RewardWallet

School of Engineering Science | Burnaby, BC · V5A 1S6
https://github.com/RewardWallet

March 30, 2018

Dr. Andrew Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: ENSC 405W Design Specification for a Customizable Reward Allocation System

Dear Dr. Rawicz:

Attached you will find our ENSC 405W Design Specification for a Customizable Rewards Allocation System. This project is designed to add reward allocation to any business by integrating their POS system into our cloud system. This easy and affordable solution is poised to give small Canadian businesses the tools needed to propel their businesses into the digital era.

These design specifications primarily focus on the detailing how the RewardWallet system prototype will be built. In addition, some sections include how the design could change to accommodate future improvements made for the pre-production model.

RewardWallet is made up of three undergraduate engineering students: Molly Bin, Nathan Tannar, Mandy Xiao. Should you have any questions or concerns, please feel free to reach out to myself at (604) 355-6292 or ntannar@sfu.ca

Sincerely,

Nathan Tannar

President and CEO
RewardWallet

Enclosure: *Design Specifications for a Customizable Reward Allocation System*

Design Specifications for the

Customizable Rewards Allocation System

Project Team:

Molly Bin
Nathan Tannar
Jia Hui (Mandy) Xiao

Contact Person:

Nathan Tannar
ntannar@sfu.ca

Submitted to:

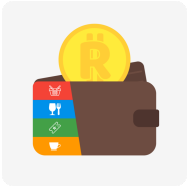
Dr. Andrew Rawicz - ENSC 405W
Steve Whitmore - ENSC 405W
School of Engineering Science
Simon Fraser University

Issued date:

March 30, 2018

Revision:

1.0



RewardWallet

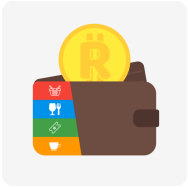
Design Specification for a Customizable Rewards Allocation System

Abstract

In today's fast paced digital world, businesses have to move quickly if they want to get an edge or keep up with the competition. Loyalty programs in particular show no signs of slowing down. “The multi-billion dollar business is just as popular as ever with Canadian consumers and companies alike” [5]. Enter RewardWallet, an end-to-end solution for small Canadian businesses that does not require any maintenance or technical knowledge for monthly subscription fee.

RewardWallet is a system that consists of 3 subsystems: an embedded system for each business, a mobile app for end-users, and a cloud server for primary computation and record storage. The embedded system acts as a bridge between the businesses current point-of-sale (POS) system, our cloud system and the users phone. It can create transaction records which are send to the user’s phone through NFC where the iOS app will complete the transaction by sending the users identification to the server. The process acts as a secure “handshake” between the business and a user so the RewardWallet system can verify valid purchases and prevent fraud. With this design any business can be enrolled into RewardWallet and out system can allocate reward points to their customers digital card based on a reward distribution model of the business’ choosing.

This design specification document details descriptions for the design and development of a Customizable Rewards Allocation System for the proof-of-concept models. The discussions and considerations are for the prototype and pre-production models marked with I and II respectively in the Functional Specification for the Customizable Rewards Allocation System [1] document. While future improvements to the system were envisioned, they are unfortunately out of scope for the current development timeline.



RewardWallet

Design Specification for a Customizable Rewards Allocation System

1 Table of Contents

1.1 Table of Figures 4
2 Introduction 6
2.1 Scope 6
2.2 Intended Audience 6
3 Complete System Design 7
3.1 System Overview 7
3.2 General Design Considerations 8
4 Embedded System Design 10
4.1 General Design Considerations 10
4.2 Security Design Considerations 10
4.3 Reliability and Performance Design Considerations 10
4.4 Usability Design Considerations 11
5 Cloud System Design 12
5.1 System Overview 12
5.2 General Design Considerations 13
5.3 Security Design Considerations 13
5.4 Standards 14
5.5 Reliability Design Considerations 14
5.6 Performance Design Considerations 14
5.7 Usability Design Considerations 15
6 Mobile App Design 16
6.1 System Overview 16
6.2 General Design 16
6.3 App Security 17
6.4 Standards 17
6.5 Reliability and Performance Design Considerations 17
6.6 Usability Design Considerations 18
7 Engineering Standards 20
8 Conclusion 21
9 Glossary 22
10 Appendix I - Test Plan 24

10.1 Unit testing 24
10.2 Integration testing 26
10.3 System testing 26
11 Appendix II - User Interface 27
11.1 Introduction 27
11.2 User Analysis 27
11.3 Tech Analysis 29
11.4 Engineering Standards 32
11.5 System Test Plan 32
11.6 Conclusion 34
12 Appendix III – 440 Planning Document 35
12.1 Introduction 35
12.2 Competition and Market Rationale 35
12.3 Risk Analysis 36
12.4 Personal Task Allocation 38
12.5 Time Management 39
12.6 Milestone Chart 40
12.7 Budget Management 41
12.8 Funding 42
12.9 Conclusion 42
13 Sources and References 43

1.1 Table of Figures

Figure 1: High Level RewardWallet System Model 7
Figure 2: Wire connections between Raspberry Pi 3 and Adafruit PN532 board. 11
Figure 3: High Level RewardSystem API Model 12
Figure 4: Security Validation Layers 13
Figure 5: Cloud System Performance Diagram. 15
Figure 6: Control flow. 16
Figure 7: Security architecture of IOS. 17
Figure 8: Flowchart of usability design. 18
Figure 9: RewardBeamer container. Dimensions: height 15cm, width 20cm, length 35cm. 28
Figure 10: RewardBeamer lid with LED cutout. Dimensions: height 1cm, width 20cm, length 35cm, cutout radius 0.3cm. 28
Figure 11: Welcome Splash Screen and Business Explore View. 29

Figure 12: Login Screen and Digital Wallet View. 30
Figure 13: NFC Reward Collection Screen and Collection Confirmation Screen. 31
Figure 14: Gantt Chart for 440 Part 1. 39
Figure 15: Gantt Chart for 440 Part 2. 39
Figure 16: Milestone Chart in Timeline. 40
Figure 17: Milestone Timetable. 41

Table 1: Reward Collection Roadmap 8
Table 2: XCode, Swift and IOS versions with device compatibility. 16
Table 3: Page data and method. 19
Table 4: Embedded System Unit Test. 24
Table 5: Cloud Server Unit Test. 24
Table 6: Mobile Application Unit Test. 25
Table 7: Integration Test. 26
Table 8: System Test. 26
Table 9: Scale of Risk Level 36
Table 10: Risk, Severity and Likelihood of Occurrence. 37
Table 11: Budget 42

2 Introduction

RewardWallet is a system that keeps track of transactions users make at a business to allocate rewards to a customer's profile based on how the business customizes their distribution model. Each customer will have a unique digital card for each business that stores reward points. By creating the hardware needed by a business, a mobile app for the user and a cloud system to bring it all together, RewardWallet gives business owners an end-to-end solution for adding a customizable digital loyalty system to their store. The design specification for RewardWallet describes how the systems requirements will be fulfilled through technical design.

2.1 Scope

his document outlines the design of the RewardWallet system and subsystems and details how said design will fulfill the functional requirements as described Functional Specification for the Customizable Rewards Allocation System [1]. This document outlines the design specifications to the components that make up the RewardWallet system. The specifications go as far as to outline the base architecture of the system that will be captured in the initial development phases. Only the requirements marked as I or II will be evaluated. The design of stage III requirements will be done post-production.

2.2 Intended Audience

his design specification was written with the intent of being read by all members of RewardWallet. Test Engineers will be use them when defining test cases to ensure each subsystem meets reliability and performance requirements. Design engineers shall use these specifications to build the system to ensure all functional requirements are met. Lastly, business owners can refer to this document as an aid should they ever need it.

3 Complete System Design

Design specifications for how the RewardBeamer, Cloud Server, and mobile app should operate in parallel to complete the system that makes up RewardWallet.

3.1 System Overview

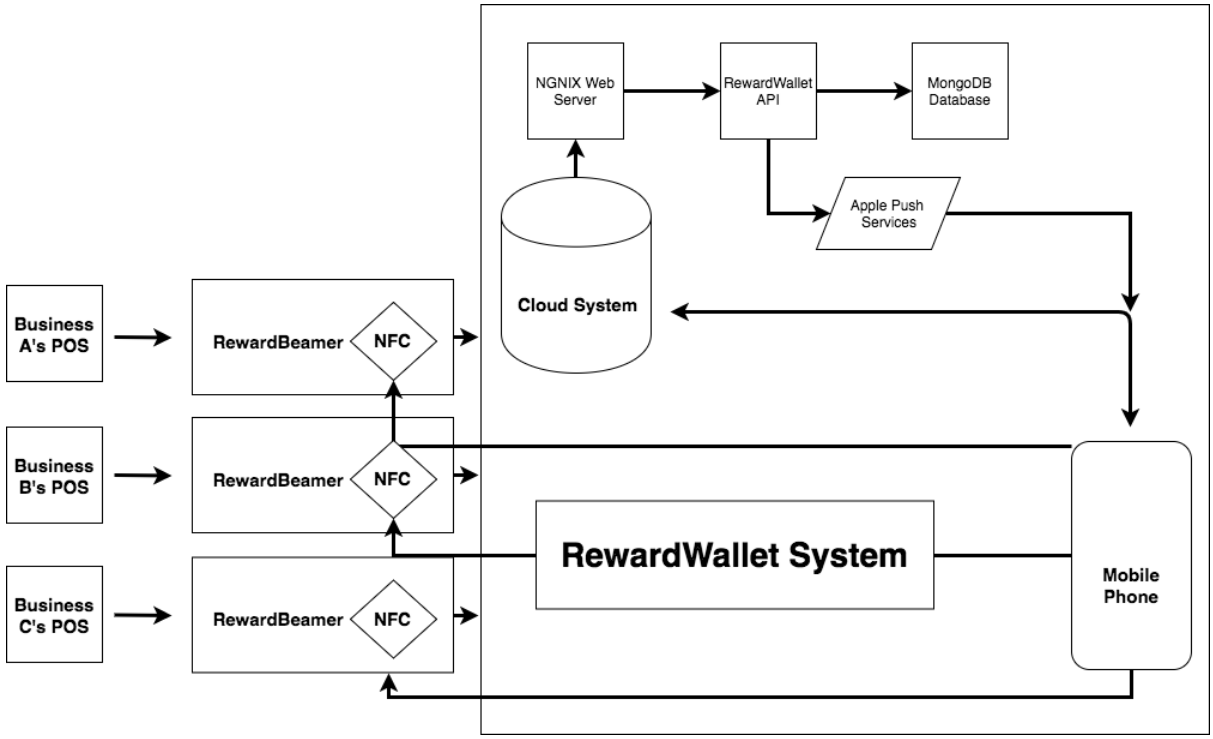


Figure 1: High Level RewardWallet System Model

RewardWallet is a system that operates across multiple separate layers. The core of the system is cloud based; however, it needs a way of communicating with the end user and a business's POS system. Enter the RewardBeamer - an embedded system running on a Raspberry Pi microcontroller that serves as an interface with the business' POS in addition operating as a reward terminal to the mobile app. When it comes time for a user to collect rewards the businesses POS system can be automated to send a request to the RewardBeamer which initializes the reward allocation process. On the cloud system the API can determine which distribution model the business is using and be ready to allocate points accordingly. This presents a system that each business can tune to their own needs.

For a successful reward point collection, a user can simply tap their phone running the RewardWallet mobile app on the RewardBeamer. Providing the transaction request that the RewardBeamer previously made was successful and had not timed out, this tap action serves as a security handshake between devices. The cloud system will then respond on the mobile

app alerting the user that they have successfully gained points. This roadmap of requests can be seen in Table 1.

Step	Device	Action
1	POS/External Device	Sends POST request to RewardBeamer to initiate a transaction
2	RewardBeamer	Sends POST request to the Cloud API including the transaction amount and business' identification code
3	Cloud API	Creates a transaction for the business that has a timeout period of 30 seconds. Returns a transaction token to the sender
4	RewardBeamer	Configures the NFC controller to act as a serial tag that is coded with the returned transaction token ID
5	Mobile App	Reads the transaction token ID over NFC and sends a POST request with the transaction token ID and current user to the Cloud API
6	Cloud API	Looks up the business associated with the transaction and uses its reward distribution model to determine the amount of points to allocate. Sends an update point balance to the user and closes the transaction.
7	Mobile App	Receives updated point balance and alerts the user the reward collection was successful

Table 1: Reward Collection Roadmap

3.2 General Design Considerations

There are a few notable solutions engineered with this design. We knew it was important that any user should be able to use their phone to wirelessly tap for reward using NFC. However, a problem presented for iPhones is that developers are restricted to only using the chip for reading purposes. Moreover, the iPhone NFC chip can only be used to read NFC tags. This is why the NFC chip on the RewardBeamer had to be programed to simulate a regular NFC tag - a very unconventional use. With this design, a transaction is first initialized on the RewardBeamer and then sent to a device for verification, rather than a device writing the users identification to the RewardBeamer to complete the transaction. Because all processing and verification is done on the Cloud API and not the other subsystems, this does approach not pose as a security risk.

Integrating the RewardBeamer with any existing POS is a challenge large enough for a separate project. Thankfully, there are already companies like Omnivore that specialize in

software for creating an API interface to all of the top POS brands. This consolidates the problem from interfacing with any POS system to just interfacing with an API. All that was left was to set the RewardBeamer to accept a POST request to initiate a transaction. This method of interfacing with the RewardBeamer will allow it to conform to the RESTful design standards, making integration easy on a POS system or any other device.

4 Embedded System Design

The Embedded System, also known as the RewardBeamer, serves as device that accepts HTTP POST requests from businesses' existing POS system. It is a component that connects the POS system and the cloud server. For the prototype design of the RewardBeamer, a Raspberry Pi 3 controller is utilized in conjunction with Adafruit PN532 board to initiate transactions and transmit records to a receiving mobile phone.

4.1 General Design Considerations

The RewardBeamer is able to initiate a transaction be receiving POST messages from POS system and immediately sends a request to the Cloud Server creating a transaction record with the transaction amount and the specific business identification number. This guarantees that the record can be retrieved at a later time for verification. After the Cloud API has created the transaction record requested, it returns a transaction token to the RewardBeamer. Since IOS does not support Host-based Card Emulation on iPhone, the development team has programmed the RewardBeamer such that it performs NFC tags for read-only message in NFC Data Exchange Format [9] [10].

Furthermore, the data exchange format NDEF is used for communication between the RewardBeamer NFC device to the iPhone [11]. This is due to the limitation to reading of specific data exchange format on IOS devices [12]. The RewardBeamer is emulating the NFC Type 4 Tag because it enables read and/or write operations in NDEF. Therefore, it is the ideal NFC Type for purpose of this device.

4.2 Security Design Considerations

The safety and security of the RewardBeamer is crucial to the business and customer users of RewardWallet. Therefore, the device takes necessary precautions to prevent any malicious activities. All codes and security information for the RewardBeamer are not made accessible to users. Business users merely have to indicate their choice of business model. Then the Cloud Server will allocate points to their customers accordingly. Thus, the users do not have to and do not have authority to modify any elements of the RewardBeamer.

4.3 Reliability and Performance Design Considerations

The development team at RewardWallet puts great focus on the reliability and performance of the product. The RewardWallet is designed to operate continuously for up to 24 hours. Since it operates under a 5V power source and a backup battery, the device will still be able to continue without disruption in the case of a power outage. All transactions are processed in real time in order to avoid any delays or miscommunications caused by those delays. As previously mentioned, the RewardBeamer communicates with the Cloud Server via network requests. However, in the case where the POST request sent by the RewardBeamer had not successfully reached the Cloud Server, the transaction will timeout after 30 seconds. This ensures that no half-completed transactions remain in the system. Similarly, if the transmission of record

reference through NFC is not successfully received by the mobile device of a customer within 20 seconds, the transaction will also timeout.

The RewardBeamer is able to handle one transaction at a time. Any attempt to use the device for a different transaction while the current transaction is still in progress will be rejected by the device. This is to make sure that no interruptions are to occur during a transaction. Therefore, minimizing the chances of duplicated transaction records or incorrect manipulations of data. In an event of system failure, the RewardBeamer is able to follow the re-initiation protocol and reload its internal programs automatically. Thus, the device will resume operation 2 minutes after failure.

4.4 Usability Design Considerations

For ease of use, the development team have made the RewardBeamer easy to set up. Users can simply plug in the device to a 5V power source. Then, it will automatically load the internal programs needed. Since the RewardBeamer makes HTTP requests to the Cloud Server, data network will be configured. The user can configure the device to be added to a local network.

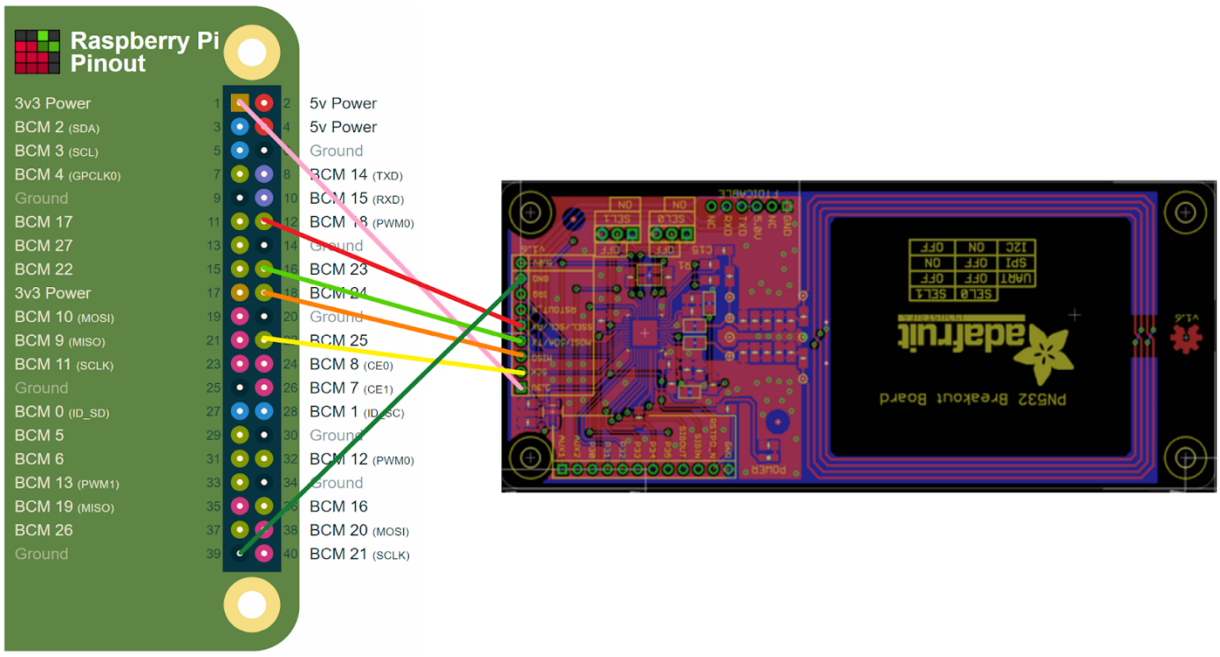


Figure 2: Wire connections between Raspberry Pi 3 and Adafruit PN532 board.

5 Cloud System Design

The Cloud System is responsible for managing the shared resources between the RewardBeamer and iOS application, ensuring data security and allocating rewards to users based on the businesses distribution model.

5.1 System Overview

The cloud system has been designed to run on a Linux operating system, a standard for backend applications. Preliminary security and routing will be handled by NGINX, a high-performance web server that will allow incoming HTTPS connections from both the embedded system and iOS application. These connections will be routed our RewardWallet API which will run all our required cloud functions in addition to interfacing with a MongoDB instance for data.

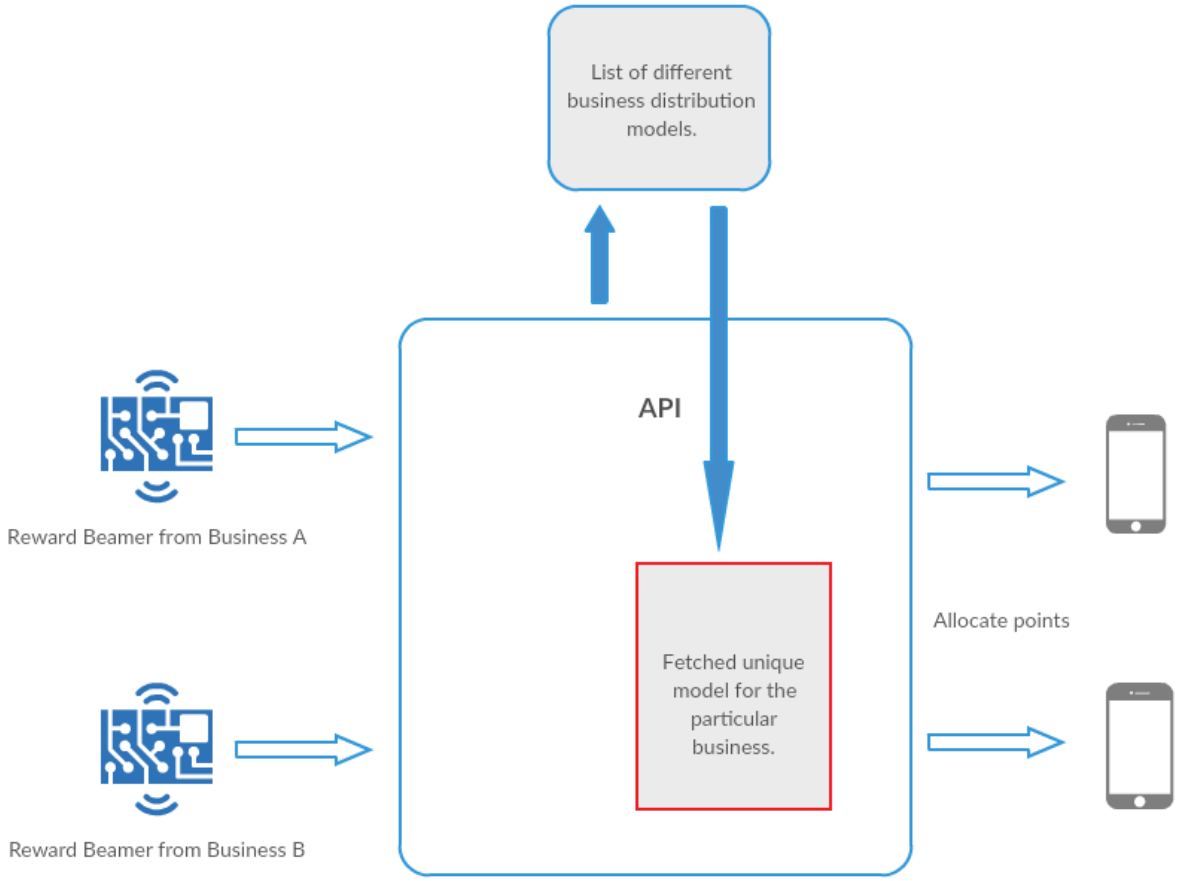


Figure 3: High Level RewardSystem API Model

5.2 General Design Considerations

Upon planning the design of the cloud system our engineering team decided to make a few general considerations. Because the decision was made to go with a primarily cloud system that handles all of the processing, the development of various cloud functions needed to be implemented. The primary functions would be to initiate a transaction, complete a transaction, and redeem rewards. Each of these steps would store records in the database. On top of use case functions, the RewardWallet API would also need user authentication, signup and a method for business owners to view transaction history (detailed in 4.7 Usability Design Considerations).

Our engineering team was most comfortable with a NO-SQL database such as the popular and free to use MongoDB. Due to its popularity we were able to find many open-source libraries that allows developers to easily interface with the database to store and retrieve objects. This allowed our team to focus their time on engineering the best security protocols to abide by and develop the necessary cloud functions that needed to be written specifically for our design.

5.3 Security Design Considerations

When building a system that gives users a form of digital currency, data security is critical for establishing trust with businesses and customers. Preventing fraud is critical, for if someone was able to maliciously collect reward points without an authorized purchase it would pose a danger to RewardWallet and all the businesses using it. For these reasons, multiple security layers have been designed. These security layers are shown in Figure 4.

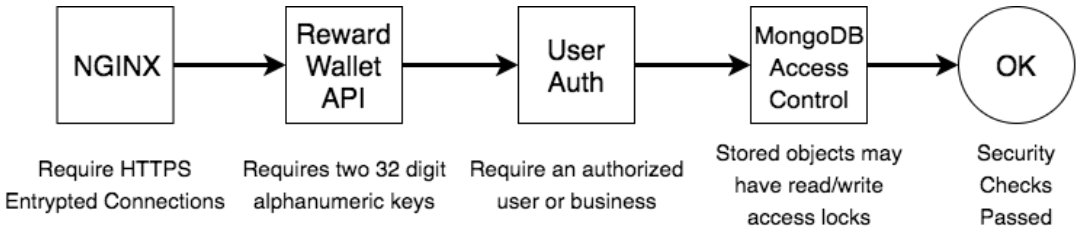


Figure 4: Security Validation Layers.

The first layer is the security that the NGINX web server provides. As previously mentioned, all incoming and outgoing connections to the RewardWallet API are protected behind the industry leading NGINX web server. By having all connections be sent through HTTPS the contents of the data are encrypted. This helps to prevent a hacker from viewing the internet traffic on a local network where they would otherwise be able to see users' login information and API keys. Before routing a connection to the RewardWallet API NGINX first tries to upgrade the connection to HTTPS before otherwise denying the request.

The second layer is the requirement of two 32-digit alphanumeric keys, sent as headers in the HTTPS connection. Without valid keys the API will refuse the incoming connection. Of the two keys, the first is an ID that must be matched and the second is to determine the overall access level the connection will have to the data. Subsystems such as the mobile app and embedded system are given client keys which require the next security layers to be passed. Alternatively, a master key can be passed which will bypass the user authentication and MongoDB access

control. This key is to be used strictly by the engineers at RewardWallet to perform maintenance or validation checks.

The third and final security layers work closely together. Objects stored in the MongoDB database can be given different access control settings which restrict read/write access. The level of read/write access a connection has is determined by the user authentication security check. Users must be validated with the email and password they used upon signup. By default, we have imposed that all transaction records only have read access to protect their validity. The only way they can be edited is by using the master key mention previously. This is the key that the RewardWallet API's cloud functions use to process and create transactions. This design helps to enforce that all computation be done on the cloud system, as the mobile app and embedded system don't have access, while also protecting data validity.

By enforcing these security layers, we require that transaction requests follow the "handshake" design shown in Table 1: Reward Collection Roadmap. It can be noted that in that roadmap all processing is done by the cloud system and requires an authenticated business and user to be successful.

5.4 Standards

Google's API Design Guide [6] has been adhered for the design of this system in addition to IEEE Engineering standards which are listed in the *Engineering Standards* section.

5.5 Reliability Design Considerations

Like most small companies, RewardWallet will be renting virtual machine instances from a cloud platform provider, such as Google, Amazon and Microsoft. This approach offloads all of the maintenance and reliability engineering for keeping a server operational while also drastically reducing RewardWallet's initial startup costs. Cloud platform providers will help to ensure uptime reliability, our team at RewardWallet just needed to ensure that the API we develop can reliably run on the server.

Because businesses can operate 24/7 we needed to ensure that RewardWallet's API had 99.9% uptime. This uptime means there should only be a max of 8.75 hours [7] that the API is unreachable per year. This can pose problems on Node.js applications, the framework being used to make the API, as it heavily consumes system resources. At peak levels a lack of available resources would cause a Node.js application to crash, a problem that could occur while RewardWallet is scaling in size and needs a more powerful server. To resolve these risks our engineering team will implement programs that monitor the API and ensure that is restarted if there is ever a crash. These programs will also automatically start the API after the server is restarted so that the API goes online after the cloud platform providers applies any maintenance.

5.6 Performance Design Considerations

For the RewardWallet's system we have chosen a design that puts all computation on the cloud system. This will help the overall performance of the system as the embedded system and

mobile app do not have to be high performing devices, nor do they need to have a strong data connection since only tiny amounts of data are sent to the system, rather than several updated objects. Devices will be able to make requests to the server at which point they are put in a synchronous queue for processing. While it is ok that the servers responses can be asynchronous, by having synchronous modification of the database data corruption is avoided without implementing a complex asynchronous queue. A visual representation of this model can be seen below.

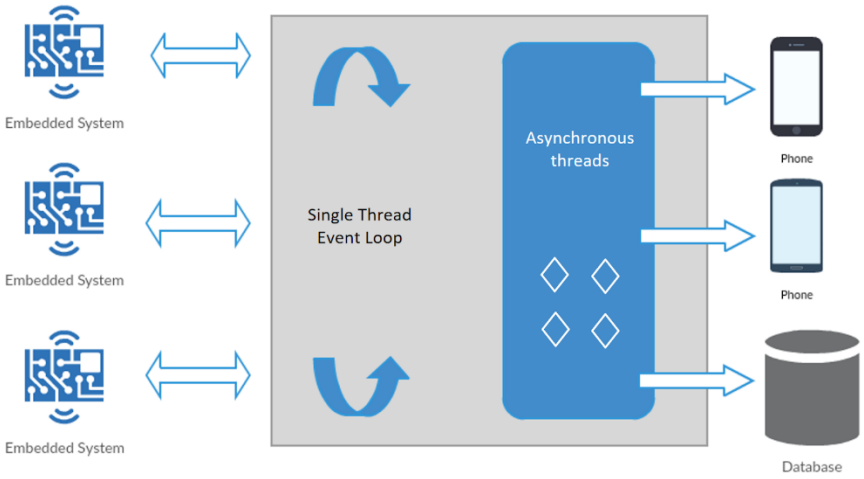


Figure 5: Cloud System Performance Diagram.

For this design, increasing the performance of the cloud system can be done by increasing the available computing power as needed, a task easily executable in any of the major cloud service providers such as Google, Amazon or Microsoft. This would help ensure that transactions are processed within seconds and the users will be able to receive confirmation as fast as possible. By utilizing an asynchronous architecture for responses and data storage high-performance can be achieved while protecting the sequencing of those actions with a synchronous event queue.

5.7 Usability Design Considerations

The key advantage our system has is that it is customizable to any business by allowing the business owner to choose the distribution model of their choice. We also want it to be easy for a business owner to go online and view a list of transactions that have been made at their establishment where they can see everything from the purchase amount, the time it was made and the customer who collected the rewards. Upon release of RewardWallet we aim to provide a high-quality dashboard to business owners where they can view transaction history, analytics, edit their distribution model and send targeted advertising to their customers. Unfortunately, this high-quality dashboard is out of scope for the initial prototypes.

6 Mobile App Design

6.1 System Overview

The Mobile app is the quickest and convenient way to collect and redeem user's rewards. The Mobile app receives transaction records through NFC and allocate rewards to users, and the mobile app client will just display and send data to the server for processing.

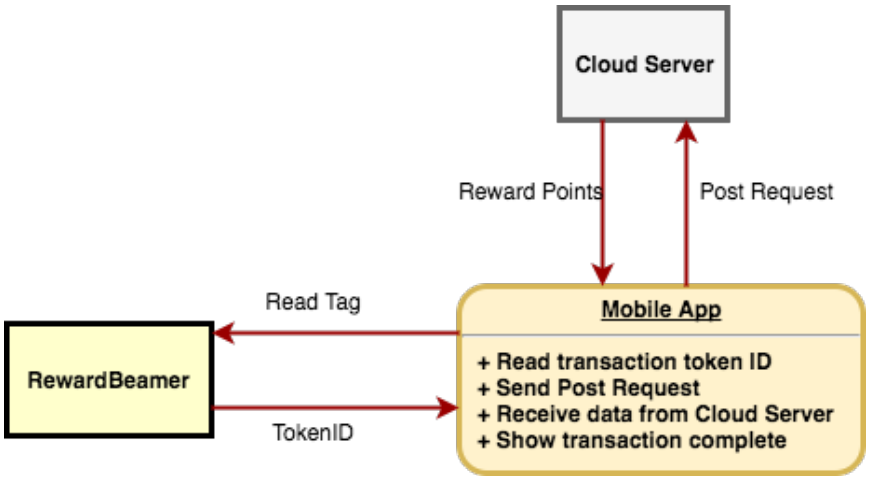


Figure 6: Control flow.

6.2 General Design

The platform of the mobile app is iOS developing by IDE XCode version 9.1.1 or newer on a Mac OS X machine. The reason of choosing iOS are desirable quality and enhanced user experience. XCode allows us to focus more on development instead of compatibility of application running on various devices.

Table 2: XCode, Swift and IOS versions with device compatibility.

XCode version	Swift version	iOS version	Devices Compatibility
9.1	4.0	iOS 11.0+	• iPhone 7 • iPhone 7 Plus • iPhone 8 • iPhone 8 Plus • iPhone X

6.3 App Security

Apple has designed the iOS platform with security at its core, and iOS and iOS device provide advanced security features which enabled by default. iOS provides layers of protection to ensure that apps are signed and verified and are sandboxed to protect user data. iOS has additional encryption and data protection features to safeguard user data. Apple uses a technology called Data Protection to further protect data stored in flash memory on the device [8]. These elements provide a stable, secure platform for apps, and users can access the app on their iOS devices without undue fear of viruses, malware, or unauthorized attacks. RewardWallet as a company have registered with Apple and join the Apple Developer Program, and the app will be reviewed by Apple to ensure for security and obvious bugs problems.

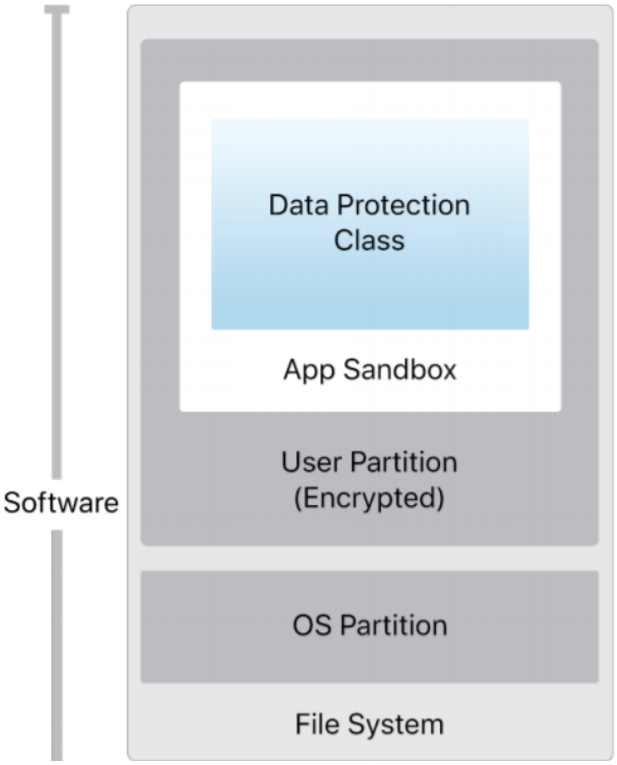


Figure 7: Security architecture of IOS.

6.4 Standards

The app must follow the guiding principle of the App Store to provide a safe experience for users to use. User interface of the app shall follow iOS Human interface guidelines [4].

6.5 Reliability and Performance Design Considerations

To ensure users get the most out of our app, as mentioned in section 6.2, RewardWallet app should be a universal app that users can use them on all of their devices. One major factor in performance design is the use of hardware compatibility of the iPhone device. RewardWallet will

confirm that our app does not contain any functions that would cause infinite loop to rapidly drain battery or generate excessive heat. Also, the mobile app will install CocoaPods that improves app launch performance by merging frameworks. One of the framework will be used in WalletView for having smooth card-like transition is DeckTransition. Smooth animation transitions will provide positive user experiences.

Similar to RewardBeamer and the cloud server, the app will also focus on the reliability of the product. Our develops will include all the possible outcomes and implement an exception handling strategy. This strategy will allow the application to catch and report an error and its detail moment before the application crashes. Moreover, the app will request a reward in the background and then update in real time. It will receive the tokenID sent by the RewardBeamer in real time. If the app does not receive the transaction ID, the app will keep trying to read the RewardBeamer until it receives the ID.

6.6 Usability Design Considerations

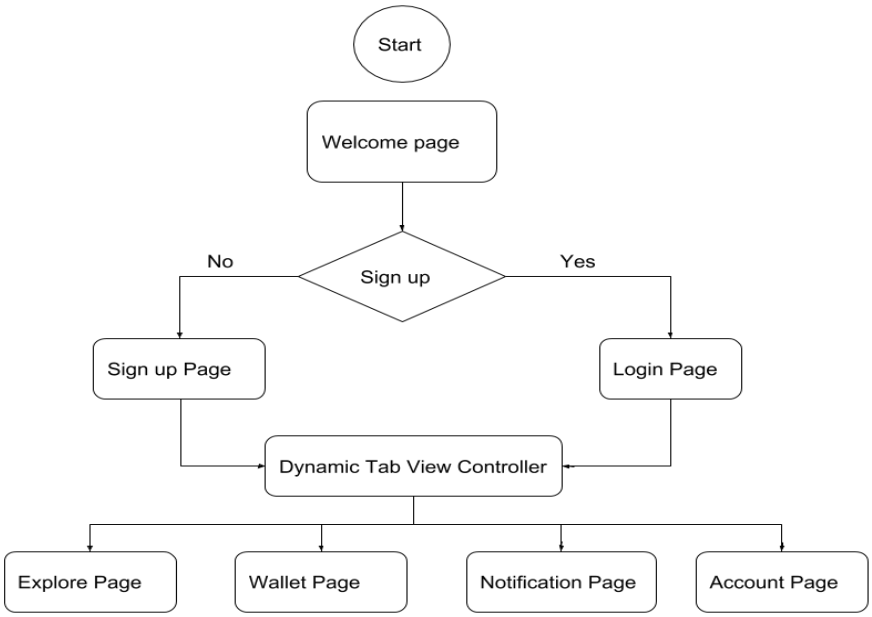


Figure 8: Flowchart of usability design.

The app shall be able to read NFC data when placing the iPhone on the top of RewardBeamer. This can be achieved by importing NFC core released on iOS 11.0. The Core NFC API from Apple has finally opened up the possibilities for NFC integration on iPhone devices. Another functional requirement is that the app shall be able to send requests to cloud system to assign record to users. This will be achieved by using HTTP method, POST and GET to communicate with cloud server. Afterwards, the app will receive an update point balance from the cloud server and display it to the user. The way to implement the function is to create the list of cards in a wallet page that contains the company name, and its reward points. The record of the transaction will pop up by an alerting view for noticing users to confirm and then save it into

notification view which contains promotions and history transaction. The app shall provide register page for users to sign up. This requirement can be achieved by creating a user model that contains email address and password.

Sign Up Page	Wallet Page	Notification Page
- Email Address **- Password**	- Card Name - Reward points	- History of transaction - Business's promotions
+ Sign up **+ Cancel**	+ Add Card + Select Card	

Table 3: Page data and method.

7 ENGINEERING STANDARDS

The Embedded System will comply with the IEEE Std 1003.13 standards for portable Realtime and Embedded Applications [15].

The Embedded System and mobile device will comply with the IEEE 1625 standards for Batteries in Mobile Computing Devices [16]. The standards ensure safety of the system devices under environments.

The iOS application will follow the commonly used Ray Wenderlich Swift coding style guideline [18]

The Cloud Server will comply with the IEEE Std 2410 standards for Biometric Open Protocol [17]. The standards ensure security regarding cloud communication with devices.

The Cloud Servers API will follow the industry leading Google Javascript coding style guideline [18]

The Cloud Server, Embedded System and mobile application will comply with the IEEE 802.15.5 standards for Telecommunications and information exchange between systems [18]. The standards ensure security of component communications using wireless networks.

The Cloud Server will comply with the Google Cloud Platforms API Design Guide [16]. The standards ensure security proper coding conventions.

8 CONCLUSION

The three main subsystems of the RewardWallet system are the Embedded System, Cloud Server and Mobile Application. Each of these subsystems are comprised of heir component-specific design specifications. The system is designed for reliability and ease to use. The complete system design detailed the importance of communication between each subsystem. The RewardBeamer, an embedded system running on a Raspberry Pi microcontroller, serves as a bridge between the existing POS and the mobile application. Furthermore, the Cloud Server determines the appropriate amount of reward allocation based on unique business distribution models. The Mobile Application is designed with elegance and updates each user accounts with points allocated or redeemed.

In the General Design Considerations section, Engineers specified the technicalities in the design of each subsystem. The Embedded System is equipped with the capabilities to perform network requests for initiating transaction and transmitting records through NFC. It is also able to emulate tag for the Mobile Application to read the transaction identifications. The Cloud Server enables business clients to execute their customized distribution models using NO-SQL such as MongoDB. The Mobile Application is developed in XCode IDE using Swift 4.0 in order to provide users with the best and newest mobile application technology.

Security requirements are also discussed in detail as the system contains highly sensitive information for both the business clients and their customers. The Embedded System keeps all sensitive information private, and any manipulation of components are done through the Cloud Server. Access to the Cloud Server requires passing a four-step security validation layer. Contents sent through HTTPS are encrypted, in addition to the two 32-digit alphanumeric keys required to complete validation. User authentication is required for access to the Mobile Application. The application does not store any sensitive information regarding its user and/or businesses.

The design of the RewardWallet system closely follows all related Engineering Standards. The prototype will be manufactured using the design specifications described in this document.

9 GLOSSARY

API: Application Programming Interface, is a set of instructions or protocols.

HTTPS: Hypertext Transfer Protocol for secure communication over a network.

RESTful: Representational state transfer web services architectural design protocol.

RewardBeamer: An embedded system running a small API that can be used to control reward allocation through its NFC module.

Distribution Model: The customizable model businesses can tune to determine how they wish to give rewards to customers; for example: cash back of 5% or 10 purchases for a free item.

POS: Point-of-Sale, referring to the instance a transaction takes place between a business and a customer.

Cloud Server: A logical server that is build, hosted and delivered through a cloud computing platform over the internet.

Embedded System: Combinations of hardware and software that perform a specific function within a larger system.

NFC: Near-field communication is a set of communication protocols that enable two electronic devices to establish communication by bringing them closer of each other.

Prototype: A first preliminary model of something.

QR Code: the trademark for a type of matrix barcode that is a machine-readable optical label contains information about the item.

Cloud Function: A function that is executed on the server through the request of a client application.

POST: An HTTP request method, normally to a web server to accept the data enclosed in the body of the request message.

10 APPENDIX I - TEST PLAN

10.1 UNIT TESTING

10.1.1 Embedded system

Table 4: Embedded System Unit Test.

Tests	Steps/Expected Result	Validation(Pass/Fail)
Server Communication	Send network requests to the Cloud Server for initiating a transaction. Receives a transaction token.	
Mobile Communication	On IPhone, open NFC Test Suite and place IPhone near device. The mobile device is able to read tag emulated by the RewardBeamer.	

10.1.2 Cloud

Table 5: Cloud Server Unit Test.

Tests	Steps/Expected Result	Validation(Pass/Fail)
Create User	Pass in a username and password as arguments; success should return an ID for the new user	
Login User	Authenticate a user with a username and password; returns an auth token	
Open Transaction	Pass a transaction amount and business ID; returns an ID to the created transaction object if a valid business	
Close Transaction	Pass a transaction ID and user ID; allocates rewards to the user for the business based on the businesses distribution model; returns an updated point balance	

Redeem Points	Pass a user ID, business ID, and a point balance to deduct points from the users digital card for the business; returns an updated point balance or an error when insignificant points	

10.1.3 Mobile app

Table 6: Mobile Application Unit Test.

Tests	Steps/Expected Result	Validation(Pass/Fail)
Sign Up Functionality	Ask Users to sign up with their valid email and password. It will require users to re-enter if their email and password are invalid.	
Login Functionality	After Users enter their email address and password, the app checks with the database. It will require users to re-enter or sign up if their email and password are not match.	
Show WalletView	After users successfully log in, a list of company cards will be displayed; each company card can have selected for details	
Read RewardBeamer	Place the iPhone near RewardBeamer, and check if the transaction ID is received; app should show a check mark once it receives.	
Update Reward Points	After the cloud server sends back the update reward points, the app should check for the recent reward balance; an updated reward should appear on the Card.	
QR Code	Open a transaction; generate a QR code for scanning; returns an updated reward points	

10.2 Integration testing

Table 7: Integration Test.

Tests	Steps/Expected Result	Validation(Pass/Fail)
NFC communication	Place the iPhone near RewardBeamer, and two devices should be able to establish communication	
Send POST request	The app should send POST request with the transaction ID when it receives from the RewardBeamer; the cloud should be able to receive the request	

10.3 System testing

Table 8: System Test.

Tests	Steps/Expected Result	Validation(Pass/Fail)
Test Transaction	Performs all the Cloud Tests in sequence with fake user data and the introduction of random error to test error response and simulate high server load when continuously executed; should not crash the server	

11 APPENDIX II - USER INTERFACE

11.1 INTRODUCTION

The User Interface Design Appendix provides a description and analysis of Customizable Reward Allocating System in terms of its design for and communication with its desired users. The main purpose of the UI design focus on the user's interaction and make it as simple and efficient as possible. In this appendix, the outline of the required user knowledge that allow them to use our RewardWallet system interact with our mobile app will be presented. The following analysis will be considered when making design choice for the UI: discoverability, feedback, conceptual models, affordances, signifier, mappings, constraints. Finally, the appendix will include the system test plan with three different scenarios that is aimed at testing how RewardWallet system can be adapted to any business.

11.2 USER ANALYSIS

11.2.1 IOS Application

The mobile app that customers use to collect points acts as a marketing tool for RewardWallet business. The product is targeting customers at all ages and both genders. As expected from the user, the mobile app is designed to deliver a great quality services for effective and enjoyable use. To achieve that goal, there are certain things the user should know to operate the device successfully.

The users need to own a smartphone that runs an updated version of iOS operating system. The system is required to have simple user interface with minimum signing up steps for users. It should be easily understood by users with the central unit menus and navigation buttons. The user interface shall follow the standard and common patterns such that users can easily identify the usage of the product. The users should be able to know how to download an iOS app use their iPhone and to wireless tap for reward using NFC.

11.2.2 Embedded System

The RewardBeamer has a target user of small business owners and cashiers. The device is intending to have an easy setup coupled with their existing POS systems. The users of this device require to have basic knowledge of powering on electronic devices. Furthermore, the users also have to have an understanding of their current POS system in order for the add-on to be used in conjunction.

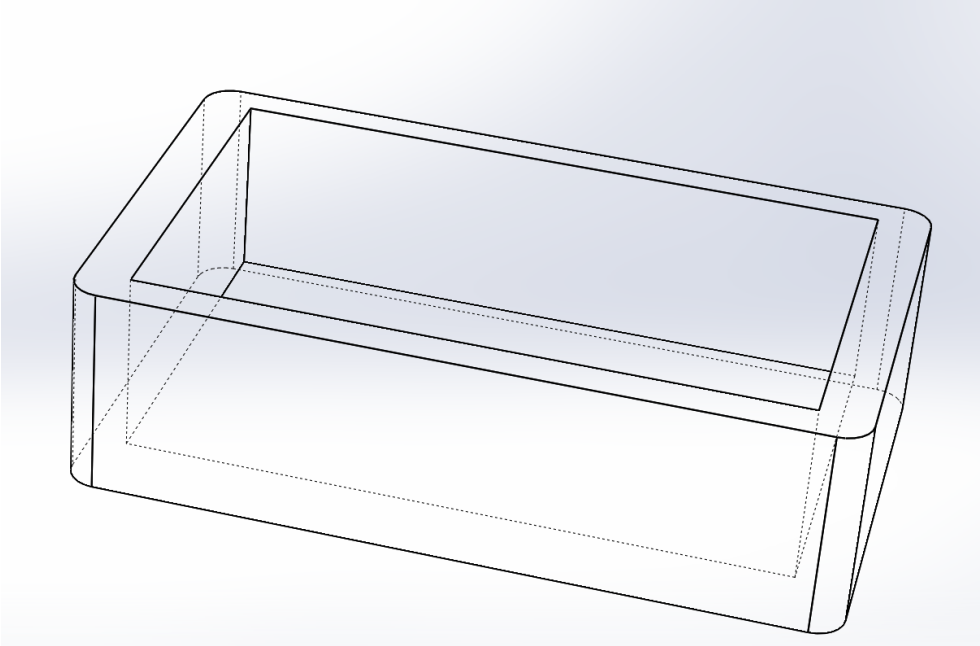


Figure 9: RewardBeamer container. Dimensions: height 15cm, width 20cm, length 35cm.

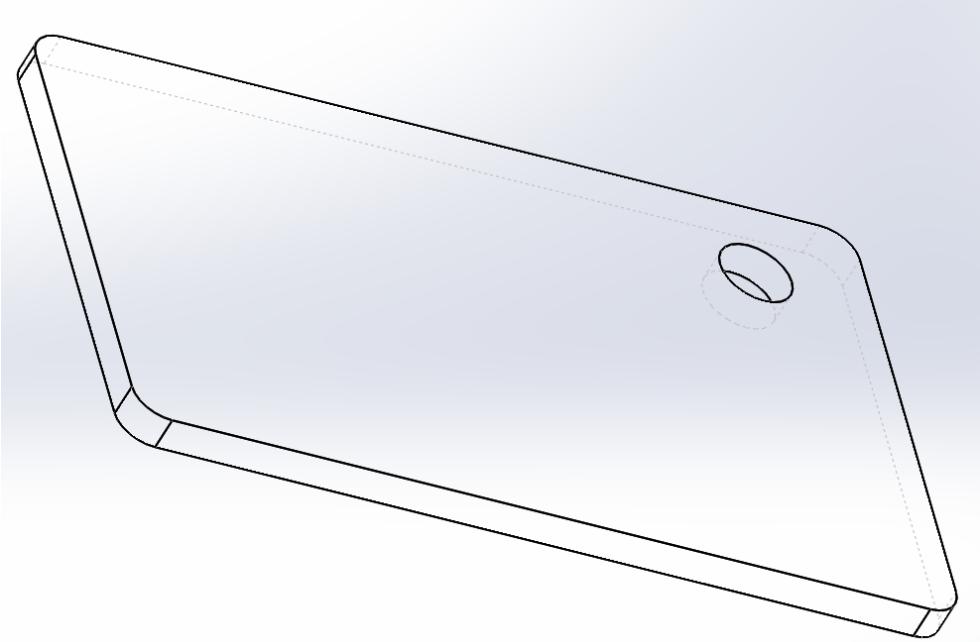


Figure 10: RewardBeamer lid with LED cutout. Dimensions: height 1cm, width 20cm, length 35cm, cutout radius 0.3cm.

11.3 Tech Analysis

According to The Design of Everyday Things proposed by Don Norman, this section will be divided into 7 subsections which follows the Seven Elements of UI Interaction [19]. These 7 principles ensure that the final product is easy to use and understand, leading to a effective interacts and communicates with the user.

11.3.1 Discoverability

Users will interact with the iOS app, which includes the standard navigation bar in the top, guiding users in how to use them. The icons and texts labeled in the bottom tab bar indicate to the users the different of operation mode. The RewardBeamer will be labeled appropriately to indicate the area of the NFC chip and where the user should tap their phone to collect rewards.

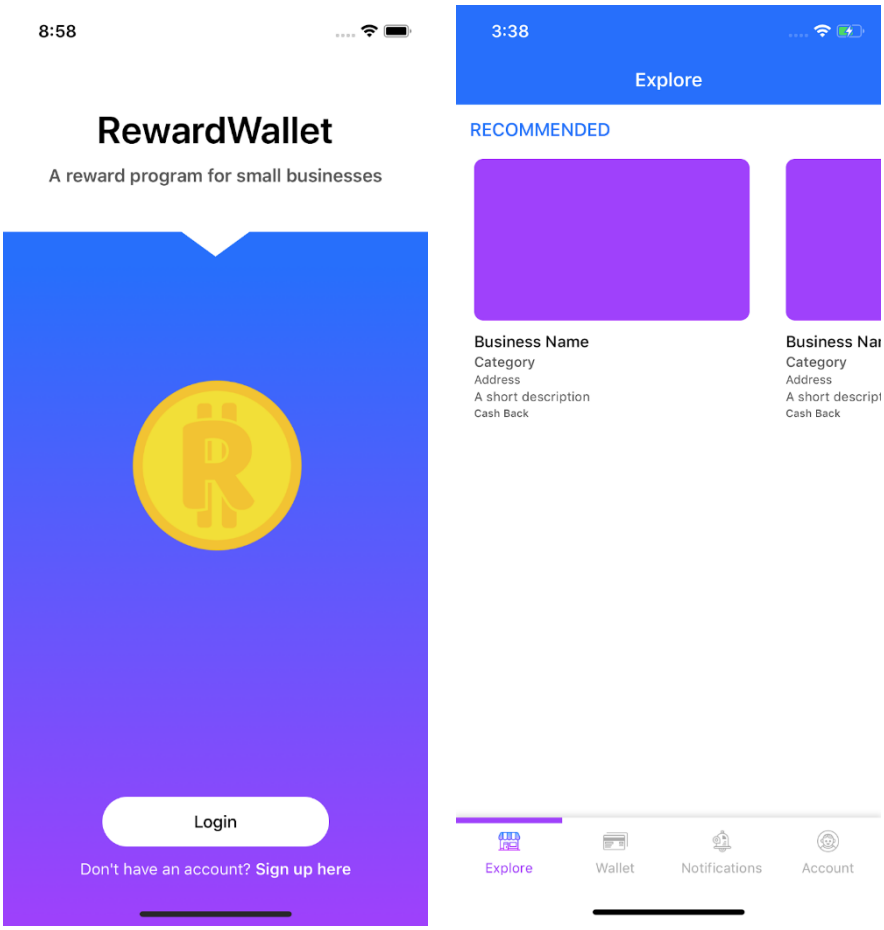


Figure 11: Welcome Splash Screen and Business Explore View.

11.3.2 Feedback

In the mobile app, upon entering the user email and password for logging in, the user is asked for a confirmation with a button labeled. The app will alert the user if their email and password are unmatched based on the data stored in the cloud. In main page of the mobile app, upon searching for the specific stores or nearby restaurants, the user is receiving a list of results. In RewardWallet system, upon placing the smartphone to the RewardBeamer and opening a

transaction, the users is receiving a notification of the transaction result. When the transaction is completed, the new balance of the reward is updated.

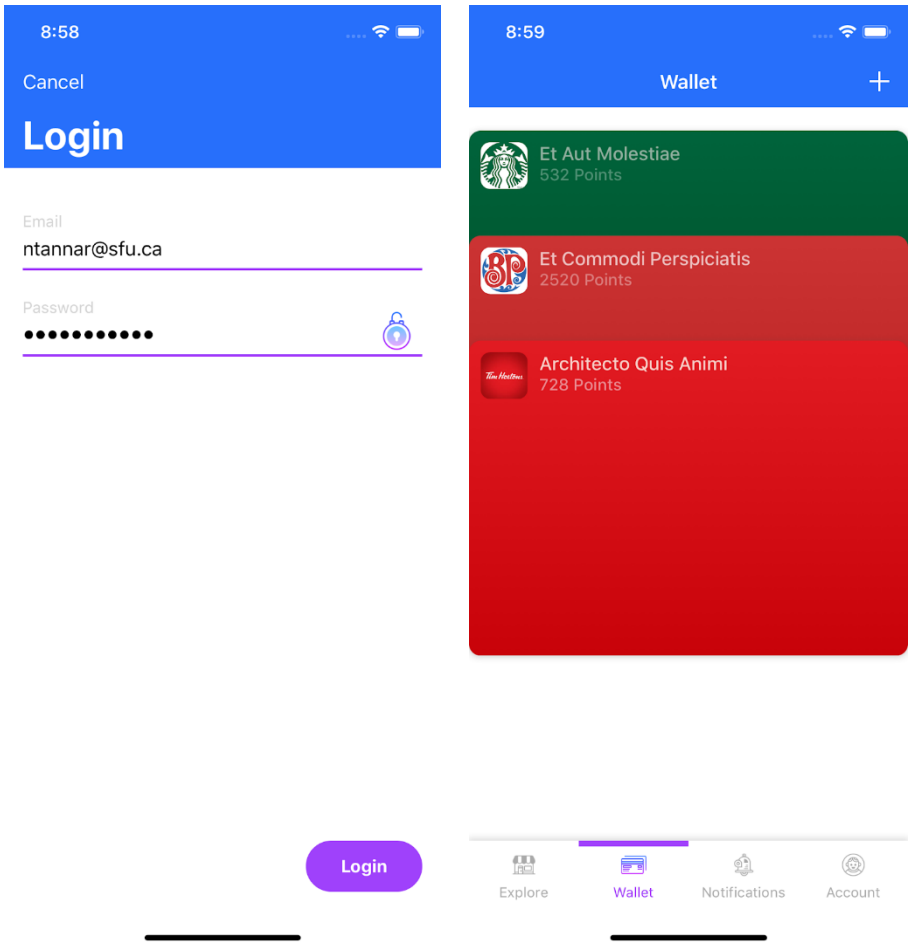


Figure 12: Login Screen and Digital Wallet View.

11.3.3 Conceptual Models

In the mobile app, the icons in the tab bar provide a good conceptual idea that help users to understand certain functionalities.

11.3.4 Affordances

The user interface for the smartphone app is designed for such that allowing users instantly identify the interactive elements on the screen, and accurately predict the result of their interaction. When users perform the transaction, they can easily place their smartphones near the RewardBeamer for allowing the NFC chip on the smartphone to read NFC tag.

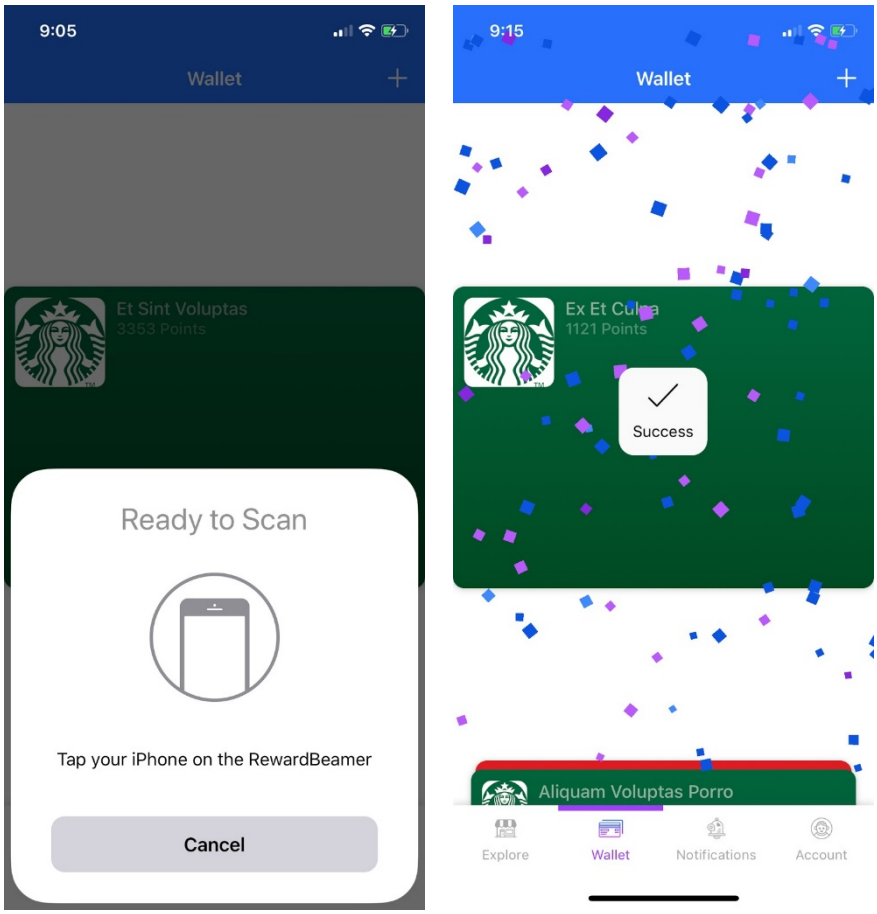


Figure 13: NFC Reward Collection Screen and Collection Confirmation Screen.

11.3.5 Signifiers

In the smartphone, the text that notify users who has not signed up for the app should sign up before using it. For the current design stage, there will be a LED light act as a signifier that indicates the transaction is completed.

11.3.6 Mappings

In the smartphone, the tab bottom line color has a strong mapping which indicates the currently selected tab. A successful completed transaction is mapping to the notification of a popped check mark.

11.3.7 Constraints

Using the latest view controller advancements in UIKit to make it easier to adapt user interface of the app to any size or orientation. XCode's Interface Builder can help automatically to create all the constraints, ensuring a set of compatible rules [22].

User is only allowed to perform one transaction at a time through the smartphone app. Transactions should not only execute in a logically correct manner, but also meet certain constraints with regard to their completion times.

11.4 Engineering Standards

The iOS application will follow the commonly used Ray Wenderlich Swift coding style guideline [20].

The RewardBeamer's programming will follow the industry leading Google JavaScript coding style guideline [21].

11.5 System Test Plan

During the ENSC 405W poster presentation in April 2018, three different use case examples will be demoed. Each use case will simulate different businesses each with their own distribution model. This will show the robustness of our system and how it can be adapted to any business.

In each scenario, a test user will be presented with a phone preloaded with a reward card for the business. Using a laptop to simulate a POS a HTTP POST request will be sent to the RewardBeamer which will allow the test user to tap the phone on the NFC tag to collect rewards.

11.5.1 Scenario A

In this scenario the business is a small coffee shop. They have decided to integrate RewardWallet into their business and have decided to go with the cash back distribution model of 5%. They chose this to promote that buying a more expensive latte should yield more points.

Distribution Model: Cash Back - X% of each transaction appears as points to the customer

Given the 5% cash back model, if the HTTP POST request indicated the transaction amount was $3.63, the test user should expect to see an additional 18 points added to the digital card.

Distribution Model	Transaction Amount	Reward Points
5% Cash Back	$3.63	18

11.5.2 Scenario B

In this scenario the business is a small pizzeria. They have decided to integrate RewardWallet into their business and have decided to go with the purchase-based distribution model of 1 point. They chose this as they wanted a simplified system where after 5 purchases customers could enjoy any pizza for free.

Distribution Model: Purchase Based - X points will be given to the customer for each transaction

Given the 1-point purchase-based model, if the HTTP POST request indicated the transaction amount was $14.23, the test user should expect to see an additional 1 point added to the digital card. It will then be shown that a transaction amount of $18.45 will also yield only 1 reward point.

Distribution Model	Transaction Amount	Reward Points
1 Point / Purchase	$14.23	1
1 Point / Purchase	$18.45	1

11.5.3 Scenario C

In this scenario the business is a large grocery store. They have decided to integrate RewardWallet into their business and have decided to go with the quota-based distribution model of 25 points for every $250. They chose this as they wanted to digitize their “$25 gift card for every $250 spent” flyer ad.

Distribution Model: Quota Based - X points will be given to the customer if they purchase $Y or more

Given the 25-point quota-based model, if the HTTP POST request indicated the transaction amount was $204.93, the test user should expect to see no additional points added to the digital card. It will then be shown that a transaction amount of $268.45 will yield 25 reward points and $512.76 will yield 50 reward points.

Distribution Model	Transaction Amount	Reward Points
25 Points / $250 Purchase	$204.93	0
25 Points / $250 Purchase	$268.45	25
25 Points / $250 Purchase	$512.76	50

11.6 Conclusion

This UI appendix outlines the different aspects of RewardWallet, including the required user knowledge and restrictions and the different elements of UI interfaces of RewardWallet. It also includes the system test plan regard the core interface and user interaction with three different business models. At this stage of the project, users can start to perform a transaction and collect rewards using NFC chip on their phones. In the next development stages, the system will be subject to testing with different types of users and feedback will be considered to improve the robustness of our system, minimize the update of collecting reward point delay, and strengthen the overall quality of the final product.

12 Appendix III – 440 Planning Document

12.1 Introduction

In today's business world, companies are utilizing the latest technology to better service their customers and growth their businesses. Most franchises create their own mobile apps that allow their customers to collect rewards and/or redeem coupons. Collecting a reward for purchasing an item is another example of instant gratification and draws customers to revisit or prefer a particular business. However, this puts smaller businesses in a disadvantage because they do not have the resources to build their own reward system. [23] Our customizable product is the system that small businesses need. By providing a low-cost solution, businesses can be empowered with a system that is customizable to their needs and deliver instant gratification to their customers.

"All good businesses are personal. The best businesses are very personal. " - Mark Cuban

The goal of this solution is to equip businesses with the tools to make them more personal and add value to a customer's purchase. By adding points customers are more likely to return to build up points and one day redeem them for a free item. Furthermore, businesses can view a history of customers who stopped by and send them personalized coupons or promotions to increase their own sales. This adds a more personal touch to marketing that can be more impactful than generic coupon flyers.

The development team behind RewardWallet can deliver this system with their combined knowledge in embedded system design, network protocol design and software engineering. With the continuous effort of this development team, such solution will be presented in August 2018.

12.2 Competition and Market Rationale

12.2.1 Market Rationale

Businesses without a loyalty program are at a huge disadvantage. According to an online survey done by more than 30,000 customers, 72% of them agree they would buy from a retailer that offers a loyalty over one without. 67% of the participants also agree that retailers who offers loyalty program will make them visit more frequently and spend more. [26]

For small businesses, hiring a development agency or freelancers to build a custom app can reach $100,000 [24] plus maintenance costs. This results in owners either opting out of offering a loyalty program or reverting to some kind of physical card with special stamps. [27] This makes it exceptionally challenging for them to compete with a loyalty program of their own against large corporations.

An end-to-end solution like RewardWallet can help small Canadian businesses keep up with larger corporations. The consumers want to see loyalty rewards and as of now the market for a solution like this is still relatively untouched. Businesses just need an easy to use solution that does not require any maintenance or technical knowledge, such as RewardWallet.

12.2.2 Competition

Although there are numerous reward programs in the market that offers customer loyalty program for small and medium businesses, most of them are not competitive due to their lack of business enrollment, app functionality, or customer engagement. However, Belly, FiveStars, and Square Loyalty are the three most well received competitors to our system.

Belly's iPad-based system records visits via QR code to collect points and pricing starts at $159 per month with higher price points offering POS integration and targeted customer acquisition. [28] Even though Belly is an easy to use app for store owners, collecting visits alone does not enable business owners to distinguish customers from spending levels.

FiveStars has POS integration of most existing POS systems and offers a full suite of features. However, with the starting price of $279 per month, it is a huge burden on small and medium businesses. [29]

Square Loyalty works with Square mobile payment system. For $25 per location per month, it is excellent for small businesses who are using Square. However, it does not provide reviews and listings of all the businesses nearby that uses Square Loyalty and therefore provides less visibility to potential customers. [30]

12.3 Risk Analysis

Table 9: Scale of Risk Level.

5 Severe	If a risk event occurs and has this rating, there will be a severe impact on the progress of the project. One or more of the critical outcome objectives will not be achieved
4 Significant	If a risk event occurs and has this rating, there will be a significant impact on the progress of the project. One or more stated outcome objectives will fall below acceptable levels
3 Moderate	If a risk event occurs and has this rating, there will be a moderate impact on the progress of the project. One or more stated outcome objectives will fall below goals but above minimum acceptable levels
2 Minor	If a risk event occurs and has this rating, there will be a minor impact on the progress of the project. One or more stated outcome objectives will fall below the goals but well above the minimal acceptable levels

1 **Minimal**	If a risk event occurs and has this rating, there will be a minimal impact on the progress of the project. There will be little or no impact on achiever outcome objectives

Table 10: Risk, Severity and Likelihood of Occurrence.

Severity	Risk	Likelihood of Occurrence
3	**Shipping delays to required infrastructure** • Delays to infrastructure such as hardware or software	35%
5	**Failure to integrate with system** • The risk that our product will fail to fit into the existing system	30%
2	**Testing takes more time than expected** • Identify the risk that specific testing need to be done before implementing the next feature	50%
4	**Change functionality overload** • A large number of change requests dramatically raises the complexity of the project and distracts key resource	25%
2	**Underestimating the budgets** • Identify the risk of inaccurate cost estimates and forecasts	10%
4	**Vulnerabilities and Malware** • Consider the risk that new functionalities always come new threats	10%

3	**Market Viability** • Through market research a need for such a system as this is believed to be wanted; however, the success will depend on businesses actually enrolling	35%
5	**User Information Breach** • A breach renders the integrity of the system useless and would dramatically harm business-to-business relations and user trust	5%
2	**Unintuitive Design** • Consider the risk that users find the app/system unintuitive and delete the app. Without users it's harder to enroll new businesses	30%
3	**New Competition Beats us to Market** • Another company could see the market potential and beat us to market, increasing the competition	15%

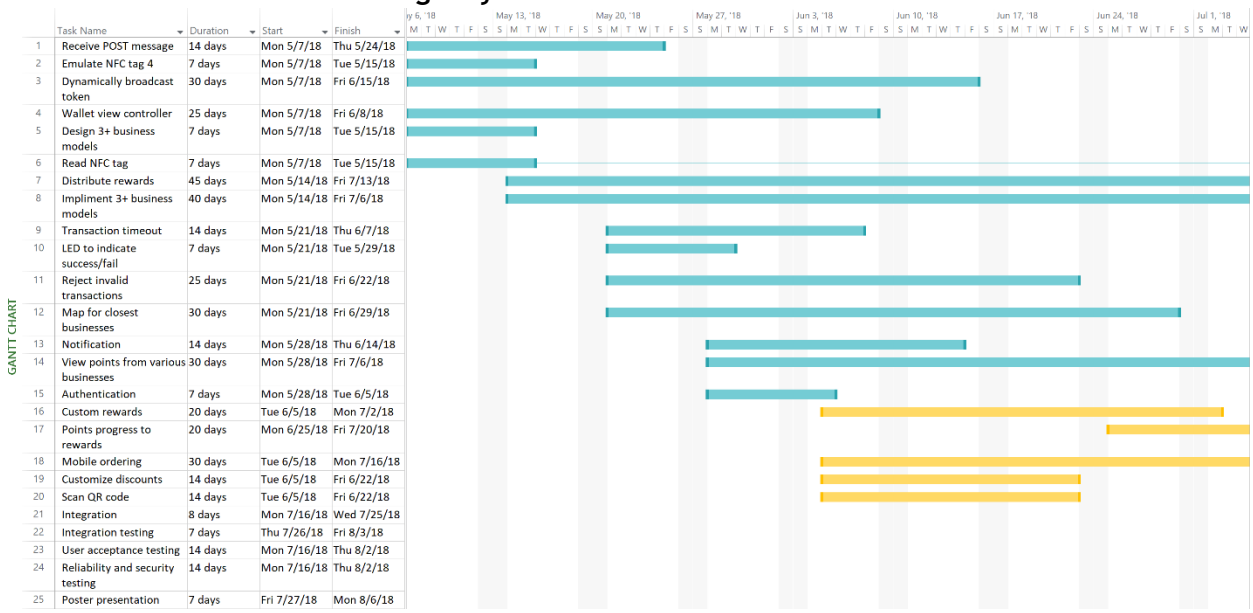
12.4 Personal Task Allocation

Proof-of-concept will be presented on April 9th, 2018. Then, engineers of RewardWallet will gather all feedback provided from professors and teaching assistants in order to improve and advance the RewardWallet system prototype in ENSC 440. In the second stage of our product development, we will focus on optimizing the existing system and increase product flexibility. Team members of RewardWallet will be assigned to various tasks throughout the development of this product.

- Nathan Tanner will be responsible for the development of the iOS application and Cloud Server.
- Molly Bin will be responsible for the development of the iOS application.
- Mandy Xiao will be responsible for the development of the RewardBeamer.

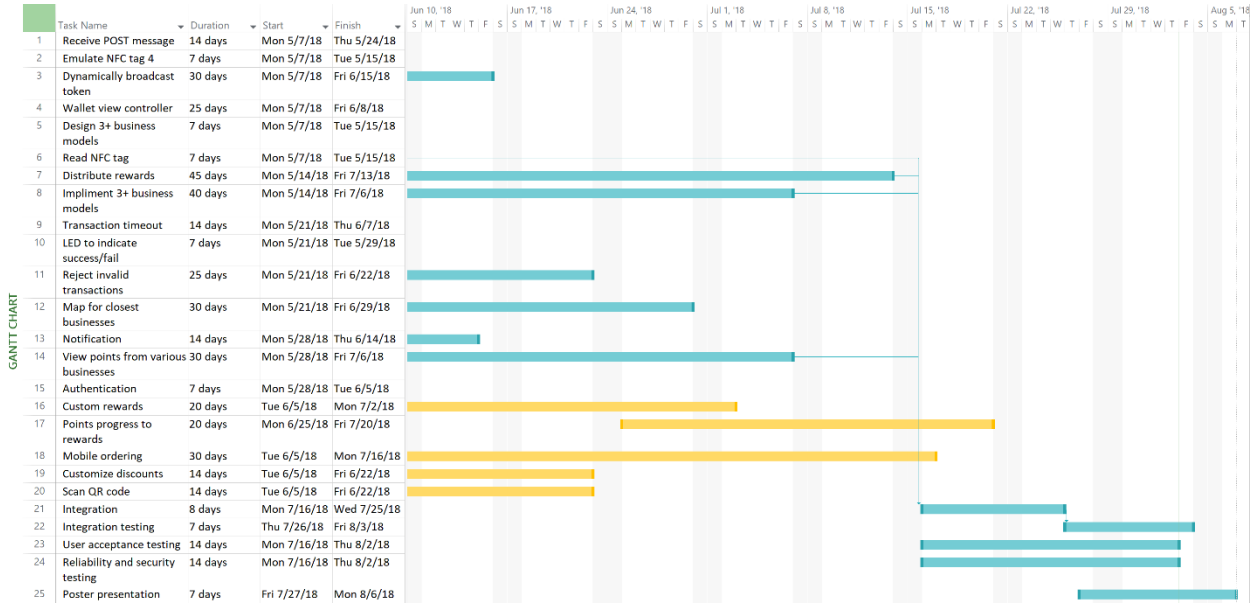
12.5 Time Management

12.5.1 Gantt Chart and Contingency Plan



	Task Name	Duration	Start	Finish
1	Receive POST message	14 days	Mon 5/7/18	Thu 5/24/18
2	Emulate NFC tag 4	7 days	Mon 5/7/18	Tue 5/15/18
3	Dynamically broadcast token	30 days	Mon 5/7/18	Fri 6/15/18
4	Wallet view controller	25 days	Mon 5/7/18	Fri 6/8/18
5	Design 3+ business models	7 days	Mon 5/7/18	Tue 5/15/18
6	Read NFC tag	7 days	Mon 5/7/18	Tue 5/15/18
7	Distribute rewards	45 days	Mon 5/14/18	Fri 7/13/18
8	Impliment 3+ business models	40 days	Mon 5/14/18	Fri 7/6/18
9	Transaction timeout	14 days	Mon 5/21/18	Thu 6/7/18
10	LED to indicate success/fail	7 days	Mon 5/21/18	Tue 5/29/18
11	Reject invalid transactions	25 days	Mon 5/21/18	Fri 6/22/18
12	Map for closest businesses	30 days	Mon 5/21/18	Fri 6/29/18
13	Notification	14 days	Mon 5/28/18	Thu 6/14/18
14	View points from various businesses	30 days	Mon 5/28/18	Fri 7/6/18
15	Authentication	7 days	Mon 5/28/18	Tue 6/5/18
16	Custom rewards	20 days	Tue 6/5/18	Mon 7/2/18
17	Points progress to rewards	20 days	Mon 6/25/18	Fri 7/20/18
18	Mobile ordering	30 days	Tue 6/5/18	Mon 7/16/18
19	Customize discounts	14 days	Tue 6/5/18	Fri 6/22/18
20	Scan QR code	14 days	Tue 6/5/18	Fri 6/22/18
21	Integration	8 days	Mon 7/16/18	Wed 7/25/18
22	Integration testing	7 days	Thu 7/26/18	Fri 8/3/18
23	User acceptance testing	14 days	Mon 7/16/18	Thu 8/2/18
24	Reliability and security testing	14 days	Mon 7/16/18	Thu 8/2/18
25	Poster presentation	7 days	Fri 7/27/18	Mon 8/6/18

Figure 14: Gantt Chart for 440 Part 1.



	Task Name	Duration	Start	Finish
1	Receive POST message	14 days	Mon 5/7/18	Thu 5/24/18
2	Emulate NFC tag 4	7 days	Mon 5/7/18	Tue 5/15/18
3	Dynamically broadcast token	30 days	Mon 5/7/18	Fri 6/15/18
4	Wallet view controller	25 days	Mon 5/7/18	Fri 6/8/18
5	Design 3+ business models	7 days	Mon 5/7/18	Tue 5/15/18
6	Read NFC tag	7 days	Mon 5/7/18	Tue 5/15/18
7	Distribute rewards	45 days	Mon 5/14/18	Fri 7/13/18
8	Impliment 3+ business models	40 days	Mon 5/14/18	Fri 7/6/18
9	Transaction timeout	14 days	Mon 5/21/18	Thu 6/7/18
10	LED to indicate success/fail	7 days	Mon 5/21/18	Tue 5/29/18
11	Reject invalid transactions	25 days	Mon 5/21/18	Fri 6/22/18
12	Map for closest businesses	30 days	Mon 5/21/18	Fri 6/29/18
13	Notification	14 days	Mon 5/28/18	Thu 6/14/18
14	View points from various businesses	30 days	Mon 5/28/18	Fri 7/6/18
15	Authentication	7 days	Mon 5/28/18	Tue 6/5/18
16	Custom rewards	20 days	Tue 6/5/18	Mon 7/2/18
17	Points progress to rewards	20 days	Mon 6/25/18	Fri 7/20/18
18	Mobile ordering	30 days	Tue 6/5/18	Mon 7/16/18
19	Customize discounts	14 days	Tue 6/5/18	Fri 6/22/18
20	Scan QR code	14 days	Tue 6/5/18	Fri 6/22/18
21	Integration	8 days	Mon 7/16/18	Wed 7/25/18
22	Integration testing	7 days	Thu 7/26/18	Fri 8/3/18
23	User acceptance testing	14 days	Mon 7/16/18	Thu 8/2/18
24	Reliability and security testing	14 days	Mon 7/16/18	Thu 8/2/18
25	Poster presentation	7 days	Fri 7/27/18	Mon 8/6/18

Figure 15: Gantt Chart for 440 Part 2.

Figures 12 and 13 detailed the project Gantt Chart for the latter four months of product development. The chart shows the estimated time for a particular task to be completed with enough spare time at the end of each tasks in case of unexpected delays. Furthermore, this time management plan minimizes dependencies amongst tasks in order to avoid delays being

propagated. One of tasks with the largest dependency is integration. All the main components of subsystems must be completed and tested before they can be integrated.

This leads into another contingency plan. Recall that in the Functional Specifications document for RewardWallet, we introduced a classification convention identifying the required deadline for each task. Here, we will use the same classification convention for consistency.

[Rn-p]
Where n is the numerical representation of said requirement and p is the priority level of the requirement. The priority levels are:

I. *Required for core architecture; modeled in the prototyped scheduled for April 2018*
II. *Required for production; modeled in the pre-production model scheduled for August 2018*
iii. *Required for post-production feature add-ons as software updates*

In Figures 12 and 13, the tasks indicated in blue are of priority Rn-ii. While tasks indicated in yellow are of priority Rn-iii. The blue tasks are of higher priority, so they are to be completed before any yellow tasks. In the case that there is a delay in the development of this product, engineers at RewardWallet will focus their efforts in completion of Rn-ii task. However, when development goes smoothly, and designated tasks are completed and tested ahead of their deadlines, engineers will start development of Rn-iii features. This allows for flexibility of time scheduling.

12.6 Milestone Chart

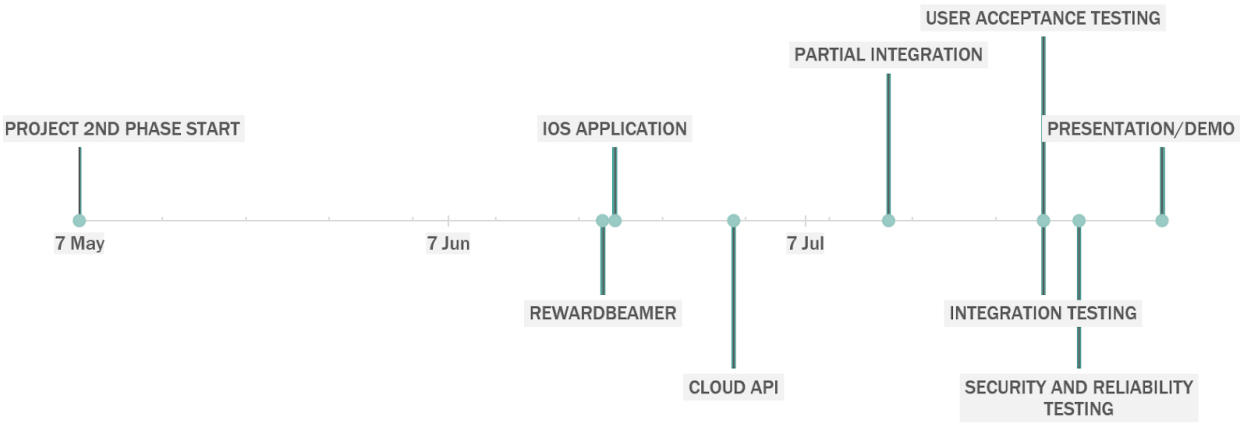


Figure 16: Milestone Chart in Timeline.

DETAILS

DATE	MILESTONE	POSITION
7-May	Project 2nd phase start	10
1-Jul	RewardBeamer	10
20-Jun	IOS Application	-20
21-Jun	Cloud API	-10
14-Jul	Partial integration	20
27-Jul	User acceptance testing	-10
27-Jul	Integration Testing	25
30-Jul	Security and reliability testing	-20
6-Aug	Presentation/Demo	10

Figure 17: Milestone Timetable.

Figures 14 and 15 demonstrated the major milestones of this project in timeline and table views respectively. Note that all key features of the three subsystems, RewardBeamer, iOS Application and Cloud API are to be completed and tested by late June and early July. This will allocate lots of time for integration, user acceptance and security testing.

12.7 Budget Management

12.7.1 Expenditures

The table below outlines some materials and tentative budget for the prototype. The budget also accounts for the appropriate 15% of contingency for any additional components that may needed during development.

Equipment List	Qty	Unit Price
Server fee	4 months	$10/month
Raspberry Pi Zero W	1	$56
Camera Module	1	$30
Miscellaneous Wires, LEDs and Resistors		$20

Estimate Shipping Costs		$40
Total		$ 186

Table 11: Budget

12.8 Funding

The initial cost in the design and implementation of this project may vary with the volume of development, and it is important to note the initial capital required will greatly exceed the cost of the finalized product. In order to ensure the success of the project and to achieve goals, it is necessary to seek early stage funding sources. We will prepare a presentation of our project to the Engineering Student Society Endowment Fund (ESSEF) and submit application to IEEE Canadian Foundation for Special Grants.

Furthermore, RewardWallet Inc plans to minimize the production costs. Some tools and components that are needed for the prototype come from previous lab kits. For the condition of our funding which does not cover all the costs, all team members agree to equally share the extra costs to ensure the success of our product.

12.9 Conclusion

RewardWallet can revolutionize small and medium businesses by providing them with state of the art technology to launch their own, highly customizable rewards point system. This innovative solution will give them an edge in competition with large franchises who already spent enormous efforts into creating their custom reward systems.

This document details the planning of the second stage of development of said system. In the Competition and Market Rationale section, we listed existing potential competitors of RewardWallet and analyzed their offered services. Then we described the competitive advantage of the RewardWallet in comparison to its competitors.

The Risk Analysis section described possible issues during the development period of the RewardWallet solution so that the development is better prepared for such problems. In Personal Task Allocation and Time Management, we detailed the project schedule and tasks allocated to each member on the team. Contingency plans are also added in the event of unexpected delays.

In the next four months, the development team at RewardWallet will dedicate their full efforts in producing a POS add-on that is reliable and low costing. It will allow businesses to flourish by awarding their customers with store points as well as updating their customers on current store promotions and personalized rewards. All these benefits are accomplished as a simple add-on to their current system, allowing businesses to invest minimally.

13 Sources and References

[1] Shaw, H. (2017). Canadians just can't seem to quit loyalty cards, despite all of the data breaches and PR headaches. [online] Financial Post. Available at: http://business.financialpost.com/news/retail-marketing/canadians-just-cant-seem-to-quit-loyalty-cards-despite-all-of-the-data-breaches-and-pr-headaches [Accessed 20 Feb 2018].

[2] How Much to Make an App. (2018). How much does it cost to make an app? - App Cost Calculator. [online] Available at: http://howmuchtomakeanapp.com [Accessed 20 Feb. 2018].

[3] Developer.apple.com. (2018). App Store Review Guidelines - Apple Developer. [online] Available at: https://developer.apple.com/app-store/review/guidelines/#safety [Accessed 20 Feb. 2018].

[4] Developer.apple.com. (2018). App Review - App Store - Apple Developer. [online] Available at: https://developer.apple.com/app-store/review/ [Accessed 20 Feb. 2018].

[5] Shaw, H. (2017). *Canadians just can't seem to quit loyalty cards, despite all of the data breaches and PR headaches*. [online] Financial Post. Available at: http://business.financialpost.com/news/retail-marketing/canadians-just-cant-seem-to-quit-loyalty-cards-despite-all-of-the-data-breaches-and-pr-headaches [Accessed 20 Feb 2018].

[6]"API Design Guide | Cloud APIs | Google Cloud", Google Cloud, 2018. [Online]. Available: https://cloud.google.com/apis/design/. [Accessed: 20- Mar- 2018].

[7]"SLA & Uptime calculator: How much downtime corresponds to 99.9 % uptime", Uptime.is, 2018. [Online]. Available: https://uptime.is. [Accessed: 31- Mar- 2018].

[8]Apple.com, 2018. [Online]. Available: https://www.apple.com/business/docs/iOS_Security_Guide.pdf. [Accessed: 31- Mar- 2018].

[9]"Core NFC | Apple Developer Documentation", Developer.apple.com, 2018. [Online]. Available: https://developer.apple.com/documentation/corenfc. [Accessed: 31- Mar- 2018].

[10]"Host-based Card Emulation | Android Developers", Developer.android.com, 2018. [Online]. Available: https://developer.android.com/guide/topics/connectivity/nfc/hce.html. [Accessed: 31- Mar- 2018].

[11]"Host-based Card Emulation | Android Developers", Developer.android.com, 2018. [Online]. Available: https://developer.android.com/guide/topics/connectivity/nfc/hce.html. [Accessed: 31- Mar- 2018].

[12]"About the NDEF Format | Adafruit PN532 RFID/NFC Breakout and Shield | Adafruit Learning System", Learn.adafruit.com, 2018. [Online]. Available: https://learn.adafruit.com/adafruit-pn532-rfid-nfc/ndef. [Accessed: 31- Mar- 2018].

[13]"raywenderlich/swift-style-guide", GitHub, 2018. [Online]. Available: https://github.com/raywenderlich/swift-style-guide. [Accessed: 31- Mar- 2018].

[14]"Google JavaScript Style Guide", Google.github.io, 2018. [Online]. Available: https://google.github.io/styleguide/jsguide.html. [Accessed: 31- Mar- 2018].

[15] Standards.ieee.org. (2010). IEEE 1003.13-2003 - IEEE Standard for Information Technology - Standardized Application Environment Profile (AEP) - POSIX(R) Realtime and Embedded Application Support. [online] Available at: http://standards.ieee.org/findstds/standard/1003.13-2003.html [Accessed 20 Feb. 2018].

[16] Standards.ieee.org. (2009). IEEE 1625-2008 - IEEE Standard for Rechargeable Batteries for Multi-Cell Mobile Computing Devices. [online] Available at: http://standards.ieee.org/findstds/standard/1625-2008.html [Accessed 20 Feb. 2018].

[17] Standards.ieee.org. (2017). IEEE 2410-2017 - IEEE Standard for Biometric Open Protocol. [online] Available at: http://standards.ieee.org/findstds/standard/2410-2017.html [Accessed 20 Feb. 2018].

[18] Standards.ieee.org. (2009). IEEE 802.15.5-2009 - IEEE Recommended Practice for Information technology-- Telecommunications and information exchange between systems-- Local and metropolitan area networks-- Specific requirements Part 15.5: Mesh Topology Capability in Wireless Personal Area Networks (WPANs). [online] Available at: http://standards.ieee.org/findstds/standard/802.15.5-2009.html [Accessed 20 Feb. 2018].

[19]"7 fundamentals of design – and how they apply to online spaces", Social in silico, 2018. [Online]. Available: https://socialinsilico.wordpress.com/2015/01/27/7-fundamentals-of-design-and-how-they-apply-to-online-spaces/. [Accessed: 31- Mar- 2018].

[20]"raywenderlich/swift-style-guide", GitHub, 2018. [Online]. Available: https://github.com/raywenderlich/swift-style-guide. [Accessed: 31- Mar- 2018].

[21]"Google JavaScript Style Guide", Google.github.io, 2018. [Online]. Available: https://google.github.io/styleguide/jsguide.html. [Accessed: 31- Mar- 2018].

[22]"Themes - Overview - iOS Human Interface Guidelines", Developer.apple.com, 2018. [Online]. Available: https://developer.apple.com/ios/human-interface-guidelines/overview/themes/. [Accessed: 31- Mar- 2018].

[23]"Amazon.com Help: Same-Day Delivery Rates", Amazon.com, 2018. Available: https://www.amazon.com/gp/help/customer/display.html?nodeId=200105970.

[24]K. Fry, "Small Business Outlook", Olympia Benefits, 2014.

[25]A. Mceachern, "What are Loyalty points worth", Smile.io, 2017.

[26]Nielsen Holdings PLC, “Get With The Program Card-carrying Consumer Perspectives On Retail Loyalty Program Participation and Perks,” New York City, New York, USA, 2016, [Online]. Available: http://www.nielsen.com/content/dam/corporate/us/en/reports-downloads/2016-reports/nielsen-global-retail-loyalty-sentiment-report.pdf

[27]M. Fruend, "Canadian Loyalty Programs Reach 175 Million Memberships, 2017 COLLOQUY Loyalty Census Shows", LoyaltyOne.com, 2017. Available: https://www.loyalty.com/news-events/latest-news /news-releases/canadian-loyalty-programs-reach-175-million-memberships-2017-colloquy-loyalty-census-shows.

[28] “Business Customer Loyalty Program | Belly.” Internet: https://www.bellycard.com/member, [Jan. 24, 2018].

[29] P. Joannis, “Fivestars Review 2018 - Merchant Maverick.” Internet: https://www.merchantmaverick.com/reviews/fivestars-review/, 03-Nov-2015 [Jan. 24, 2018].

[30] “Create a Loyalty Program with Square | Square Support Center.” Internet: https://squareup.com/help/us/en/article/3952-create-a-loyalty-program-with-square, [Jan. 24, 2018].

RewardWallet