2014

# Marketing with Mobile Push Notifications in a Location Specific Context

Nathan Taylor

*Grand Valley State University*

# Marketing with Mobile Push Notifications in a Location Specific Context

By
Nate Taylor
August 5, 2014

# Marketing with Mobile Push Notifications in a Location Specific Context

By
Nate Taylor

A project submitted in partial fulfillment of the requirements for the degree of

Master of Science in

Computer Information Systems

at

Grand Valley State University

August 5, 2014

---

**Dr. Jonathan Engelsma** **Date**

## Table of Contents

# Abstract

Businesses have employed different tactics over the years to market directly to individual consumers. These range from blind mailing flyers to direct targeted Internet advertisements. Many forms of marketing are effective, but none of them incorporate contextual information based on a person's current location in real time. Apple's iBeacon technology and the Bluetooth LE protocol make this easier than ever. Beacons can be placed in physical locations throughout a store and emit pulses that mobile applications can subscribe to. In this project, a highly scalable marketing system using iBeacon and the iOS platform was built to send push notifications to users based on their proximity to the University Book Store. Content was provided to people to draw them into the bookstore for the purpose of increasing foot traffic and ultimately increasing sales. Analytics were captured to answer questions about how many people walk through the building housing the book store, how many users received notifications and how many acted on them.


# Introduction

Businesses have been providing promotional materials in many forms throughout the years such as standard/electronic mail, brochures and coupon booklets. These forms of advertising can be effective but cost lots of money and use lots of paper. They also lack any real sense of locational context when it comes to the recipient. What if there was a way to provide coupons or information to users based on their current location in real time? Apple's iBeacon technology that utilizes the Bluetooth LE protocol is a perfect fit for solving this problem. Where GPS fails to provide location context indoors, iBeacon does not. [1]

In order to understand some general uses of beacons, let's step outside the retail space. Imagine a large art show similar to Art Prize in Grand Rapids, Michigan. Different forms of art work are scattered throughout the downtown area. Visitors casually stroll across town viewing the different fixtures. At some stops, artists are available to discuss their inspiration or how they created the piece. At other stops small place cards are available with related information. Now let's bring beacons into the picture. Imagine that a beacon is placed by every single piece of art that is part of the contest. A mobile application is built to subscribe to the broadcasts generated by each beacon; therefore, the application can tell your location in relation to the art pieces that are close by. If it can tell which piece is closest, it could automatically pull up a profile of the artist, offer links for more information and provide a button to vote.

In terms of retail, the applications are very interesting. Beacons could be used to navigate through store departments, offer promotions based on location inside the store, automatically pay for purchases or place a pre-order upon entry. [1] These ideas could lead to a more autonomous and personalized shopping experience.

For my project, I partnered with a retail business, the University Book Store at Grand Valley State University. The main goal in working with the store was to provide a low cost solution to increase foot traffic through the store and ultimately increase sales. We brainstormed and came up with few different ideas on how to do this.

- Put beacons in different departments in the store to provide coupons and information specific to that department. When a user walked into the department, their screen would show this information.
- Put a beacon near the book store that would automatically take the user to the book store's Facebook page so they could view the 'Like Us on Facebook' promotion.
- Put beacons in the entry ways to the building where the book store is located. When a user walks past the beacon, it will send a push notification to their phone with a message. Upon clicking or swiping the message, a mobile application would open that takes the user to a page with the latest promotion displayed.

We chose to work on the last option for several reasons. First, putting beacons outside the book store could potentially draw foot traffic into the book store and provide promotions without having the user actually enter the store. Secondly, this solution allows for dynamic content to be displayed based on the current promotion. Different promotions would display for different days. Lastly, utilizing push notifications gets the user's attention right away, so this could help with impulse purchasing. Along with implementing beacons and push notifications, we chose to add a tab in the existing GVSU Laker Mobile application for the book store. The tab contains links to the book store's web pages as well as the current promotion. We chose to do this so users would have another way to view the current promotion outside of push notifications.

This project proposal was very interesting to me because it incorporated social aspects. Push notifications have not traditionally been used for advertising purposes. This experiment allowed me to see what level of tolerance users would have for receiving promotion notifications on a somewhat regular basis. Their reaction could be viewed as positive if they use the coupons provided or it could be viewed as advertising spam. It also could make some users uncomfortable receiving notifications based on their current location. They could feel that it is an invasion of their privacy.

# Background and Related Work

Neil Hughes from *AppleInsider* reported earlier this year that many grocery stores such as Safeway and Giant Eagle are rolling out beacon technology. This technology will be used to provide in-store advertising. To start, they are only targeting customers as they walk through the entrance doors, not in each and every aisle. From the article, it sounds like this has been a success so far. [2] It appears that this retail application is very similar to my work with the book store. However, in order for the application to

function in real-time as they advertise, the user must have their loyalty mobile application open or activate the display on the screen. This stems from the way that mobile applications can interact with beacons in the foreground and the background. While in the foreground, applications can constantly poll for updates to beacons' locations and get real time results. This capability is called ranging. However, while the application is in the background it can only check for beacon statuses about every fifteen minutes or when the screen display is awoken. This makes getting the user's attention much more difficult when you require instantaneous notifications.

This previous year, a Grand Valley State University student demonstrated the uses of iBeacon to track spatial usage in the new library facility. [4] He placed Estimote beacons throughout the building to find out how many students resided in an area at the current time. The current occupied areas were shown on a monitor in the library and on the GVSU Laker Mobile application so students could choose study areas based on the amount of people already using those areas. His application also gathered data from the beacons and saved it in a repository for data mining purposes. Useful information was gathered to analyze usage patterns in the building. Some metrics that were captured included: how long users stayed in an area, number of people passing by the beacons and regular occupancy times in the library. This data seems very useful when determining how to organize areas of the building and making sure staffing levels are adequate based on usage patterns. [4] Overall, this project seemed to be successful in demonstrating the effectiveness of iBeacon at detecting users. It was able to show concrete evidence of varying usage patterns that one would expect, such as the highest volume of students existing on exam week. One weakness of this study is the need to estimate the number of users in a location based on certain factors. The targeted user must have an iPhone with Bluetooth LE, Bluetooth turned on and the Laker Mobile application downloaded and running in the background. It seems that some manual counting of students would still be necessary to establish an estimated ratio of users providing data and others who are not. This ratio could vary over time and would still need to be spot checked periodically.

# Program Requirements

The solution built for the university book store sends push notifications to potential customers utilizing iBeacons and the existing GVSU Laker Mobile application. The notifications are triggered based on the person entering the beacon region and contain current promotional material for the store. In order to avoid pushing updates to the mobile application on a very frequent basis, a content management system was created to supply the promotional material. Along with the push notifications in the mobile application, a static tab was built containing information regarding the book store including the current promotional image. This was included to provide a way for users to see the current promotion outside of receiving a push notification. Analytics were also implemented to determine how many people walked through the book store's building, how many users received the notification and how many acted on it. Manual metrics were also gathered in the form of coupon redemptions at the store.

The following sections detail high level requirements and why they are important to this project.

## Scalability

Typically when a developer makes changes to a mobile application, they need to resubmit the application to the applicable store for an update. In this case if we wanted to update an image, notification text or even add another beacon, traditionally we would provide an update to the application store. In order to make the experience much easier and less time consuming for administrators, a content management system (CMS) was built. The CMS saves information for beacons, notifications and promotions in a database. When the mobile application starts up it gets the list of beacons from that database and listens to them. For example, if the book store wants to use a beacon in another campus location, they simply place the beacon there and add a new beacon entry in the CMS with the beacon's identification number. That's it. The next time each user starts the GVSU Laker Mobile application, the list of subscribed beacons will refresh. The same concept applies to notifications for each beacon. Different beacons can be configured in the content management system to push different notifications. So, if the downtown campus book store is running a different promotion than the main campus, different notifications with different text and images can be sent.

## User Interaction

Since this application is truly a form of advertising, we do not want to annoy the user by spamming them constantly with notifications every time they walk within the range of a beacon. The CMS contains a setting for each beacon that indicates how often a user should be notified. The minimum is one day, but the recommended time period is two weeks. This is a typical cycle for the book store promotions and will ensure that the user is only getting notified once per promotion. Pushing notifications too often will result in users ignoring them, so we must be mindful of this.

## Customizable Notifications

As mentioned above, a CMS was created to make notifications customizable. Notifications can be changed to have one of three different behaviors. The most typical use case is to open a page in the application with some promotional text and an image of a coupon. Another use case is to display a page with text only. The third use case is to open the user's browser and take them directly to a specific web page. For example, if the book store is running a promotion where 'liking' them on Facebook will result in a coupon, the notification would take them directly to the book store's Facebook page.

## Performance

In the current world of mobile applications, users have little patience for applications that are slow. According to a study by AppDynamics, almost ninety percent of users have uninstalled a mobile application due to poor performance. Furthermore, users tend to ditch applications that are slow and move on to use a competitors application if it performed better. This goes even as far as switching banks based on the quality of their mobile application. [5] Needless to say, performance is very important. Making sure that a notification is sent promptly and loaded quickly is of the utmost importance in this application. To achieve a high level of performance, the book store tab and the notification screen do not block the UI thread. This means that users can navigate away from the page even before it is fully loaded. Also, promotion images used are only reloaded once per hour so that users can seamlessly navigate from one tab to another without waiting for the image to load. Of course, the best way to ensure performance is to have requests to the server complete quickly. Therefore, guidelines were placed around the size of images loaded to make sure megabytes weren't being transferred with every call. Grand Valley's own application/web server was used to have more fine grained control over the performance as well.

## Ability to Analyze Data

In order to measure the effectiveness of this application, data must be gathered on its usage. Different scenarios in the application trigger events that are sent to the server. Events that are tracked include:

- A user enters the region for a beacon – This will help analyze the foot traffic in the area where the beacons are placed. Beacons could then be moved to different locations to find the optimal place with the most foot traffic.
- A notification is sent to a user's device – We can see how many single users actually receive notifications.
- Users click on the notification – We can tell what subset of users from the previous event actually follow through and act on the notification. This will help measure overall effectiveness.

Through all of these user events that are tracked, no personal data is gathered from the user. A unique device ID is kept to link the events together from above but this cannot be traced back to the device. User privacy was of utmost importance when building this application.

# Implementation

The implementation of this solution required several parts, a content management system (CMS), a mobile application and a RESTful service

## Content Management System

The content management system was written in Ruby and utilized the Rails framework. [7] Ruby on Rails was a brand new technology that I had no prior experience with. Rails was chosen because the previous semester's iBeacon project utilized the same framework. This new application would be integrated into the existing application and function as separate web pages. We thought of using Java and Google's AppEngine to write this application originally, but it made more sense to use the existing code for support reasons. Building all server side logic for iBeacon functionality into a single application would help future graduate assistants find everything in a common location.

The Rails application is hooked up to a database using the ActiveRecord framework. [6] ActiveRecord allows developers to build models that will automatically propagate to the database as well as create the database tables themselves without writing a single line of SQL.

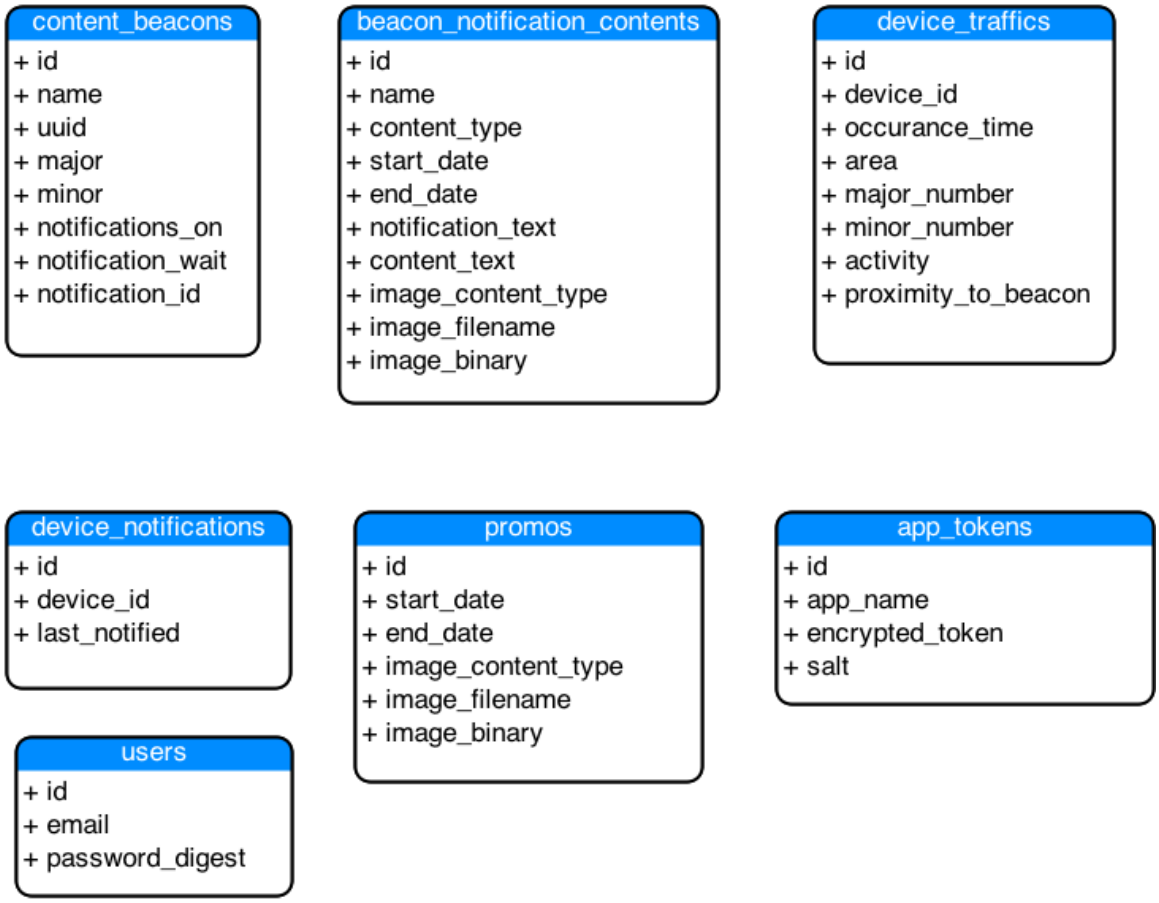


**Figure 1. Database Tables for Content Beacon Application**

In Figure 1, you can see a list of the database tables and their associated attributes. PostgreSQL was the DBMS chosen because it works very well with Rails. Below is a short description on the function of each table:

- Content_beacons – Holds a list of all beacons used in the application. Beacons are uniquely identified by a combination of the UUID, major and minor columns. Beacons can be turned on and off and also have a 'wait' setting to indicate how long before a user can receive another

notification. Beacons are paired with a single notification to know what to display on the mobile device.

- Beacon_notification_contents – Houses all the different notifications that can be applied to a beacon. Notifications have a defined date range where they are valid. Other properties of a notification include the actual text sent to the device as well as content/link text and images to be displayed on the screen.
- Promos – Has data for the promotions that are displayed on the static book store tab on the mobile device. Each promotion contains a valid date range and image information.
- Device_notifications – This table holds data that indicates the last date/time that a user was sent a notification. This is used to determine if another notification should be sent to the user.
- Users – The users table houses the user names and encrypted passwords of everyone able to access the content management system.
- Device_traffics – Holds all analytics information pertaining to mobile application usage. This table existed before the content beacon functionality was built into the application. The CMS simply leverages the same structure to track beacon usage.
- App_tokens – Has information pertaining to API keys used by the RESTful layer to authenticate when posting data to the application. Each application is given a name and a unique token that grants them access.

The content management system includes an authentication process for security purposes. This is to prevent intruders from posting malicious notification text and image content that is sent out to mobile application users on campus. In order to gain access to the CMS a Rails task must be run on the server. The task takes two parameters: an email address and a password. During execution, Rails uses the BCrypt library to encrypt the password and store it securely in the database. Change password functionality is available in the application so users can set their own passwords to whatever they like after logging in for the first time.

The front end of the application used the Twitter Bootstrap CSS library for styling. [8] Bootstrap provides default styles for lots of HTML elements. It also has out-of-the-box CSS classes that you can add to your HTML to make it look nice. Bootstrap is an easy way to get a stylish website up and running in a minimal amount of time. Along with Bootstrap for styling, jQuery and its form validation plugin were used for form validation throughout the site. The plugin provides out-of-the-box error messages and form submit handlers that require the user to enter valid data before submitting. It is a handy way to do quick form validation on a website. I had previous positive experiences with both of these libraries so they seemed like good candidates for this application.

## The RESTful Service

A RESTful service was built to handle two parts of this solution: communication between the server and mobile application as well as providing a way to post analytics to the server. REST stands for REpresentational State Transfer and refers to a set of guidelines when communicating between clients and servers. REST piggybacks on the HTTP protocol and is the standard style of service built when accessing server data through mobile applications (among other uses). The service built is characterized by a series

of endpoints that are available over the web as URI's. Each endpoint is designed with a specific purpose, to retrieve or post data to the server. Here is a list of endpoints that the service offers:

| Endpoint | HTTP Verb | Parameters | Purpose |
|---|---|---|---|
| /api/beacon/all | GET | N/A | Returns a JSON representation of all beacons. Currently used to register beacons on the mobile application. |
| /api/beacon/byId/{id} | GET | ID – The beacon identifier | Gets the details for a specific beacon by ID. |
| /api/beacon/notification/{id} | GET | ID – The notification identifier | Gets the details for a specific notification by ID. This is used by the mobile application to determine notification details to send when a beacon region is entered. |
| /api/beacon/promo | GET | N/A | Gets the current promotional image. If more than one promotional image is valid, it just returns the first one. |
| /api/beacon/post_notification/{id}/{uuid}/{major}/{minor} | POST | ID – The device ID<br>UUID – The beacon ID<br>Major – The beacon major version<br>Minor – The beacon minor version | Posts a notification to the server that the specific device has entered a region. The database is checked to see if the device is eligible for notification. If so, a JSON response is sent back indicating its validity. |
| /api/device_traffics | POST | App name – Requires the app name for | This endpoint is used for analytics purposes. The mobile application posts |

| | | | |
|---|---|---|---|
| | | authentication API token – Required for authentication Tracking info – Tracking information such as device ID, beacon, activity name | messages when specific events happen. |

## Mobile Application

For this project to be successful, an existing and active mobile application user base was required. GVSU's Laker Mobile application was the perfect candidate. It currently has around 30,000 users with availability in both the Google Play store and the Apple App store. The iOS application was chosen for its stability and seamless integration with iBeacons and the Bluetooth LE protocol.

There were two parts that were implemented in the mobile application, the static tab in Laker Mobile and the dynamic notification view. The static tab was built to fit in with existing functionality in Laker Mobile. It was added to the list of views that could be added to a user's "tab" bar that is located on the bottom of the screen. This view is mostly static content including a book store logo, link to the mobile site and icons for the various social media sites. The interesting functionality from this view comes from the promotion image that resides on the page. The image is loaded dynamically from the RESTful service detailed earlier. The image data is sent over in JSON format with the image being a base 64 encoded string generated from the actual image bytes. This method works very well except in cases where the image uploaded in the content management system is high resolution. The larger the image file size, the longer it takes to transfer the bytes to the user's phone and therefore, the longer it takes to render the image on the screen. The user sees a spinner GIF while the image is loading and their screen is not locked. They are still free to move about the application while the call completes. Once completed, the application only reloads the image once per hour. There is a check in the "viewDidAppear" method that only calls the RESTful service when the time limit has expired. This provides smooth transitions if the user moves back and forth between tabs. If no promotion is currently available or the service call fails, text is simply shown on the screen stating "No promotion image is currently available."

**Figure 2 - Estimote Beacon**

The implementation for the beacons content notifications is a bit more complex than the static tab. Estimote, the manufacturer of the beacons used, provides an SDK that can be used to access the signals transmitted by the beacon. [9] The SDK is essentially a wrapper around the CoreLocation framework in iOS. There are several types of monitoring for beacons provided in iOS. In order to receive data from the beacon, you must first register an Objective C class as a delegate. See the appendix item, LMBeaconContentManager.h for the Objective C code. A beacon manager instance (ESTBeaconManager) must be created to set the delegate and some high level options (see appendix item LMBeaconContentManager.m – snippets). Once this is created each beacon must be registered using a ESTBeaconRegion object. The beacon region is created using the UUID, major and minor versions, which in this application, are provided from the RESTful service. There are several types of monitoring a beacon manager can perform. The details and overridable subscription methods are described below:

- Boundary crossings – The beacon manager can tell when you come in range or move out of range of a beacon. The Estimote SDK provides two methods – didEnterRegion and didExitRegion to check for these conditions. Enabling the notifyEntryStateOnDisplay property allows the application to "wake up" if it's running in the background to check beacon status if the user turns on the phone display. This is particularly useful because in the event the user does not turn the display on, this event could take up to fifteen minutes to be recognized.
- Ranging – Ranging is only available while the mobile application is running in the foreground. It constantly polls the area for any applicable beacons. This capability is extremely useful if you want to tell exactly how close you are to a beacon or even relative proximity. It can also tell you if you are in range of more than one beacon. Because ranging constantly polls for updates, it tends to drain the device's battery faster, which is why it cannot run in the background. The overridable method is didRangeBeacons.
- State determination – When the device crosses an applicable region boundary, the state of the beacon can be determined. This monitoring is also useful for determining relative proximity to

beacons. Developers can trigger this callback manually by calling requestStateForRegion. The overridable method is didDetermineState.

For this application, we only needed to determine when a user entered the boundary of a given beacon. Therefore, we subscribed to the didEnterRegion callback method. When a user crosses the threshold of a registered beacon, didEnterRegion is called which, in turn, calls the RESTful service to see if a notification should be sent to the user. If not, the application does nothing and continues to run in the background. If so, another call to the service is made to retrieve details on the notification including the text that should be displayed and the type of content to show on the screen. When a local notification is generated, the user can either have Laker Mobile in the foreground or the background. If the application is running in the foreground, the view will automatically switch to a view called the "beacon content." The view will be displayed differently based on the response from the RESTful service. The three different scenarios are text only, image and link. If one of the first two scenarios is matched, the "beacon content" view is displayed with text only, or text and the image. If the link scenario is matched, a web browser will pop open with the link provided from the service. While Laker Mobile is running in the background, rather than immediately open the application, the user must swipe or select the displayed notification. If they choose to do so, the application goes through the scenarios above again. In this instance, the choice is left entirely to the user. They are not required to act on the notification; they need only do so if they are interested in the promotion.

Throughout the new code added to Laker Mobile, there are calls to the RESTful service indicating that the user performed some sort of interesting action. The specific interesting actions were detailed earlier in this paper. All analytics calls are non-blocking, asynchronous calls to the service; therefore, the user will not notice them happening. If these calls happen to fail because the user can't connect to the service, the events are simply discarded. Analytics are implemented this way as to not affect the user experience in any fashion.

# Results, Evaluation, and Reflection

First and foremost, time was a limiting factor when evaluating results as part of this project. Because of time taken to develop the software, publish the application and coordinate beacon installment, the project ran for only seven days before results were gathered. In this short period of time, I believe that the application was moderately successful. Results were gathered based on individual device identification numbers so that nothing can be traced back to a specific user. The number of individual devices that actually received notifications during the seven day trial period was eight. A number of factors may have contributed to this number being low. First, results were collected during the summer at the university when the volume of students is significantly lower than other times of the year. This isn't to say that many people did not visit the university, but those who did have a lower probability of having the Laker Mobile

application installed on their device. Secondly, a number of requirements must be met by each individual user for them to receive the notification: they must have an iPhone, have Bluetooth turned on and have the Laker Mobile application installed. These criteria may minimize the potential number of users who even have the possibility of getting a notification. Thirdly, applicable users must either stay in the range of the beacon for fifteen minutes or turn on their display while entering the beacon region. In my opinion, this also significantly lowers the odds that a user will get a notification. Despite all of these obstacles to overcome, analytics still show that people saw the applicable marketed content. Fifty percent of distinct user devices that received notifications acted on them.

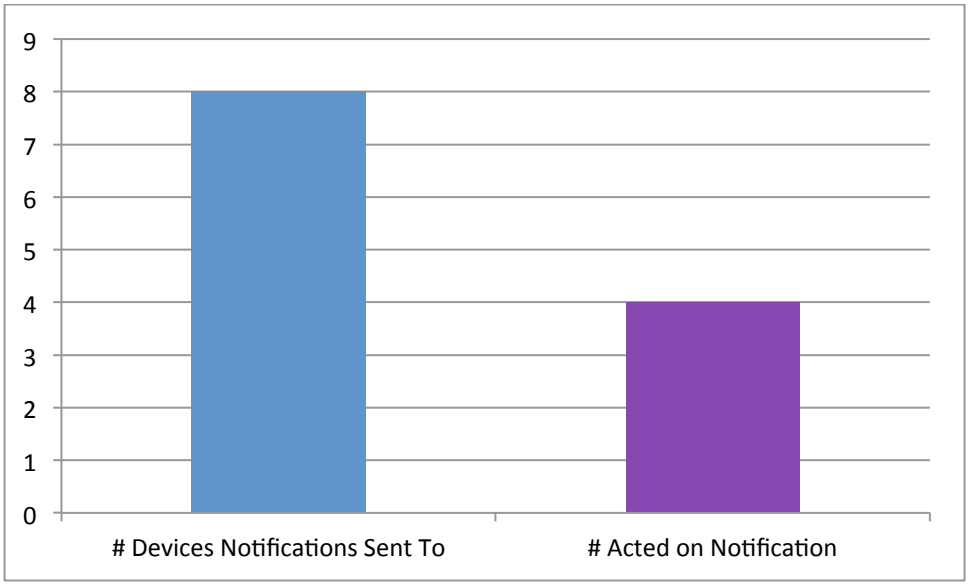


**Figure 3 - Users acting on notifications on**

I believe that attracting fifty percent of users to open up an advertisement on their phone is a successful form of marketing. Taking the time to click the notification means that they are genuinely interested. The next step would be to track the number of successful coupon redemptions in correlation to the number of notifications acted on. Unfortunately, the data for this was unavailable as the book store has no automated way to track redemptions for the current promotion. Even so, putting the content in a personalized and localized context begs the user to take action more than other forms of marketing such as billboards on the side of the road.

For the purpose of testing in a short time frame, the number of days between notifications was set to a single day. This helps us analyze notifications deeper than on a per device basis and dig into the patterns of how often users move through the beacon region.

| Beacon Name | Total Number of Times Region Entered | Total Number of Notifications Sent | Total Number Acted on Notification |
|---|---|---|---|
| Kirkhoff Beacon | 9 | 6 | 2 |
| Kirkhoff Beacon 2 | 8 | 5 | 3 |
| Total | 17 | 11 | 5 |

As you can see from the table above, users entered the region a total of seventeen different times. They only received notifications eleven of those times. This indicates that users are walking through the beacon areas more than once per allowed time period (one day). By looking at this data and individual device identification numbers, we could find out if a single user is consistently walking through the beacon region and if they are acting on notifications more than once. Since the "acted on" count in the table above is five and the individual device "acted on" count is four, this indicates that one user acted on the notification multiple times. This is important to gauge whether or not customers will become annoyed with notifications after the first reception and elect not to click on them. Furthermore, from the data above we can analyze beacon placement within the building. The number of times a user entered the region between both beacons was about equal. Kirkhoff Beacon had nine and Kirkhoff Beacon 2 had eight. It would be interesting to add more beacons or move each of the beacons to see if they can gain a higher count of users walking through their respective areas.

Going forward I would like to try some new things to see if I could get better results. As mentioned before, strategic beacon placement would be one thing to try. If beacons could be placed in the most crowded area or at points where people typically turn on their phone display, it could maximize potential of users receiving notifications. For instance, if the building that houses the book store has a coffee stand, it would be smart to put a beacon near that since people would be more apt to turn on their phone displays while they are waiting in line. It would also be beneficial to see if the university has usage data on walking patterns through the building where the book store resides. If one entrance is more heavily used, multiple beacons could be used to ensure the entire walkway is covered.

In order to get a better idea of how notifications are used, I would recommend that analytics continue to be gathered during a busier part of the year. The low amount of foot traffic during the summer and the short results gathering period contributed to a small data set. Since the data set is so small, it is hard to assume that the results can be applied on a broader scale. Even so, the first results appear promising. A fifty percent chance of a person viewing a book store advertisement in direct proximity to the book store seems like it would have an increase in foot traffic and in sales as well. The book store should be involved in gathering metrics such as an increase in foot traffic and coupons redeemed.

In terms of the solution that was built for the book store's requirements, I would say that this project has been a success. The architecture developed is extremely scalable and can be used to add many more beacons in different locations. If a battery fails on a beacon, it can easily be swapped out for a new one. The mobile application works seamlessly with the RESTful service and does not require frequent updates for new beacons. Analytics are tracked and recorded with minimal impact on user performance.

# Conclusions and Future Work

Overall, the project was very successful in terms of architecture. Initial results are promising but need more time to develop and further analytics are necessary before any concrete conclusions can be reached. The book store's goals were met however. A low cost, flexible solution with minimal maintenance was built.

One open issue that should be taken into account and fixed is a limitation on the size of images that can be uploaded. Instead of coaching users, a hard size limit should be set in the content management system to prevent the transfer of large amounts of data between the CMS and the mobile application. This would provide optimal performance and guard against user error.

To reach a larger audience, I suggest that the changes made in the iOS platform should be made in the Android platform as well. Also, analysis should be done on traffic patterns through the building that houses the book store to find optimal locations for beacon placement. These two ideas will ensure that the most people possible will receive notifications and subsequent promotions from the book store. The store could also do some marketing of this feature in the Laker Mobile iOS application. They could encourage students to use the application to find the latest deals in the store. They could also urge people to download the application for those deals if they have not done so already. The higher percentage of people that travel through the building where the book store is located with Laker Mobile, the better the chance they will be reached by the notifications.

# Bibliography


1. Elgan, Mike. WhyApple's 'indoor GPS' plan is brilliant. 2013 Sept 14. Details can be found at http://www.computerworld.com/s/article/9242393/Why_Apple_s_indoor_GPS_plan_is_brilliant?pageNumber=1
2. Hughes, Neil. US grocery chains rolling Apple's iBeacon tech out to dozens of stores. 2014 Jan 6. Details can be found at http://appleinsider.com/articles/14/01/06/us-grocery-chains-rolling-apples-ibeacon-tech-out-to-dozens-of-stores
3. iOS Developer Library – Region Monitoring and iBeacon. Details can be found at https://developer.apple.com/library/ios/documentation/userexperience/conceptual/LocationAwarenessPG/RegionMonitoring/RegionMonitoring.html
4. Serpoosh, Sam. Contextual Computing and Context Awareness Occupancy and Traffic Monitoring. 2014 Apr.
5. AppDynamics. Mobile App Performance Increasingly Critical – AppDynamics Releases 'App Attention Span' Study Which Shows Nearly 90 Percent Surveyed Stopped Using an App Due to Poor Performance. Details can be found at http://www.marketwatch.com/story/mobile-app-performance-increasingly-critical-appdynamics-releases-app-attention-span-study-which-shows-nearly-90-percent-surveyed-stopped-using-an-app-due-to-poor-performance-2014-06-12
6. Ruby on Rails – ActiveRecord Framework. Details can be found at http://guides.rubyonrails.org/active_record_querying.html
7. Ruby on Rails. Details can be found at http://rubyonrails.org/
8. Twitter Bootstrap. Details can be found at http://getbootstrap.com/css/
9. Estimote SDK. Details can be found at http://estimote.com/api/

# Appendix

## LMBeaconContentManager.h

```
#import <Foundation/Foundation.h>
#import <ESTBeaconManager.h>

@interface LMBeaconContentManager : NSObject <ESTBeaconManagerDelegate>
- (void)registerContentBeacons;
- (NSTimeInterval)getHoursSinceLastUpdated;
@end
```

## LMBeaconContentManager.m - snippets

```
-(void)registerContentBeacons{
    self.beaconManager = [[ESTBeaconManager alloc] init];
    self.beaconManager.delegate = self;
    self.beaconManager.avoidUnknownStateBeacons = YES;

    NSMutableURLRequest* request = [NSMutableURLRequest requestWithURL:
                                    [NSURL URLWithString:ALL_BEACONS_URL]];
    [request setHTTPMethod:@"GET"];
    (void)[[NSURLConnection alloc] initWithRequest:request delegate:self];
}

ESTBeaconRegion *region = [[ESTBeaconRegion alloc] initWithProximityUUID:uuid
                                                                   major:[dict valueForKey:@"major"]
          minor:[dict valueForKey:@"minor"]                                identifier:[dict
          valueForKey:@"name"]];

region.notifyEntryStateOnDisplay = YES;

[self.beaconManager startMonitoringForRegion:region];
[self.beaconManager startRangingBeaconsInRegion:region];
```