
Num_Ratings	Int
VenmoID	String
Gender	"Male" or "Female"

Bet	
Name	String
Description	String
Amount	Int
Challenger	ProfileID
	Name
Challengee	ProfileID
	Name
Category	String or [String]
Completed	Boolean
DateOpened	Date
DateClosed	Date
FInished	Boolean
Winner	Boolean
Arbitration	Boolean
Accepted	Boolean
Rating	Double
Accepted	Boolean
Completed	Boolean
Confirmed	Boolean

Categories
[String]

Friends	
Key	ProfileUID
Challenger	ProfileID
	Name
Challengee	ProfileID
	Name

Usernames	
Key	UserName
Value	Boolean

Bet_Locations	
G	String
L0	Float
L1	Float

Figure 20: Detailed Descriptions of the Database and its Fields