Ping Based Online Digital Marketing Service

COMP 4801 - Project Final Report (v1)

Project By:
Apoorva Katta (3035123617)

Project Supervised by:
Dr. Chuan Wu

[15 April, 2018]
Abstract

With the rising usage of smartphones on the street, physical display advertisements in public spaces are losing their reachability to the public as a result of dropping public attention to reality. Moreover, advertisers are investing large sums of money to advertise their products or services in public space. Henceforth, this project has resulted in the successful development of a combination of web and mobile applications to enable advertisers to digitally market their goods or service on an Online-to-Offline (O2O) service. Moreover, this project also achieved its aim to generate revenue by requiring advertisers to pay for the designed service. Furthermore, this report provides justifications on the idea as well as methodologies used to implement the idea. Adding on, this report also illustrates on the system architecture used and the approach to implement the idea. To sum up, 2 public interfaces were developed for public use.

1. A web interface for advertisers to publish their adverts digitally.
2. A mobile interface for both Apple and Android smartphone end-users to access adverts digitally.

Please note that the 2 public interfaces are still in development process and will be released to the public after thorough testing and resolving all potential legal disputes.

This report mainly addresses the completed stages of development of the system architecture.
Acknowledgement

I would like to thank Dr. Chuan Wu for helping me to conceptualize my idea. Also, I would like to thank Mr. Keith Chau for guiding me in writing this report.
## Contents

1 – Introduction ........................................................................................................................................... 6  
1.1 – General background with respect to the real world ................................................................. 6  
1.2 – Problem statement and motivation ......................................................................................... 7  
1.3 – Objective and scope ...................................................................................................................... 9  
1.4 – Deliverables ................................................................................................................................. 13  
1.5 – Rough outline of the remaining parts of the report ............................................................... 13  
2 – Related works ................................................................................................................................... 14  
2.1 – Similar applications .................................................................................................................... 14  
2.2 – Differentiating from similar applications .................................................................................. 15  
3 – Implementation approach ............................................................................................................... 16  
3.1 – Brief overview ............................................................................................................................. 16  
3.2 – System architecture ..................................................................................................................... 16  
3.3 – Phases of development .............................................................................................................. 19  
4 – Completed phases of development ............................................................................................... 20  
4.1 – Phase 1 – Research and project specification ........................................................................... 20  
4.2 – Phase 2 – Database implementation .......................................................................................... 21  
4.3 – Phase 3 – Backend server implementation ............................................................................... 22  
4.4 – Phase 4 – Client web application implementation ....................................................................... 27  
4.4 – Phase 5 – Client mobile application implementation ................................................................. 30  
4.6 – Phase 6 – Testing and deployment ............................................................................................... 31  
5 – Future development ......................................................................................................................... 32  
5.1 – Phase 7 – Releasing the application to the public ....................................................................... 32  
6 – Discussion .......................................................................................................................................... 33  
6.1 – Risks, Challenges and Mitigation ............................................................................................. 33  
6.2 – Limitations ..................................................................................................................................... 33  
7 – Project schedule ............................................................................................................................... 34  
7.1 – Project development schedule ................................................................................................. 34  
7.2 – Project management deliverables schedule .............................................................................. 35  
8 – Conclusion ......................................................................................................................................... 36  
9 – References .......................................................................................................................................... 37
List of Figures

Figure 1 – Average daily usage of smartphones (US) ................................................................. 6
Figure 2 – Specialized walking lanes for smartphone users in China ............................................ 6
Figure 3 – Advertisements in Time Square, New York, US ......................................................... 8
Figure 4 – 2D posters advertisements at Causeway Bay MTR station, HK ................................. 8
Figure 5 – Illustration on the 2d advertising pings .................................................................... 10
Figure 6 – Apple Wallet Pass on Apple Wallet ......................................................................... 10
Figure 7 – Android Pay Card on Android Pay ........................................................................... 10
Figure 8 – Illustration on no mobile notification when smartphone is not in ping range .......... 11
Figure 9 – Illustration on mobile notification when smartphone is in ping range ..................... 12
Figure 10 – Technical Illustration on the architecture of the system ........................................ 18
Figure 11 – Illustration on the formation of JSON Web Tokens (JWT) token ............................. 23
Figure 12 – OOGO user account system ................................................................................... 27
Figure 13 – OOGO service dashboard system .......................................................................... 28
List of Tables

Table 1 – Routes implemented on the backend server .............................................................. 24
Table 2 – Project Development Schedule .................................................................................. 33
Table 3 – Project Management Development Schedule ............................................................ 34
1 – Introduction

This section commences with factual information on the general background of the project idea, followed by problem statement, motivation, specifications and deliverables of the project. In addition, this section also outlines the remaining sections of the report.

1.1 – General background with respect to the real world

Over the last decade, there has been a significant rise in the usage of mobile phones [1]. Referring to a 2016 study conducted in the US by an analytics firm Flurry shown in Figure 1, on average, US citizens are spending more 5 hours on their smartphones on a daily basis in the year of 2016 [2]. Importantly, 5 hours per day accounts for more than 20 percent and 100 percent rise in daily usage compared to years of 2013 and 2015 [2]. Moreover, in correlation to the increasing usage on smartphones, pedestrian deaths due to mental and visual distraction have increased by 9 percent [3]. Furthermore, the increasing number of such nature of accidents has resulted in the Honolulu government to impose a USD 50 fine against pedestrians walking with their eyes glued to their smartphone screens [3]. Similarly, as shown in Figure 2, a district governing body in China introduced a pedestrian walkway for mobile phone users [4]. Also, since 2013, Hong Kong’s MTR have introduced anti-phone announcements to encourage public to not keep their eyes only on their smartphones [4]. These arguments provide justification on the idea that the general public is using their smartphones on the streets while commuting.

Figure 1 – Average daily usage of smartphones (US)

Figure 2 – Specialized walking lanes for smartphone users in China
1.2 – Problem statement and motivation

Extending from section 1.1, it is possibly safe to assume that the general public is losing its focus from reality while on the streets to their smartphones. Moreover, as shown in Figure 3 on the following page, a significant number of businesses physically advertise their goods or services. However, it is important to understand that firms advertising their products or services in public spaces are possibly losing their effectiveness as a result of decreasing advertisement reachability. In simpler terms, public advertisements are not able to gain people’s attention on the street as the public is glued to their smartphone screens while walking on the streets. Moreover, firms are spending a large sum of money for advertising in public spaces. For example, as shown in Figure 4 on the following page, a travel agency advertises potential travel locations in the world on a selection of continuous 12 physical 2-dimensional posters advertisement in Hong Kong’s MTR Causeway Bay station. Moreover, the firm has to pay HKD 35,700 per week for a minimum of 3 weeks to advertise their service to passing commuters [5]. Hence, with potentially decreasing effectiveness of traditional public advertisements, it is safe to assume that the travel firm is unable to gain significant returns from their advertisement expenditure. Therefore, this project aimed to generate revenue by helping firms to effectively advertise their products or services to the general public in public spaces through its paid Online-to-Offline (O2O) service.
Figure 3 – Advertisements in Time Square, New York, US

Figure 4 – 2D posters advertisements at Causeway Bay MTR station, HK
1.3 – Objective and scope

The objective of this project is to take advantage of people’s usage habits of their smartphones in public spaces by enabling marketers to digitally market their product on an Online-To-Offline (O2O) service using Global Positioning System (GPS) and Mobile Wallet Technologies. In simple terms, Online-To-Offline (O2O) service is used to classify services that transfer online user traffic to offline user traffic, i.e. directing online users to physical shops. This was achieved by developing and releasing two public interfaces, a web interface and a mobile interface. Note that the project will be developed for worldwide use.

The web interface will enable marketers to create virtual pings that hold their advertisement data on a 2-dimensional map. As represented by the red location icon and the green circle in the purple circle in Figure 5 on the following page, the pings will be a circular area with a certain radius that will be centered at the user defined pair of longitude and latitude values. Moreover, the web interface will allow marketers to create different types of pings to suit their desired purposes; such as Text Ping, Coupon Ping, Membership Card Ping and Ticket Ping. The functions of these different types of pings are listed as below;

1. A Text Ping will hold text based promotions.
2. A Coupon Ping will hold one-time-use offer promotions.
3. A Membership Card Ping will aid loyalty programme registration of new nearby customers.
4. A Ticket Ping will aid in distributing certain transit or event tickets.

For Coupon Ping, Membership Card Ping and Ticket Ping, the marketer will need to design a mobile wallet pass/card so that mobile interface users can store their coupons, membership cards and tickets on their mobile wallet. The web interface provides such service for Apple Wallet passes (refer to Figure 6) and Android Pay cards (refer to Figure 7).
Figure 5 – Illustration on the 2d advertising pings in the purple circle

Figure 6 – Apple Wallet Pass on Apple Wallet

Figure 7 – Android Pay Card on Android Pay
The mobile interface will periodically retrieve and notify end-users about nearby digital advertising pings based on their GPS location. To illustrate the idea, as shown in Figure 8, there is no notification displayed on the smartphone of a user when the smartphone is not in the range of the advertising ping. Furthermore, as shown in Figure 9, a notification is displayed on the smartphone when the smartphone is within the range of the advertising ping. Moreover, the mobile interface will also allow manual retrieving of nearby digital advertising pings. For Coupon Ping, Membership Card Ping and Ticket Ping, the mobile interface will aid end-user in registering for a corresponding Apple Wallet pass or Android Pay card. Also, a service for in-app purchase of certain pass/card will be provided through Apple Wallet or Android Pay payment options.

Figure 8 – Illustration on no mobile notification when smartphone is not in ping range
Figure 9 – Illustration on mobile notification when smartphone is in ping range
1.4 – Deliverables

To achieve the objective of this project, the following deliverables are to be delivered at the end of this project.

→ One web application interface for marketers to enable them to create, retrieve, update and delete advertising pings and design mobile wallet passes and/or cards
→ Two mobile applications interface for end-users to enable them to retrieve advertising pings and register for mobile wallet passes or cards
→ One final report to justify and describe the new service
→ One final presentation to pitch the new service to the public

1.5 – Rough outline of the remaining parts of the report

The remainder of this report will proceed as follows. Firstly, there will be some analysis on some similar works that are currently being used or have been used in the past. Secondly, there will be some elaboration on the methodologies that will be used to complete this project. Then, the status of the project will be addressed as well as some discussion on difficulties encountered and limitations. Lastly, there will be a conclusion to summarize the main focus of this report.
2 – Related works

This section includes research on similar applications that are already in the market as well as comparison of those applications to the one suggested by this report.

2.1 – Similar applications

Note that, only functionality based downsides are mentioned in the analysis of these applications to different these applications from the proposed application.

1) Van Leuween Mobile App [6] – Van Leuween is a New York, USA based ice cream vendor that started a campaign in 2014 that notified nearby app users about $5 ice cream campaign to attract them to visit their shop. Their app incorporated PayPal’s mobile payment service for digital payments. Such initiative was well accepted in public as the vendor was able to make 5% of its sales digitally. However, the downside to this application is that the app only allowed marketing of a specific product by a specific firm.

2) Nearbuy.com Mobile App [7] – Nearbuy.com is a New Delhi, India based digital marketing firm that allows its users to manually search for nearby offers based on region rather than dynamic location. However, the downside to this application is that its users have to manually search for nearby offers rather than automatically getting notification on them as it is not using any GPS technology. Also, the app is limited to within India use only.

3) Thinknear Marketing Solutions [8] – Thinknear is a California, USA based company that designs and develops personalized mobile apps for various companies that enable these firms to advertise to nearby people based on their locations. However, the downside to this is that, one application will only work for one firm.

4) HoKoBuy Mobile App [9] – HoKoBuy is a Hong Kong based company that allows marketers to advertise their products on its web and mobile interface. However, similar to Nearbuy.com, HoKoBuy does not use GPS technology to advertise its content to its mobile application users.

5) Eatigo Mobile App [10] – Eatigo is a Hong Kong based mobile application that advertises discounts at various restaurants based on reservation timings. However, similar to Nearbuy.com and HoKoBuy, this mobile application also does not advertise based on real-time location of its user.
2.2 – Differentiating from similar applications

Based on a couple of weeks of research on location based marketing applications, it was found that there were not any popular location based marketing application in the current market that used both GPS and Mobile Wallet technologies and allow multiple marketers to advertise their product on one interface. Henceforth, this project will address all the downsides mentioned in section 2.1, to differentiate from these applications as well as succeed over them.
3 – Implementation approach

This section includes description of the designed system architecture as well as the phases of development of the project.

3.1 – Brief overview

This project was developed incrementally. Moreover, the development of this project was divided into six ordered major phases; the six ordered major phases are research and project specification, database implementation, backend server implementation, client website application implementation, client mobile application implementation, testing and deployment. Note that, each ordered major phase relies its preceding ordered major phases to be completed.

3.2 – System architecture

This project is a combination of three key web development processes; front-end, back-end and database development. In terms of software engineering, a combination of these 3 web development process is known as full-stack project. Moreover, MEAN stack and Python Django are two commonly used frameworks to develop full stack projects. The noticeable difference between the two frameworks is that MEAN stack uses JavaScript and Python Django uses Python. Moreover, as the developer of this project is vastly familiar with the MEAN stack framework over the Python Django framework, this project was be developed using the MEAN stack. MEAN stack framework consists of MongoDB, ExpressJS, Angular 4 and NodeJS for database, webserver, frontend and backend respectively.

Moreover, as Angular 4 can only be used to make web interfaces, Apple and Android mobile applications were made outside of this framework. The Apple and Android applications were developed using XCode and Android Studio respectively. Furthermore, to isolate the database from the main application, CouchDB was used instead of MongoDB. CouchDB is a no-SQL based database that provides its functionalities over HTTP requests. The reason behind the isolation of the database is to allow easy deployment of the server without the requirement of restarting the database service.

Moving on, both the backend and database server will be developed as Representational State Transfer Application Programmable Interface, in short REST API. To better
understand REST API, consider it as a waiter, the server as the kitchen and the web-app client as a customer. Commonly, a customer requests an order to the waiter. Then the waiter carries the request to the kitchen. After that, the kitchen responds with the order to the waiter. Finally the waiter carries the order to the customer. Putting the analogy into context, web-app and mobile clients create HTTP request and send it to the server REST API. The REST API will execute the request and respond to the web-app and mobile clients accordingly.

Referring to Figure 8, all client applications use HTTP and JSON body for request-response query cycle to communicate with the server REST API on the Amazon Web Service EC2 Virtual Machine IP address. IP addresses are used to identify the target computer, or virtual machine in this case, on the internet. Moreover, for data security, the database REST API is allowed access only from the IP address of the designated Amazon Web Service EC2 Virtual Machine. This will block hackers from accessing the database remotely and manipulating it. In contrary, the server REST API will be allowed access from any IP address given valid user authentication details, i.e. API access token. The API access token is basically a JSON web token, also known as JWT. JSON web tokens are used to identify and check for the user credentials before execution of request. Also, note that to enable easy scalability, the webserver will be hosted on Amazon Web Services’ EC2 virtual machine. A virtual machine is basically a virtual computer located at a remote location.
Figure 10 – Technical Illustration on the architecture of the system
3.3 – Phases of development

This subsection roughly describes the six phases of development.

1) Research and project specification – This phase includes in-depth research of the project and finalizing project specification.

2) Database implementation – This phase includes designing and implementing CouchDB database.

3) Backend server implementation – This phase includes development of backend core logic functionality using NodeJS and ExpressJS. Adding on, it also includes CouchDB database integration, Apple Wallet integration for Apple Wallet Passes, Android Pay integration for Android Pay Cards, Slack integration for error reporting, and other trivial tasks. Slack is a chatting web-app that has an open-end endpoint for sending message to a particular internal chatroom.

4) Client website application implementation – This phase includes development of a frontend website application using Angular 4, HTML5 and CSS. Apart from the functionality stated in section 1.3 for the web interface, the web application will include visual error reporting and other trivial add-ons.

5) Client mobile application implementation – This phase includes development of two mobile applications for Apple and Android devices. Both the mobile application will have the same functionality as stated in section 1.3 for the mobile interface. The Apple and Android applications will be developed using XCode and Android Studio respectively.

6) Testing and deployment – This phase will includes user acceptance test (UAT) to verify whether the application is market ready or not. If needed, changes will be made to implementation of previous phases based on user feedback. After the completion of the testing process, the project will be deployed to a live environment for the public to start using. Note that, a domain name for the web application will be purchased after the whole application is completed.
4 – Completed phases of development

This section includes description of the three completed phases of development.

4.1 – Phase 1 – Research and project specification

In this phase, two major sub-tasks were completed. The first of the two major sub-tasks to be completed was market research. It is important to evaluate the current market to determine whether the market will accept or deny the suggested application of this project. Therefore, after the research was completed, it has been determined that there are multiple marketing and advertising based applications that the public has accepted and started using. One of these applications is HoKoBuy. In this application, merchants can post their goods and services on the system for the public to view and purchase. Moreover, more than 75,000 people have downloaded the application system’s mobile application for use over the last six months. Henceforth, market research suggested that the application suggested in this report had good prospects to survive in the market.

The second of the two major sub-tasks that was completed was finalizing the project specification. In this sub-task, with respect to the completed market research, the objective and scope of the project was finalized. In addition to that, the application’s initial system architecture was designed. Section 3.2 of this report illustrates and describes the designed system architecture.

This task was completed on schedule without any difficulties or limitations.
4.2 – Phase 2 – Database implementation

Similar to phase 1, this phase was also divided into two major sub-tasks. The first of the two major sub-tasks that was completed is Amazon Web Services (AWS) Elastic Compute Cloud (EC2) virtual machine setup. AWS EC2 has been used to create a remote virtual computer that can be accessed over the internet and perform computational tasks on cloud.

The reason behind choosing AWS EC2 is straightforward; its system downtime is lower than its competitors and it has a streamlined data recovery system that allows its users to recover data if AWS EC2 system crashes. This means that, the application will run consistently over longer time intervals with limited interruptions that block its execution. Adding to those points, one other reason for choosing AWS over its competitors in the cloud service industry is that it was developed before most of them and so has significant stakes in the market. Hence, to a high extent, the possibility of AWS stopping its services is lower than its competitors. Henceforth, AWS EC2 virtual machine is being used for the suggested application.

Moreover, as Linux OS generally yields the best server response time, the virtual machine runs on the Linux OS instead of Windows Server OS and other available operating systems.

The other of the two major sub-tasks was to setup database server on the AWS EC2 virtual machine. As decided in phase 1 system architecture design, CouchDB was the designated database server for this application. Henceforth, CouchDB v2.1 was installed on the server.

Unlike phase 1, there was one difficulty encountered in AWS EC2 setup. The issue was that the AWS EC2 virtual machine was not accepting TCP connection for any ports other than the SSH port 22. After a few days of research on the issue, it was figured that AWS EC2 virtual machines block all other connection except the SSH port 22 connection by default. The issue was resolved by adding custom inbound connections for AWS EC2 connection firewall and adding port listeners on relevant ports on the AWS EC2 virtual machine.

Despite the difficulties encountered, this phase was also completed on schedule.
4.3 – Phase 3 – Backend server implementation

In this phase, I completed the backend server implementation using REST API logic. As stated in section 3.2 system architecture, I used ExpressJS 4.14.0 and NodeJS 6.10.0 to setup my backend server. A backend server is used to provide server side functionality to the frontend, i.e. web or mobile, interfaces. The role of a backend server in a web application system is to implement the user authentication, core business logic and database access through HTTP endpoints.

One of the key decisions that was made in this phase was to whether use an HTTP server over a HTTPS server, or the vice versa. In the end, it was decided to use the more secured option of HTTPS server, as it allows the application to protect user data from being intercepted on the TCP connection to and from the server. To set up an HTTPS server, a Secure Sockets Layer (SSL) certificate is required. The SSL certificate is used to verify that the request is being sent to the intended application and not to an impersonating application. Adding to that, the SSL certificate is also used to send encrypted data over a TCP connection. Moreover, before any data is sent over a TCP connection, client applications, for example, Google Chrome and Postman, verifies the SSL certificate with a known and globally accepted Certificate Authority (CA). The concerning request is sent only when the CA responds confirms the authenticity of the SSL certificate or the client user has added the SSL certificate to his trusted certificate on his local machine. For development, a self-signed SSL certificate was created to start a HTTPS server. The downside of using a self-signed SSL certificate is that the self-signed SSL certificate fails the CA verification and hence need to be manually added to trusted certificate on the client application. When the suggested application is market ready, a CA authorized SSL certificate will be for the HTTPS backend server.

With HTTPS server, the client is able to verify and identify the server. So the next key decision that was made was to implement client verification and identification for the server. There are two approaches, they are client generates verification token or the server generates verification token and sends it to the client. A popular way to authenticate client is to create and send JSON Web Tokens (JWT) token on the server. A JWT token consists of three segments; header, payload and signature that are delimited by a dot string, i.e. “.”. The header segment is base64URL encoded and includes the hashing algorithm used to
encrypt the signature. The payload segment is base64URL encoded and includes the client identifier on the database, i.e. userId, and the expiry time of the token, i.e. exp. The signature segment is the hashed string of base64URL encoded header and base64URL encoded payload delimited by a dot string, i.e. “.”, with the client secret stored on the server, i.e. userSecretOnServer. The signature is hashed using the hashing method given in the header segment. The format of the JWT token is illustrated in Figure 11. The server uses a JWT token to validate and identify the client. This is achieved by the following steps. Step 1, the server decodes the base64URL encoded string from the payload segment of the JWT token and extracts the userId value. Step 2, the server accesses the database and retrieves the userSecretOnServer and uses it to verify the token. Step 3, if the token is valid, the server responds back with the requested data for the user. If the token is invalid or expired, the server responds back with an error. Note that user login, register, reset password request and reset password endpoints do not check for JWT token.

After configuring the backend server to meet the applications needs, I implemented create, retrieve, update and delete functionalities for user, pings and mobile wallet passes/cards. In addition to this password reset endpoints and email verification were also implemented. These functionalities are used by the web and mobile interfaces to fulfill the objective of the application. Moreover, since the backend server is be allowed to interact with user defined scripts, i.e. not the designated web and mobile application, error handling was also added.
A node package, named sendgrid, is being used to send email to users dynamically. The reason why an external service is required is because the application does not include a mail server. Moreover, the package sendgrid is being used for development only to check the flow of the application. As using an external mail server creates vulnerability for the application, before deployment, a google suite business mail server will be purchased and be used for the application in production stage.

Also, to add more functionality to the application, analytics functionality was also implemented. At this point, the functionality is limited to number of passes registered for a ping. In future, more functionality will be added.

Note that billing functionality has not been implemented yet as all major online payment system require a registered business and bank account. Hence, the billing functionality will be implemented right before the deployment of the public interfaces. Possible platforms for online payments are PayPal, Epoch and Stripe.

Here is a list of endpoints on the backend server:

<table>
<thead>
<tr>
<th>Endpoints</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ping</td>
<td>To check if backend server is running or not</td>
<td>Completed</td>
</tr>
<tr>
<td>/pingAuthorized</td>
<td>To check if backend server is running or not</td>
<td>Completed</td>
</tr>
<tr>
<td></td>
<td>To check if the attached JWT token to the request headers is valid or not.</td>
<td></td>
</tr>
<tr>
<td>/user</td>
<td>To retrieve user data from the application database that is associated with the given username</td>
<td>Completed</td>
</tr>
<tr>
<td>/user/register</td>
<td>To register new user to the application database</td>
<td>Completed</td>
</tr>
<tr>
<td></td>
<td>To send an email to the user to verify their given email for registration</td>
<td></td>
</tr>
<tr>
<td>/user/emailVerification</td>
<td>To verify the associated user account email</td>
<td>Completed</td>
</tr>
<tr>
<td></td>
<td>To allow the associated user to login to the system</td>
<td></td>
</tr>
<tr>
<td>Route</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>/user/login</td>
<td>To check if the associated user account is in the application database or not</td>
<td>Completed</td>
</tr>
<tr>
<td></td>
<td>To return a valid JWT token for the associated user to allow him to access restricted endpoints</td>
<td></td>
</tr>
<tr>
<td>/user/update</td>
<td>To update the associated user account in the application database</td>
<td>Completed</td>
</tr>
<tr>
<td>/user/resetPasswordRequest</td>
<td>To send reset password email for the associated user account</td>
<td>Completed</td>
</tr>
<tr>
<td>/user/resetPassword</td>
<td>To reset password for the associated account</td>
<td>Completed</td>
</tr>
<tr>
<td>/user/deactivate</td>
<td>To deactivate user account in the application database</td>
<td>Completed</td>
</tr>
<tr>
<td></td>
<td>To remove associated ping and pass data to the user account in the application database</td>
<td></td>
</tr>
<tr>
<td>/ping/get</td>
<td>To retrieve all associated ping data to the user account from the application database</td>
<td>Completed</td>
</tr>
<tr>
<td>/ping/get/:id</td>
<td>To retrieve the associated ping data to the user account from the application database</td>
<td>Completed</td>
</tr>
<tr>
<td>/ping/create</td>
<td>To create ping data in the application database</td>
<td>Completed</td>
</tr>
<tr>
<td></td>
<td>To update associated user account with the ping data reference in the application database</td>
<td></td>
</tr>
<tr>
<td>/ping/update/:id</td>
<td>To update ping data in the application database</td>
<td>Completed</td>
</tr>
<tr>
<td>/ping/invalidate/:id</td>
<td>To invalidate ping data in the application database so that the ping data is still available to the user but the ping is not active</td>
<td>Completed</td>
</tr>
<tr>
<td>/ping/delete/:id</td>
<td>To delete ping data in the application database so that the ping data is not available to the user and hence the ping is not active</td>
<td>Completed</td>
</tr>
<tr>
<td>Route</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>/ping/report/:id</td>
<td>To report misused pings and block the misused pings from being sent to other users</td>
<td>Completed</td>
</tr>
<tr>
<td>/pass/:id</td>
<td>To get selected pass data</td>
<td>Completed</td>
</tr>
<tr>
<td></td>
<td>Note that this route is the default value of the barcode on the mobile wallet pass/cards</td>
<td></td>
</tr>
<tr>
<td>/pass/issueRequest/:id</td>
<td>To send pass issue requirements to the requester, i.e. user defined data on the pass and payment (if applicable)</td>
<td>Completed</td>
</tr>
<tr>
<td>/issue/:id</td>
<td>To issue the pass with send pass data</td>
<td>Completed</td>
</tr>
<tr>
<td>/invalidate/:id</td>
<td>To invalidate pass to indicate that the pass has been used or has expired</td>
<td>Completed</td>
</tr>
<tr>
<td>/analytics</td>
<td>To calculate and respond with analytical data</td>
<td>Partially Completed</td>
</tr>
<tr>
<td></td>
<td>Need to add more analytical data</td>
<td></td>
</tr>
<tr>
<td>/billing</td>
<td>To use online payment services to implement billing logic</td>
<td>To do</td>
</tr>
</tbody>
</table>

Table 1 – Routes implemented on the backend server

One of the significant difficulties I faced during the development of the backend server was about getting used to asynchronous programming logic, i.e. callbacks and promises. It was very confusing at first. But with time it got easier.

Similar to phase 1 and 2, this phase was also completed on schedule.
4.4 – Phase 4 – Client web application implementation

Please visit this hyperlink (https://13.250.32.222:80/) to access the OOGO client interface. Please contact me at apoorvakatta@gmail.com if the server is down.

This phase started on a bad note. Google rejected my application of issuing Android Pay cards as their business logic was conflicting with my intended use of their API. Unlike Apple Wallet, the Android Pay system had been designed to be the sole issuer of its cards as it aims to directly connect to the actual card advertisers, i.e. target clients of this project. Thus, Android Pay blocks any sort of middleman type application from creating a separate platform to issue Android Pay cards. Henceforth, as this project has been limited to creating only Apple Wallet passes, the Android application previously proposed is out of the scope of this project.

Moving back to phase 4 of development, an online dashboard based interface was designed to give advertisers a platform to create Apple Wallet passes. This platform was named “OOGO” which is an abbreviation for “Offers On GO”. This platform consists of two major user interface/user experience elements, the OOGO user account system and the OOGO service dashboard system.

Figure 12 – OOGO user account system
To avoid cluttering this report with further user interface images, I have added a hyperlink that will direct you to the OOGO web platform at the beginning of this sub-section. Please access the web platform while reading this sub-section.

Firstly, the OOGO user account system, illustrated in Figure 12. This system consists of all the conventional user account system requirement; and they are login, register, forget password, reset password and verify email. All the requests to the backend server in this section do not require any authorization.

Secondly, the OOGO services dashboard system, illustrated in Figure 13. The system has been designed to implement the following interfaces; ping interface, analytics interface, billing interface and account interface.

The ping interface has been designed to enable clients to design, update and manage pings. For managing pings, a table has been shown to specify the specifications of all the launched pings. On the table, the client can update, invalidate and delete pings. Moreover, one of the key business logic was also added at this stage of development. The logic is that pings created by free client users are automatically invalidated after 2 days. The way this is implemented is as follows. A ping that has been running for more than 2 days will be invalidated the next time the free client user logs in. For the period between the end of the 2 days limit and the next time the free client user logs, the mobile application will receive the
ping but will not notify the end-user about them. Also, each free client can have up to 5 free pings for testing purposes. To create more pings, the user will have to purchase more pings from the billing interface. Moreover, buying more pings will allow the user to have unlimited time pings. The analytics interface has been designed to visualize key statistical data for the client user to provide them with more information their pings. The billing interface has been designed for the client users to purchase OOGO services. The user interface has been designed to enable the client user to modify their profile on the OOGO system.

As indicated in the methodology section, the web application was implemented using Angular 4. Angular 4 uses TypeScript instead of conventional JavaScript for client-side scripting. Also, one of the main benefits of using Angular is its 2-way data binding between html form element values and TypeScript. As this was my first time using TypeScript, a lot of time was spent understanding the implementation of Angular 4 using TypeScript and debugging the written code. Also, in comparison to its previous version AngularJS and Angular 2, Angular 4 has very limited functionality and online helping resources. For example, the HttpClient module in Angular that handles AJAX HTTP and HTTPS requests does not support the use of self-signed certificates. As a result, I had to use a bypass to execute jQuery function in a TypeScript function.

Lastly, the system has been designed in such a way that common web-security vulnerabilities, for example cross-site scripting (XSS) and SQL/No-SQL injection, are minimized. Cross-site scripting is a security issue in which a perpetrator injects malicious codes on the server. SQL/No-SQL injection is a security issue in which a perpetrator is able to unfairly access the database.

As coding in TypeScript took longer than expected, this phase of development was finished 2 weeks after the proposed the completion dates/
4.4 – Phase 5 – Client mobile application implementation

In this phase, the OOGO iOS app was designed. As the previous phase was completed late, the development done in this phase was limited and subject to time constraints.

Firstly, GPS integration, mobile wallet integration and notification integration were implemented using Apple iOS SDKs. Next, the core and intuitive logic were implemented, i.e. receiving nearby passes from the backend server and notifying user of their presence. Also, to avoid repeated ping notification, the mobile app filters out already notified pings before forwarding presence notifications on the mobile.

Also, an in-app form system was added to dynamically add user entered data to the pass before creation. Also, for paid passes, the user will have to pay for them before he could create their mobile wallet pass. Note that the payment system will be added after a company has been registered so that online payment options may be used for in-app payments.

In addition to that, to avoid misuse of the service, a report system was added to the mobile interface. For pings that have been reported for 5 times, they will automatically be invalidated.

Lastly, this phase was completed 1 week after schedule and the OOGO iOS app is still being reviewed by Apple for public release.
4.6 – Phase 6 – Testing and deployment

With the time constraint, this phase only limited UAT testing was done. In the limited testing, a significant number of test users commented that there was no need of text pings as most users will prefer coupons pings over the text pings. Hence, text pings were removed from the implementation.

The project is still in deployment stage and the remaining works have been added to the Future Development section.
5 – Future development

This section will elaborate on the future development of the project.

5.1 – Phase 7 – Releasing the application to the public

This phase includes the remaining work of phases 1 to 6. Hence, this phase will require code clean up and logic loopholes analysis. The logic loopholes analysis will be done manually. Next, if applicable, all potential legal risks will be mitigated. Once that is done, a Certificate Authority (CA) authorized certificate will be registered for the applications. Next, a dedicated mail server will be purchased for applications email services. And Lastly, a company would be registered in Hong Kong for the application so that online payment system can be implemented. Once everything mentioned above is completed, the applications will be released to the public.
6 – Discussion

This section provides a discussion on the risks, challenges and mitigation, and limitations.

6.1 – Risks, Challenges and Mitigation

With such a massive project, risks and challenges are inevitable. The major challenge for this project was to design user interface (UI) and user experience (UX). Success of both web and mobile interfaces heavily depend on these two elements and the functionality. Henceforth, with the limited time given, it is extremely challenging to achieve satisfiable level of quality. Furthermore, to mitigate such risks and challenges, qualified personal were consulted for assistance. Also, significant time was spent on researching for best used practices to mitigate common risks.

6.2 – Limitations

There were and still are two major limitations of this project. The first of the two major limitations is related to the Global Positioning System (GPS) technology. GPS technology only works on 2-dimensntional maps, i.e. longitude and latitude. Hence, it needs to be understood that there is no possible approach to differentiate between pings for different altitude. Note that, this limitation cannot be resolved as there are no available mobile technologies that will resolve this limitation. The second of the two major limitations is related to end-user preferences. The mobile application requires end-users to turn on mobile data and location features all the time. However, there is no way to force or guarantee that end-user will accept that and turn on these features all the time due to monetary or mobile battery concerns. Note that, since turning on mobile data and location features is completely up to the user, this limitation cannot be resolved. Henceforth, these were and still are the two major limitations of this project.
7 – Project schedule

This section provides information on the schedule of the project.

7.1 – Project development schedule

Table 2, shows the list of development phases ordered periodically. Also the current status of each phase of development is shown in the table.

<table>
<thead>
<tr>
<th>Dates</th>
<th>Milestones</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>01 Sep 2017 – 01 Oct 2017</td>
<td>Phase 1 – Research and Project specification</td>
<td>Completed</td>
</tr>
<tr>
<td>02 Oct 2017 – 10 Oct 2017</td>
<td>Phase 2 – Database implementation</td>
<td>Completed</td>
</tr>
<tr>
<td>21 Nov 2017 – 10 Feb 2018</td>
<td>Phase 4 – Client website application implementation</td>
<td>Completed</td>
</tr>
<tr>
<td>11 Feb 2018 – 30 Mar 2018</td>
<td>Phase 5 – Client mobile application implementation</td>
<td>Completed</td>
</tr>
<tr>
<td>01 Apr 2018 – 10 Apr 2018</td>
<td>Phase 6 – Testing and deployment</td>
<td>In Progress</td>
</tr>
</tbody>
</table>

Table 2 – Project Development Schedule
### 7.2 – Project management deliverables schedule

Table 3, shows the list of project management deliverables sets ordered periodically. Also the current status of each of the set is shown in the table.

<table>
<thead>
<tr>
<th>Dates</th>
<th>Milestones</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 Sept 2017 –</td>
<td>Project Management Deliverables Set 1 –</td>
<td>Completed</td>
</tr>
<tr>
<td>01 Oct 2017</td>
<td>Project proposal and Project website</td>
<td></td>
</tr>
<tr>
<td>25 Dec 2017 –</td>
<td>Project Management Deliverables Set 2 –</td>
<td>Completed</td>
</tr>
<tr>
<td>07 Jan 2018</td>
<td>First project presentation</td>
<td></td>
</tr>
<tr>
<td>08 Jan 2018 –</td>
<td>Project Management Deliverables Set 3 –</td>
<td>Completed</td>
</tr>
<tr>
<td>21 Jan 2018</td>
<td>Project interim report and preliminary implementation</td>
<td></td>
</tr>
<tr>
<td>05 Apr 2018 –</td>
<td>Project Management Deliverables Set 4 –</td>
<td>Completed</td>
</tr>
<tr>
<td>15 Apr 2018</td>
<td>Project final report and finished implementation</td>
<td></td>
</tr>
<tr>
<td>10 Apr 2018 –</td>
<td>Project Management Deliverables Set 5 –</td>
<td>Scheduled</td>
</tr>
<tr>
<td>15 Apr 2018</td>
<td>Final project presentation</td>
<td></td>
</tr>
<tr>
<td>15 Apr 2018 –</td>
<td>Project Management Deliverables Set 6 –</td>
<td>Scheduled</td>
</tr>
<tr>
<td>01 May 2018</td>
<td>Project exhibition deliverables – TBC</td>
<td></td>
</tr>
<tr>
<td>TBC</td>
<td>Project Management Deliverables Set 7 –</td>
<td>Scheduled</td>
</tr>
<tr>
<td></td>
<td>Project competition deliverables – TBC</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3 – Project Management Development Schedule**
8 – Conclusion

In conclusion, to exploit the increasing usage of smartphones in public spaces, this project is set to achieve its aims to deliver a web-application and two mobile applications in the coming month or two. Moreover, the developed application will be generating revenue as some monetary rules have been implemented into these applications. The web-application targets advertisers by enabling them to create virtual location based advertisements. The mobile applications aims to target the general public by showing them advertisements based on their locations. Furthermore, Global Positioning System (GPS) and mobile wallet technologies were used to implement the designated interfaces. However, the downside of using these technologies is that GPS technology only works on 2-dimensional planes and not all users have mobile data and location service running on their phone throughout the day.

The entire project had been divided into six ordered major phases; research and project specification, database implementation, backend server implementation, client website application implementation, client mobile application implementation, and testing and deployment. The first three phases; research and project specification, database implementation, backend server implementation were completed on schedule. The next two phases client website application implementation and client mobile application implementation were completed after schedule by 2 weeks and 1 week respectively. The remaining phase, phase 6, is partially completed and the remaining work has been pushed to phase 7, releasing the application to the public.

Moreover, as Android Pay had rejected my request to develop Android Pay cards, any implementation related to Android were removed from the scope of the project.

Henceforth, the project was almost completed and will require further work after the end of this project before releasing the applications to the public.
9 – References


