# Web Application: Noshspoon

## CS 4750 Database Systems Semester Project Documentation

Elizabeth Chang (hc3gf), Jinnie Park (jp7hk), Linda Xiong (rx5zv), Cynthia Zheng (xz7uy)

FILES TODO:

1. Add_address.php
2. Add_item.php
3. Add_review.php
4. Address_modal.php
    a. Html formatted for address inputs
        i. recipient name
        ii. Street
        iii. City
        iv. State
        v. zip
5. Cart.php
6. Check_coupon.php
7. Check_out.php
8. Delete_item.php
9. Edit_item.php
10. Edit_listing.php
11. Export_json.php
12. Header.php
    a. `<?php include 'header.php'; ?>` in these files:
        i. Cart.php
        ii. Search.php
        iii. My_listings.php
        iv. Review.php
        v. Index.php
        vi. My_orders.php
13. Index.php
    a. `header("Location:index.php");` in these files: this link explains what header does
        i. logout.php
        ii. login.php
        iii. Post.php
    b. Inside assets/js/add_item.js:
        i. `document.getElementById('alert-indicator').appendChild(div);`
           `window.location.href = "index.php";`
    c. `<?php include 'header.php'; ?>`
    d. `<?php include 'nav_bar.php'; ?>`

e. `<?php include 'index.html'; ?>`

14. My_listing.php
15. My_orders.php
16. Nav_bar.php
    a. <a class="navbar-brand" href="index.php">Noshspoon</a>
17. New_item.php
18. Place_order.php
19. Remove_cart.php
20. Review.php
21. Review_modal.php
    a. Html format for review
        i. Score
            1. 1
            2. 2
            3. 3
            4. 4
            5. 5
        ii. Review
22. Search.php
    a. Inside nav_bar.php :
        i. <form class="navbar-form navbar-left" role="search" method="POST" action="search.php">
23. To_cart.php
    a. Inside assets/js/add_item.js
    b. Check if logged in. if not logged in: `echo 'not_logged_in';`
~~24. Library.php~~
~~25. Post.php~~
~~26. Register.php~~
~~27. Login.php~~
~~28. login.html~~
~~29. Logout.php~~
~~30. index.html~~

Inside assets folder:

31. add_address.js
32. add_item.js
33. add_review.js
34. [front end js ex. auto-generated bootstrap.js.]
35. check_out.js
36. Edit_listing.js
    a. Functions:
        i. Indicate(data): if(data==success) alert ‘success’. Else if(data==Failed) alert ‘failed’. Else if(data== Not logged in) alert ‘you are not logged in’
        ii. editItem(item_id)
        iii. deleteItem(item_id)

37. Npm.js -- autogenerated via the `commonjs` Grunt task. You can require() this file in a CommonJS environment.
38. remove_cart.js
39. CSS folder
40. Fonts folder

# Part 1: Project Information

## 1.1 Introduction

## 1.2 Requirements Document

### Functions

### Requirements by Sections:

#### Buyer:

1. Log in: verify username and password; stay on the same page if either field is wrong or invalid; redirect to home page if both fields are correct; if username does not exist, customer is not a member, direct to the member registration page; enable sessioning (create session and retrieve session info); logout (destroy session).
2. Register: redirect to the register page; perform an insert query to the user table; if the username is found in the user table, redirect to the register page (usernames cannot be the same); after registered, redirect to login page.
3. Search items?????

#### Admin:

### User Story

### Use Case

1. Searching and purchasing item:

    a. Search chilli pepper, then add to cart
    b. Search lychee jelly, then add to cart
    c. Check out
    d. Enter coupon
    e. pay
2. Sell item -- fill in the form that includes the following:
    a. Name: __
    b. Price: __
    c. Picture: __
    d. Quantity: __
    e. Type: __
    f. Description: __

# Part 2: Design Process

## 2.1 Design Decisions

## 2.2 ER Diagram

___insert pic___

## 2.3 Database Schema

## 2.4 Proof that the Database is 3NF

# Part 3: Evaluation

## 3.1 Testing Procedures

Each of our team members registered as users on our website. As different users, we checked to make sure all functions are working properly. After an action is performed, we checked whether the database is updated accordingly. For example, after registering a user, the user table should have a new row with the correct attributes. If the database is modified, we checked if the application responds correctly. For instance, after adding a new item in the item table, item

information should be correctly displayed on the website. Another testing method we did was comparing common use cases and performance with that of other online shopping and recipe websites such as Yamibuy.

## 3.2 Sample SQL Queries

### 1. Get order information from DB:

_____fill in_____

### 2. Get everything in a user's stock:

____fill in____

### 3. Post an item:

_____fill in_____

## 3.3 Sample Data