2016

Project Proposal: PlayList

XAVIER CRAFT

10/7/16

Hanover College, Computer Science

Introduction

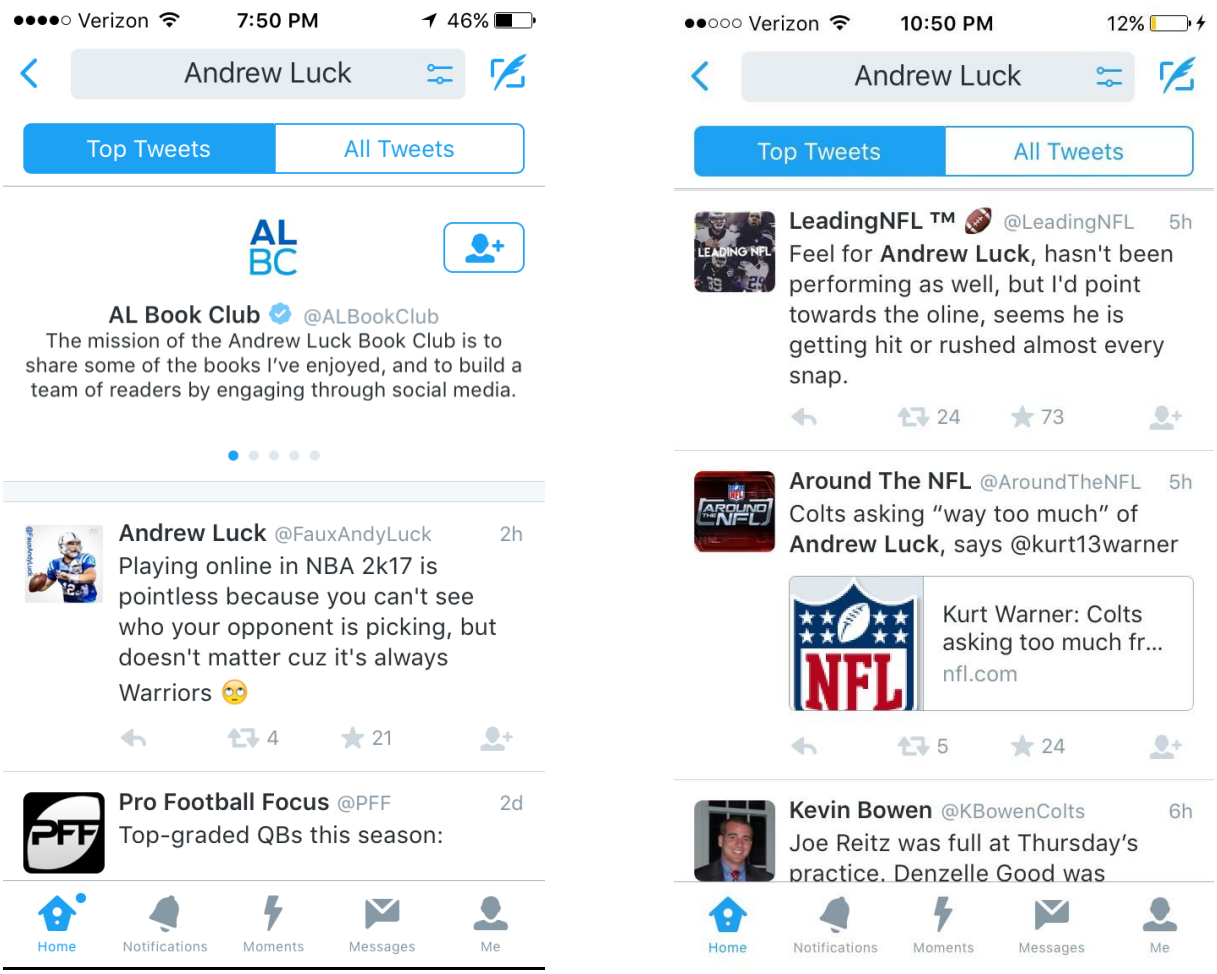
The plan for my senior project is to develop an iOS application called PlayList. PlayList is an application that will work with Twitter to curate relevant sports-related tweets based on an individual user's preferences. Users will be able to login directly with their Twitter account's username and password. I envision this application as a way to quickly scroll through relevant sports updates for all of a user's favorite sports, teams, and athletes. For example, I would love to be able to open up PlayList on a Sunday morning during the NFL season and be able to quickly navigate to all of the relevant football updates for that day.

I am creating this application because I feel that other applications for receiving sports updates have too much going on and are hard to navigate. ESPN, CBS Sports, and Team Stream are all examples of other sports update applications currently in app stores. They have a lot of great information but it takes quite a few clicks to navigate through it all. None of these applications offer a feed of summarized and automated updates in 140 characters or less.

Although twitter offers many of the same updates that my application will, I feel that PlayList is the better option because it has a more focused purpose than Twitter. Twitter is a social media platform where you can find a post of song lyrics, a link to a news article, and a funny video all on the same feed while my application will focus solely on sports. If a user were to go to Twitter and run a search for Andrew Luck (the quarterback for the Indianapolis Colts football team) he or she would find all recent tweets from any user that mention his name. This could be just a random user trash talking him, a parody account pretending to be him, or a legitimate

update. My application is for users that don't want to have to scroll past every random tweet on Twitter in order to get to the relevant sports updates they want.

Figure 1

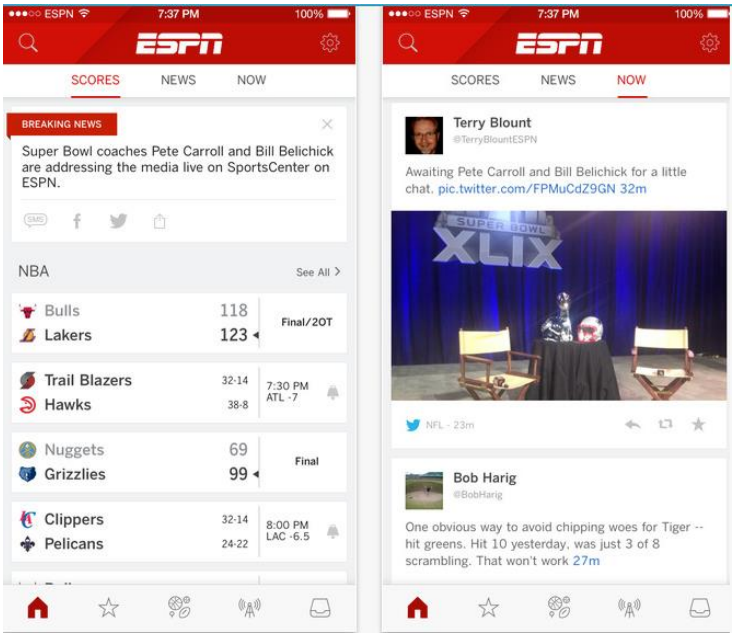


This application will be for iOS devices because I want to be able to use this application on my own smart phone, and I have wanted to have this sort of functionality on my phone this for quite some time.

Background

ESPN, CBS Sports, and Team Stream are all great mobile applications out on the market today, but I feel that they don't bring a level of simplicity that the general user wants. The ESPN application is probably the most popular sports-update applications currently out there. It has three main tabs. The Scores tab, the News tab, and the Now tab. The Scores tab shows all of the scores for games currently being played. This is a nice feature, and it is quick to scroll through and find specific scores. The news tab has articles and videos about current events in the sports world. This is a nice feature as well but it is time consuming to have to watch videos or read articles to get the information you want. The Now tab shows all of the most recent articles that have been posted.

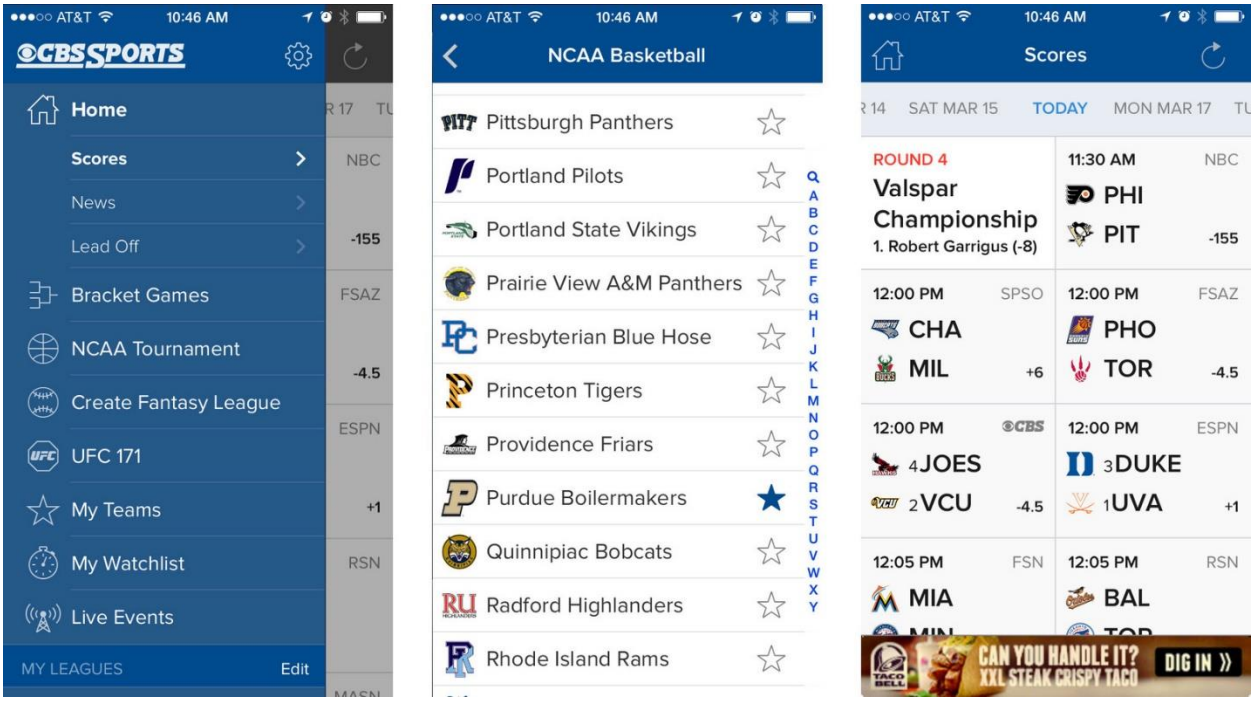
Figure 2



ESPN Application

The CBS Sports application is very similar to the ESPN application. It focuses mostly on the scores of each game happening that week in each respective sport. It also has fantasy sports built into the application. The option of having fantasy leagues is a great idea and definitely increases downloads, but it also causes problems with the servers during big games when a lot of people are using the app at the same time.

Figure 3



CBS Sports Application

Team Stream is also a great application, and it is probably the second best sports-update application out there. You can follow your favorite sports and teams in the app to get a customized list of articles and videos to scroll through and view. This application also features a

scores tab that tracks the scores of all your favorite teams each week. Once again this is a nice application, but it is time consuming to have to watch videos or read articles to get the information you want.

Figure 4

Team Stream Application

I believe my application will have a much better user experience for sports fans desiring quick, focused updates on their favorite sports, teams, and players than other similar applications on the market.

Project Details

My senior project will consist of three main parts; the scraper function that will retrieve tweets based on a user's preferences using Twitter's API, a filter function for removing the junk tweets that unavoidably come along with the relevant ones, and the graphical user interface. These three parts will work together seamlessly to bring you the best sports updates whenever you need them.

The first piece to the puzzle is the scraper function. I am going to need a function in my application that is capable of retrieving tweets based on a user's preferences that will be collected and stored locally on their phone. I imagine that it will take an array of strings in order to get all of a user's preferred updates at the same time. The scraper function will use the Twitter API to accomplish this. It will allow me to pull all relevant sports updates from whatever sources I choose. Twitter graciously gave the public two types of access to their API, the REST API and the Streaming API. The REST API gives us programmatic access to the ability to read and write Twitter data. The Streaming API is used to collect large amounts of data in real time. In order for PlayList to work I am going to need to use both of these APIs, but much more so the Streaming API. I plan on using the REST API when the application first starts in order to let a user sign into their personal account and then the Streaming API from then on to make sure the application's feed stays up-to-date.

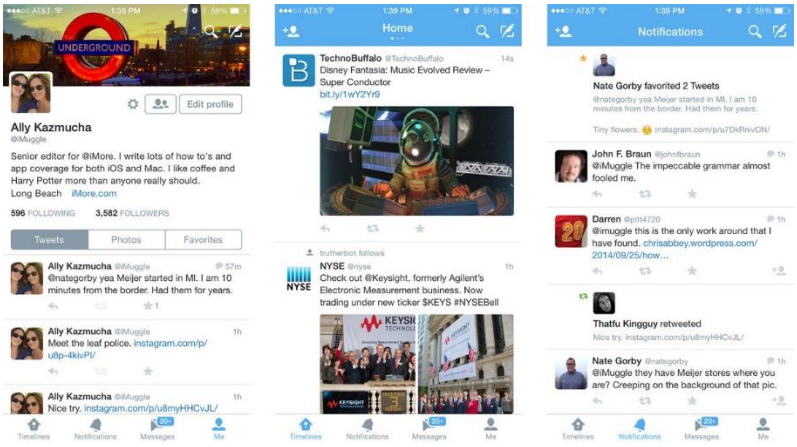
The filter function is the next part of my senior project. The filter function will be used to sort through the large bulk of tweets that the scraper function will be retrieving. The scraper function will be able to grab relevant tweets on its own but some junk tweets will inevitably come

along with them (refer to images on page 2). This function's purpose is to remove those junk tweets before they make it to a user's feed. The remaining relevant sports tweets will then be sent to the GUI to be displayed for the user.

The final part of my application will be the graphical user interface. This is the most important part from a user's perspective because it will be the only way they can see or use any of the data that the application will be pulling from Twitter. It is also how the users will set their preferences. It's important to have a pleasing GUI so that users are not annoyed by an ugly interface. In order to do this, I plan on modelling my application's layout based on the standard Twitter interface. It is visually pleasing and easy to use, which ensures that you are not losing users just because they find your GUI to be confusing.

I plan on having two main tabs in my application, the feed tab and the user tab. The main tab in my application, the feed tab, is where a user's sports-update tweets will be displayed. I want that feed to update around every 30 seconds so that a user can get real-time updates. Below are a few images of what Twitter's feed looks like and is what I will base my GUI off of.

Figure 5



Twitter Application

The second tab in my application is the user tab. This where users will set and update their sports-feed preferences. Users will be able to specify which particular sports, teams, and players they want to get updates about on their feed. A user will also be able to access the settings for the app from this tab. This will include managing Twitter login information and notification settings.

Before I could begin working on this project I needed to get up to speed on some of the technologies that I would be using. I have started learning Swift, the new programming language for iOS applications and built some rudimentary programs in order to get a better idea of how I am going to build PlayList. I have also worked through a lab from Professor Skiadas' data wrangling class in order to get a better understanding of accessing the Twitter API and how I will be able to do so in Swift. Soon I will begin learning how Swift interacts with the Twitter API and then start building and testing my scraper function.

While my PlayList application is currently simple in design and proposed features, I hope that after implementing these essential pieces I will be able to add more features, like the ability to post to your sports feed or adding a scores tab. I think that this application will have a great user experience and will be very useful. I hope that I can put this app on my own device and actively use it for years to come and possibly one day deploy it to the Apple store for general use.

Project Timeline

Week 1: Research iOS applications that use the Twitter API and find out how they do so in the Swift language. Begin trying to access and retrieve tweets myself.

Week 2: Start building and testing my first prototype of the scraper function.

Week 3: Finish scraper function.

Week 4: Test how to filter the group of tweets my scraper function will retrieve with each query to the Twitter API.

Week 5: Develop filter function capable of effectively filtering out all irrelevant updates.

Week 6: Figure out what kind of layout/design I want to use in PlayList.

Week 7: Start coding the GUI and displaying relevant tweets output by the filter function.

Week 8: Add the User tab and start implementing the login feature using twitter credentials.

Week 9: Implement preferences feature.

Week 10: Polish application and make sure that all essential features have been thoroughly tested and work together seamlessly.

Proposed Grading Rubric

	Scraper Function	Filter Function	GUI
A	Function effectively retrieves all relevant sports tweets for a user's preferences and minimalizes irrelevant ones	Function effectively removes all irrelevant updates retrieved by the scraper function	Seamless GUI with smooth functionality and pleasing design. Also easy to navigate
B	Function effectively retrieves all relevant sports tweets for a user's preferences along with some irrelevant ones	Function removes most irrelevant updates but sometimes slightly irrelevant or unwanted tweets slip through	GUI has pleasing design but some flaws. Still easy to navigate
C	Function retrieves a majority of relevant sports tweets but misses some along with some irrelevant ones	Function does an effective, but not great job of removing all unwanted tweets	GUI very average looking and not very easy to navigate
D	Function retrieves around half of relevant sports tweets along with quite a few irrelevant ones	Filter still lets through a majority of unwanted tweets from the scraper function	Functionality is limited and displays incorrectly regularly. Not easy to navigate
F	Function retrieves more irrelevant tweets than relevant ones	Does not filter out unwanted updates	GUI is completely unusable

References

Corpuz, John (2016). *15 Best Sports Apps.* Toms Guide. Retrieved November 28th, 2016, from http://www.tomsguide.com/us/pictures-story/599-best-sports-apps.html#s4

Twitter Documentation (2016). Retrieved October 1st, 2016, from https://dev.twitter.com/overview/documentation

Swift Documentation (2015). Retrieved October 1st, 2016, from http://nshipster.com/swift-documentation/