**DT265 Systems Analysis and Testing Phase I Submission**

**Semester 2 2015/2016**

**Gavin O’Neill - D15127205**

**The Happy Sumo Japanese Restaurant**



Table of Contents

[1. General Project Information 1](#_Toc445219280)

[2. Research Conducted 4](#_Toc445219281)

[3. Model Developed 5](#_Toc445219282)

[3.1 Use Case Diagram 5](#_Toc445219283)

[3.2 Use Case Narratives 6](#_Toc445219284)

[3.3 Activity Diagrams 8](#_Toc445219285)

[4. Entity Classes 10](#_Toc445219286)

[5. Guidelines for developing required artefacts 11](#_Toc445219287)

# 1. General Project Information

**1.1 Team Details**

My group has a total of five members, the others being:

* Zheer Zhang – D15126205 – DT8265
* Kevin Walsh – C07393946 – DT8265
* Godwin Otite – D13126293 – DT8265
* Niall Thompson – D15127211 – DT8265

**1.2 Project Description**

**The Happy Sumo Japanese Restaurant**

As a group we decided to develop an online system for a Japanese restaurant, located in Dublin, which would allow its customers to place orders online for either delivery or collection and also give users the facility to book a table in the restaurant. This restaurant is an established restaurant already, however, the proprietors feel it is now essential to have an online presence to stay in touch with their main competitors.

In carrying out this project, I made the assumption that the restaurant has the staff and infrastructure to handle the additional business that will be generated by the new website including delivery drivers, vehicles and training for staff on how to use the system.

The restaurant already uses Just-Eat.ie to take online orders but the owners would like more control of their content, connect the website to social media accounts for marketing purposes and also have additional features such as booking tables, contact forms, etc.

Staff will be alerted by the system when an order is received and must then manually accept the order before it can be finalised. This will allow the staff to reject any order containing any item which is out of stock and the staff member can enter an explanation to assist the customer. The customer will receive a message advising him of the situation and will be prompted to update their selection by the system.

For delivery orders there will be a specified delivery area served by the restaurant, designated by the relevant Dublin area codes. A customer will be required to enter their area code when making a delivery order and if they do not fall under the required area then the order will not be processed.

The system will have a built-in function for handling bookings which can be edited by the staff. The staff will decide how many 2, 4, 6 or 8 person tables the restaurant will have available and the system will allocate bookings automatically. For odd number bookings the system will assign that customer to a table of the next biggest even number and the staff can then organise the restaurant seating to their own preference. For bookings of more than 8 people the customer will be prompted to phone the restaurant.

With regard to restaurant stock, I contemplated including an automated stock ordering function within the system, however, unlike other businesses such as retail, the restaurant doesn’t scan items in and out as they are used so it is almost impossible to manage. The restaurant also orders a lot of ingredients fresh on a daily basis from local suppliers who are not set up for business to business e-commerce.

*[Project Description – Include:*

*The proper name used to identify this project*

*Briefly outline what your project is about*

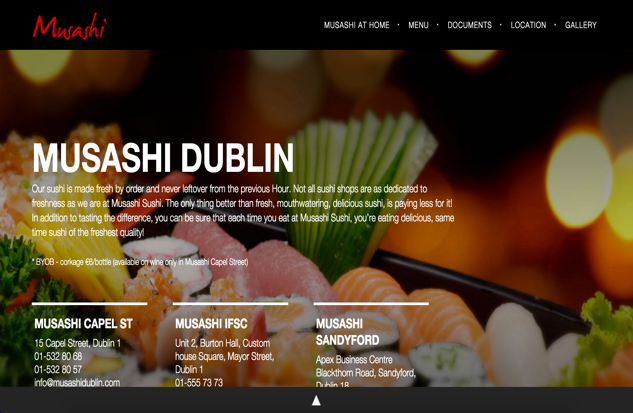
Outline briefly any assumptions/constraints you are making in relation to this project. E.g. third party providers etc.

*Outline any restrictions on the scope you are considering. This may be related to what you chose not to include as well as issues related to the nature of the product being developed E.g. focusing solely on addressing personal customers not business to business.]*

# 2. Research Conducted

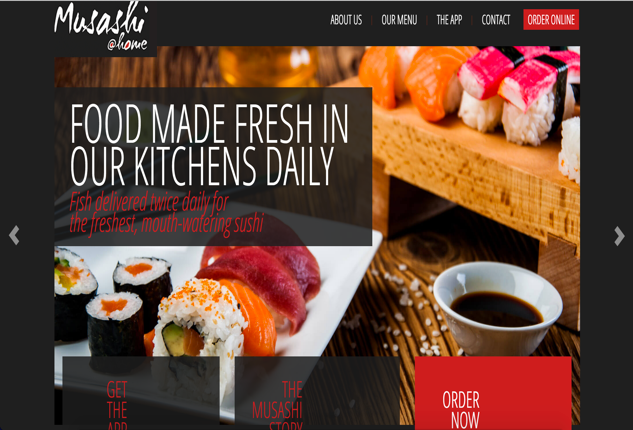
**The Competition**

To start off my research into the main competitors of the restaurant, I tried a simple Google search for Japanese food delivery which yielded surprisingly few results. It seems that most places either use the third party provider Just-Eat.ie for their online orders or if they do have a website, it only has functionality to book a table. In the end, I finally found two competitors with their own stand-alone food ordering capabilities which I will now discuss further.



First, we’ll look at Musashi which has three restaurants in Dublin and a contemporary looking website. What immediately strikes me as unusual is that on the main website (<http://musashidublin.com/)> you can view the menu, details of the restaurants and book a table, however, when you click the option for “Musashi at Home” you are re-directed to another website (<http://www.musashiathome.ie/>) designed with the sole purpose of taking your food order.

Main website

**

Both of Musashi’s websites are well laid out, very minimalistic with only the essential information displayed. They are easy both to navigate and to use for experienced and novice users alike.

Food ordering website

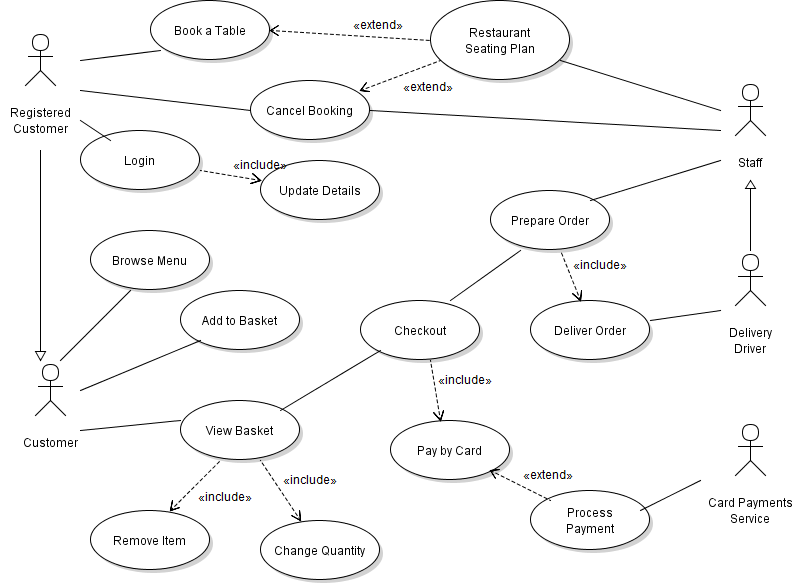
The second competitor website I looked at belongs to a restaurant named Michie Sushi, which also has three locations around Dublin city. What I noticed first off is that the homepage (<https://www.michiesushi.com/>) doesn’t have any link in the navigation bar to either the food menu or contact details and in fact to see the menu you must click the “Order Online” button. The restaurant appears to have won lots of awards for it’s food and the website shows this through large images and graphics on every page. I understand why they wanted to do this but the amount of images, etc, was too much in my opinion and distracts from the food ordering aspect.

From their ordering page you would first need to select your nearest store location, you could add items to your basket and continue from there to the checkout. You were then presented with an option to login or continue as guest which is how I plan to design the website for ‘The Happy Sumo’. I noticed that some links on this website would lead to an unfinished page, such as when clicking on ‘Delivery Information’, with just text displaying ‘Coming Soon’. This seems a little unprofessional to me and as a user I would be unimpressed that a website didn’t provide this seemingly important piece of information.

There is no function to book a table on the website so this is an are that ‘The Happy Sumo’ can take a step ahead of it’s competitor.

# 3. Model Developed

## 3.1 Use Case Diagram

**In the above Use Case diagram I have two different actors that are specialisations of another actor. First off ‘Registered Customer’ is a specialisation of ‘Customer’ because I wanted it to inherit all of the base actor’s use cases while making it clear that only a ‘Registered Customer’ can manage table bookings. Secondly ‘Delivery Driver’ is a specialisation of ‘Staff’ because only this person can deliver an order and they share all use cases of ‘Staff’ too.

The ‘View Basket’ use case has two includes relationships with the abstract use cases ‘Remove Item’ and ‘Change Quantity’. These abstract use cases are only seen in conjunction with ‘View Basket’. The same can be said for the ‘Login’ use case which has an includes relationship with the abstract use case ‘Update Details’.

‘Process Payment’ has an extends relationship with ‘Pay by Card’ and this is because it will only come into play if ‘Pay by Card’ is enacted. It extends the functional capabilities of the original use case. The same can be said about ‘Restaurant Seating Plan’ which is only used when the ‘Book a Table’ or ‘Cancel Booking’ use cases have been initiated.

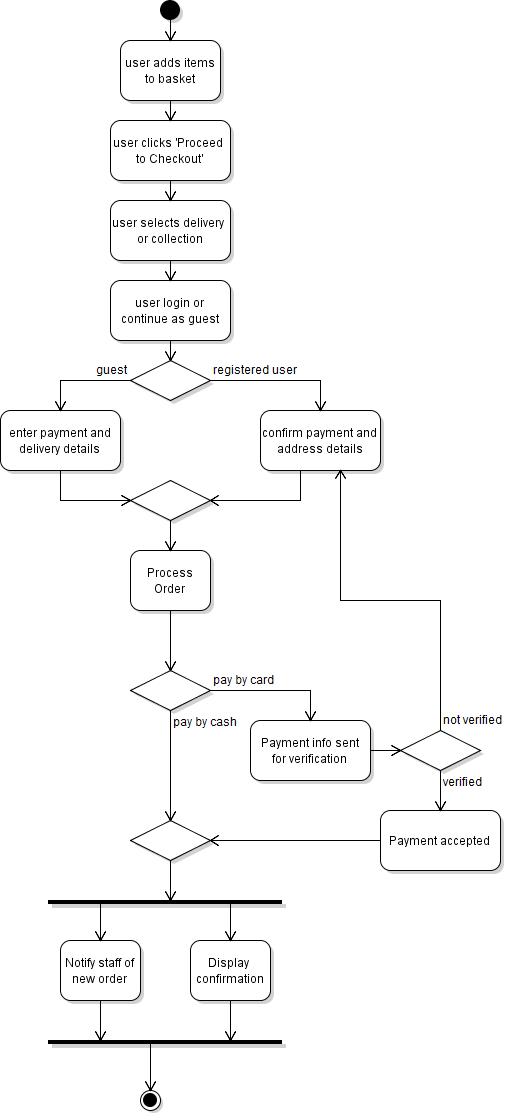
## 3.2 Use Case Narratives

|  |
| --- |
| **Use Case Name:** Checkout |
| **Intent:** Complete transaction and process order through the system |
| **Pre-condition:** User has added their desired items to the basket |
| **Use Case Initiation:** User clicks ‘Checkout’ option from ‘View Basket’ screen |
| **Dialog:**   1. User is viewing their basket and clicks ‘Proceed to Checkout’. 2. System asks user to select delivery or collection. 3. System then asks user to login or continue as guest. 4. User logs in and is presented by the system with their saved payment and delivery information. 5. User confirms or edits their information and then confirms the order. 6. System processes order. 7. Upon completion user is shown on-screen confirmation message by system and restaurant staff receive notification of new order from the system.   **Alternate Flow:**   1. User hasn’t selected any items so system displays “Your Basket is empty, please add an item(s) to continue”.   4. User has chosen to continue as guest so is required to enter delivery address and payment details if paying with card.  5. User cancels the order, system returns to menu screen with items remaining in the basket.  6. If user has opted to pay by card then system passes the payment information to the payments partner for validation and processing. If this process doesn’t complete then system displays this message to the user and requests the customer to try again. |
| **Use Case Termination:** The use case is finished once the transaction is completed. The user can exit back to the website by closing the last window which the system will have displayed. |
| **Post-condition:** Normal termination results with the order being processed through the system for the restaurant staff to fulfil.  If the order is cancelled at any stage nothing will pass through the system. |

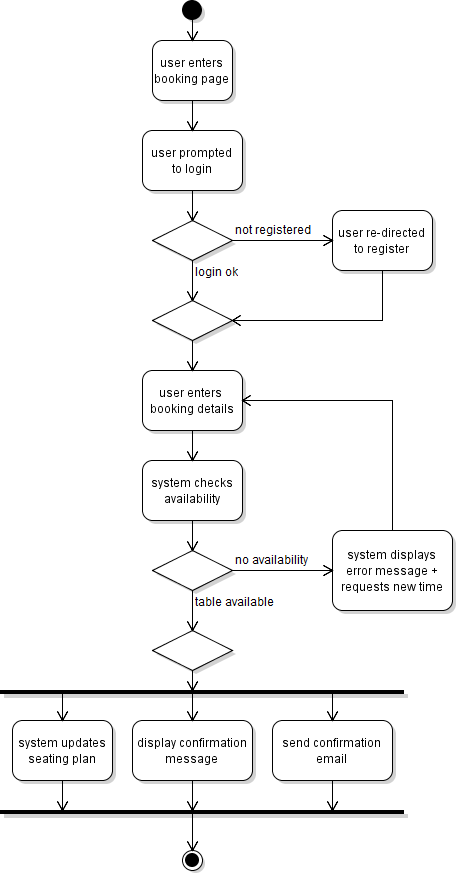
|  |
| --- |
| **Use Case Name:** Booking a table |
| **Intent:** Reserve a table in the restaurant |
| **Pre-condition:** User is on the website homepage. User must be registered to use this feature |
| **Use Case Initiation:** User clicks the button to “Book a Table” |
| **Dialog:**   1. System presents the user with a login screen. 2. User enters their login details which are verified by the system. 3. System displays the user’s saved contact information and requests booking date, time and number of people for table reservation. 4. User must confirm or update their contact info and choose their desired reservation details. 5. System validates the user’s input and checks the seating plan for availability. 6. The seating plan for that date is updated in the system with the user’s request. 7. The system then displays a confirmation message on-screen for the user and informs that an email has been sent to their registered email address. 8. The staff will be able to view the updated seating plan whenever they need.   **Alternate Flow:**   1. If the user is not already registered they will not be able to continue and are re-directed to the registration page.   5a. Data received by the user is in an incorrect format so user is asked for details to be re-entered.  5b. There is no availability at the chosen date/time so system displays this message to the user and requests an alternate selection. |
| **Use Case Termination:** Normal termination occurs when the system is updated with the reservation and user is notified.  If the user has never registered and is unwilling to do so then this use case is aborted.  If the user exits the booking page before the system displays a confirmation message then the use case is aborted. |
| **Post-condition:** The user receives a confirmation email regarding his reservation. The restaurant seating plan is updated within the system and can be viewed by the staff at any time. |

## 3.3 Activity Diagrams

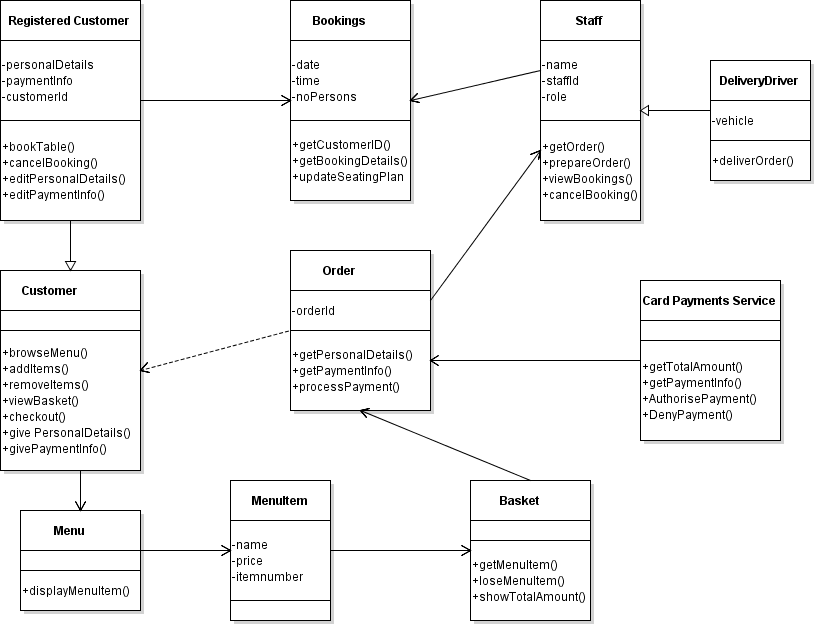
## **Checkout**



**Booking a Table**



# Entity Classes



For my above first cut class diagram I used both my use case diagram and narratives as an aid to deriving the classes involved. I tried to break down the requirements and identify the nouns as these are usually the entity classes. Typically entity classes would be described as modelling the key concepts of the system, they hold long lasting information and can be used in multiple use cases. In my own diagram this would apply to ‘Registered Customer’ whose info will always be kept, ‘Order’ as these are stored for accounting purposes, ‘Staff’ whose information will be kept long term and ‘MenuItem’ which once created can be stored long term and added/removed from the menu when desired.

In placing operations into each of my classes I first tried to make a list of all the action or doing verbs from my narratives and used these to figure out what I needed in my diagram. The operations describe the relationships between the different classes and objects in the system.

# 5. Guidelines for developing required artefacts

*[Include here your guidelines of issues to bear in mind or any advice you would give to novices when* ***developing the type of artefacts*** *you were required to develop.*

*These are guidelines for creating the artefacts for any project. You are expected to use your experience of developing the artefacts for this project to derive* ***general guidance – hints, tips, advice, things to avoid, useful explanations, etc****.]*