

## **Restaurant O SIX JOINT Online Ordering System with Anti-SQL Injection**

**Tang Jie Yi, Nurul Hidayah Ab Rahman\***

Faculty of Computer Science & Information Technology,  
Universiti Tun Hussein Onn Malaysia, Parit Raja, 86400, MALAYSIA

DOI: <https://doi.org/10.30880/aitcs.2021.02.02.013>

Received 15 June 2021; Accepted 09 September 2021; Available online 30 November 2021

**Abstract:** In the era advanced technology, e-commerce is becoming more popular all over the world including the food industry. Therefore, some restaurants have changed their traditional way of ordering food to an online ordering system. With the online ordering system, the operation of restaurants has become more efficient and it is convenient for customers to place an order quickly. In addition, online ordering also became a necessary condition for restaurants to survive, especially in the situation of coronavirus pandemic to reduce contact. However, the online ordering system is based on web service that has its own vulnerabilities such as SQL injection attacks. Therefore, an online ordering system which can minimize the risk called Restaurant O Six Joint Online Ordering system with Anti-SQL injection is proposed. The proposed system is developed in a web platform and adopts the Object-Oriented Software Development model as the methodology of this project. In this proposed system, the modules developed will be categorized for three target users which are user admins, staff and customers and another module which represents all of the target users. The significance of the project is to protect the web-based system from SQL injection attack by complying with the OWASP requirements. Moreover, other security mechanisms also implemented on the proposed system to achieve confidentiality, integrity, availability and authentication triad.

**Keywords:** e-commerce, online ordering system, OWASP, SQL injection

### **1. Introduction**

In this advanced modern technology era, e-commerce has become one of the world-famous commodity trading or service platforms as it has created new digital markets where prices are more transparent, the markets are global and efficient transactions [1]. Besides, the outbreak of coronavirus pandemic has a wide impact on e-commerce. According to TheStar, online retail sales in Malaysia increased by 28.9% since the start of MCO in April 2020 [2]. This would include online ordering food because of lockdown [3]. Therefore, a web-based ordering system will ease the ordering process and meet the SOP customized by the government. Furthermore, customers could access the website and make their order.

Although the restaurant online ordering system improves the quality of order experience for customers, some security problems should be considered. This is because most of the online web-based

---

\*Corresponding author: [hidayahar@uthm.edu.my](mailto:hidayahar@uthm.edu.my)  
2021 UTHM Publisher. All rights reserved.  
[publisher.uthm.edu.my/periodicals/index.php/aitcs](http://publisher.uthm.edu.my/periodicals/index.php/aitcs)

system does not fulfill the safety requirement and most of these systems are using database as their persistent storage of data [4]. A report revealed that the details of hundreds or thousands of credit cards from at least six Southeast Asian countries, including Malaysia, have been leaked online in an apparent massive data breach [5].

SQL injection is the most common threat that attacks the database server in the web-based system [6]. It must be taken seriously because this threat is also at the list of Top 25 Most Dangerous Software Errors [7]. Most of the web-based systems are vulnerable to the SQL injection. In general, SQL injection occurs when an unauthorized user gains access to the system by inserting some malicious inputs or leverages the syntax and capabilities of SQL which can affect the execution of predefined SQL commands [8]. This attack may cause the data inside the database to be modified and it against the integrity of data. Therefore, SQL injection must be resolved to make sure that the web-based system cannot easily be accessed by unauthorized users.

According to Open Web Application Security Project (OWASP) general guidelines, one of the countermeasures is focusing on the authentication form [9]. When the authentication is performed by the user using a web form, the user's identity is checked by comparing the username and password with the database. It provides chances for attackers to access into the system by brute force attack. Another potential solution is to limit login attempts and the account will be locked down if the limit of login attempts is over.

There are three issues identified from the web-based system which are manual ordering food processes, increasing web-based threats and sensitive data is leaked or altered by unauthorized access. In order to solve the issues, a proposed system, Restaurant O SIX JOINT Online Ordering System with Anti-SQL Injection will be developed. Another system which is vulnerable to SQL injection will be implemented as the comparison to initiate the impact of SQL injection. There are three objectives of this project – to design, to develop restaurant O SIX JOINT online ordering system with anti-SQL injection, and to test the effectiveness of anti-SQL injection for proposed systems by conducting the SQL attack simulation.

The modules developed in this proposed system will be categorized for three target users which are user admins, staff and customers and another module which represents all of the target users. The proposed system helps in overcoming the limitations of manual ordering systems and minimize the potential vulnerabilities that SQL injection may exploit. The countermeasures used to prevent the SQL injection, including using parameterized queries, input validation and minimize staff privileges. Strong password policy, limit login attempts and encryption password also implemented on the proposed system in order to achieve the security features like confidentiality, integrity and availability.

## **2. Literature Review**

### **2.1 E-commerce**

E-commerce, also known as electronic commerce refers to the use of the Internet to buy and sell goods or services, as well as transfer of funds and data for the execution of these transactions. E-commerce is not solely the sale of physical products online, but it includes any kind of commercial transactions that is facilitated through the Internet. Thus, it is very important to ensure the security of the e-commerce system by integrating the security attributes such as confidentiality, integrity, availability, authenticity, non-repudiation, privacy and encryption [10].

E-commerce has a direct impact on a firm's relationship with suppliers, customers, competitors and partners as well as how firm market products, advertise and use brands. According to a study by the globe newswire, the global e-commerce market will reach an estimated 6.07 trillion US dollars, with a compound annual growth rate (CAGR) of 11.34% from 2020 to 2024 [11].

However, with the rapid growth of e-commerce in our daily life, cyberattacks and security threats also arise. These security threats must be taken seriously because they can easily lead to user sensitive information being attacked by potential attackers and against the confidentiality, integrity and availability of information. The common security threats of e-commerce include distributed denial of services (DDoS), SQL injection, firewall loophole, cross site scripting (XSS), session hijacking [12].

## 2.2 SQL Injection

SQL injection can be defined as a technique for exploiting web applications by using data provided which normally comes from attackers through the input boxes to gain access and make changes to the SQL queries, but without first stripping potentially harmful characters [13]. With this potential vulnerability, attackers will flow commands to a web application's underlying database and destroy the functionalities or confidentiality. The vulnerabilities that will cause the SQL injection attack, including improper input validation, generous privileges, improper handling error messages, multiple statements and more [14].

SQL injection attack is a common website attack technique in which attackers insert custom or inappropriate SQL statements into a query via the data entries of the web system such as input field in login page or URL to get unauthorized access to resources. Improper handling of input validation in web systems is a leading cause of attackers to use SQL injection attacks successfully [15].

The example of SQL statement is:

```
SELECT * FROM employee WHERE userid='112' and password='aaa' OR '1' = '1';
```

Apart from '1' = '1' will always be true, other examples such as " or ""=" will also be usable to inject the SQL query. The SQL select statement will return all records. Therefore, no matter what is the user's username or password, the attacker still can access the system as the manager. This is a case of theft and infringement of data privacy.

## 2.3 Comparison with Existing System

Existing system of online ordering system is studied to learn more about the related features and to examine the security mechanism that should be improved and implemented in this study based on the OWASP Web Application Security Quick Reference Guide 2.0.

### 2.3.1 Flipdish System

Flipdish offers a free website for any restaurant to make their own online food ordering website [16]. Flipdish is a cloud-based solution design and people are able to take order. Flipdish does not require users to register before proceeding to check out. It does not provide input validation for user input fields such as checking the verification of CVV numbers. The password setting does not meet the password policy. An error message generated by the back-end database are displayed on the client side when the users entered wrongly. All the evidences are shown in Appendix A.

### 2.3.2 Cloud Waitress System

Cloud Waitress can be used for cloud and as SaaS software [17] which allows users to connect. However, Cloud Waitress does not have a strong password policy which allows the attackers easily to gain access to an account by brute force attack. It allows the users to use the previous password as a new password which make the password more vulnerable to be guessed. It does not validate the required user input which easily leads to SQL injection attacks too. All the evidences are shown in Appendix B.

The studied existing systems are very important for the analysis by comparing with the proposed system. The output of the comparison is important to determine the differences between the systems. Table 1 shows a comparison between existing systems and the proposed system to clarify their

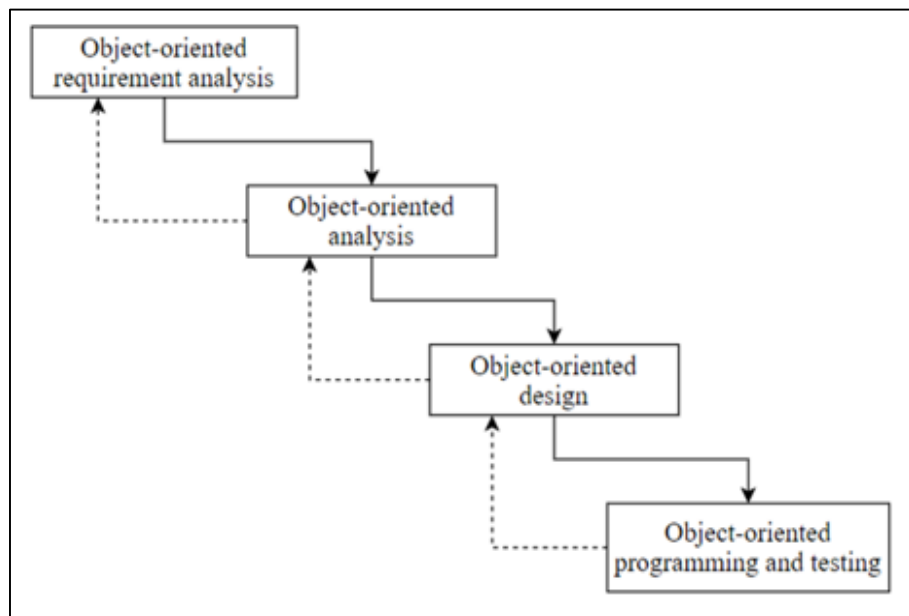
similarities and differences. The features are referred from the OWASP Security Testing Guidelines by only choosing some relevant statements [18].

**Table 1: Comparative Summary of Existing System and Proposed System**

Systems	Flipdish System	Cloud Waitress System	Proposed System
Complexity of password	✗	✗	✓
Saving username and password	✓	✓	✓
Verification during password change	✗	✓	✓
Locking account after few attempts	✗	✗	✓
Captcha Implementation	✗	✗	✓
Log out user after a period of time	✗	✗	✓
Input validation	✗	✗	✓
Privilege Escalation	✓	✗	✓

### 3. Methodology

Object-oriented Software Development (OOSD) method is a methodology that is suitable for developing software application which centralizes on the object of a problem [19]. The proposed system deployed in a scalable and flexible way to handle many kinds of challenges during the development process. Software development is able to make changes on each of the phases in this model according to the necessities to achieve the objective.



**Figure 1: Object-Oriented System Development (OOSD) Model [20]**

Figure 1 shows the process of OOSD Model and clearly illustrates how this methodology works in deployment of the proposed system. The four phases of the methodology are object-oriented

requirement analysis, object-oriented analysis, object-oriented design, object-oriented implementation and testing.

### 3.1 Object-oriented Requirement Analysis and Object-oriented Analysis Phase

In the phase, the requirements are collected to achieve the aspect of objectives and scope of the proposed system. In this project, interview will be carried out for collecting information. The restaurant owner requires to answer interview questions which are shown in Appendix C. Subsequently, the collected information is analysed and the problems of the current system are being defined. First, the ordering process requires the waiter to take the order and the waiter needs to key in the food ordered by the customer. Second, the food menus cannot update immediately because the cost of updating the menu is quite expensive especially when the number of food menus is too much. Third, the possibility of human errors will be quite high especially in taking wrong orders from customers.

After the problems are defined, the objectives and scopes of the proposed system have been identified. There are three objectives in this project which are to design, to develop and to test the proposed system. The scope of the project is focused on the user admin, staff and customers.

In object-oriented analysis phase, except the required software and hardware requirements, system and security requirements of the proposed system will also be analysed. For system requirement, the target users of this project have been determined. The basis function modules for all target users are analysed which are register, login, menu, place order and manage accounts. The proposed system focuses on user admin, staff and customer in Restaurant O SIX JOINT.

For security requirements, the main focus on this system implementation is describing the possibility of the security threats occurred especially the SQL injection. The STRIDE model is used to model the potential security threats to the proposed system to ensure that the application developed meets the security directives of the CIA triad.

### 3.2 Object-oriented Design Phase

Object-oriented design is the second phase in the OOSD model. Both of the models developed in the system analysis phase and the complete architecture designed are necessary in order to develop a complete design model. Unified Modeling Language (UML) diagram is used to simplify the complex process of the proposed system design. UML diagram that has been used, including class diagram which shown in Appendix D, activity diagrams, use case diagram and sequence diagrams. Entity relationship (ER) diagram along with data dictionary and user interface design also created for the proposed system design which shown in Appendix E and Appendix F respectively.

### 3.3 Object-oriented Implementation and Testing Phase

In this phase, it includes two phases which are implementation and testing. The implementation phase of the proposed system, including the setup the coding environment and writing code based on the designs in the previous phase. For the testing phase, it is performed to ensure the proposed system works well in a secure environment and whether it will be attacked by SQL injection. Unit testing, system testing and user testing will be carried out to ensure the system functions as expected. The test cases consist of four basic function test module which is tested the functionality of the system is able to store and manage all target user data, food data and order data will be shown in Table 2. The four basic modules will be tested and the expected result will be listed along with the actual result in Table 3.

**Table 2: Test Module**

Module	Description
Register & Login	Test the functionality of system is able to store and manage the user data
Manage Food	Test the functionality of the system is able to store and manage the food data and must able to add, update and delete
Manage Cart	Test the functionality of the system is able for customer to add, update, delete the food items in the cart
Manage Account	Test the functionality of the system is able for customer to edit their personal information

**Table 3: Test Cases**

Test Module	Description	Expected Results	Actual Results
Register & Login	All the information entered in required field with valid format	Registered user being verify and able login to the system	Pass
	Information with incorrect or invalid format or malicious code to be submitted	The system validates and prompts a message for the incorrect or invalid format of inputs	Pass
Manage Food	Food information entered with the correct format even when inserting or updating	The system will prompt the success message of adding or updating the food	Pass
	Food information entered with incorrect or invalid data when insert or update	The system will prompt the error message of incorrect or invalid format	Pass
Manage Cart	Add one of items with quantity to the cart	The ordered item is well-displayed with correct quantity	Pass
	Update the quantity of food in the cart	Quantity of the item in the cart being updated	Pass
	Delete one of food items from the cart	Food item will be removed from the cart	Pass
Manage Account	The data with valid format to be updated	The system validates and prompts the success message of account updated successfully	Pass
	The invalid format of data to be updated	The system validates and prompts a notification of incorrect format od data entered	Pass

#### 4. Results and Discussion

In this project, the main focus of security in system developing is describing the possibility of the security threats occurred especially the SQL injection. There are two systems implemented to test on the impact of the SQL injection, one is free from SQL injection and other one is not. Table 4 shows the attack plan on both vulnerable and secure system based on the three scenarios, including login, search and URL.

**Table 4: Attack Plan**

Scenario	Malicious Input
Login	' or 1=1 -- " or '='
Search	' or 1=1; drop table test; -- ' or 1=1 --
URL	SearchItem.jsp?search-term=%27+or+1%3D1+--+

#### 4.1 SQL Injection Attack Plan Results

The attack plan results of two system based on three scenarios are presented in Table 5 and Table 6.

**Table 5: Attack Plan Results (Vulnerable System)**

Scenario	Malicious Input	Result
Login	' or 1=1 -- " or '='	The first user account had been logged without proper information
Search	' or 1=1; drop table test; -- ' or 1=1 --	Table test has been dropped from database All of the food information retrieved from database and display in front of the user
URL	SearchItem.jsp?search-term=%27+or+1%3D1+--+	All of the food data displayed at client side

**Table 6: Attack Plan Results (Secure System)**

Scenario	SQL Command	Result
Login	' or 1=1 -- ' or '='	Unable to login to the system
Search	' or 1=1; drop table test; -- ' or 1=1 --	Table did not drop from database No data is found
URL	SearchItem.jsp?search-term=%27+or+1%3D1+--+	No any effect to the system

#### 4.2 Discussion on the SQL Attack Plan Results

This section discusses the results of SQL attack plan to both vulnerable and secure system which can be shown in Table 7.

**Table 7: Comparison of Code Implementation between Vulnerable System and Secure System**

Scenario	Login
Vulnerable System	<pre>String id = request.getParameter("id"); String password = request.getParameter("password");  String sql = null; if (id.contains("ADM"))     sql = "SELECT * FROM user_admin where ADM_ID = '" + id + "' AND ADM_PASSWORD = '" + password + "'"; else     sql = "SELECT * FROM customers_vul where CUS_PHONE = '" + id + "' AND CUS_PASSWORD = '" + password + "'";  PrintWriter writer = response.getWriter();  Connection connection = null; Statement statement = null; ResultSet results= null;  dbconnector dbConn = new dbconnector(); try {     connection = dbConn.getConnection();     statement = connection.createStatement();      results = statement.executeQuery(sql);      Connection connection = null;     PreparedStatement statement = null;     ResultSet results = null;      String sql = null;     if (id.contains("STA")) {         sql = "SELECT COUNT(*) FROM staff where STA_ID=? AND STA_PASSWORD=?";     } else if (id.contains("ADM")) {         sql = "SELECT COUNT(*) FROM user_admin where ADM_ID=? AND ADM_PASSWORD=?";     } else {         sql = "SELECT COUNT(*) FROM customers where CUS_PHONE=? AND CUS_PASSWORD=?";     }      connection = dbConn.getConnection();     statement = connection.prepareStatement(sql);      statement.setString(1, id);     statement.setString(2, password);      results = statement.executeQuery();</pre>
Secure System	

Table 7: (cont.)

Scenario	Search
Vulnerable System	<pre>String sql = "select F_NAME, PRICE, IMAGE, F_ID from food_item natural join food_category " + "where F_Name LIKE '%" + word + "%' OR CA_NAME LIKE '%" + word + "%'";  try { connection = dbConn.getConnection(); statement = connection.createStatement();  rs = statement.executeQuery(sql);  while(rs.next()) { Items GetItems = new Items();  GetItems.setF_NAME(rs.getString(1)); GetItems.setPRICE(rs.getString(2)); GetItems.setIMAGE(rs.getString(3)); GetItems.setF_ID(rs.getInt(4)); retrievedSomeItems.add(GetItems); } }</pre>
Secure System	<pre>String sql = "select F_NAME, PRICE, IMAGE, F_ID from food_item natural join food_category " + "where F_Name LIKE ? OR CA_NAME LIKE ? ";  try { connection = dbConn.getConnection(); statement = connection.prepareStatement(sql);  statement.setString(1, "%" + word + "%"); statement.setString(2, "%" + word + "%");  rs = statement.executeQuery();  while(rs.next()) { Items GetItems = new Items();  GetItems.setF_NAME(rs.getString(1)); GetItems.setPRICE(rs.getString(2)); GetItems.setIMAGE(rs.getString(3)); GetItems.setF_ID(rs.getInt(4)); retrievedSomeItems.add(GetItems); } }</pre>
Vulnerable System	<p style="text-align: center;">URL</p> <pre>&lt;h2 class="section-title"&gt;Search&lt;/h2&gt; &lt;form action = "SearchItem.jsp" method = "get"&gt; &lt;input type="text" name="search-term" class="text-input" placeholder="Search..."&gt; &lt;button type = "submit" style="font-size:30px"&gt;&lt;/i&gt;&lt;/button&gt; &lt;/form&gt;</pre>
Secure System	<pre>&lt;h2 class="section-title"&gt;Search&lt;/h2&gt; &lt;form action = "SearchItem.jsp" method="post"&gt; &lt;input type="text" name="search-term" class="text-input" placeholder="Search..."&gt; &lt;/form&gt; &lt;/div&gt;</pre>

The main difference of two systems' coding is onto SQL command which shown in Table 7. In the login and search scenario for the vulnerable system, create statement is used to retrieved the data. As the data comes from the user input, then the user input will combine with the SQL statement to form a SQL query. If the user enters the malicious input like ' or 1=1 -- , the command will always return true and hence all data will be fetched. The secure system, on the other hand, prepared statement is used to parameterize the queries. Using prepared statement, the SQL query statement will be pre-compiled and then the values are bind to the statement later before the execution of the statement. The database server can recognize the type of commands and the data inputs will not change the intent, which mitigate SQL injection attacks. In this case, the SQL command will not be modified and user can only login by using the proper id and password.

Third scenario is about the URL. The selection of form method is necessary. For the vulnerable system, GET method is used to pass the data and hence the data will be on the URL of the webpage. With that, user can easily modify the parameter value which will launch different kind of results. But in the secure system, the data will be passed by using POST method where user is unable to modify the data easily through URL because the parameter value does not display in the URL.

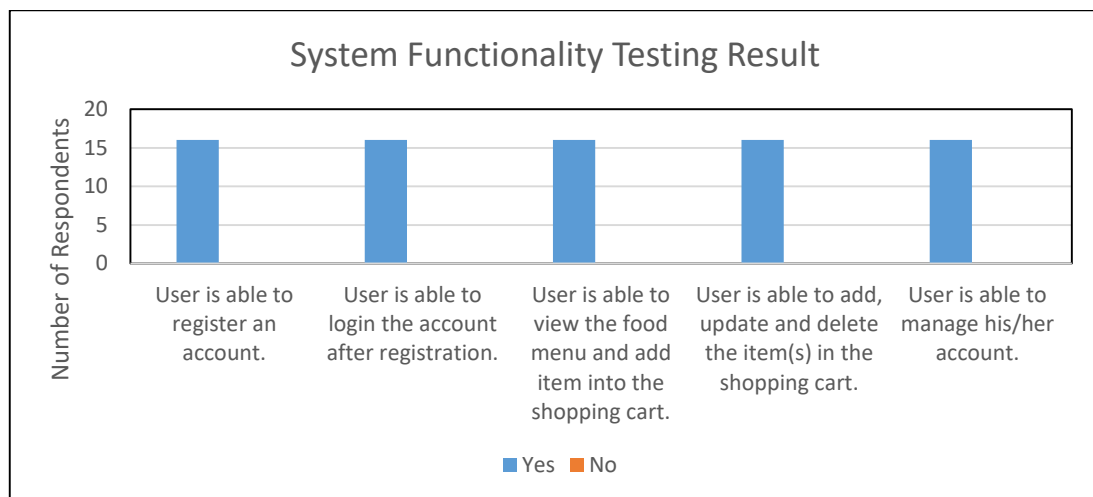


Besides, input validation and limit the user privilege also can prevent this critical threat of SQL injection. Input validation verifies and validates the inputs entered by the user and tests whether it is applicable or not [21]. This validation also included the length and format of the input. If the user's input consisted of some sensitive words or values which may affect and modify the result of the query commands, this can be considered as invalid input which disables the opportunities for the malicious users to attack the system.

Next, privilege escalation will exploit the system vulnerabilities to elevate the access to the system. It can be categorized into vertical and horizontal. Further, minimize staff privilege also considered as one of the prevention method of SQL injection [22]. For the proposed system to counter privilege escalation, admin is free to manage the privilege for each staff based on their current position. The system should make sure that the user who interacts with the database on behalf of the web-based system has minimal permissions. Enforce minimum permissions on the database and ensure that each application has its own database credentials to protect the application from SQL injection attacks.

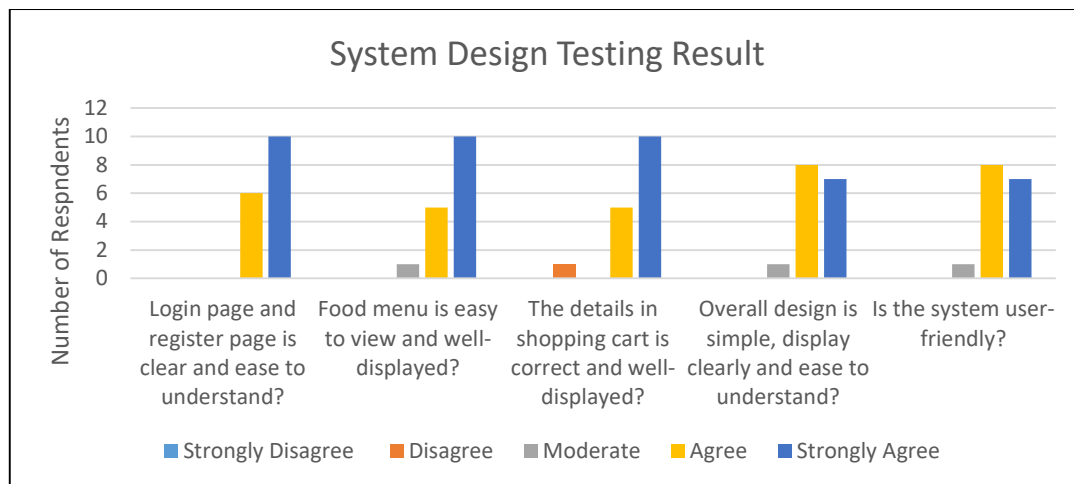
#### 4.3 User Acceptance Form Results

The user acceptance of the system is collected through online survey questionnaire using Google Form that has been distributed to respondents for user acceptance testing. There are 16 respondents involved to test the system. The results of user acceptance form include system functionality, the design of the system and the security usability which shown in Figure 2, Figure 3 and Figure 4.



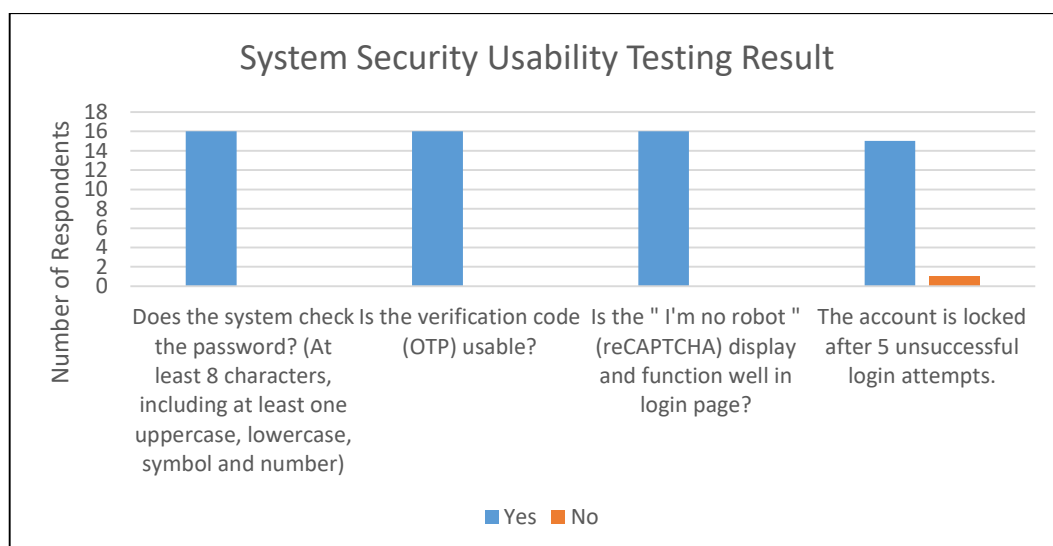
**Figure 2: System Functionality Testing Result**

Figure 2 present the results of system functionality. From the Figure 2, there are 16 respondents who agree that they are able to register, login, view the food menu, manage their shopping cart and also their own account.



**Figure 3: System Design Testing Result**

Figure 3 shows the results of system design. 10 respondents are strongly agreed with login and register page is clear and ease to understand, food menu is easy to view and well-displayed and details in shopping cart is correct and well-displayed. Besides, 8 out of 16 respondents agree with the overall design is simple, display clearly and ease to understand and system is user-friendly. As a conclusion, all the respondents are agreed with the system design.



**Figure 4: System Security Usability Testing Result**

The result of system usability is shown in Figure 4. From Figure 4, the security features of strong password policy, verification of OTP value and reCAPTCHA works for all of the 16 respondents, but the security feature of lock account after 5 unsuccessful attempts works for 15 out of 16 respondents.

## 5. Conclusion

In this project, the target restaurant is named as O SIX JOINT. The current ordering system in this restaurant does not come out with a computerized system which may cause a lot of inconvenience. The timeline to develop this project is within the outbreak of the novel coronavirus, therefore the manual ordering system in this restaurant does not inspired by the customers and the restaurant owner. With that, an online ordering system needs to be implemented for this restaurant. However, based on the researches, the web-based system in these past few years had been damaged hugely with the threat of SQL injection which is the top one ranking within the OWASP and top 25 most dangerous software errors.

After referring to some findings and resources, some prevention ways had been found and discussed in the result and discussion as the result of those findings. Along with the implementation of this project, the methodology used is the Object-Oriented Software Development (OOSD) which is a general guideline to obey for smooth development of the system. This methodology was chosen because of the proper management of the phases of software processes and also the object-oriented approach using which encourages the codes reusability and related much on the Java, the programming language chosen for implementing this web-based system.

Apart from SQL injection, there are some other countermeasures applied to enhance the security features of the implemented system. Some brief examples are Google reCAPTCHA and lock account after several invalid attempts for reducing brute-force attack, password encryption with strong password policy and session management which aid in preventing the sensitive information be exposed to the public. The security features might need to increased or improved from time to time for preventing the new tricks from attackers in future.

### Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support and encouragement throughout the process of conducting this study.

### References

- [1] K. C. Laudon and C. G. Traver, *E-Commerce 2017: Business, Technology, Society*, Boston: Pearson, 2017.
- [2] TheStar, "Malaysia's online retail sales up 28.9% in April," 11 Jun 2020. [Online]. Available: <https://www.thestar.com.my/business/business-news/2020/06/11/malaysia039s-online-retail-sales-up-289--in-april>. [Accessed 12 Nov 2020].
- [3] L. Laguna, S. Fiszman, P. Puerta, C. Chaya and A. Tárrega, "The impact of COVID-19 lockdown on food priorities. Results from a preliminary study using social media and an online survey with Spanish consumers," *Food Quality and Preference*, vol. 86, 2020.
- [4] A. Petukhov and D. Kozlov, "Detecting Security Vulnerabilities in Web Applications Using Dynamic Analysis with Penetration Testing," *Computing Systems Lab, Department of Computer Science, Moscow State University*, pp. 1-120, 2008.
- [5] R. Minion, "Report: Details of 37,145 Credit Cards In Malaysia Leaked Online," 6 March 2020. [Online]. Available: <https://rojakdaily.com/news/article/8827/report-details-of-37-145-credit-cards-in-malaysia-leaked-online>. [Accessed 12 Nov 2020].
- [6] E. Lebanidze, "Securing Enterprise Web Applications at the Source: An Application Security Perspective Perspective," OWASP, 2006. [Online]. Available: <https://docplayer.net/7147520-Securing-enterprise-web-applications-at-the-source-an-application-security-perspective.html>. [Accessed 14 Nov 2020].
- [7] S. Technical and W. Paper, "SQL-Injection Technical White Paper," *Matrix*, pp. 1-7, May 2006.
- [8] J. Clarke-Salt, *SQL Injection Attacks and Defense*, Rockland, MA: Syngress Media, 2009.

- [9] OWASP, "Testing for SQL Injection," 2020. [Online]. Available: [https://owasp.org/www-project-web-security-testing-guide/latest/4-Web\\_Application\\_Security\\_Testing/07-Input\\_Validation\\_Testing/05-Testing\\_for\\_SQL\\_Injection](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/05-Testing_for_SQL_Injection). [Accessed 12 Nov 2020].
- [10] Le, D. N., R. Kumar, B. Mishra, J. Chatterjee and e. Manju Khari, *Cyber Security in Parallel and Distributed Computing: Concepts, Techniques, Applications and Case Studies*, New York: John Wiley & Sons Inc, 2019.
- [11] ReportLinker, "Global E-Commerce (Online Shopping) Market: Insights & Forecast with Potential Impact of COVID-19 (2020-2024)," 15 June 2020. [Online]. Available: <https://www.globenewswire.com/news-release/2020/06/15/2047792/0/en/Global-E-Commerce-Online-Shopping-Market-Insights-Forecast-with-Potential-Impact-of-COVID-19-2020-2024.html>. [Accessed 14 Nov 2020].
- [12] M. I. Al Ladan, "E-commerce Security Challenges: A Taxonomy," *Journal of Economics, Business and Management*, vol. 4, no. 10, pp. 589-593, 2016.
- [13] K. Spett, "SQL Injection: Are you web applications vulnerable?," SPI Dynamics, 2002. [Online]. Available: <https://docplayer.net/2926118-Sql-injection-are-your-web-applications-vulnerable.html>. [Accessed 16 Nov 2020].
- [14] A. Tajpour, S. Ibrahim and M. Sharifi, "Web Application Security by SQL Injection DetectionTools," *International Journal of Computer Science Issueses*, vol. 9, no. 2, pp. 332-339, 2012.
- [15] I. Balasundaram and E. Ramaraj, "An Authentication Mechanism to Prevent SQL Injection Attacks," *International Journal of Computer Applications*, vol. 19, no. 1, pp. 30-33, 2011.
- [16] D. Frietist, "De Frietist Automates Online Ordering Using Flipdish," Flipdish, 2019. [Online]. Available: [https://www.flipdish.com/wp-content/uploads/2019/07/de\\_frietist\\_testimonial.pdf](https://www.flipdish.com/wp-content/uploads/2019/07/de_frietist_testimonial.pdf). [Accessed 13 November 2020].
- [17] SourceForge, "CloudWaitress Reviews and Pricing 2020," 2020. [Online]. Available: <https://sourceforge.net/software/product/Cloud-Waitress/>. [Accessed 15 Nov 2020].
- [18] L. Casey, "OWASP Web Application Security Quick Reference Guide," The OWASP Foundation, 2013. [Online]. Available: [https://owasp.org/www-pdf-archive//OWASP\\_Web\\_Application\\_Security\\_Quick\\_Reference\\_Guide\\_0.2.pdf](https://owasp.org/www-pdf-archive//OWASP_Web_Application_Security_Quick_Reference_Guide_0.2.pdf). [Accessed 16 Nov 2020].
- [19] M. Seidl, M. Scholz, C. Huemer and G. Kappel, *UML@ classroom: An Introduction to Object-Oriented Modeling*, Cham: Springer International Publishing AG, 2015.
- [20] A. R. Hevner, "Object-Oriented System Development Methods," *Advances in Computers*, vol. 35, pp. 135-198, 1992.
- [21] O. P. Voitovych, O. S. Yuvkovetskyi and L. M. Kupershtein, "SQL Injection Prevention System," *In 2016 International Conference Radio Electronics & Info Communications (UkrMiCo)*, pp. 1-4, 2016.


[22] M. A. Mohd Yunus, M. Zainulariff Brohan, N. M. Nawati, E. S. Mat Surin, N. Azwani Md Najib and C. W. Liang, "Review of SQL Injection: Problems and Prevention," *JOIV: International Journal on Informatics Visualization*, vol. 2, no. 3-2, pp. 215-219, 2018.

## Appendix

### Appendix A: Existing system, Flipdish lack of security interface

**Payment**

< Back

 Add a New Card


Card Number  
85742695458623675


Exp Date: 12/2222      CVV Number: 18552

Error  
Please check the credit card number.  
Please check the card expiration date.

Add Card

**Change password**

Your current password  
123456 

Set a new password  
123456 

Must be 6 characters or more

CANCEL    CHANGE

Bank account

Choose account type

Company


Individual / sole proprietorship

Bank name  
maybank

Account name  
lalalala

⊗ Missing required param: external\_account[routing\_number].

### Appendix B: Existing system, Cloud Waitress lack of security interface



Login    Register

leejongfeng@gmail.com

...

Accept Terms & Conditions - [View](#)

ⓘ Minimum of 6 characters for password

Continue

**Change Password** ^

123456

123456

Success!

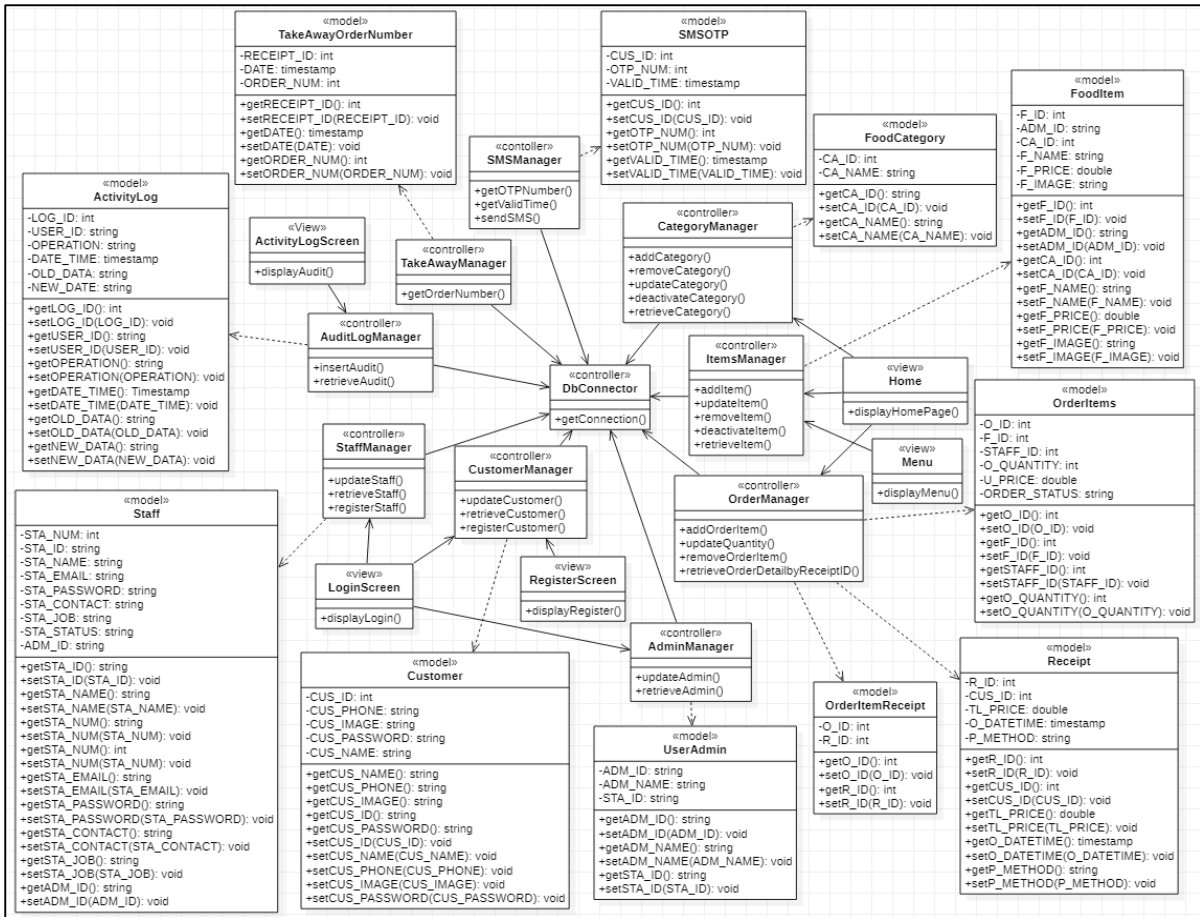
The image shows two side-by-side screenshots. The left screenshot is a booking form with fields for name (demo), email (dsdsd@djdnd.ccuvn), phone number (4841322855222698/855222), and number of people (6). A red box highlights the phone number field, and an arrow points to it from the label 'phone number'. Below the form is a 'Submit Booking Request' button. The right screenshot is a 'Booking Receipt' for #1094, showing a progress bar from 'Un-Confirmed' to 'Confirmed'. It lists booking details: Booking For (03/11/2020, 12:30 pm), No. Of People (6), Placed (03/11/2020, 11:31 am), Name (demo), E-Mail (dsdsd@djdnd.ccuvn), and Phone (4841322855222698/855222). The phone number is highlighted in red. The store address is 351 Lonsdale Street, Melbourne, Victoria 3000, Australia.

### Appendix C: Interview Questions

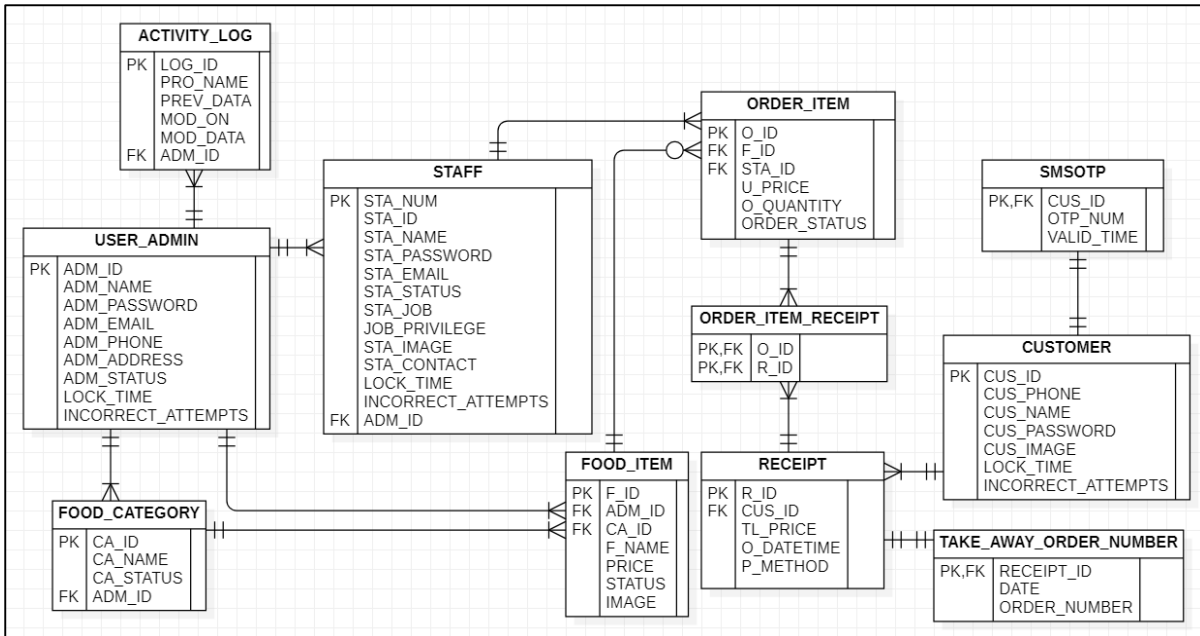
#### Questions for interview session

1. **What are the food categories sell in o six joint?**  
The food categories sold in the restaurant, including spaghetti, salads, baked rice, grilled and chops, snacks and drinks.
2. **What is the method used by restaurant for customer to place order?**  
The customers can choose either order through waiter or use the printed menu to place their orders.
3. **Is there any difference before and during coronavirus for customer to place order?** No, same as usual.
4. **What are the problems faced by customers and staffs when using manual ordering?**  
Staffs may misunderstand the requirements of the customers. Sometimes, the customers will place wrong orders.
5. **Except manual ordering, any other kinds of ordering system provided for the customer to make their order?**  
The customers are able to make their orders through Foodpanda, hungry app or GrabFood app.
6. **May I know how your restaurant promotes your meals?**  
Their promotions will be promoted through Facebook or Instagram.
7. **How does the restaurant update customers with the latest information?**  
The staffs will inform the latest information to the customers before or after they make their ordering.
8. **If there is an online ordering system which is specially designed for your restaurant, are you going to use it?**  
Maybe yes.
9. **If there is a new system, what are the features or functionalities you expect to be implemented in this new system?**  
The system should be simple and must provide more than one language.
10. **What are the preferred user interface design for the new system?**  
The background color should be soft and comfortable.
11. **Which staff will be allowed to use the new system?**  
The system can only use by cashier and the restaurant's manager.

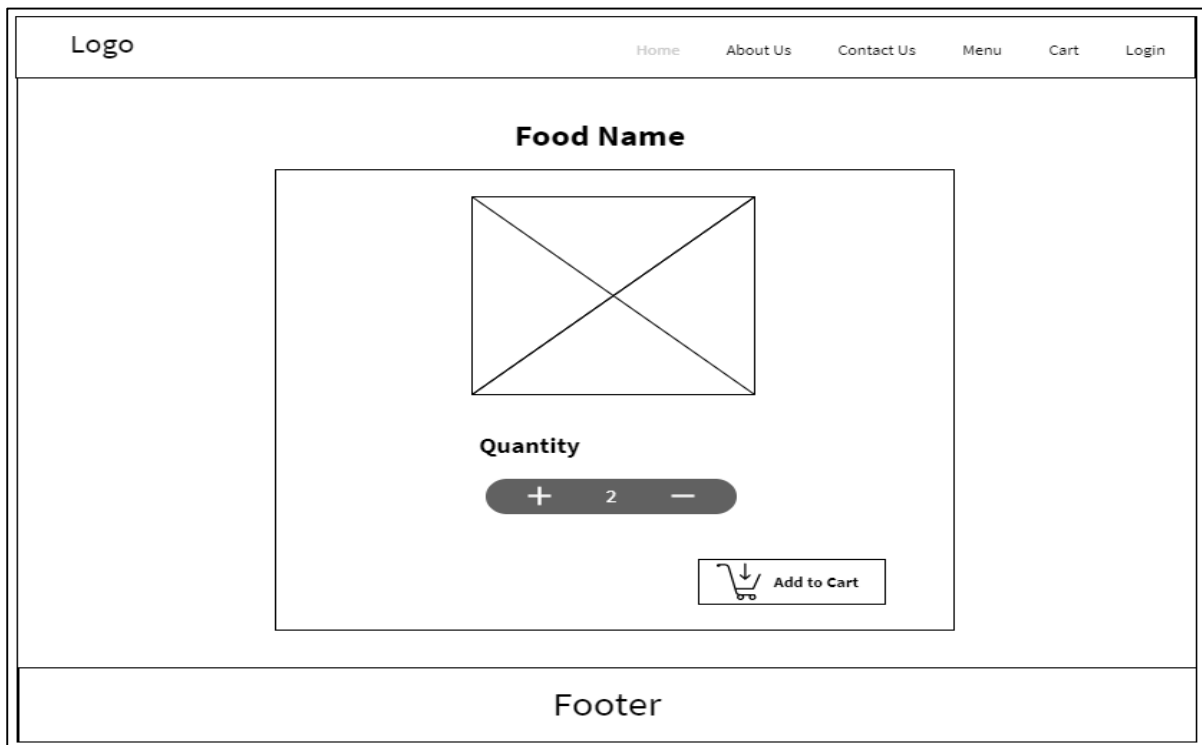
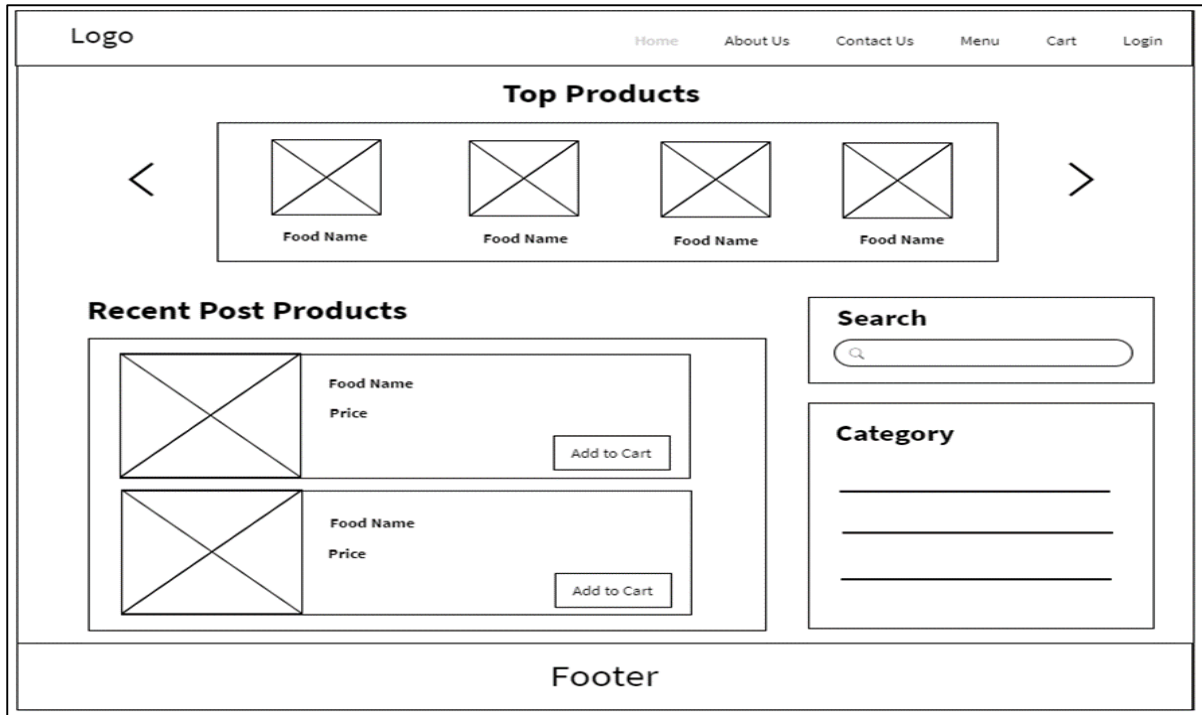
### Appendix D: Class Diagram for proposed system



Appendix E: Entity Relationship Diagram (ERD) for proposed system









Appendix F: Six significant user interfaces of proposed system





Logo
Home   About Us   Contact Us   Menu   Cart   Login


### Cart List

Product Image	Product Name	Unit Price	Quantity	Subtotal	Action
	_____	RM 14.90	2 <input type="text"/>	RM 29.80	
	_____	RM 16.90	1 <input type="text"/>	RM 16.90	
	_____	RM 16.90	1 <input type="text"/>	RM 16.90	

Total Purchase    RM 63.60


Sales tax @6%    RM 3.82





Amount due        RM 67.42


Continue Shopping

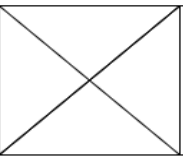
✓ Proceed to Checkout

### Footer

Logo
 Name

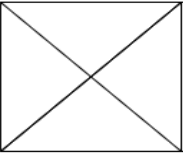
-  Cashier
-  Manage Order's Details
-  Manage Account
-  Logout

### Products



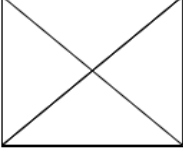
Food Name

Price



Food Name

Price



Food Name

Price

### Category

---



---



---



---



---



---

Logo
 Name

- Dashboard
- Food Category
- Food Item
- Sale Report
- Manage Staff
- Manage Account
- Audit Log
- Logout

### Manage Food Category

+ Insert

Category Name	Category Status	Modified By	Update	Delete
Snack	Available	tang		
Chop	Available	tang		
Salad	Available	tang		
Rice	Available	tang		

Logo
 Name

- Dashboard
- Food Category
- Food Item
- Sale Report
- Manage Staff
- Manage Account
- Audit Log
- Logout

### Check Audit

**Select:**

User ID	Operation	Date & Time	Old Data	New Data
Jong	Update Food Item ...	19/12/20 19:06:12	RM14.90	RM22.90
Feng	Delete Staff	19/12/2020 12:07:00	Lee	