Front-End Problems

|  |  |
| --- | --- |
| Sub-Problem | Explanation |
| Making sure all the pages are responsive | Do to this I will need to use media queries to ensure that all the elements are responsive and look suitable for mobile devices |
| Ensure that all links and buttons work | We can ensure that all hyperlinks work when testing the pages |
| Create a web socket connection | We can do this when the page loads |
| Parse the JSON data | We do this as alongside the on-load event and we can use the on-data event to ensure that we receive the JSON from the server |
| Stringfy JSON objects | We can do this we when want to send the JSON to the server |
| Send the JSON to the server | We can do this by doing websocket.send() method |
| Output the menu items to the page | After receiving the menu items from the JSON string and parsing them we can display them to the page either by putting them into a table or creating paragraph tags and putting them into a list |
| When the user clicks on the item or button add it to the basket | We can attach an event to the paragraph as we insert into the list so when the user clicks on the paragraph it will add it to the basket.  In local storage we can store the basket list. When a user adds an item, we store it in there, therefore when the user leaves the page, we save it in local storage |
| Meal deal behaviour | Meal Deal Behaviour   * Cycle through the basket array to check to see if there is a meal deal (Sort the array by price) * Have three different arrays for the different types * To check whether there is a meal deal there will be some sort of counter to see if each type of item appears e.g., if there is a main, drink and snack in the basket array. * Once we have determined that there is a meal deal, I will add the items into another array, to do this I will get the id of that item and then get the item based on the id and store it into another array * I will then delete the items from |
| When the user clicks on the basket send then to the basket page | This can be checked during testing, and we can do this by having an anchor element for the icon |
| When the user adds an item to the basket have a counter go up on the basket icon | We can do this by having a number above the image and always have a counter in the local storage and retrieve the number of items from the basket and display it to the user |
| Display the items on the basket page | As the list will be stored in local storage, we can retrieve the list from there and display it. |
| When the user checkouts the basket take them to the log in/checkout page | We can do this by adding an anchor element to the checkout button |
| When the user makes an error when logging in/checking out show a specific error | We can do this by having a paragraph element below each input element so when one of the text boxes or input element is incorrect, we can display the error below. |
| Validate the text boxes | We can do this by having a length and presence check for the text boxes. |
| Verify the input boxes | We can do this by asking the user to enter in their student ID for a second time providing verification |
| When the user clicks login or checkout finalise their order (and validate the text boxes) | We can do this by doing the validation mentioned above and we can add an anchor element to the button |
| Display the users final order on the webpage | We can do this by storing the user’s basket into and object along with the user’s details and when the user moves to the orders page, we can show them the order details, the basket items and the customer’s name and ID. We will also display the amount of meal deals the user has ordered. |
| When the user goes onto the gallery page show them a slideshow with pictures of the college | We can do this by creating a slideshow |
| Create a slideshow | We can have multiple pictures on the html page but have them set to “display: none” and when the user clicks onto the arrows it will “move” the slide over via an amination and then set the next picture to “display: block” |
| Create an item object | We do this so we can store multiple attributes into the item object that we want to access such as name and price |
| Create a meal deal object | We do this so we can store multiple attributes into the meal deal object that we want to access such as items in the meal deal |
| Create a user object | We do this so we can store multiple attributes into the user object that we want to access such as name and student id |
|  |  |
|  |  |