# The University of Sydney
# Faculty of Engineering
# School of Electrical and Information Engineering

## Year S1, 2022

A thesis submitted in partial fulfilment of a Bachelor of Engineering / Master of Engineering Degree

| | |
|---|---|
| **Student Name** | Kuan Li |
| **SID** | 490443479 |
| **Unit of Study Code and Name** | **ELEC4712 Thesis A** |
| **Supervisor** | Dong Yuan |
| **Title** | Menudo: BLE Based Table Ordering System |

**The University of Sydney**

**SCHOOL OF ELECTRICAL AND INFORMATION ENGINEERING**

# PROJECT CLEARANCE FORM

**Unit of Study Code and Name**

ELEC4713 Thesis B

**This is to certify that my student**

**Student Name** Kuan Li

**SID** 490443479

**Has:**

- **Returned all books and reference material;**
- **Returned all equipment and keys; and**
- **Tidied their work place.**

**SUEIE Academic supervisor:**

**Signature:** **Date:**

**Name:**

**External supervisor (if applicable):**

**Signature:** **Date:**

**Name:**

# Menudo: iBeacon Based Table Ordering System

Kuan Li
Software Engineering, USYD
Sydney, Australia
490443479
kuli0394@uni.sydney.edu.au

## Abstract

The whole world goes wireless. And digital. One technology is right in the centre of these two megatrends: Bluetooth Low Energy (BLE). Instead of physical beacons with batteries that need to be deployed, virtual BLE is an ultra-low power *digital* version of Bluetooth, act as the beacon point that is added to existing *wireless* Wi-Fi networks for accurate proximity calculation. BLE beacon represents the one of the least invasive technologies with no monitoring of the *exact location* of a cell user but rather the *relative distance* between his device and that of another. It is regarded as the cornerstones of indoor positioning.

82% of smartphone users say they consult their phones on purchases they're about to make in-store. (*Proximity-Marketing-What-How-Why*, n.d.) Research from the National Restaurant Association (NRA) showed that 60% of adults agreed that restaurant technology increased their convenience. (*HOW TABLE ORDERING APPS HAVE TAKEN OVER RESTAURANTS*, n.d.) These show a growing need to make a difference with potential enhancement of table ordering system.

A cleaner, neater way will be illustrated in detail in this project. The intention of this project is to use BLE Technology to design and develop an innovative *table ordering system* namely **Menudo**, where sellers can broadcast the restaurant information and diners can detect the menu instantly to their mobile device, which means no extra hardware is required during the entire dining experience. In short, diners can preview dishes before entering the restaurant followed by a smoother ordering experience after entering the restaurant. It is developed under iOS environment for better user experience in the lifecycle including *iBeacon, Swift,* and *Apple Pay*. Finally, the seamless connections among smart devices and the cloud are achieved with *Serverless* framework for an auto-scaling applications on *AWS Lambda.*

# Acknowledgments

Thanks to my supervisor, **Dr. Yuan Dong** for his guidance and support throughout the project and the precious advice for my future career.

Thanks to my friends with insightful suggestions for any mistakes and possible improvements.

Thanks to **Lin Shi** my beloved.

# Table of Contents

# List of Tables



# List of Figures



# Glossary

| Term | Definition |
|---|---|
| Bluetooth Low Energy (BLE) | Bluetooth low energy (BLE) technology, also known as Bluetooth Smart or Bluetooth 4.0, uses less energy than standard Bluetooth wireless communications. |
| AWS Lambda | AWS Lambda is an event-driven, serverless computing platform provided by Amazon as a part of Amazon Web Services. |
| Serverless | A serverless architecture is a way to build and run applications and services without having to manage infrastructure. |
| Amazon Simple Storage Service (S3) | S3 provides scalable object storage through a web service interface. |
| Universally unique identifier (UUID) | A universally unique identifier (UUID) is a 128-bit label used for information in computer systems. It is for storing objects in a database, where every object needs a unique ID. |

| | |
|---|---|
| lifecycle | The application life cycle constitutes the sequence of events that occurs between the launch and termination of application. |
| Swift | Swift is a programming language used to create apps for iOS, macOS, tvOS, and watchOS. |
| JSON Web Token (JWT) | JWT is a proposed Internet standard for creating data with optional signature and/or optional encryption whose payload holds JSON that asserts some number of claims. |
| Model-View-Controller Design Pattern (MVC) | Model–view–controller is a software architectural pattern commonly used for developing user interfaces that divide the related program logic into three interconnected elements. This is done to separate internal representations of information from the ways information is presented to and accepted from the user. |
| Create, read, update and delete (CRUD) | In computer programming, create, read, update, and delete are the four basic operations of persistent storage. |
| RFID (Radio-frequency identification) | Radio-frequency identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to objects. |
| QR (Quick response) | A QR code is a type of matrix barcode. A barcode is a machine-readable optical label that can contain information about the item to which it is attached. |
| NFC (Near-field communication) | Near-field communication (NFC) is a set of communication protocols that enables communication between two electronic devices over a distance of 4 cm. |
| REST API | Representational state transfer is a software architectural style that describes a uniform interface between physically separate components, often across the Internet in a client-server architecture. |
| Application Programming Interface (API) | API is a collection of software functions and procedures. They allow two applications to interact with each other without any user intervention. |
| Unified Modelling Language (UML) | UML is a rich language to model software solutions, application structures, system behaviour and business processes. |
| Command-line interface (CLI) | Command-line interface is a text-based user interface (UI) used to run programs, manage computer files, and interact with the computer. |
| Postman | Postman is an API platform for developers to design, build, test and iterate their APIs. |
| Entity Relationship Diagram (ERD) | An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities. |

# Chapter 1 Introduction

## Necessity of digital table ordering

"The digital transformation in March and April (2020) was on a larger scale than the digital transformation we have witnessed during the past two years" – Satya Nadella, CEO of Microsoft. The pandemic completely altered routine of a customer and accelerated the digitalisation of daily life. Way departed from traditional table service, recent restaurant tech that's grown due to the pandemic offers diverse tailored order approaches (kiosks, mobile, app, website, and online ordering solutions), anything hassle-free, contactless, and personalized, and customized options for different restaurant segments. National Restaurant Association (NRA) showed 60% of adults agreed that restaurant technology, including *(1)* smartphone apps, *(2)* table ordering systems like tablets for payment processing, and *(3)* self-service kiosks increased their convenience. (*HOW TABLE ORDERING APPS HAVE TAKEN OVER RESTAURANTS*, n.d.) It surely makes a necessity to a step towards digital table ordering.

## Traditional payment to mobile payment

Contactless, cashless payments become standards in a post-COVID world. Technology is driving the mobility of payment gateways to satisfy with customer demands for commerce. Smartphones have become the default payment gateway, and most models now have built in modules to ensure extra security. 92% of Australians own and use a smartphone, which is becoming a tool for consumers to do everything from research to payments. In addition to external devices that can be swiped or tapped, phones themselves are becoming portable cash machines, with Apple, Samsung and Android all introducing mobile wallets into their product suites. (*Mobile and Contactless Payment Options*, n.d.) Cash payments have been steadily declining in Australia, with under 27% of consumer payments now made in cash. This decline can largely be attributed to the swift rise in contactless payments options, such as tap-and-go and digital wallets like Apple Pay. (*10 Dining Trends To Watch In 2022 | Industry Report*, n.d.)

## A more secure way

Current table ordering solutions provide customers with scannable QR codes or tappable RFID stickers containing link to restaurant menu site to make orders. There is news from time-to-time reporting about menu fraught with data risk. Local restaurant Balmoral Pasture in Mosman had encountered the malicious data collection when using Me&U as the contactless table ordering solution. (Richards, 2021) It comes with security risks if the legitimate QR code gets replaced with malicious one which is easy to achieve but hard to detect. Existing solutions are cumbersome and can be easily exploited. On the other hand, proximity marketing via beacons only allows unidirectional data transfer hence the user information can never be leaked. Combining proximity marketing with table ordering ensures higher level of security.

# Chapter 2 Background

## Target Audience

The target audience for Menudo are diners intended for a more convenient and secure table ordering experience.

## User Stories

*Table 1 User stories*

| | NAME | AGE | PROFESSION |
|---|---|---|---|
| **PERSONA 1** | WenYuan, Shi | 55 | Retired, bricklayer |
| | WenYuan used to eat out with family occasionally, but after he heard more and more news reporting on invasively data risks of commonly seen digital menus, he cooked at home more often. He reminded his children take extra attention to suspicious stickers, QR codes and websites, however, risks were hard to be identify. | | |
| **PERSONA 2** | Gordon, Liang | 21 | Student |
| | Gordon is an environmentalist who promoted the digital menu replacing printed menu when it first came in public view. There existed a lot of waste in traditional table ordering experience: (1) waster of paper when regular change of menus, (2) usage of non-recyclable credit cards, (3) single-use receipt and kitchen notice slip. The digital menu was an all-rounder to solve all these problems. When he learnt that beacon-based table ordering can even eliminate the stickers or tablets, he felt the excitement to see it coming. | | |
| **PERSONA 3** | Chao, Li | 34 | Café owner |
| | Chao is the owner of Babyccino Kids Café (https://goo.gl/maps/5xL7D7onvYf4qEhn7). Babies and kids were causing a lot of problems on newly installed QR code digital menu. Food is often spilled on the sticker on each table, kids would peel the edges, or draw on it to blur the code. The stickers were worn out quickly. He did not enjoy the convenience and comfort as he thought when he made the decision to replace paper menus. | | |

## Market

According to a survey by Shopper, Aussie households saved over *$140* billion because of the pandemic; *40%* of Australian consumers are looking forward to spending on dining out. The post-lockdown desire for revenge spending will provide a much-needed revenue boost for hospitality businesses. (*Australian Consumers to Go on "revenge Spending" Sprees Once Covid-19 Restrictions Are Lifted*, n.d.)

In the report by Lightspeed notes that only 4.9% of Lightspeed venues were utilising QR codes for ordering in August of 2020. This figure has steadily increased, topping out at 15.2% in August of 2021. Customers who order via QR table ordering spend on average 25% more than traditional dine-in orders. (*10 Dining Trends To Watch In 2022 | Industry*

*Report*, n.d.) 57% of respondents stated they would choose a venue based on the availability of QR technology. When ordering, iPad POS systems benefit both vendor and customer. 72% of single-location restaurant owners show interest in mobile point of sale according to Software Advice. (*How Many Restaurants Are Using an IPad POS System?*, n.d.)

Menudo perfectly fits in with the current market needs. Digital menu has been gradually accepted by the public for both restaurant owners and customers. Convenience of contactless payment becomes dominating by the push of pandemic. The rising of QR code menus show the potentiality of digital table ordering. There is no doubt that the revival of social activities including dining out after pandemic is the hotbed for an upgraded table ordering system.

## Existing Solutions

**QR code mobile ordering, Mr Yum, Me&U(Me&U, n.d.)**
Using a smartphone camera to scan a QR code on the menu at participating venues, diners can access a digital version of the restaurant's menu on online platform. QR tags must be visible and has limitation of needing a line of sight to scan. On the other hand, QR code is effective in low cost and easiness to both use and generate.

**Passive RFID devices Table Tracker(Table Tracker, n.d.)**
This ordering system containing three components: a touch-screen PC monitor, Table Tracker devices and a map of the restaurant. Customer places the Table Tracker device on the table, it sends that table's exact number to the kitchen display via RFID tags installed under the table. Instead of laser scanners for barcodes and optical scanners for QR codes, RFID tags use radio waves to transmit the information stored in them.(*Crash Course: RFID, Barcodes, QR Codes—What's the Difference?*, n.d.) Passive tags are "powered" by electromagnetic energy sent from an RFID reader. There is a possibility for passive RFID tags to last a lifetime since the don't require battery or any internal power source.

**NFC Tags - TagThose(*NFC Food Ordering Tags*, n.d.)**
Hover your phone over the sticker and the data will quickly be sent to your phone for an existing mobile food ordering service in place. NFC technology utilizes only an alternating magnetic field, meaning that no power is emitted in the form of radio waves. This prevents any interference from occurring between similar devices. Due to its' larger storage space, NFC devices can store and transmit more data than RFID devices that can only carry simple ID information.(Salmane, 2021)

## Technologies

**Bluetooth Low Energy (BLE), iBeacon**
The BLE is designed for devices that do not require large amounts of data transfer and is intended for short-range wireless transmission with low power consumption and cost(Dahlgren & Mahmood, 2014), (Siekkinen et al., 2012). Consumer BLE devices do not

report the channel a particular beacon is heard on, and so BLE beacon fingerprinting to date has not tended to use the channel information.(Powar et al., 2017) The signal fluctuation in BLE is a consequence of the random use of the advertisement channels. BLE has become the de facto standard for smartphones to communicate with wearable devices and with nearby IoT devices.(Cha et al., 2018) Similar to Wi-Fi, the localization methods like proximity, trilateration, and fingerprinting can be realized using a set of BLE. (Subedi & Pyun, 2020)

The advantages of BLE include (1) wide consumer device support; (2) low power continuous background scanning; (3) simple, low battery powered with a lifespan for years and can be deployed anywhere; (4) support for high density deployments with minimal interference; and (5) instantaneous beacon reporting at the handset (no fingerprint smearing). BLE is suited for applications that work well with a periodic transfer of data and reduce a significant amount of battery usage. This makes BLE suitable for IoT (Internet of Things) and proximity marketing- related applications.

iBeacon is based on Bluetooth low energy proximity sensing by Wireless transmitters bringing attention to their location by broadcasting a signal with a unique identifier at regular intervals. Apple develops the iBeacon profile (natively supported in iOS), and its signal contains three main pieces of information, namely, UUID (Universally unique identifiers), Major, and Minor(Spachos & Plataniotis, 2020) which is a 25 byte payload that is set as the manufacturer data field of a BLE advertisement (Welles, n.d.).

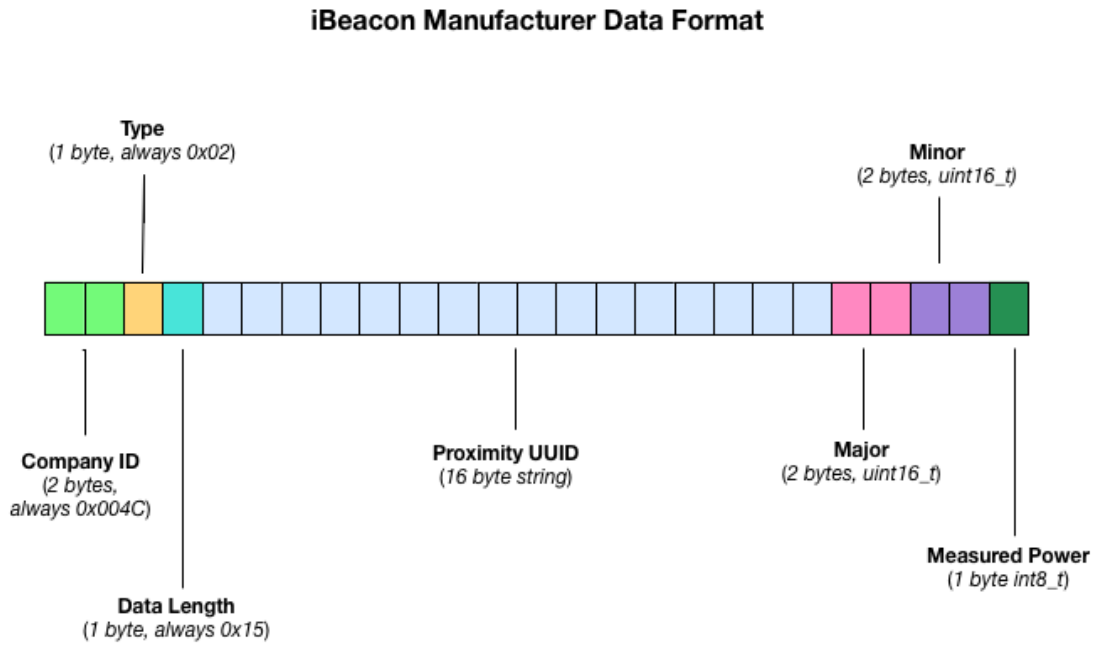



*Figure 1 iBeacon Manufacturer Data Format*

There are few things to be highlighted for using iBeacon in Menudo. The positioning accuracy of the iBeacon-based method is RMSE 2.75 m. Hence the restaurant counter is ideally to be placed at the entrance. In this project, detection nearby restaurants is done by

recognising in-built iBeacon UUID. Next, each restaurant corresponds to an iBeacon Major value so that customer is able to receive and view the specified menu. For security concerns, iBeacon does not allow beacon ranging in background. As a result, iBeacon-based applications have the nature to avoid bombardment of ads or spams by simply close the app or minimise to the background. (*IBeacon Background Monitoring*, n.d.) Furthermore, iBeacon with unknown UUID cannot be detected 10/*11/2022 23:56:00*. It ensures that only authenticated restaurants will be added to the detection list and broadcasting menus. It prevents from illegal broadcasting and detection happening, which guarantees the customer's security.

### Serverless, Amazon Web Services (AWS) Lambda, Function as a Service (FaaS)

Compared to traditionally deployed web applications in a *serverless* architecture, the developing team does not have to maintain servers or infrastructure. The server code is deployed with the help of a *cloud provider*, who is responsible for handling the operation of running the back-end logic, maintaining, scaling as per the traffic, and security. (*How to Build a Serverless Backend with AWS Lambda*, n.d.) *AWS Lambda* is the cloud provider utilised in this project. It is regarded as an event-driven compute service which allows running code for backend service without provisioning or managing servers.

*Function as a Service* (FaaS) is the serverless architecture that is implemented in deployment of server code. With *FaaS*, simplified and modular approach server code with one function corresponds to each operation. Each Lambda function runs in its own container. When a function is created, Lambda packages it into a new container and then executes that container on a multi-tenant cluster of machines managed by AWS. (*AWS Lambda The Ultimate Guide*, n.d.)Functions are triggered by events of *HTTP* requests from the client, database operations, file uploads, scheduled events and so on. Each endpoint of the *REST API* is going to be a separate serverless function deployed on *AWS Lambda*. In this modulation, users only pay for functionality that is used. However, this project is still in the free tier since only tiny resources are required.

### Amazon S3

*Amazon Simple Storage Service* (S3) provides object storage through a web service interface, using scalable storage infrastructure. In this project, item image is uploaded in S3 with S3 compatible API provided by *Swift* for applications. A unique s3 bucket URL is generated, and image will be retrieved and downloaded with the URL.

### JSON Web Token (JWT) Authentication

*JWT* is a proposed Internet standard for creating data with optional signature and/or optional encryption whose payload holds JSON that asserts some number of claims. The user gets a token upon logging in, and then sends that token back to the endpoint on every request. It makes it easy to store additional user information directly in the token, not just the access credentials. you don't have to access the datastore for getting user information, which can decrease operational costs significantly changing the shape of the data stored in JWT tokens during development is faster, and that enables easier experiments, implementing JWT is easier than reading and writing sessions.

Downside, JWT tokens are larger than average session keys, so your clients may be sending more data to your endpoints overall Implementing authentication via JWT in a production app certainly requires spending extra time on ensuring that the tokens are used correctly, that you only store the most necessary information in the tokens, and that you are keeping your encryption keys safe. (*Strategies for Implementing User Authentication in Serverless Applications*, n.d.)

In Menudo, token is added to headers with "Authorization" header name, which will be processed by serverless backend and decode the user_id to perform authenticated actions.

### Capabilities: Sign in with Apple, Apple Pay

A capability grants your app access to an app service that Apple provides, including Sign in with Apple, Apple Pay and Wallet. To use app services, provision is required in the app, adding a capability with Xcode's project editor that configures the app service correctly.

*Sign in with Apple* is the fast, easy and more private way to sign into third-party apps and websites using the *Apple ID* that you already have. UUID The user identifier property of *ASAuthorizationAppleIDCredential* is a stable unique identifier between an Apple ID and Developer Team. Application revocations will not affect the stability of the user ID. This also means that among a developer team's set of applications the user ID will be the same, making it optimal for synchronization among a developer's apps. *Apple Pay* is a mobile payment service that allows users to make payments in person, in iOS apps, and on the web. The transactions can be verified whether they are completed successfully.

In Menudo, sellers require login while customers don't. Seller will receive a JWT each time logging in, once the JWT is expired, seller is required to re-login the app. As for client app, an UUID will be assigned to each customer, and it will be packed into order information when sending orders to the seller. Apple Pay is a mocked version in Menudo since there is no legitimate merchant license or business permit in this project development. Hence all transactions will fail, and orders will be sent before success in transaction.

# Chapter 3 Design

## Features

Menudo table ordering system consists of:

**Seller App**

- Activate in-built iBeacon in iOS devices and broadcast menu for nearby customers
- Edit restaurant profile
- Edit restaurant menu which will be auto update on client app
- Keep tracking of order status

**Client App**

- Detect nearby restaurants by recognising in-built iBeacon UUID
- Receive and view the menu by corresponding iBeacon Major value
- Online payment
- Send order information to seller

## System Architecture

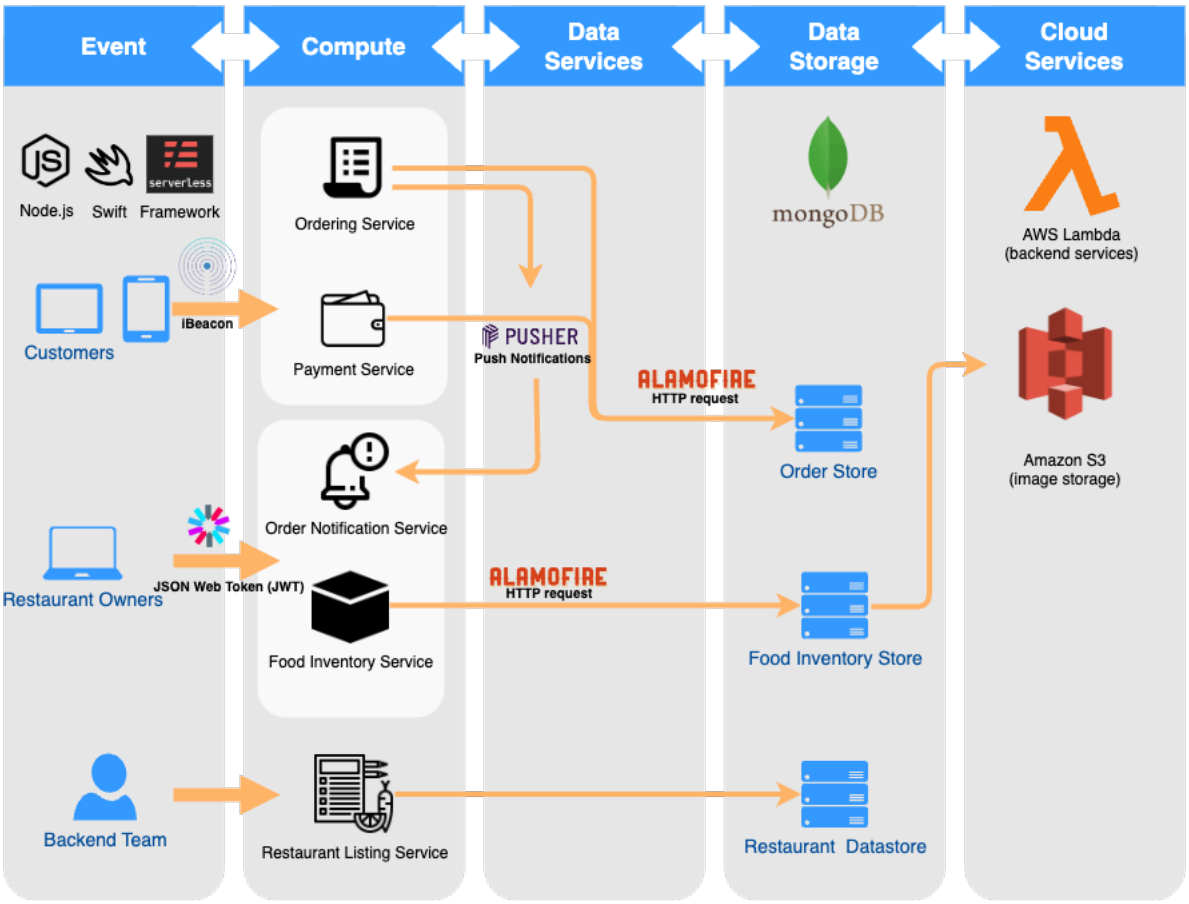



*Figure 2 Menudo System Architecture Overview*

## Development Tools

**Swift, v5**
*Swift* is a powerful and intuitive programming language created by Apple for building apps. *SwiftUI* is a modern way to declare user interfaces for any Apple platform. iOS development target is required above 15.0.0 and below 16.0.0.

**Node.js, v14.18.0**
*AWS Lambda* supports different runtimes such as Node.js, Java, Python, .NET Core and Go for you to execute a function. *Node.js* is to develop back-end services to write the functions or *CRUD* operations of the application.

**CocoaPods, v1.11.3**
*CocoaPods* is a dependency manager for Swift and *Objective-C* Cocoa projects, which is similar to npm for JavaScript package management.

**npm, v6.14.15**
*npm* is the default package manager for the JavaScript programming language runtime environment Node.js.

**Alamofire, v5.0**
Swift-based, *HTTP* networking library. It provides an elegant interface on top of Apple's Foundation networking stack that simplifies common networking tasks.

**AWSS3, v2.15.3**
The AWS SDK for iOS provides a library and documentation for developers to build connected mobile applications using AWS. AWSS3 is one of the modules allow getting access to existing S3 bucket.

**jsonwebtoken, v8.2.1**
An implementation of JSON Web Tokens. It is used in Menudo to generate token with user id and authenticate and decode token to user id.

**mongoose, v6.7.0**
Mongoose is a MongoDB object modelling tool designed to work in an asynchronous environment. Mongoose supports both promises and callbacks. It is used in Menudo backend to get connection with the database.

## Environment setup

### Backend

The serverless backend *is deployed* and can be called at any time. However, if you like to make modifications on it, you need to first

*sls config credentials --provider aws --key ACCESS_KEY --secret SECRET_KEY*

with your AWS user access key and secret key, and then

*npm install*

to install all packages required, next

*sls deploy*

to deploy the newest version of Menudo serverless backend.

### Frontend

You are required to use Xcode to run the project. Go into project folder, run

*pod install*

to install all packages required, next open the *MenudoClient.xcworkspace* / *MenudoSeller.xcworkspace* with Xcode, select a build simulator or iOS device to see the application. If you would like to build on actual device, the device should be in iOS version above 15.0.0 but below 16.0.0.

## Data Model

This section outlines the data schemas in diagram, highlighting entity relationships, data types, objects and structures.

### Data Type Tables

*Table 2 Data Type Table - Seller*

| FIELD NAME | DATA TYPE | REQUIRED | DESCRIPTION |
|---|---|---|---|
| **id** | String | Yes | Restaurant owner id |
| **full_name** | String | No | Restaurant owner name |
| **restaurant_name** | String | No | Restaurant name |
| **email** | String | No | Restaurant owner email |
| **uuid_major** | Number | Yes | Restaurant's beacon id |

*Table 3 Data Type Table - Item*

| FIELD NAME | DATA TYPE | REQUIRED | DESCRIPTION |
|---|---|---|---|
| **id** | String | Yes | Item id |
| **seller_id** | String | Yes | Item's seller id |
| **name** | String | Yes | Item name |
| **price** | Number | Yes | Item unit price |
| **image** | String | No | Item image URL |

*Table 4 Data Type Table – Order*

| FIELD NAME | DATA TYPE | REQUIRED | DESCRIPTION |
|---|---|---|---|
| **id** | String | Yes | Order id |
| **seller_id** | String | Yes | Seller id |
| **customer_id** | String | Yes | Customer id |
| **items** | [String] | Yes | List of items in the order |
| **total_price** | Number | Yes | Sum of costs for the order |

### Class Diagram and Entity Relationship

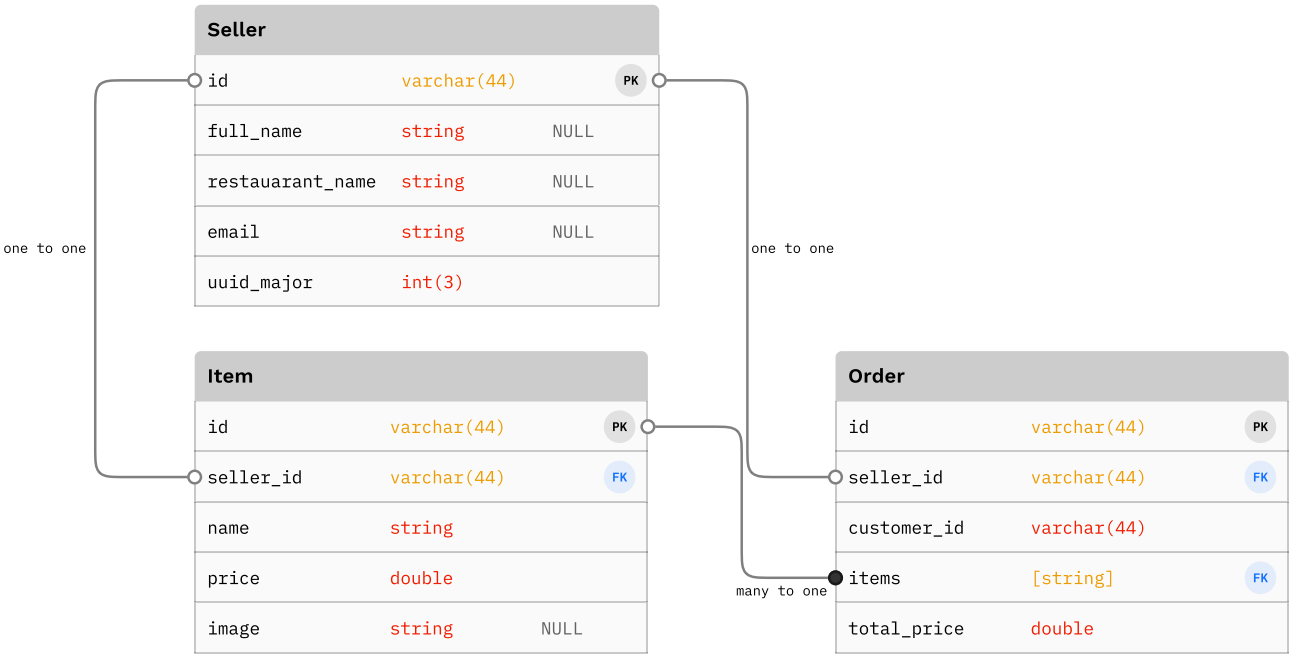



*Figure 3 Menudo Entity Relationship Diagram*

Note:
(1) ID include *seller_id*, *item_id*, *customer_id* and *order_id* are generated with *UUID()* function in Swift, hence they have the same length of 44 characters.

(2) Uuid_major complies to iBeacon Major which is an unsigned integer value between 1 and 65535. It is restricted to 3-digit identifier for simplicity in Menudo.

## API Endpoints

*Table 5 Backend API Endpoints - Auth Handler*

| FUNCTION NAME | HTTP REQUEST | PARAMS | AUTH | DESCRIPTION |
|---|---|---|---|---|
| **login** | Post | user_id | No | Login with user id |
| **register** | Post | user_id, full_name, email, uuid_major | No | Register account |

| | | | | |
|---|---|---|---|---|
| **me** | Get | N/A | Yes | Get account info with auth token |
| **register_if_not_exist** | Post | user_id, full_name, email, uuid_major | No | Register account if not exist, otherwise sign in |
| **edit_profile** | Put | full_name, email, restaurant_name | Yes | Edit profile with auth token |
| **users** | Get | N/A | No | Get all users in database |

Note: the actual authentication is done by *sign in with apple* module, the backend of Menudo only generates a JWT to disable the account for long time inactivity and does not store any password.

*Table 6 Backend API Endpoints - Item Handler*

| FUNCTION NAME | HTTP REQUEST | PARAMS | AUTH | DESCRIPTION |
|---|---|---|---|---|
| **upload_item** | Post | Item_id, seller_id, name, price, image url | Yes | Upload item, with auth token |
| **edit_item** | Put | Item_id, name, price, image url | Yes | Edit existing item with auth token |
| **delete_item/{id}** | Delete | Path parameter: item_id | Yes | Delete item by item_id with auth token |
| **inventory/{id}** | Get | Path parameter: user_id | No | Get inventory by seller_id |

*Table 7 Backend API Endpoints - Order Handler*

| FUNCTION NAME | HTTP REQUEST | PARAMS | AUTH | DESCRIPTION |
|---|---|---|---|---|
| **order/{id}** | Get | Path parameter: order_id | No | Get order by order_id |
| **orders** | Get | N/A | Yes | Get orders with auth token |
| **send_order** | Post | Order_id, seller_id, customer_id, items (array), total_price | No | Send an order |
| **complete_order/{id}** | Delete | Path parameter: order_id | Yes | Complete an order by order_id with auth token |

## Wireframes

### Menudo Seller

| PAGE | FUNCTIONALITY |
|---|---|
| SIGN IN | register/login |
| MENU LISTING | view/delete items |
| PROFILE | edit profile |
| ITEM UPLOAD | add item |
| ORDERS LISTING | view/complete orders |

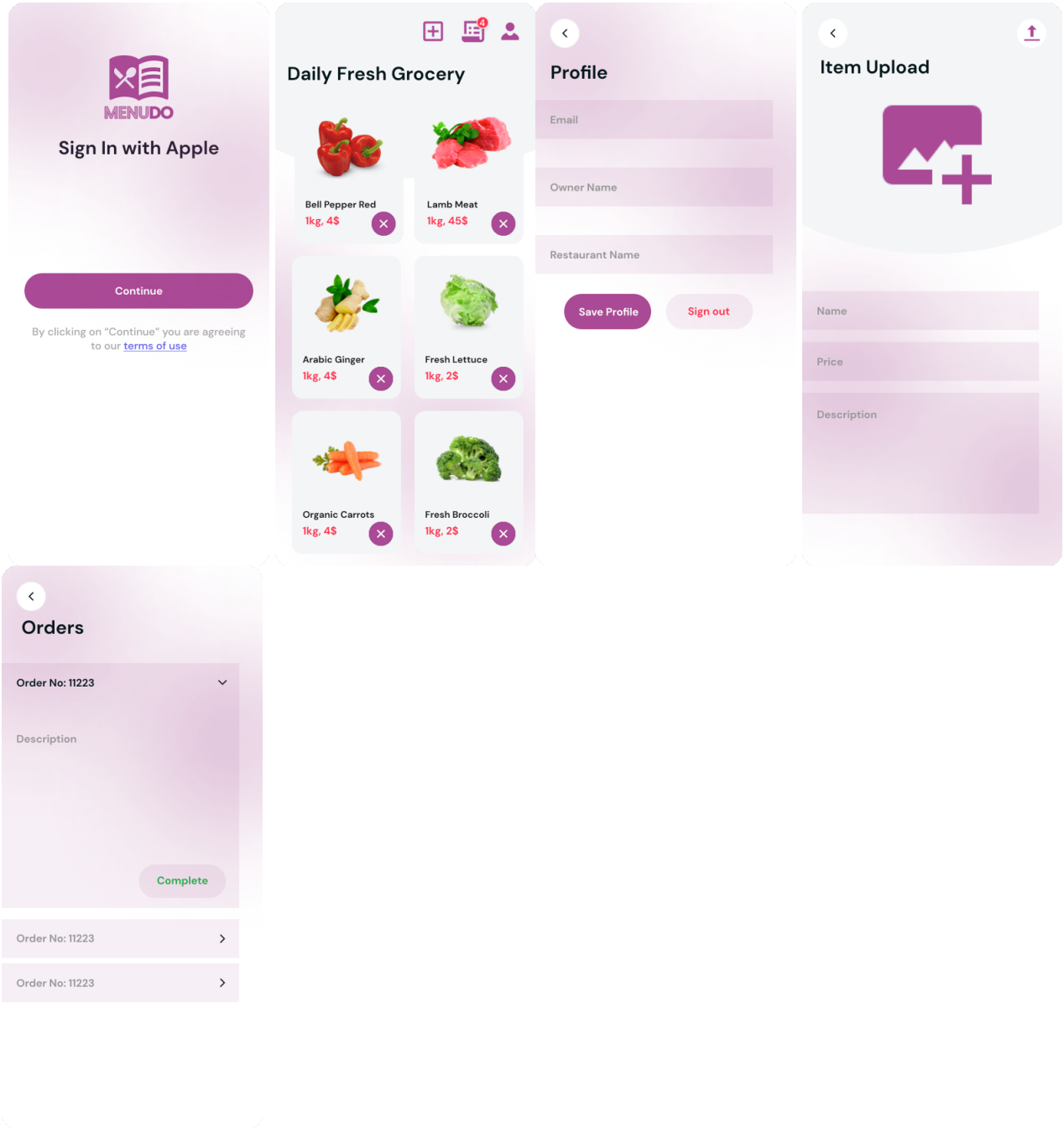



*Figure 4 Wireframes - Menudo Seller*

**Menudo Client**

| PAGE | FUNCTIONALITY |
|---|---|
| SEARCH RESTAURANT | beacon detection |
| WELCOME | Landing page |
| MENU LISTING | view/add item to cart |
| SHOPPING CART | view/delete item, make payment, send order |

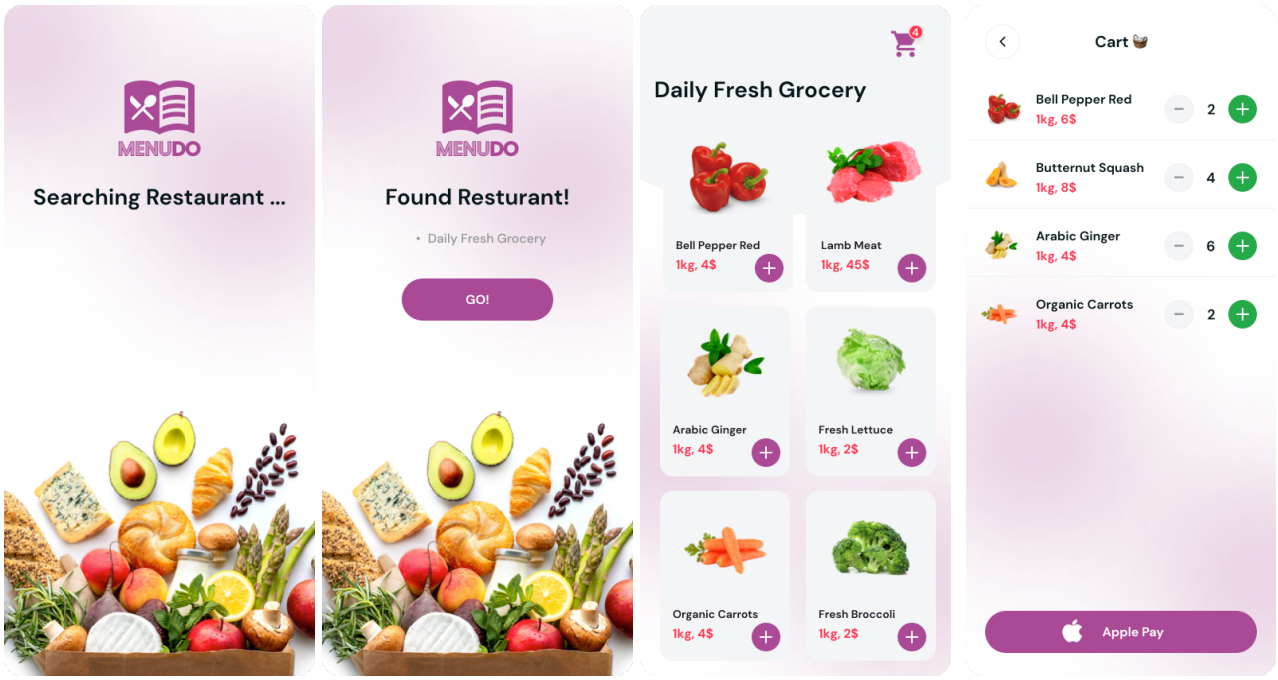



*Figure 5 Wireframes - Menudo Client*

## Frontend Design Pattern

The frontend design complies to MVC design patterns. The Controller is a mediator between the View and the Model, where View and Model are independent and reusable. Controller is non-reusable for other projects since it contains certain application specified logics. (*IOS Architecture Patterns*, n.d.)

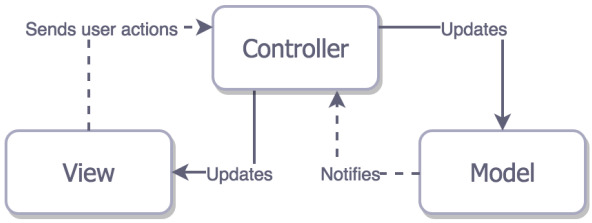



*Figure 6 iOS Design Pattern - MVC*

Menudo Seller and Menudo Client follows the similar design patterns, containing four main packages:

**Models**: Upload data to database in format. Interpret data from database in swift struct.

**Components**: Modules to be integrated into views. Components do not handle any logic but only visualise the values passed in.

**Managers**: Handle all traffics between frontend and backend, using http request to serverless api endpoints.

**Views**: Each view represents a page in user interface. Controller in MVC is integrated in views.

# Frontend UML

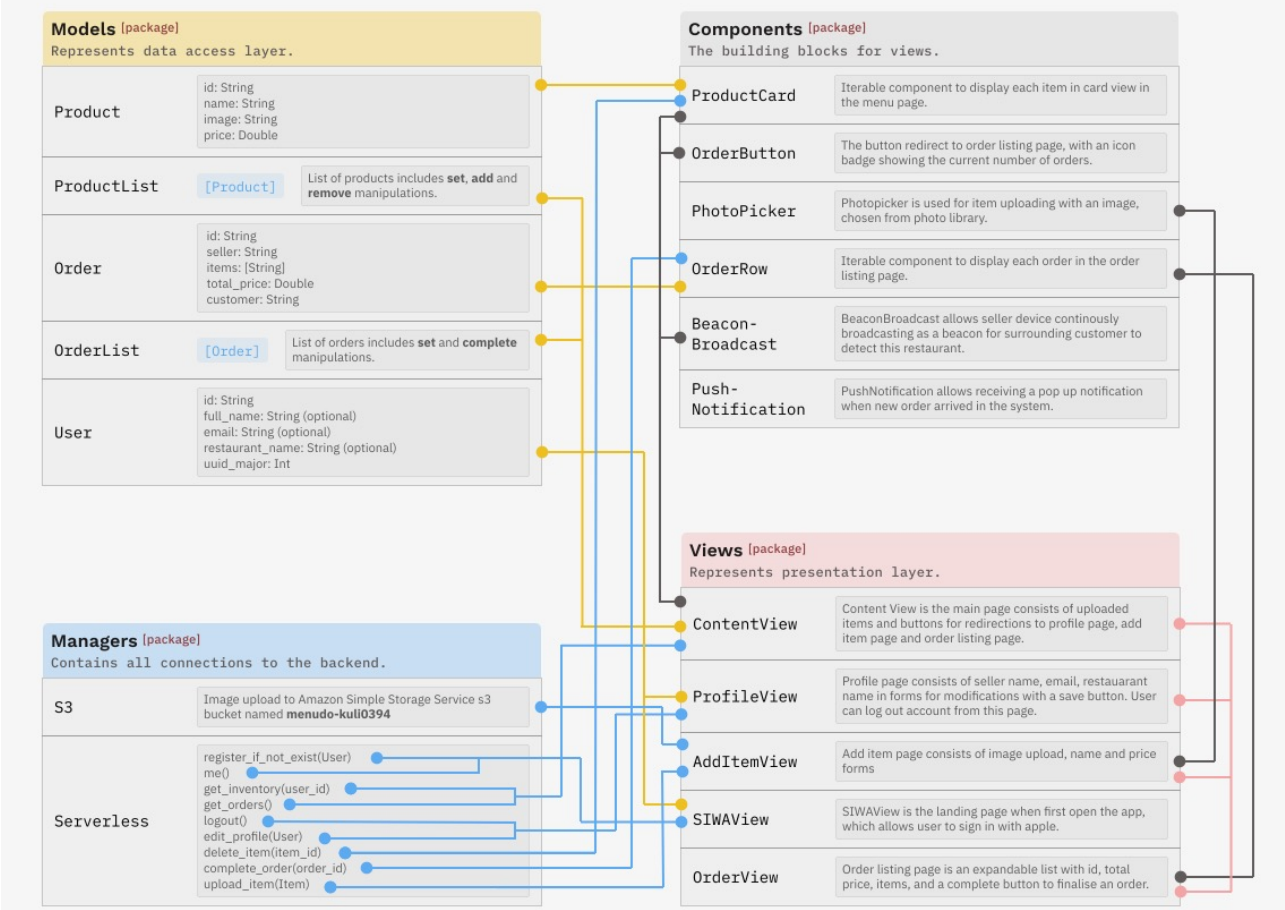



*Figure 7 UML - Menudo Seller*

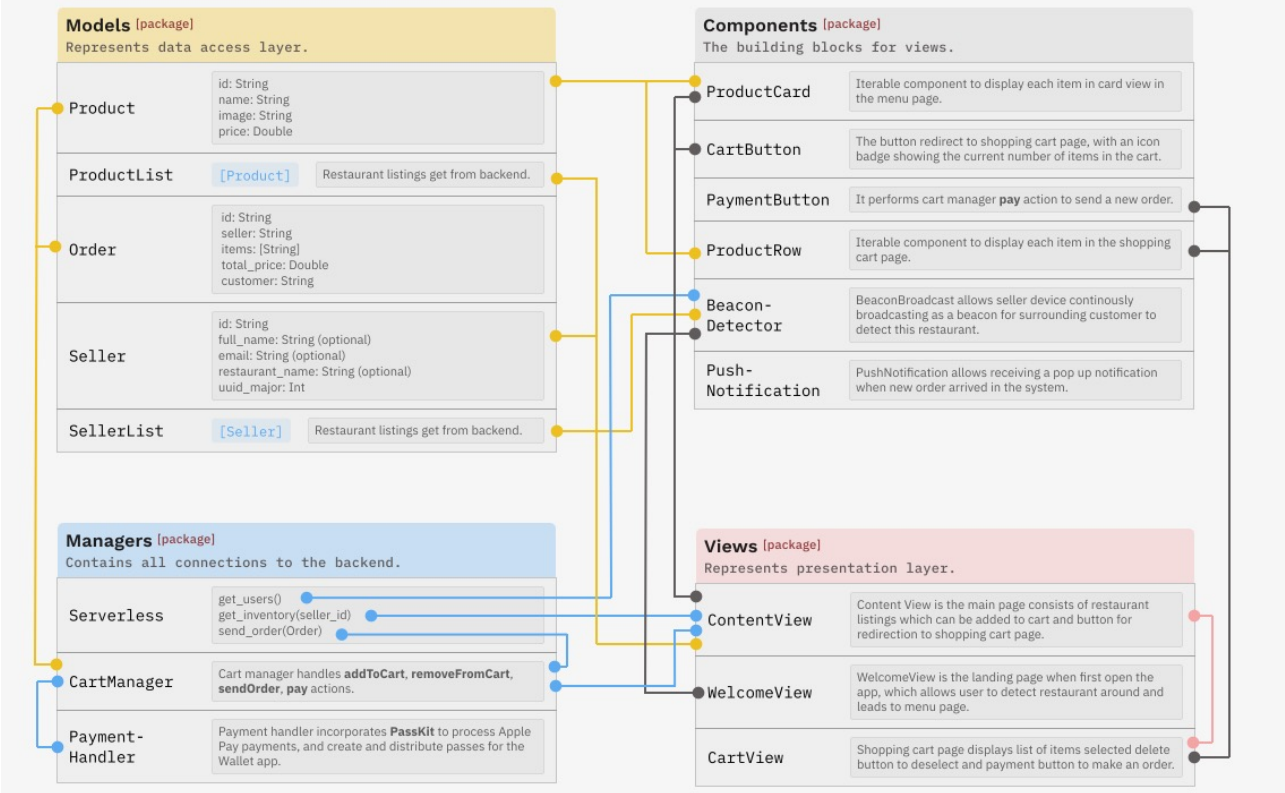



*Figure 8 UML - Menudo Client*

# Chapter 4 Analysis

## Troubleshooting

### Backend Debug

After each deploy, CLI *serverless invoke* of new functions added will give compiling errors in code. If the function does not require headers use CLI

*serverless invoke -- function function_name --data "(params)"/ --path params.json*

to debug. If the function requires headers, use Postman to test each API endpoint functional, where Body is the same as params in serverless invoke command.

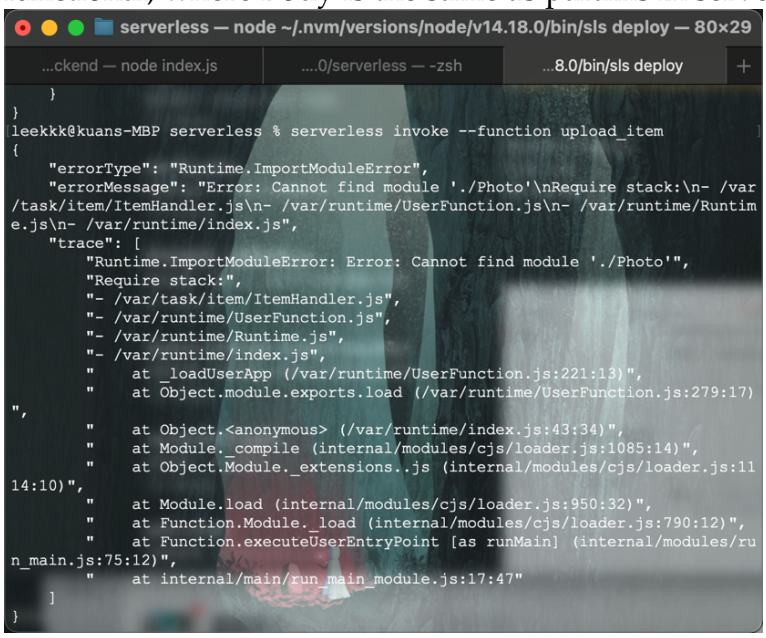



*Figure 9 Troubleshooting - CLI*

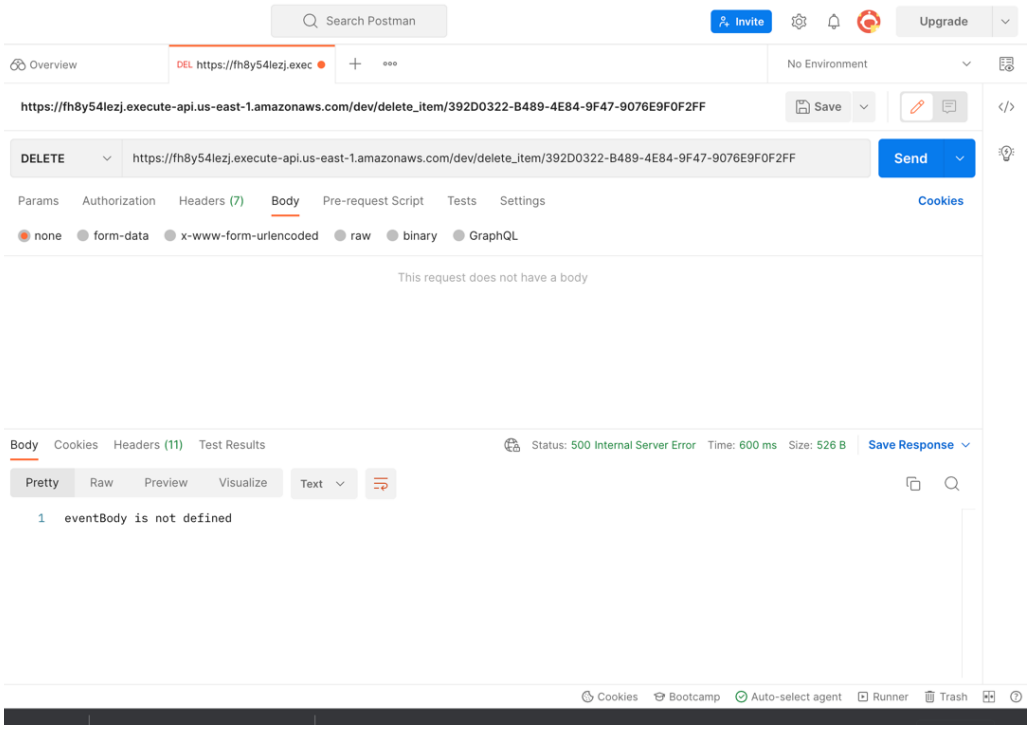
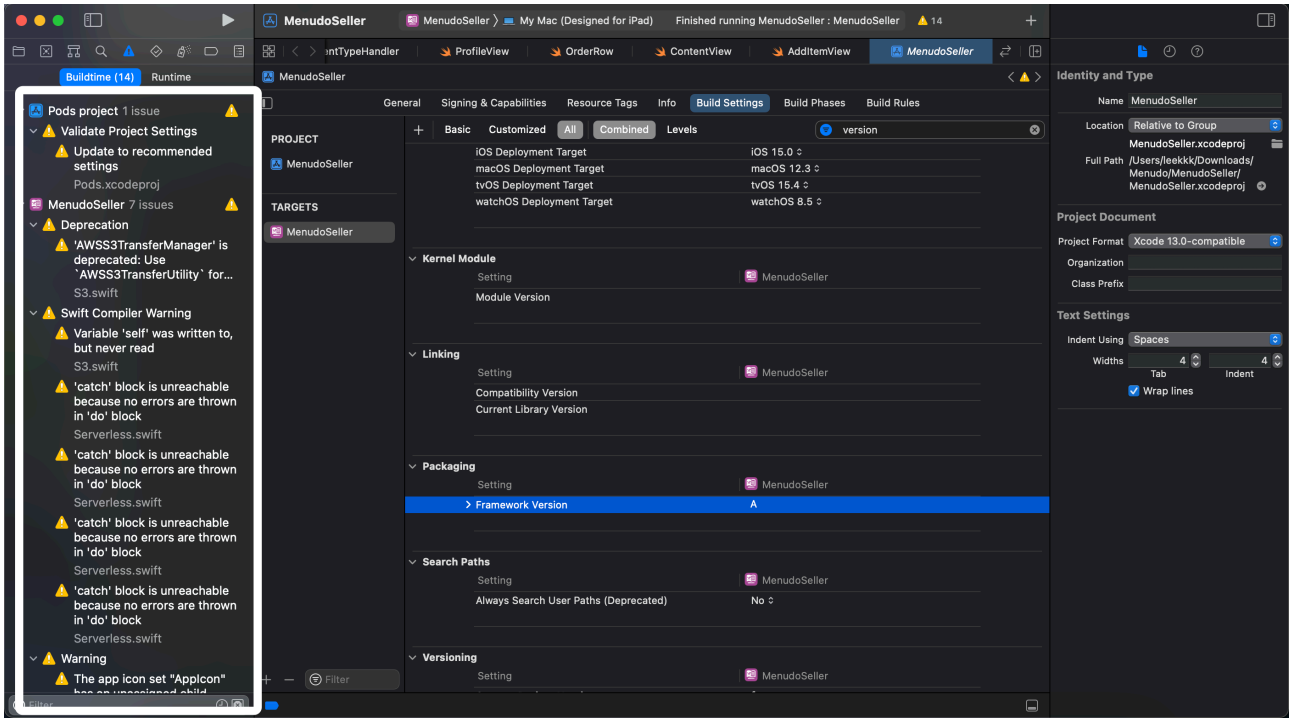
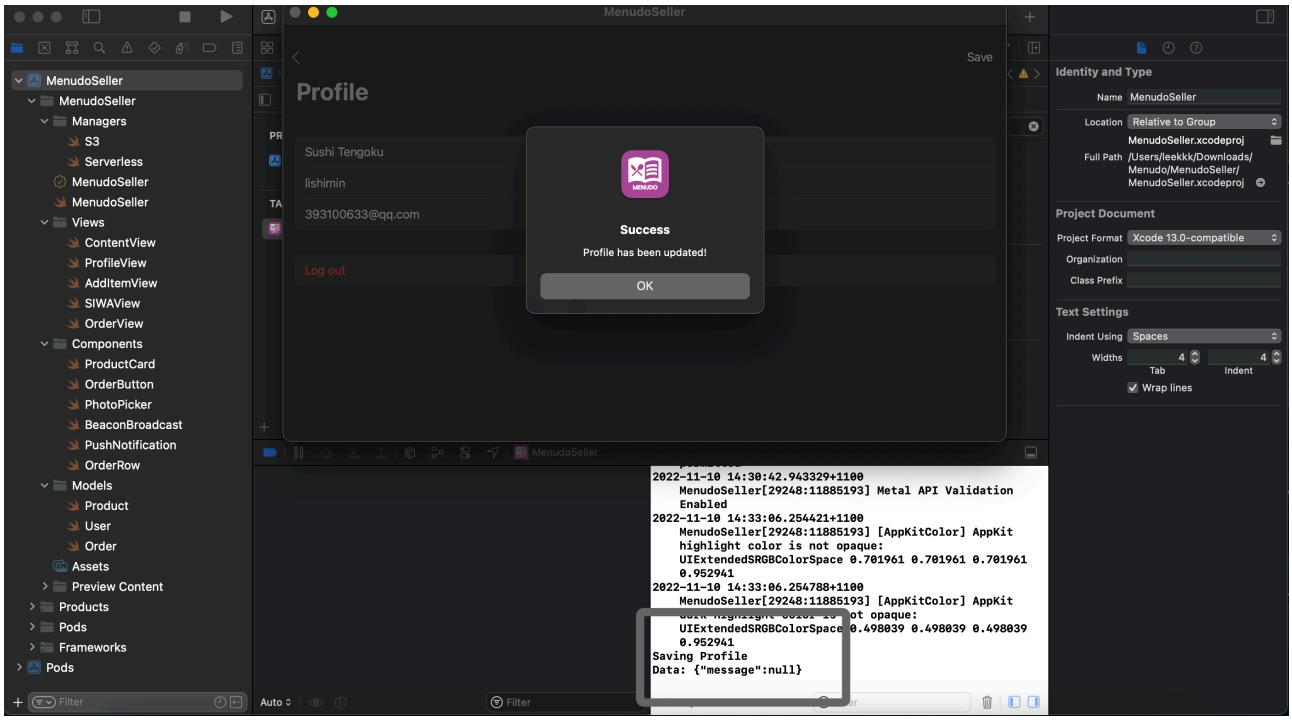



*Figure 10 Troubleshooting - Postman*

## Frontend Debug, Integration Debug

The build time warnings and errors are for Swift related problems, such as formatting, null pointer exception, infinite loop. The print output is debugging for connection to backend to display error or success response. Do-catch block is also used in code to avoid exceptions occur. The system is simulated on both of iPad and iPhone to ensure the layout and user interactions are compatible in cross platforms of iOS environment.



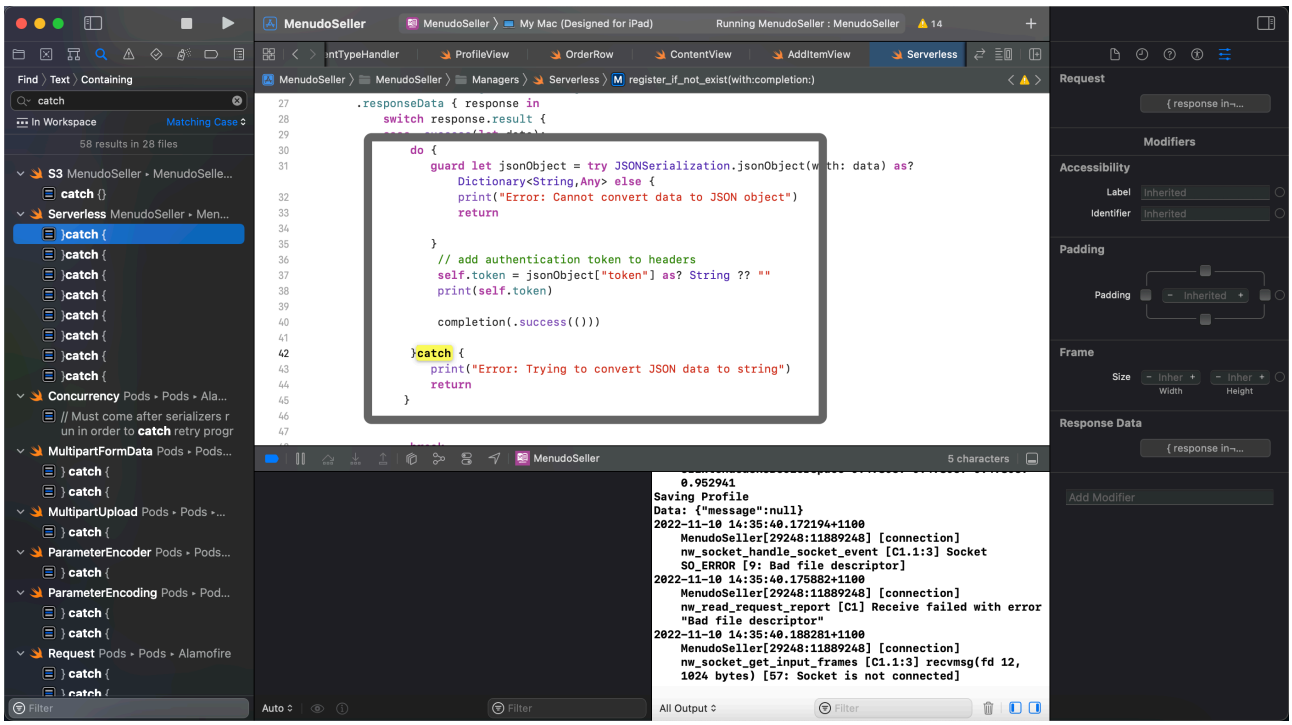


*Figure 11 Troubleshooting - Swift*

In addition, the user acceptance review gives me precious feedback to mend the flaws of Menudo. Here are a few examples.

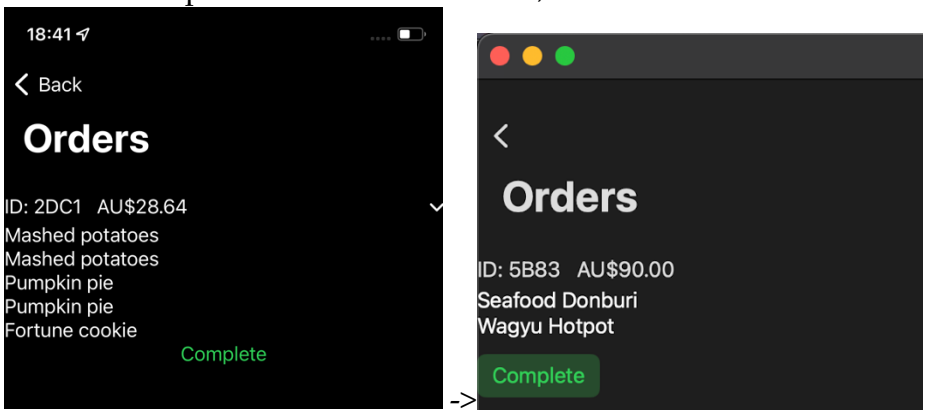
(1) *RemoveFromCart* function will delete all items of the same id, which means if user added the same item multiple times, delete one of them will in fact delete all. The logic has been mended to delete single item only.

```
18
19      func removeFromCart(product: Product) {
20          products = products.filter { $0.id != product.id }
21          total -= product.price
22      }
23
```

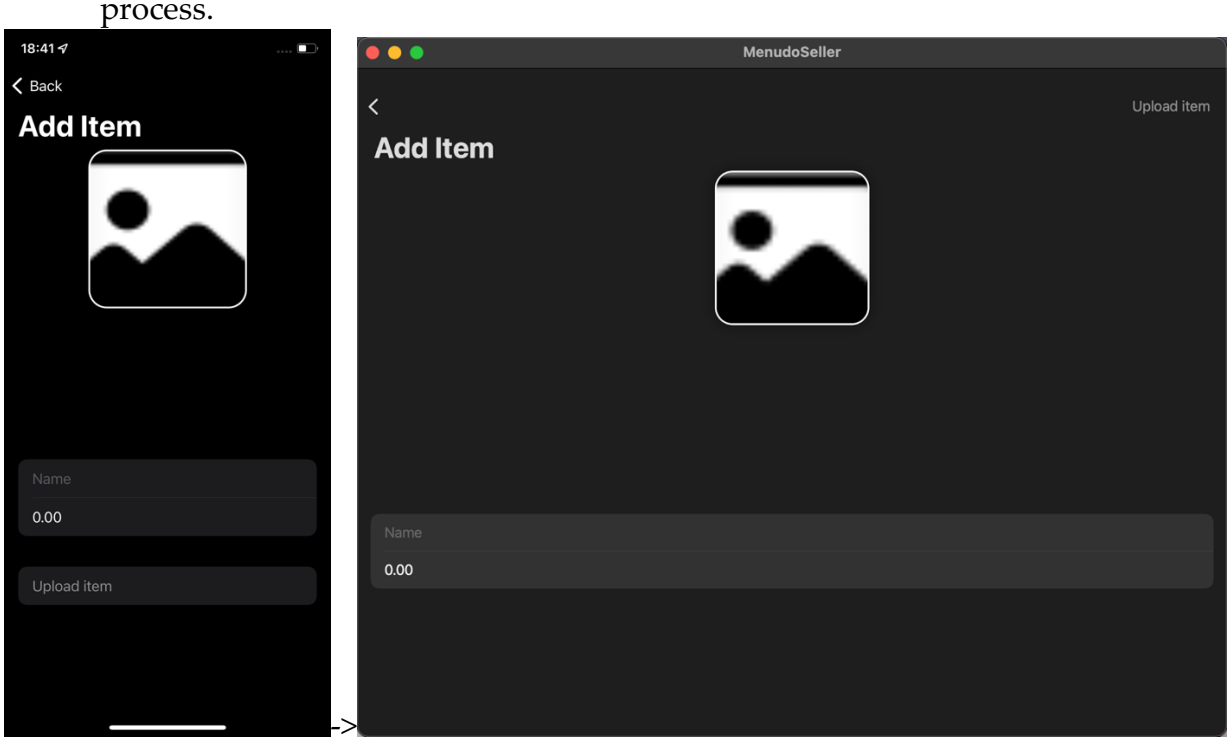
->

```
24      func removeFromCart(index: Int) {
25          let product = products.remove(at: index)
26          total -= product.price
27      }
28
```

(2) *Complete* button in orders does not look like a button to tap on, more alike a status description. After modification, it becomes more button like.



(3) Upload item does not obey the common layout with the similar look to forms, it has been moved to top right corner. And button needs to be disabled during uploading process.




```
72                  print("Saving Profile")
73              }
74              .alert(isPresented: $presentAlert) {
75              Alert(
76                  title: Text("Success"),
77                  message: Text("Profile has been updated!"),
78                  dismissButton: .default(Text("OK")) {
79                      presentationMode.wrappedValue.dismiss()
80                  }
81                  )
82              }
83              .buttonStyle(.plain)
84              .disabled(disableForm || inProgress)
85          }
86      }
```

Note: Tests are made on different devices so the layouts are different.

## Unique Selling Points

(1) Menudo is a novel table ordering system with *no external hardware*. No installation or placement of equipment is required to implement this solution. All existing solutions require hardware such as RFID and QR code stickers, or tablets with monitor. The only hardware will be the smart devices which come with potential customers and restaurant's existing POS system. As a substitution, the in-built BLE beacon will be play an important role in the Menudo implementation.

(2) Restaurant owners' side, Menudo will save expense on human resources to deliver menus and taking orders.

(3) Diners' side, Menudo will give the advantage of previewing menus before choosing a restaurant, order promptly in an elegant and convenient way.

(4) The backend is handled by the *AWS lambda*. The *serverless* architecture, the developing team does not have to maintain servers or infrastructure. The cloud provider is responsible for handling the operation of running the back-end logic, maintaining, scaling as per the traffic, and security.

## User Acceptance Review

*Table 8 User Acceptance Review*

| | NAME | AGE | PROFESSION |
|---|---|---|---|
| **PERSONA 1** | WenYuan, Shi | 55 | Retired, bricklayer |
| | WenYuan tested the MenudoClient.<br>WenYuan said Menudo was a user-friendly application for elders with obvious fonts and styles. It was a relief to solve the data leaking risks with beacon broadcasted menus. There was upgrade in convenience for menu browsing, ordering and payment. | | |
| **PERSONA 2** | Gordon, Liang | 21 | Student |
| | Gordon tested both MenudoClient and MenudoSeller.<br>Gordon thought the overall performance is good, but the system was a bit lagging when processing data. It was in-built technology he had never noticed before as an Apple user. It was a magic for him to remove hardware completely during the whole dining experience. He also provided lots of suggestions to improve in user interface. | | |
| **PERSONA 3** | Chao, Li | 34 | Café owner |
| | Chao tested the MenudoClient and MenudoSeller.<br>Chao was impressed by the system. Menudo could bring great comfort and ease into the industry. It perfectly solved the problem of doodled or damaged stickers to be regularly replaced. Even though Babyccino was using iPads to handle payments and orders, Menudo was only suitable for iOS users which was definitely not yet applicable in real world situations. As a seller, more components could be integrated into the system, such as monthly revenue report. | | |

## Challenges, Concerns

### Privacy

Most BLE beacons have a static IP which everyone in the transmission area can receive when broadcasting. Hence, an attacker can mimic a trusted beacon, by using the same trusted IP and have access to private information. However, as mentioned in Technologies iBeacon section, beacon ranging is not allowed in background, and iBeacon with unknown UUID cannot be detected. As a result, iBeacon-based applications have the nature to avoid bombardment of ads or spams by simply closing the app or minimise to the background. It also ensures that only authenticated restaurants will be added to the detection list and broadcasting menus. Consumers will be only receiving useful information rather than the feeling of privacy has been invaded.

### Installation

Retailers in particular have had a tough time getting consumers to download their apps, and that has limited the reach of beacons in malls and other retail environments. (*Beacon Marketers Have New Ways around App Download Requirement*, n.d.) What is more, current privacy concerns are solved based on iBeacon which is only supported in iOS devices. iPhone only has an 18% market share in the global smartphone industry as of Q1 2022. (*26+ IPhone User & Sales Statistics (Fresh Data 2022)*, n.d.) This means Menudo is targeted at 18% of customers and restaurant owners who have to install the app and keep it in phones for future dining. Installation is going to be the crucial challenge of promoting Menudo in the market.

### Security

Although detection of unknown family of beacons is not allowed among Apple devices, it is possible using a third-party beacon scanning software. UUID and Major, Minor values will be revealed. It is called *Piggybacking*: An attacker listens to a beacon and captures the profile (e.g. UUID, Major, Minor) and adds them to another application without consent. There is a possibility of unknown user adds himself/herself into the restaurant list. However, the broadcasting information is not simply a message or website URL, if the hacker cannot mock identically to data format of registered restaurant in Menudo, the information will not be processed and seen by customers.

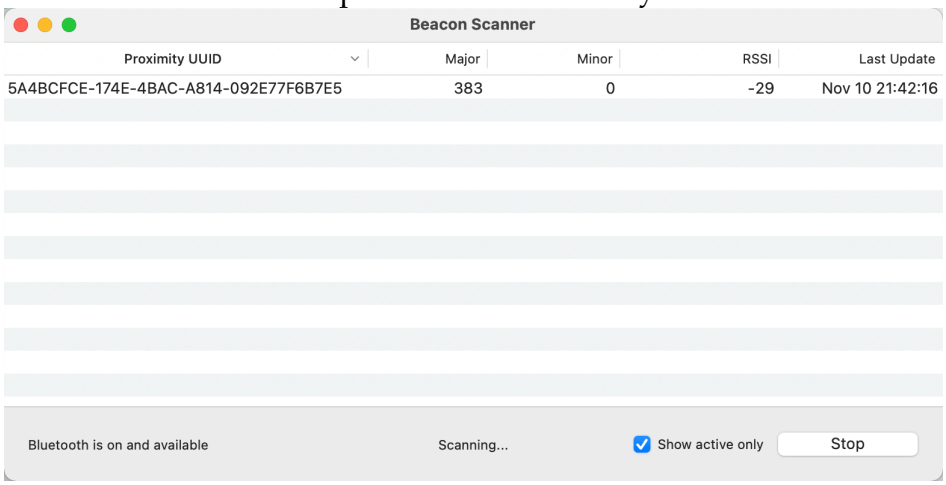



*Figure 12 Beacon Scanner*

## Improvements

### Top selling, Revenue report, Analysis

For seller side, there are a lot of functionalities can be developed considering real world usage. Top selling dishes can be displayed at the top of the menu page; customer habits can be stored for personalisation and customisation; restaurants can be provided with revenue report monthly or quarterly with detailed analysis which will be beneficial to further development for the enterprise.

### Push Notifications

When new order is sent, a pop-up push notification can be used to notify the seller to process it. In fact, modules related to push notifications have been added to backend and frontend, but it is not tested functional yet due to the time limit.

### Cross platform compatibility

Menudo completely relies on iOS environment. However, most people are not using Apple devices. Making Menudo compatible with Android devices would be a tough problem to solve and improve on.

### Range

Menudo supports detection within a couple of meters, which is "near" in *CLProximity* value to describe distance to a beacon. So that customer can only view a restaurant's menu at its door front. It can be implemented with "far" in in *CLProximity* value, which means greater than 10 meters away. In that case, customer will be able to receive a list of restaurants in a food court and select the one he/she interested in.

# Chapter 5 Conclusion

**Menudo** is an innovative table ordering system based on BLE Technology. With Menudo, sellers can broadcast the restaurant information, while diners can detect the menu instantly to their mobile device. No extra hardware is required during the entire dining experience. In this report, the design and development journey of Menudo is illustrated in detail. **Chapter 1 Introduction** gives a general overview of *status quo* and the opportunities in the industry. **Chapter 2 Background** further discusses the *target audience* and *user stories* of the application; *market* and *existing solutions* in the real-world situations; *technologies* going to be used in the application development. **Chapter 3 Design** takes the major weight in this report. It can be categorized into frontend and backend of the Menudo application. *Features* and *system architecture* are the distinguishing characteristic of Menudo ordering system. *Development tools* and are the libraries and packages used in this project. *Environment setup* explains the procedures to run the application on your terminal. *Data model* and *API endpoints* are defined and developed at the backend. *Wireframes*, *design pattern*, *UML* are all related to frontend implementation. *Integration* of frontend and backend is mentioned throughout the design chapter. **Chapter 4 Analysis** focuses on discussion after the application is implemented. *Troubleshooting* provides evidence of debugging process in the project. *Selling points* of Menudo are listed; *user acceptance review* gives a sneak peek of the application in normal people's eyes. A step forward, challenges, concerns and possible improvements are elaborated at the end of the report. It was a tough and challenging journey for me mingled with lots of fun.

# Bibliography

*10 Dining Trends To Watch In 2022 | Industry Report*. (n.d.). https://www.lightspeedhq.com.au/blog/10-dining-trends-to-watch-in-2022-industry-report/

*26+ iPhone User & Sales Statistics (Fresh Data 2022)*. (n.d.). https://www.demandsage.com/iphone-user-statistics/#:~:text=Key%20iPhone%20Statistics%20(2022)&text=Over%202.2%20billion%20iPhone%20units,industry%20as%20of%20Q1%202022.

*AWS Lambda The Ultimate Guide*. (n.d.). https://www.serverless.com/aws-lambda

*Beacon marketers have new ways around app download requirement*. (n.d.). https://marketingland.com/beacon-market-matures-marketers-drop-173651

Cha, S.-C., Chen, J.-F., Su, C., & Yeh, K.-H. (2018). A Blockchain Connected Gateway for BLE-Based Devices in the Internet of Things. *IEEE Access*, *6*, 24639–24649. https://doi.org/10.1109/ACCESS.2018.2799942

*Crash Course: RFID, barcodes, QR codes—What's the difference?* (n.d.). https://h41369.www4.hp.com/taw/article/UA/GB/TAW_000187

Dahlgren, E., & Mahmood, H. (2014). *Evaluation of indoor positioning based on Bluetooth Smart technology* [Chalmers University of Technology Department of Computer Science and Engineering]. https://hdl.handle.net/20.500.12380/199826

*How many restaurants are using an iPad POS system?* (n.d.). https://bouncepad.com/blogs/news/how-many-restaurants-are-using-an-ipad-pos-system

*HOW TABLE ORDERING APPS HAVE TAKEN OVER RESTAURANTS*. (n.d.). https://lunchbox.io/learn/restaurant-software/table-ordering-app

*How to build a serverless backend with AWS Lambda*. (n.d.). https://pusher.com/tutorials/serverless-backend-aws-lambda/

*IOS Architecture Patterns*. (n.d.). https://medium.com/ios-os-x-development/ios-architecture-patterns-ecba4c38de52

*Me&U*. (n.d.). https://www.meandu.com/

*Mobile and contactless payment options*. (n.d.). https://digitalready.tas.gov.au/resources/mobile-payment-options/

*Mr Yum*. (n.d.). https://www.mryum.com/

*NFC Food Ordering Tags*. (n.d.). https://www.tagthose.com/nfc-food-ordering-tags/

Powar, J., Gao, C., & Harle, R. (2017). Assessing the impact of multi-channel BLE beacons on fingerprint-based positioning. *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 1–8. https://doi.org/10.1109/IPIN.2017.8115871

*Proximity-marketing-what-how-why*. (n.d.). https://www.unacast.com/post/proximity-marketing-what-how-why

Richards, D. (2021, October 31). Balmoral Pasture Menu Now Fraught With Data Risk. *Channel News*. https://www.channelnews.com.au/balmoral-pasture-menu-now-fraught-with-data-risk/

Salmane, K. (2021, May 6). *RFID vs. NFC: What are the 5 Key Differences?* https://blog.nortechcontrol.com/rfid-vs-nfc

Siekkinen, M., Hiienkari, M., Nurminen, J. K., & Nieminen, J. (2012). How low energy is bluetooth low energy? Comparative measurements with ZigBee/802.15.4. *2012 IEEE*

*Wireless Communications and Networking Conference Workshops (WCNCW)*, 232–237. https://doi.org/10.1109/WCNCW.2012.6215496

Spachos, P., & Plataniotis, K. (2020). BLE Beacons in the Smart City: Applications, Challenges, and Research Opportunities. *IEEE Internet of Things Magazine*, *3*(1), 14–18. https://doi.org/10.1109/IOTM.0001.1900073

*Strategies for implementing user authentication in serverless applications*. (n.d.). https://www.serverless.com/blog/strategies-implementing-user-authentication-serverless-applications/

Subedi, S., & Pyun, J.-Y. (2020). A Survey of Smartphone-Based Indoor Positioning System Using RF-Based Wireless Technologies. *Sensors*, *20*(24), 7230. https://doi.org/10.3390/s20247230

*Table Tracker*. (n.d.). https://www.lrsus.com/location-tracking/table-tracker/

Welles, M. (n.d.). *BeaconScanner—IBeacon Scanning Utility for OSX*. https://github.com/mlwelles/BeaconScanner

## Diagram template references

System Architecture
https://app.diagrams.net/
Wireframe
https://www.figma.com/community/file/1117545712205927220
UML
https://www.figma.com/community/file/1080957210729895033
Relational Database diagram
https://www.figma.com/community/file/976371040144444424