Project Description

The high-fidelity prototype for our food ordering system was written in HTML, CSS, JavaScript and jQuery using the WebStorm IDE. We also utilized additional resources to support our design, such as Bootstrap (for styling and responsiveness), Lunr (for search indexing) and myCart (for price and quantity management). Throughout the development process, the system was tested and debugged using the latest version of Chrome.

The horizontal prototype was built to resemble our low-fidelity prototype as much as possible. However, some modifications were made in order to correct the issues reported by the users of our low-fidelity prototype. The system's text is now shown in proper sentence case instead of all uppercase to make reading easier for non-native English speakers. Interactive elements, such as buttons and the search bar, were reduced in size to eliminate wasted space. Descriptions were added to each food item and their quantity defaults to 1 rather than 0. The system's buttons were also colored to make them more noticeable and easier to find for the user.

Our first vertical prototype was built to implement the 'Browse Menu and Order Food Items' task. Each selectable food category is listed under 'Menu' along the left side of the screen. The selected category will remain highlighted to indicate which type of food items the user is currently viewing. The right side of the screen displays all the food items that are available for the selected food category. The images of each food item were specifically sized to be large enough so that the user does not have to click to enlarge them.

The descriptive text, 'Qty' drop down menu and 'Add' button are located below each image. The low-fidelity prototype originally had them located to the right of each image. However, we found that this arrangement did not work well in reality during development. The original arrangement limited the amount of images that could be viewed without scrolling and the only solution was to significantly shrink the images, which we did not desire.

A clickable 'Cart' link can be found at the top of the screen where the user can quickly see how many items they have ordered (indicated in the red bubble) and the total cost of their order (tax included). The 'Cart' information is updated immediately when the user selects a quantity and clicks 'Add' under the desired item. A 'Request Help' link is also included; its intended function is to alert a restaurant staff member when a user requires assistance with the system. Both the top and left sections of the system's interface remain in a fixed position when the user scrolls through the list of food items. This was done so that the user does not have to constantly scroll up to select a different food category or view the cart.

Foods items can also be ordered by using the search bar located above the 'Menu'. The user can type in keywords (such as *tuna* or *salmon*) and the corresponding results will show up immediately on the right side of the screen. There is no need to press 'Enter' since the results are generated the moment a keyword is detected. We originally intended to have the search function evaluate the query as the user types in each letter, but we found that the results were often broad and unrelated until an entire keyword was given (e.g. *sal* would return results with *salmon* and *salad*). The resulting food items that are shown after searching can be added to the 'Cart' in the same way as browsing by category.

The second vertical prototype implements the 'Pay for Food Order' task. The user can initiate the payment process by clicking the 'Cart' link, which generates a modal popup that displays all of the food items added by the user. From here, the user is able to review their order and change the quantity of each food item by using the up and down arrows of the number spinner. The ability to completely

remove a food item from the cart list can also be executed by clicking the red 'X' button, which was a suggestion made by a low-fidelity prototype user. The 'Cart' popup can be closed if the user wishes to add more items or they can click 'Checkout' to select a payment method.

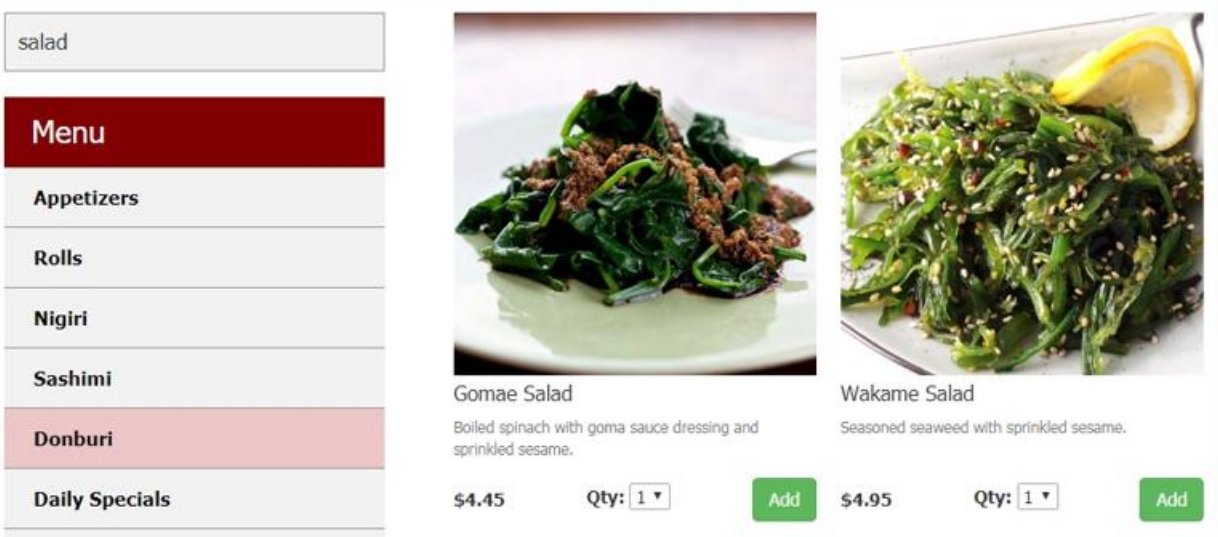

Figure 1. Results returned by the search function.

After clicking 'Checkout', a list of buttons are shown that each correspond to a different payment method. When either 'Pay by Debit Card or Credit Card' or 'Pay by Gift Card' is chosen, the user is presented with labelled form fields, each showing examples of the expected input format. Validation is performed on each field to ensure that the user provides valid information before proceeding. After filling all fields, the user can then complete their order by clicking 'Submit'. There is also an option to 'Pay at Counter' if the user wishes to pay by cash. In this scenario, the system would print an order ticket which the user gives to a cashier who will process the cash payment.

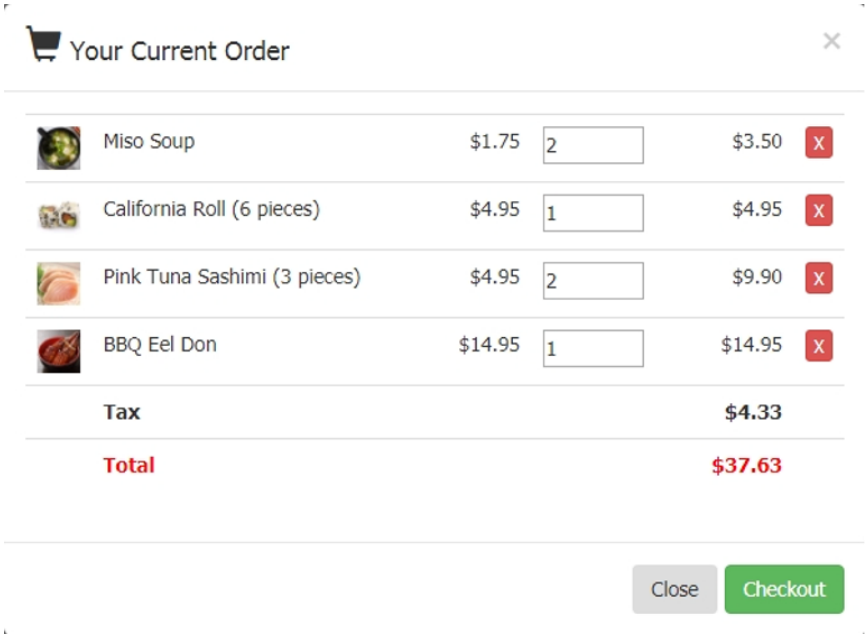

Figure 2. Modal popup displaying the user's 'Cart' items.

The low-fidelity prototype had no means of returning to the previous screen if the user wished to change their order or payment method. To correct this issue, ‘Go Back’ buttons were added to each screen of the payment task. The low-fidelity prototype also had a feature to accept and produce coupons for the user. The marker for Milestone 2 commented that the feature lacked depth and did not contribute to the system’s interaction in a meaningful way, so it was removed during development of the second vertical prototype.

In terms of problems with our prototype, a few technical issues were observed. In the ‘Cart’ modal popup, the number spinner for quantity does not accept direct input of numbers. Additionally, clearing the number in the spinner alters the total price of the order, but does not remove the corresponding item from the cart. Lastly, searching and adding items to the cart work fine in Chrome and Firefox, but these same functions do not respond when running the system in Safari.