USAble Life Programming Exercise

Goal: Create a browser-based application for restaurant servers to be able to place orders.

Required screens:

1. Place meal order
   Functionality: Allow user to select/de-select menu items, select/de-select discount, display running totals, and submit an order. A sample order might consist of 2 Cokes, 1 hamburger, and 2 orders of fries. A sample discount might be a Veteran's Discount of 10%. See #10 under Business Requirements for totals.

2. View orders placed
   Functionality: Allow user to view all orders placed. Display should include date/time, server name (first and last name), sub-total, discount amount, pre-tax total, tax amount, and total.

Optional:

1. CRUD (create, read, update, delete) screens for Discounts, Taxes, Employees, and Menu Items.
2. Order details (shows menu items and prices as well as summary information).

Technical Requirements:

1. Javascript
2. C#, if using backend.
3. Browser based application (demo using Chrome)
4. System is capable of running in Visual Studio

Business Requirements:

1. Servers will use Chrome browser.
2. Discounts may be of fixed amounts or percentages of the total. Examples: Veteran's Discount = 10%, Night Owl Discount = $2.00.
3. Only one discount may be applied to an order. A discount is not required.
4. A Discount cannot be more than the total cost. Example: if the total is $2.50, then using a $5.00 will make the total cost $0.00. Any discount selected applies to the entire order and is not dependent on what is ordered. There are no "Buy One Get One Free" or 10% off one-item discounts.
5. All taxes will be applied to all items. E.g., there is no tax specifically on alcohol.
6. Multiple taxes may be applied. Example: a city tax of 4%, a state tax of 3%, and a county tax of 1% would result in a total tax rate of 4% + 3% + 1% = 8%.
7. System will calculate and display the price of the order including discounts and taxes.
8. Orders must include at least one item.
9. Order may include multiple instances of the same item such as 3 Steaks.
10. Taxes will be applied after discounts are applied. Example: A $100 order with a 10% discount results in a total of $90.00 before taxes. If taxes are 5%, then the total collected would be

$90.00 + $4.50 = $94.50. The place order screen should show all these values (Sub Total, Discount Amount, Pre Tax, Tax Amount, Total).

Data: - The following items may be hardcoded or loaded from a data source:

1. Names and prices of menu items
2. Names, types (fixed amount or percentage), and amounts of discounts
3. Names and percentages of tax types (Example: city tax = 4%)
4. System will have an automated method to provide starting information such as discounts, taxes, users, etc. It should also generate orders so the orders placed view can be demonstrated.

Out of Scope: - The following items are not required:

1. Inventory control or tracking of any kind. Assume endless supply of all items.
2. Cash drawer control. Assume customers pay up front using another system. This system is simply for placing and reviewing orders.
3. Payroll - including hours worked.
4. Any collection or tracking of tips. Assume the payment center handles this.
5. The ability to cancel or modify a previously placed order.
6. The ability to split the check among multiple people. Assume the payment center takes care of this.
7. The ability to collect optional information. Example: When ordering a steak, you need not collect details such as rare, well-done, etc. You need not collect a dressing type for salad etc.
8. Notes or comments of any kind.
9. Reporting other than, the ability to see all orders placed and all order details. You need not provide printing, exporting, etc.
10. Database backup/restore, import/export, monitoring etc.
11. Security beyond simply tracking which server placed an order and when.
12. Internationalization of dates and currency. You can assume United State dollars and use the computer date/time.
13. Direct use by customers. Assume servers will place all orders.
14. Tracking customer loyalty via rewards cards.
15. Age verification (must be 21 to buy alcohol)

Validations:

1. Discount names must be alpha numeric and 1 to 25 characters.
2. Discount names must be unique.
3. Discount types must be Fixed or Percentage. If the type is Fixed, then the Amount must be a number in the range 0 to 100 with no more than 2 decimal places. If the type is Percentage, then the Amount must be an integer value in the range 1 to 100.
4. Tax names must be alpha numeric and 1 to 25 characters.
5. Tax names must be unique.
6. Tax percentages must be integer values from 1 to 100.
7. Menu Item prices must be numbers in the range 0 to 100 with no more than 2 decimal places.
8. Menu item names must be alpha numeric and 1 to 25 characters.
9. Menu item names must be unique.
10. Employee should have a first and last name that are letters and 1 to 25 characters long.

Options:

1. You may use any method (EF, NHibernate, ADO, etc.) to access the data, persistence is not required.
2. You may use any thin client library such as React, Angular, Knockout, etc.
3. You may use any CSS library.
4. You may use any code generators.

If any of your code requires pre-processing or transpiling, please include instructions on how to do so with the project.