

**EE4717 Web Application Design**

**Project Report**

F33-DG08

A Saravanan U1322025E

M Rubern Chakravarthi U1322592K

Project Title: How’s Catering Online Ordering System

Summary of Project:

The Online Ordering System is a web-based ordering system that supersedes the current phone based ordering system that is in place. Customers are able to easily view the entire menu and are able to interactively select the package and dishes that they wish to order.

Contents

[Chapter 1 Application Requirements and Specifications 1](#_Toc434080305)

[Requirement 1 - Provide a web platform for How’s Catering 1](#_Toc434080306)

[Requirement 2 – Provide the full menu 1](#_Toc434080307)

[Requirement 3 – Online Ordering 1](#_Toc434080308)

[Requirement 4 – Transaction Enquiry 1](#_Toc434080309)

[Chapter 2 Functional Requirements and Specifications 2](#_Toc434080310)

[Requirement 1 - Provide a web platform for How’s Catering 2](#_Toc434080311)

[Requirement 2 – Provide the full menu 2](#_Toc434080312)

[Requirement 3 – Online Ordering 2](#_Toc434080313)

[Requirement 4 – Transaction Enquiry 2](#_Toc434080314)

[Chapter 3 Design of the Web Application 3](#_Toc434080315)

[Site Map 3](#_Toc434080316)

[Story Board 4](#_Toc434080317)

[Home Page 4](#_Toc434080318)

[Menu 4](#_Toc434080319)

[Transaction Enquiry 5](#_Toc434080320)

[Contact Us 6](#_Toc434080321)

[Ordering 7](#_Toc434080322)

[Wireframe 8](#_Toc434080323)

[Database 9](#_Toc434080324)

[Database Design Considerations 9](#_Toc434080325)

[Web Application Testing Plan 12](#_Toc434080326)

[Testing Requirement 1 12](#_Toc434080327)

[Testing Requirement 2 12](#_Toc434080328)

[Testing Requirement 3 12](#_Toc434080329)

[Testing Requirement 4 12](#_Toc434080330)

[Chapter 4 Implementation 13](#_Toc434080331)

[Index 13](#_Toc434080332)

[Menu 14](#_Toc434080333)

[Ordering 15](#_Toc434080334)

[Transaction Enquiry 16](#_Toc434080335)

[Chapter 5 Testing of the Web Application 17](#_Toc434080336)

[Incremental Process 17](#_Toc434080337)

[Agile Programming 17](#_Toc434080338)

[Normalize.css 17](#_Toc434080339)

[Achieving Test Requirements 17](#_Toc434080340)

[Testing Requirement 1 17](#_Toc434080341)

[Testing Requirement 2 18](#_Toc434080342)

[Testing Requirement 3 18](#_Toc434080343)

[Testing Requirement 4 18](#_Toc434080344)

[Chapter 6 Conclusion 19](#_Toc434080345)

Appendix A - Wireframe Diagrams

Appendix B – Source Code Listing

Appendix C - Contributions

# Application Requirements and Specifications

The objective of building the web application is to create a web presence for How’s Catering and setup an online catering ordering system. The web application aids the user in ordering process by computing the cost in real time and showing images of items chosen immediately. This system would supersede the current phone based ordering system the company has in place. The requirements for the application are listed below.

## Requirement 1 - Provide a web platform for How’s Catering

Users should able to easily identify that the website belongs to How’s Catering and understand that the company provides high quality catering services.

## Requirement 2 – Provide the full menu

Users should be able to view the full menu that How’s Catering offers easily. In addition to that, users would be able to be able to see an image of the items in the menu which will further help them in taking decisions.

## Requirement 3 – Online Ordering

Users should be able to efficiently and smoothly order using the website.

## Requirement 4 – Transaction Enquiry

Users should be able to enter the transaction number provided at the end of the transaction and view the details of the orders.

# Functional Requirements and Specifications

The following paragraphs will detail how the application requirements specifications will be achieved.

## Requirement 1 - Provide a web platform for How’s Catering

The landing page of the web site will be designed such that users who view the page will be able to easily identify How’s Catering through the usage of the company logo as well as the match to the company’s color palette. A slideshow of the most popular dishes ordered by the customers as well as the company mission will be clearly shown on the page.

## Requirement 2 – Provide the full menu

The entire menu will be stored inside the SQL database. A separate page will be dedicated to showing the menu. The page would be broken down into different parts so as to reduce the time taken for each query.

## Requirement 3 – Online Ordering

How’s Catering sells it’s catering through different tiers of packages. The online ordering system will allow a customer to select the tier of package that he/she wishes and customize the menu within the parameters of the tier. After which the customer will be able to enter the delivery details as well as contact information. The payment portal will handle the payment. After confirmation of successful payment has been made, a transaction reference number will be provided and the information will be stored inside the database.

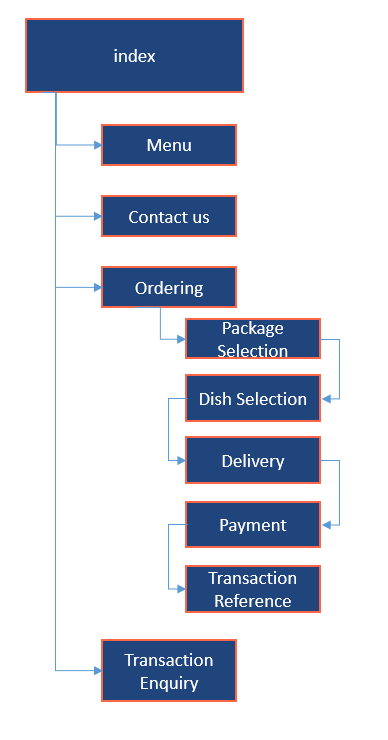
## Requirement 4 – Transaction Enquiry

Using the transaction reference number, the customer will be able to query for details of their orders efficiently. They will need to key in the number into the box and submit, after which an SQL query will be performed and the details of their transaction will be returned.

# Design of the Web Application

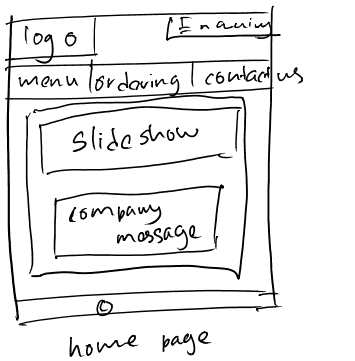
The following content show cases the design of the web application.

## Site Map

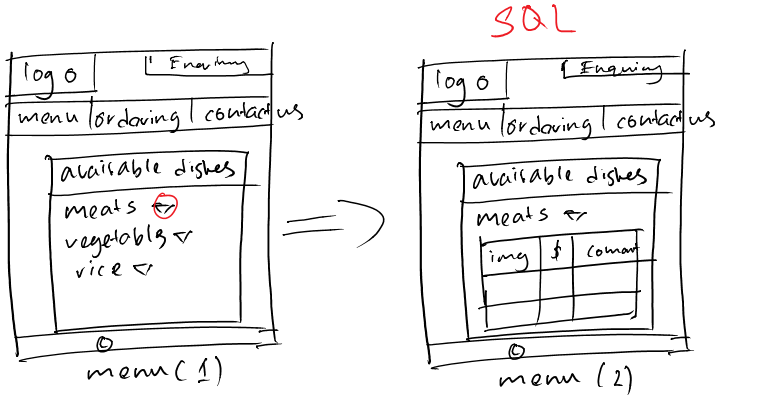


## Story Board

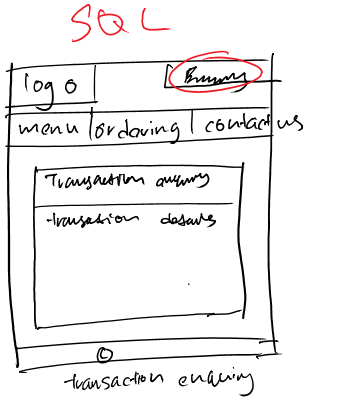
### Home Page



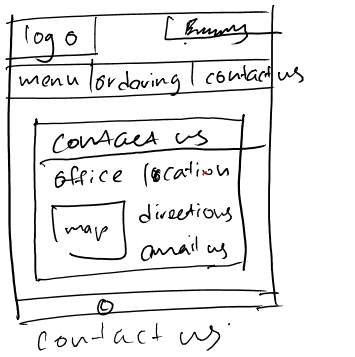
### Menu



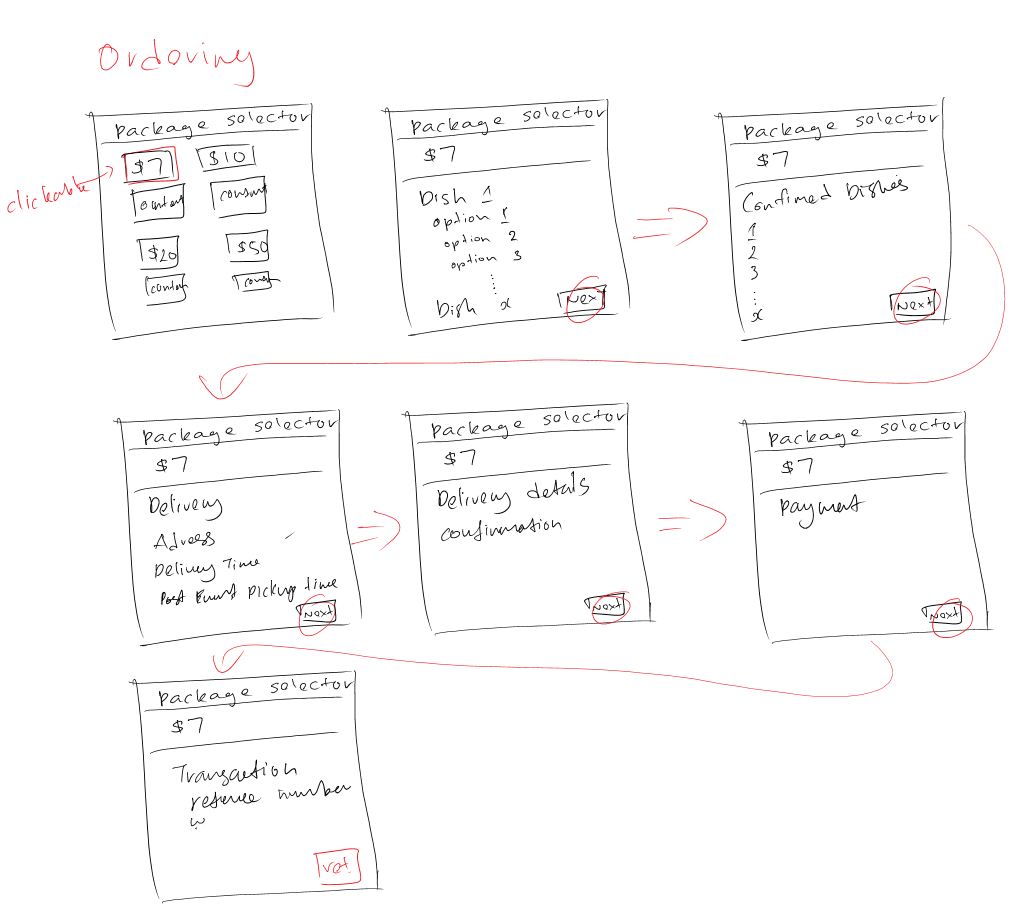
### Transaction Enquiry



### Contact Us



### Ordering



## Wireframe

Due to the size of the wireframe diagrams, they have been contained in Appendix A – Wireframe diagrams.

## Database

### Database Design Considerations

When designing the database, the following steps were used:

1. Gather Information
2. Identify the objects
3. Model the objects
4. Identify the types of information for each object
5. Identify the relationship between objects

#### Gather Information

In order to create a database, it is necessary to have a good understanding of the job that the database is expected to perform. To that end, we performed some business analysis on actual catering companies and tried to understand how they would have implemented a paper based system and tried to adapt that to a database system.

#### Identify the objects

During the process of gathering information, we identified the key objects that would be managed by the database.

* Packages
* Dishes
* Transactions

#### Model the Objects

Due to the nature of the project, it was decided that we would use a relatively simple model for the database.





#### Identify the types of information for each object

After the primary objects were identified, the next step was to identify the types of information that must be stored for each object.

Packages

* Package\_id as int
* Package\_name as char
* Package cost as tinyint
* Main\_1\_choices as char
* Main\_2\_choices as char
* Opt\_1\_choices as char
* Opt\_2\_choices as char

Transactions

* Transaction\_id as int
* Contact\_name as char
* Contact\_number as int
* Contact\_email as char
* Delivery\_adress as char
* Delivery\_time as char
* Collection\_time as char
* Pax as int
* Payment\_method as char
* Payment\_amount as int
* Payment\_content as text

Table

* Dish\_id as int
* Dish\_type as char
* Dish\_name as char
* Dish\_description as char
* Dish\_img\_location as char

#### Identify the relationships between the objects

Due to the relatively simple nature of the workflow, we did not need to create any relationships between the objects.

## Web Application Testing Plan

The objective of a testing plan is to provide a framework to ensure that the application requirements and specifications are met.

### Testing Requirement 1

Requirement 1 is to ensure that users are able to easily identify that the website belongs to How’s Catering and understand that the company provides high quality catering services. To ensure that this requirement is met, we will use a slideshow on the landing page to showcase the dishes available as well as place the company strategically around the web page. Also, the company colors will be used.

### Testing Requirement 2

Requirement 2 is to provide the full menu with images so that users will be able to view and decide on an item easily. To achieve this, the database will be used to hold the image location so that it can be easily displayed.

### Testing Requirement 3

Requirement 3 is to ensure that users would be able to efficiently and smoothly order using the website. To achieve that, we will break down the business function into smaller parts and ensure that they are implemented cleanly.

### Testing Requirement 4

Requirement 4 is to allow users to enter the transaction number provided at the end of the transaction and view the details of the orders. To achieve this, we will test the function to ensure that the output is generated.

# Implementation

The following sections will describe the main parts of the website, namely the index, the menu, the ordering pages and the transaction enquiry.

## Index

The index page serves as the template for all other pages. We used HTML5 elements such as *nav* and *section* when designing the layout.



Figure Simple Diagram to show layout elements used

The wrapper element is a *div* element that we used CSS to style to contain our content and allow us to control the maximum width easily. This allowed us to create a stable and easily controlled layout to deliver the rest of the content.

For the actual index page, we included some functionality such as a Javascript based picture slideshow as well as a text scroller to provide a brighter landing page for potential customers.

## Menu

The menu was implemented using PHP to interact with the database to dynamically generate a table containing the available items.

Figure 2 Some sample code to show how we dynamically generated the menu

<?php

session\_start();

@ $db = new mysqli('localhost', 'f33ee', 'f33ee', 'f33ee');

if (mysqli\_connect\_errno()) {

echo 'Error: Could not connect to database. Please try again later.';

exit;

}

$query = "SELECT \* from dishes";

$result = $db ->query($query);

$num\_results = $result->num\_rows;

echo "<table align='center' border='0'>";

for ($i=0; $i <$num\_results; $i++) {

$row = $result->fetch\_assoc();

echo "<tr><td>".$row['dish\_id']."</td>";

echo "<td>".$row['dish\_name']."</td>";

echo "<td>".$row['dish\_type']."</td>";

echo "<td>".$row['dish\_description']."</td>";

echo "<td><img src='".$row['dish\_img\_location']."'width='400' height='200'></td></tr>";

}

$result->free();

$db->close();

?>

By dynamically populating the table by accessing the database, it allows the webmaster to simply update the database to update the menu.

## Ordering

The ordering function is one of the main requirements of an online purchasing system. The activity diagram below describes the use case of a user ordering a package.



Figure User ordering a package use case activity diagram

The ordering system uses both SQL *select* and *insert* queries. The select statements are used to populate the dishes upon package selection. The insert statement is used to add the transaction into the database. There is Javascript form validation being performed whenever there is a user modifiable input.

## Transaction Enquiry

The transaction enquiry function is a simple SQL *select* query to the database to return the information held in the transaction table based on the transaction reference number given to the user.

# Testing of the Web Application

## Incremental Process

The testing of the page was continuously done from the very beginning of the project. There was continuous and iterative process of testing to check if each modification and addition to the code works.

A simple example would be the process of designing the overall layout of the webpage. Initially the layout was done fully by the use of dividers. But through the process of testing, many errors and limitations surfaced. Div elements results in a automatic line feed. This was initially solved by the use of <span>. However this was also tested to be not as efficient in dealing with the layout of the page.

Section (<section>) and article (<article) were used in coding the body of the pages. Reference was mage from <http://www.w3schools.com/tags/default.asp> in choosing the suitable element to come out with the framework.

## Agile Programming

There were lots of components that were involved in this project. Different components of a page itself was split in-between the group members and some components were worked on simultaneously. Changes made by one person had to be fitted in to another person’s code. This made it a challenging task to code simultaneously as a small incompatibility or change that was done affected the page display and at times resulted in the page not being displayed completely. It resulted in lots of time wasted to identify the errors.

We decided to use github to share the files, which helps to manage and combine the codes when two people are editing the same page at once. This saved lots of time from combining the codes.

## Normalize.css

The testing process of the pages included the testing of the same page on different web browsers. When these web pages are made live for customers to use, it is up to customer’s preference to use whichever Internet browser of their choice. As a web page designer our goal is to make sure that the page is displayed the same way that it is supposed to be in all available browsers.

Therefore, research was done to suite the page to be displayed on different browsers the same way as much as possible. Normalize.css is included in all pages in addition to main.css that was created by us to control how the page looks like. Normalize.css has codes specific to browsers so that all browsers will render a page as similar as possible.

## Achieving Test Requirements

### Testing Requirement 1

We were able to achieve this requirement through placing the company name strategically around the web site and using the company colors. We also used some testers to view the website and provide their comments and feedback to ensure that the requirement has been met.

### Testing Requirement 2

We were able to achieve this requirement by ensuring that the code was written correctly and able to display the menu and the images.

### Testing Requirement 3

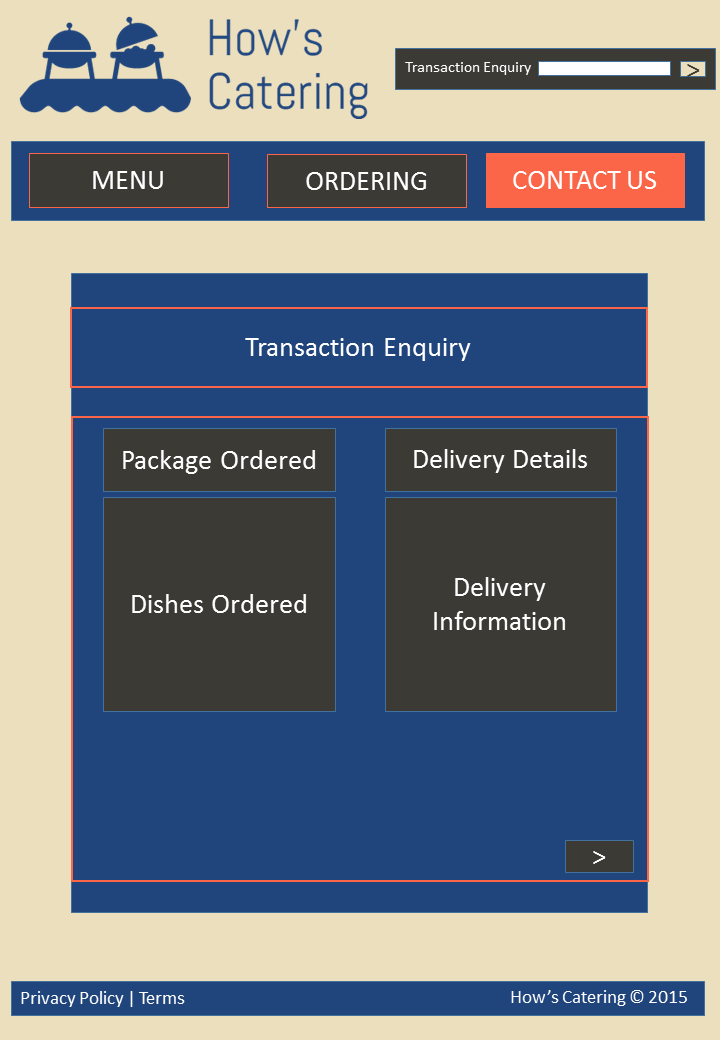
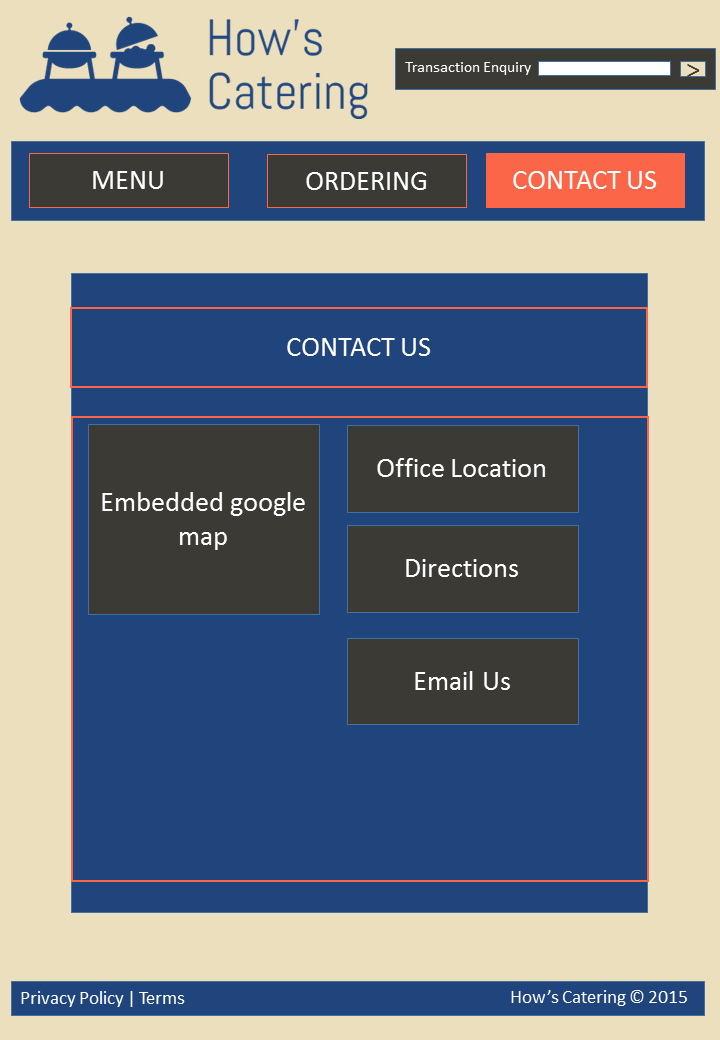
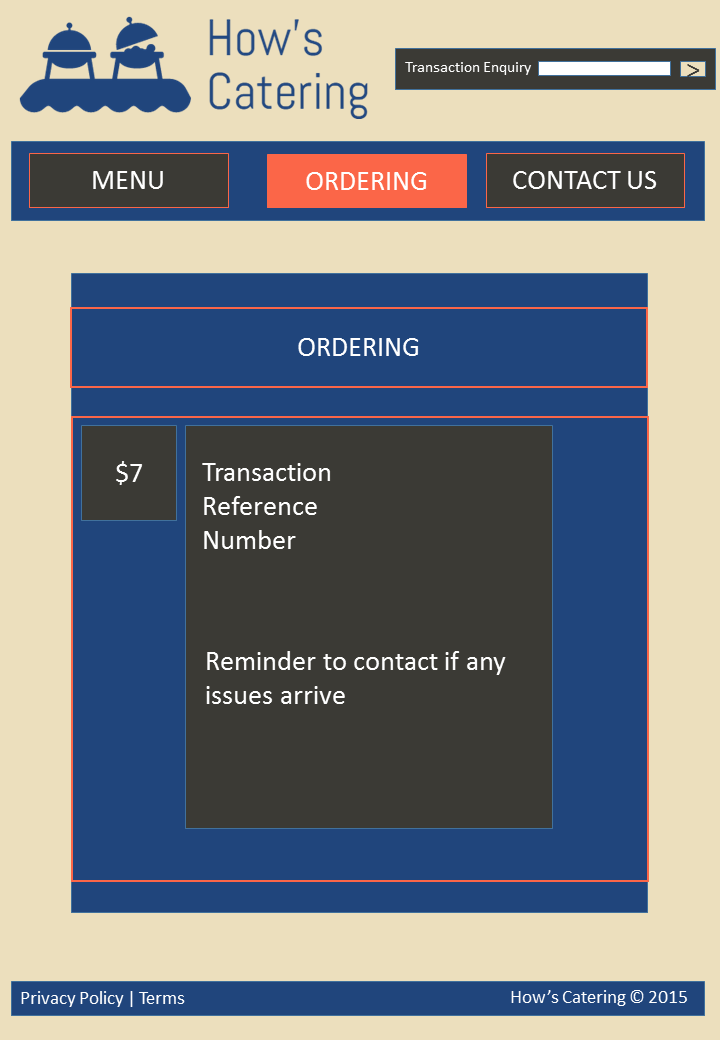
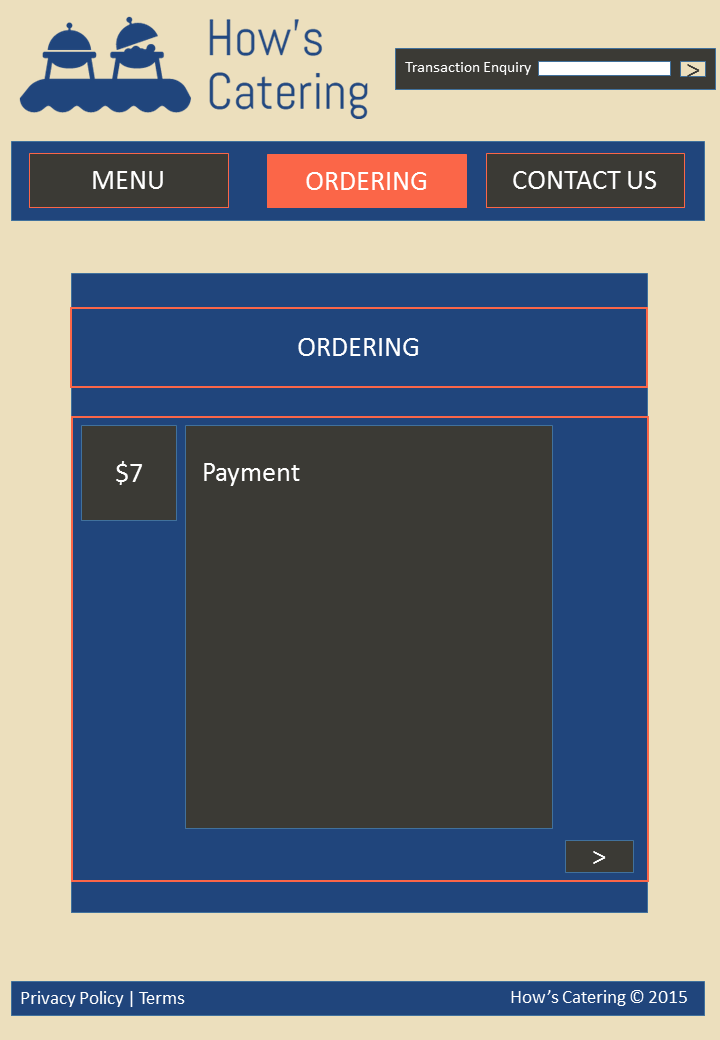
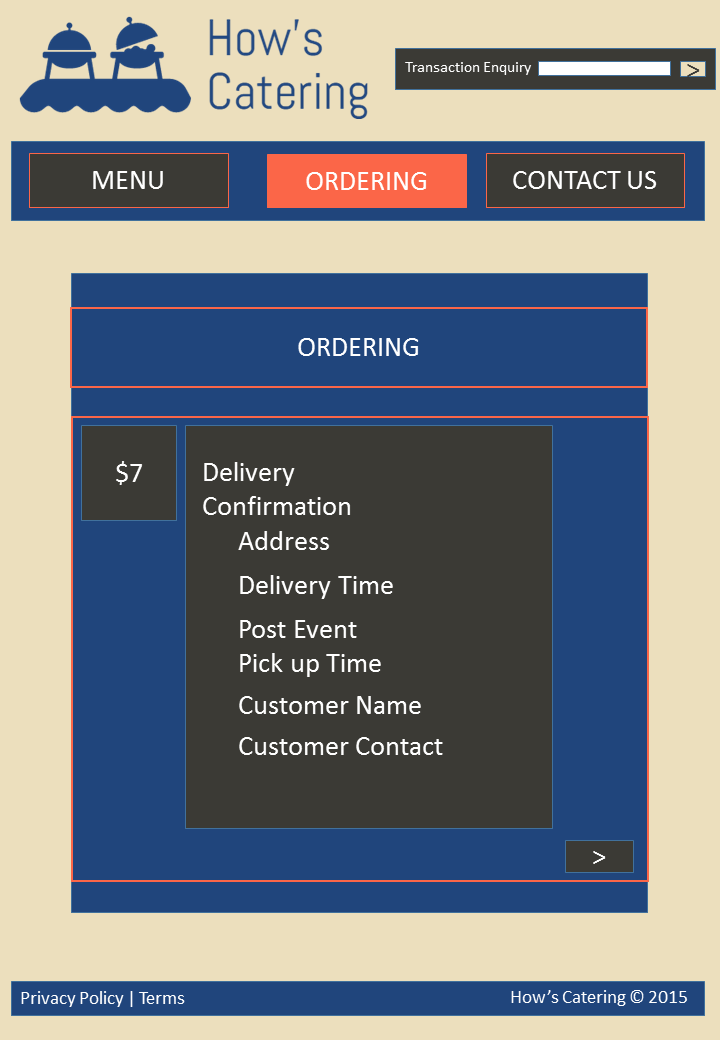
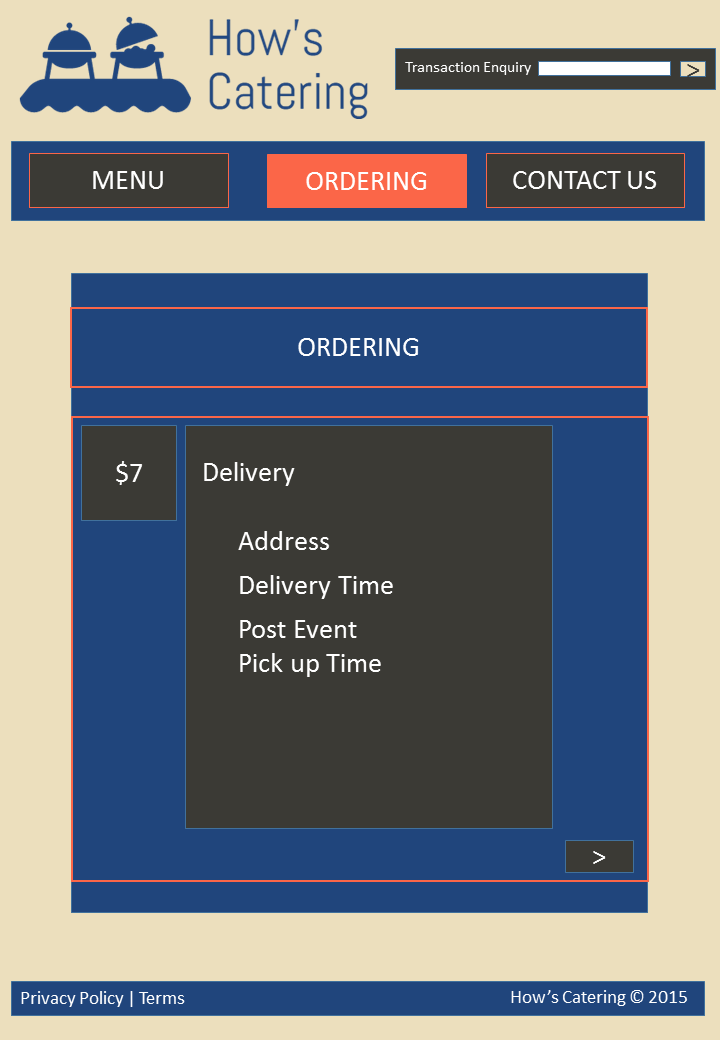
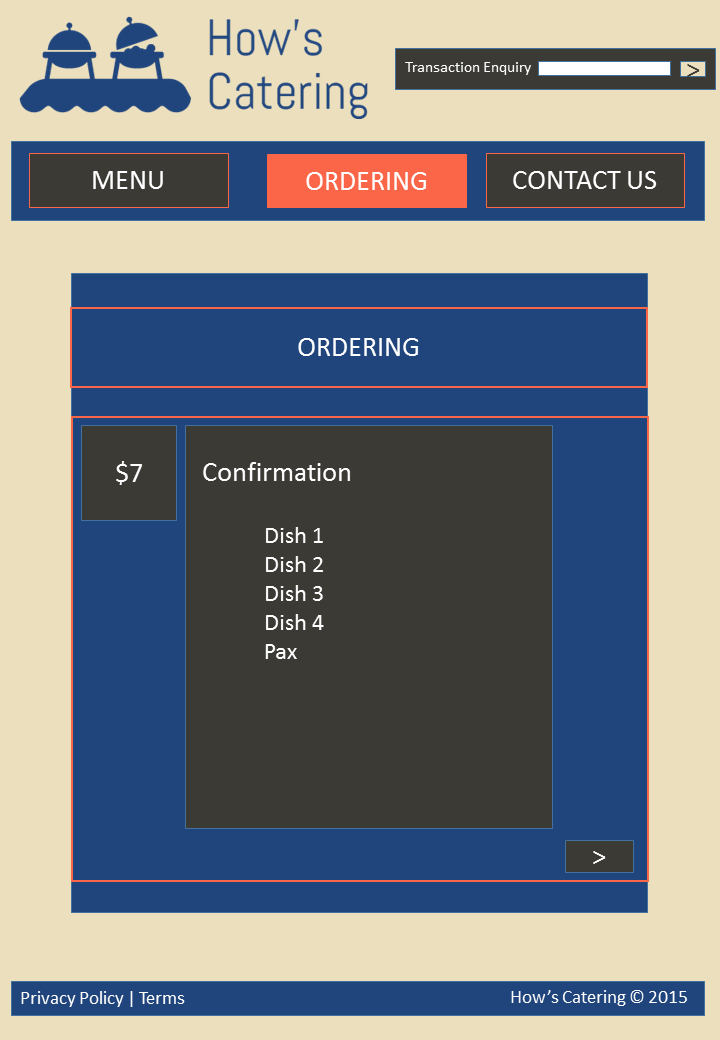
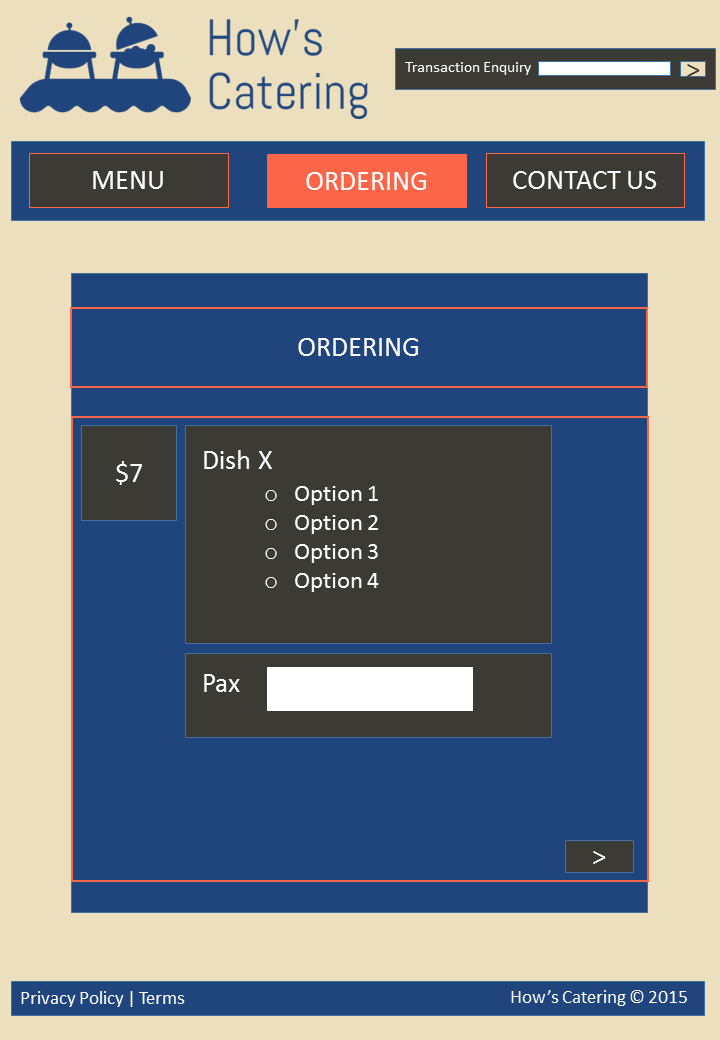
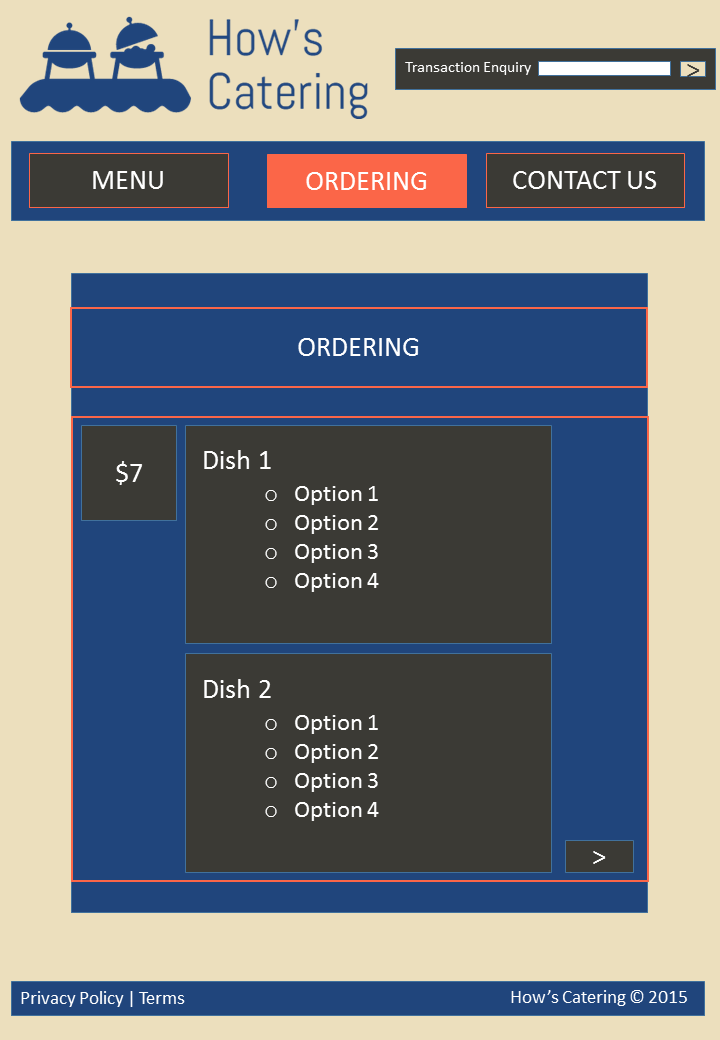
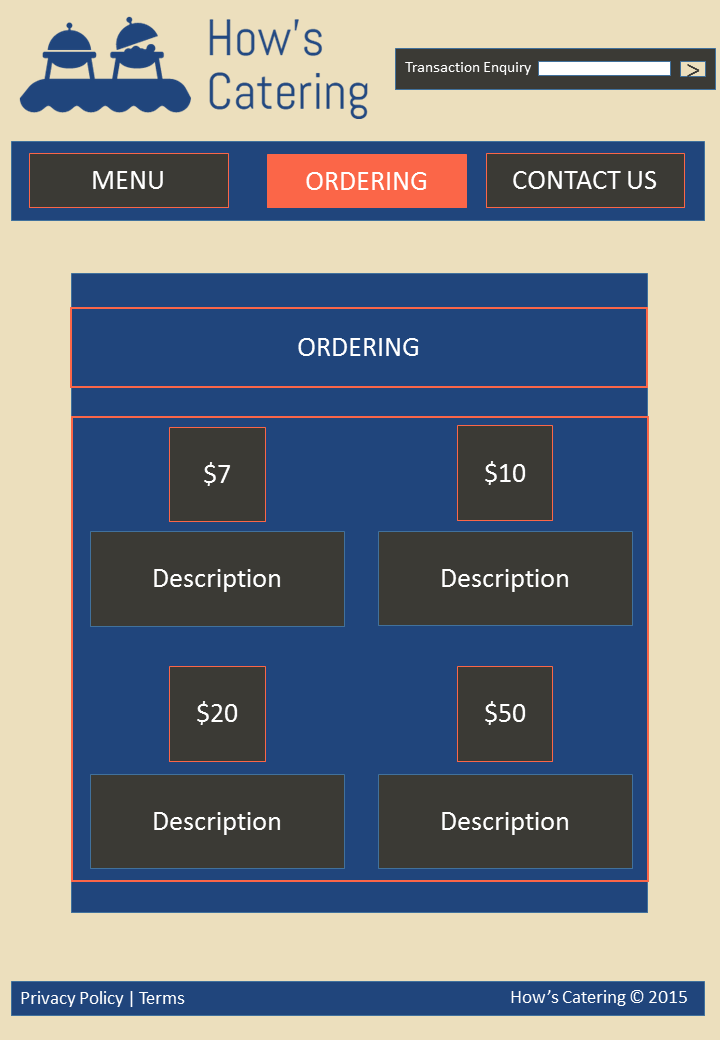
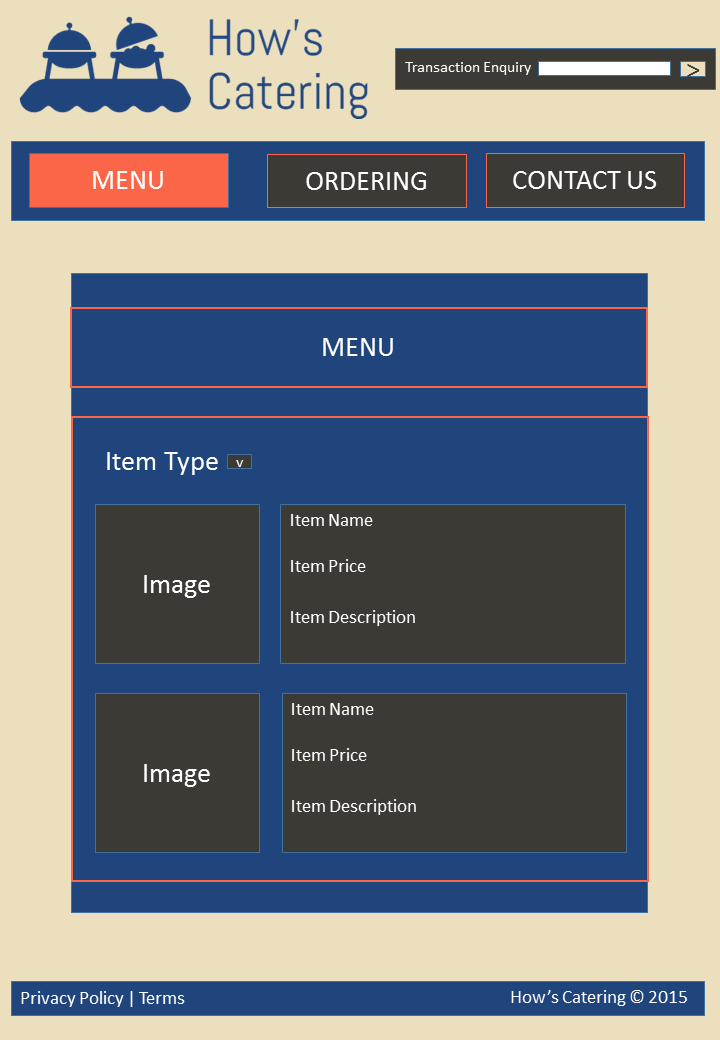
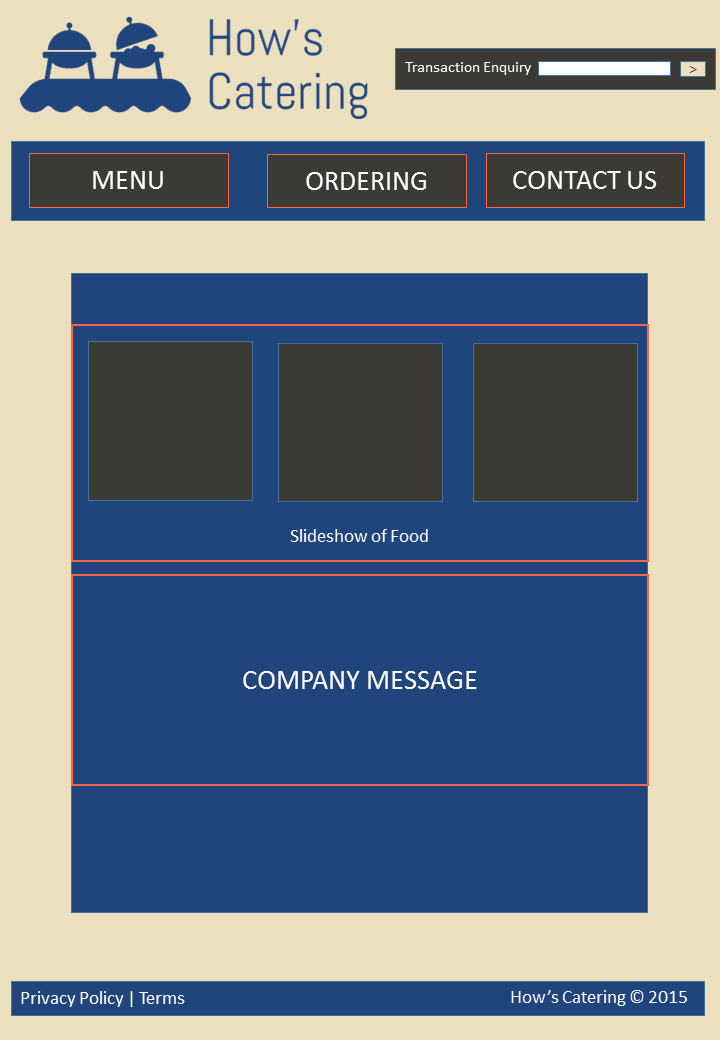
For this requirement, we used several testers to test the ordering process and give feedback and comments regarding it. After several rounds of iteration, we were able to come up with a process that was acceptable.

### Testing Requirement 4

We were able to achieve this requirement by ensuring that the code was written correctly and the transaction reference could be input and the details displayed.

# Conclusion

We were able to achieve all of the requirements that were specified during the design phase of the project. Through this project, we were able to learn how to meld both front end and back end web technologies to deliver a functional and aesthetically pleasing web application to a fictional customer.



Appendix B - Source Code Listing

Appendix C - Contributions

|  |  |
| --- | --- |
| Web Application – Segments of the Project | Name |
| Planning | BOTH |
| Web Application – index.html | BOTH |
| Web Application – menu.php | SARAVANAN |
| Web Application – ordering.php | RUBERN |
| Web Application – transaction\_equiry.php | BOTH |
| SQL | BOTH |
|  |  |