CEN 4010 Principles of Software Engineering

Spring 2023

Milestone 3

Website called OnlyTable that allows customers to book reservations and order online from a restaurant. It also allows employees to view reservations and online orders that have been placed, as well as allowing them to remove items that are out of stock.

Avalanche, OnlyTable

Project Website Link: https://wip.d2nc7095y5s7h0.amplifyapp.com/

Group 4

Steven Luongo – Team Lead – sluongo2022@fau.edu

Sadie Shank – sshank2018@fau.edu

Jaden Badal-Campbell - jbadalcampbe2020@fau.edu

Isabella Costa – icosta2021@fau.edu

Brian Rudowitz – brudowitz2021@fau.edu

3.21.23

History Table

Row #	Revision Date	Revision Description
1	3.18.23	Began working on the document.

2	3.20.23	Added significant content.
3	3.21.23	Added in the diagrams and more content.
4	3.22.23	Edited the document.
5	3.26.23	After working on implementation, changed the database set up section.
6	3.30.23	Implementation Finished, Edited Document and Submitted

Executive Summary

We are creating a website for a restaurant called “OnlyTable”. This website will allow the restaurant to display their menu and prices, allow customers to reserve tables and place online orders, and allow employees to view orders and reservations and remove menu items that are out of stock. It will be user friendly and easy to navigate for both the customers and the employees.

“OnlyTable” is the name of our website, which is a catchy name that customers will be able to remember when making reservations / online ordering. It will walk them through the process and be a clear website that is easy to navigate, unlike other websites that can be difficult, have many popups, and have too many confusing pathways for a customer to follow.

This project is important because it can often be difficult to create reservations at a restaurant due to faulty websites that have a lot of glitches and are difficult to navigate. Online ordering can also be a nightmare for a customer because websites tend to hide menu prices, force the latest deals and combos onto customers, and are difficult to simply see the items available for purchase as some can be out of stock. The major enhancement that our website provides is an easy way for employees to remove menu items that are no longer available. Restaurants today struggle with supply chain issues and when customers order items that they do not have, it creates an intense situation to deal with for both customer and employee. Our website will provide an easy solution to these problems with an easy-to-use

interface that walks the customer through the process and allows the restaurant itself to quickly see who has booked a reservation and who has placed an online order.

Competitive Analysis

OnlyTable	Competitors (like Pollo Tropical and McDonalds online ordering sites, and fancy restaurants like Ruth Chris' Steak House that have call-in reservations or online reservations through a third-party site)
Easy for customers to use. Allows customers to reserve a table on the website instead of calling ahead. Has the reservation page up front so customers do not have to navigate around to find it.	Confusing website that does not directly point customers to the page for reservations. Sometimes leaves out the reservations page and customers have to call ahead to reserve a table. This can be confusing and lead to customer service issues. More employees are needed to handle reservations.
Interactive Interface. Is designed appropriately so customers are viewing a pleasant interface.	Not designed properly. Links are located in odd locations. Colors are not chosen appropriately and are annoying to navigate, especially for those with accessibility problems.
Easy for restaurants to use. Allows restaurant employees to view reservations and orders easily and update menu options.	Is difficult for restaurant employees to access and view the reservations / online orders. Makes restaurants less efficient and leads to issues when supplies run out for certain menu items.

Easy to integrate for any particular restaurant. Is not entirely customized to only one restaurant. Any restaurant can take the design and input their personal menu to use it.	Is difficult for another restaurant to use the design. It is too tailored towards a specific restaurant. Too complicated that another restaurant is better off creating their own from scratch than reusing another one. Third-Party sites are used instead which could lead to data security concerns as well as being confusing for a customer to be redirected.
Ample documentation in case features need to be edited or another restaurant wishes to use the application. All documentation is available in the same location in GitHub where everything is together so it is easy to find what is needed.	Little documentation available on the project. If something needs to be fixed, it is difficult to find where the error is and which application is needed to be able to fix it.
Easy to order online. All menu items are listed conveniently with their prices included. The shopping cart allows for quick information inputting and employees can see what was ordered when. It tells customers the approximate time until their order is ready.	Some menus do not have their prices included, so it makes it difficult for customers on a budget to intelligently order. It does not tell customers and approximate time until their order is ready so they are left guessing. Employees see a confusing arrangement of orders, whereas our website will list the orders in the order they were placed.

OnlyTable will have an easy-to-use design and an interactive interface which competitors' products do not. Our product will be a lot easier to use. We will not add any random information and links that are not important to making a reservation, viewing the menu, and ordering. Competitors tend to include filler information on their webpages that make it difficult for users to find the information and functions they need. They contain ads for website revenue as well. Our website will make revenue simply from the restaurants that pay a fee to use the site and have it slightly altered to match their restaurant. Our design will be versatile and easy for any restaurant to adapt to so they can use it as well.

Our website will have all its files stored in GitHub which will make it easier for future software developers to make any changes that they see fit because all of the documents will be in one location. This is also helpful to the customers because they can see what is being created as we create it. This allows for customer / developer transparency. With most applications of this sort, it is difficult for them to be useful for other companies to use or edit because there is no documentation, so they do not know how the application was created and will have to start from scratch building it up. Our website is great in this area because it will be easy to see where the problem is if it has to be updated in the future since all documentation is available in the same area.

The major advantage of our website is the ease of use for not only the customers, but also for the employees. Employees should not have to deal with updating a website that needs an entire training session to be able to do. It should be an intuitive design that any restaurant employee can immediately use to update menu items, view reservations, and view online orders on the fly.

Our website takes out the hassle and chaos for both customers and employees in the restaurant industry, saving them time and money.

Data Definition

Term	Meaning	Usage
OnlyTable	The online reservation / online ordering website.	Used when referring to the website that we are developing for restaurants to use.
Customer	The user who is making the reservation or placing the order.	Used to refer to the person who is using the website to interact with a restaurant and not the employee.
Employee	The restaurant employees only.	Used to refer to the person who is utilizing the website for the restaurant's purposes and is representing the restaurant. They are accessing information about who has reservations and who placed online orders and are updating menu items.
User	Both customers and employees.	This term is used to refer to both customers and employees when referring to a feature that impacts both.
Menu Page	The page that has all of the menu items included on it.	Customers are able to use the menu to add items to their cart when online ordering and also to view it ahead of time if placing a reservation so they are aware of items offered and their pricing.

Cart Page	Unique to each customer. Includes the items that the customer has added to online order. Has an order button to place an order.	Allows the customer to view the items that they are ordering and allow them to place their order and input payment information.
Reservation Page	The page that allows the customer to input the date and time that they want to place a reservation. It shows the available time slots for a reservation to be made.	Allows the customer to place a reservation. Shows when reservations can be made for. Does not allow customers to make reservations for an already booked slot.
Reservations Made Page	Allows the employee to see all reservations that have been made, at what time and date they have been made, and who made them.	This is only available for an employee to view. They can see all the information about a reservation.
Orders Placed Page	Allows the employees to see all the orders that have been placed and at what time.	Employees are able to remove orders from this page as they are filled and see the orders in the order they were created.
Update Menu Items Page	Used by employees to add / remove menu items.	This is only available to employees and consists of a form for an employee to either add or remove a menu item.

Tables

Menu – This is on the Menu page.

1. Objects –
    - itemId
    - menuName
    - menuPrice
2. Functions –
    - addItemstoCart() - From the Menu page, the customer is able to add items to their cart.

Reservation – This represents the reservations page.

1. Objects –
    - reservationDate
    - reservationTime
    - customerName
    - noPeople
2. Functions –
    - createReservation() - From the reservations page, you can create a reservation.
    - searchReservations() - You can also search for a reservation to book one.

Employee – This is for the employee login.

1. Objects –
    - employeeUsername - One username and password for every employee at the restaurant. This is the restaurant's global username / password.

- employeePassword

2. Functions –
    - logIn() - An employee has to login to the employee account.
    - updateMenu() - They can update the menu by adding / deleting items.
    - checkReservation() - They can check the reservations that have been made.
    - checkOrders() - They can check the orders that have been placed.
    - deleteOrders() - They can delete orders once they are filled.

Order Details – This is on the orders placed page for the employee to see.

1. Objects –
    - orderId
    - productid
    - productName
    - quantity
    - unitCost
    - subtotal
2. Functions –
    - calcPrice() - It calculates the total price, also on the cart page.

Orders -

1. Objects -
    - id
    - cardNumber
    - createdAt

- name
- card_cvv
- card_ex
- total

Items on Orders - Connects the Items that were ordered (Order Details) to the customer's information in Orders

1. Objects -
    - orderId
    - assignedAt
    - assignedBy
    - itemId

Overview, Scenarios, and Use Cases

This webpage will allow customers to view the menu, put in an online order, and make a reservation. It will also allow employees to access the reservations made information to see who has made what reservations, see what orders have been placed, and update menu items in the case of adding / removing items. A customer will be able to see the reservations, menu, and the cart pages while the employee will be able to see all of those as well as the reservations made, order placed, and update menu pages. In this section, we will outline the different situations that might occur and how the customer / employee will use the system for each case.

Scene	Use Case
A customer wishes to make a reservation.	They navigate to the reservations page and put in their information. They search for the available times using the number of people that will be in their group and pick an available time slot. They then click the submit button. Their reservation is stored in the system and the spot is taken out of available reservation spaces so no other customer can book a reservation for that time. The employees will see the new reservation created in the reservations created page.

A customer wishes to place an order.	They navigate to the menu and put the items they want in their cart. Each menu item has the corresponding price, so the customer can make an intelligent decision. They click on their cart, input their payment information and their name, and click ‘place order’. Their order has the price for the order listed. The customer will see the estimated time of pickup which is based on the time that the order is placed, so it will take longer during peak dinner time and take less time during off-hours. The employees will see the new order put on the order page and they can begin making it.
An employee wishes to check the reservations.	The employee navigates to the reservations made page and sees all the reservations, who made them, and when they are for. This allows the employees to plan accordingly for restaurant space.
An employee wishes to view the orders placed.	They navigate to the orders placed page and see all the orders that have been made. They can see the name of the customer who placed it and at what time it was placed as well as the order itself. They can input the id number

	of each order to remove it from the system when it is finished.
A menu item is out of stock.	An employee clicks on the update menu items page and inputs the id number of the item in the remove form. They then remove the item from the menu. This item is saved in a backup menu database so it can easily be added later.
A menu item that was out of stock is now back in stock.	An employee clicks on the update menu items page and inputs the id number of the menu item in the add form. The menu item is added back in because it was saved in the backup database. If there is no matching id number for the item in the backup database, the user will be told so. They will then input the information for this item into the bottom form and it will be added.

Initial List of High-Level Functional Requirements

1. Priority 1

   The ability to make reservations. This will be a data entry form that the customers can fill out to pick a time to reserve. This reservation will reflect in the reservations made page. The time slot that the customer chooses will be removed from the available times database so future customers cannot make a reservation for that time. The customer will input their name and the number of guests for the reservation so the employees know who made it.

   The customer will be able to input a date and a number of people and click search. This will relate to the database where it will select all the available reservations which the restaurant will be responsible for inputting. This will display on the screen, the customer can select the one they want, input their credentials, and click reserve.

2. Priority 1

   The ability to add items to the cart. This will be a functionality that will occur on the menu page. Customers will scroll through the menu and add the items that they want. These items will then be reflected in the cart where customers can check those items out. Each item will have its price labeled on it.

   The database will add each item to a temporary table to store the user's menu items. On the cart page, the database will have a select all statement to show all the items that the user added.

3. Priority 1

   The ability to place an order. This will occur in the cart. The customer will click on their cart and see the items they have added. They will then input their billing

information and click place order. The order will show the total price that the customer will be paying. The order will then be visible to the employees on the orders placed page who will be able to fill the order and know who made it. The customer will be shown the estimated pick-up time which is based on the time that it was ordered. If it was ordered during a typically busy time for the restaurant, then it will take longer.

The database will have the order temporarily stored where it can retrieve it to show on the cart page. The order along with the customer's information will be displayed on the orders placed page for the employees to see.

4. Priority 2

   The ability to check the reservations made. The employees will be able to access a reservations made page that will show all reservations as well as the customer that made the reservation. This will help the employees to plan their restaurant availability and they will know who is scheduled to come in when, so they can properly prepare for the customers and allow walk-in customers to have the unfilled tables. This page will simply show a table of reservations that have been made along with the number of guests that are coming.

   It will be connected to the one reservations table and will show all the reservations that have been made.

5. Priority 2

   The ability to check the orders placed and remove them when finished. The employees will be able to see all orders that have been placed in the order that they were placed. This will be beneficial to the employees who work with the food, so

they are able to plan accordingly. They will have access to a button to remove the orders once they have been filled so it will not clog up the website with filled orders. To remove orders, it will have the ability to select items from the table that the website shows which will link to a delete statement that will take it out of the database.

6. Priority 3

   The ability to remove and add menu items when needed. If a menu item is out of stock, an employee will be able to go to the update menu items page and remove it. When the item comes back in, they will go to the same page and add it. If the item that they are adding is an entirely new item, there will be a form available to add it to the whole system.

   This will be accomplished by a simple select and delete statement. The deleted items will be moved to a new table so they can easily be brought back.

7. Priority 2

   An interactive interface is a functionality that means our website will be designed appropriately so it is pleasant to view. It will not use confusing graphics or blaring colors and, instead, will have a thoughtful design. This will be essential for accessibility as well because it will make it easier for customers with disabilities to navigate quickly through.

List of Non-Functional Requirements

1. Easy-to-Use design will be a non-functional requirement of our website. This simply means that the website will have the links to the reservations, menu, and cart easily viewable so the customer can see exactly where they need to go. The customer will be able to easily navigate the website because there will not be any filling meant to distract users like advertisements. They will view the site as an easily accessible and a navigable platform.
2. Accessibility will be a priority for our website design. We will pay attention to requirements such as making our site readable for a screen reader, having content placed in sensible locations to avoid difficulty navigating, and having text in larger fonts to make sure it is easy to read. Any images or graphics that are used will have descriptive alt-text included so those with low vision can view the site as effectively as other users can. This will be a requirement for our website because it will attract more customers because a lot of websites are difficult for some people to navigate. We will remove any difficulties that people would normally have with other sites.
3. We will have our text in a color that is readable for those that are color blind. We will avoid using colors that some users might find difficult to read. Our site will avoid the use of unnecessary decorations that distract a user.
4. The site will have high-speed performance. It will load and add content to the databases quickly. This is especially important for OnlyTable because a customer will not stay long to use a faulty website. If they input information and it is not reported properly, then reservations and online orders can be messed up. This would cause the restaurants to have issues with customer relations, and can reflect poorly on their reputation.

5. Availability is a key feature of our site. It will be available to users of any restaurant that chooses to participate. The databases will run efficiently so the users can add their information and have it processed properly. Accessibility runs into availability as well because in order for a website to be properly available to all customers, it has to be accessible to everyone, so this website will strive for both. It will be available on many different browsers and operating systems.
6. Data Security is an important feature because without secure data, our customers will not be willing to use our site. If a customer is concerned that their data is at risk of a breach, they will not be willing to enter their personal information, especially when it comes to credit card information when ordering online. If a customer does not enter their information for fear of a lack of data security, then there will be no online orders which will damage the restaurant's business. To combat this, we will ensure that we use high-quality database products like MySQL which will provide us the platform to store and access data from.

High-Level System Architecture

The software products and tools we are going to use are as follows:

- GitHub – We will use GitHub to store all of our files in one location. This is a convenient file hosting site because we can collaborate as a group more effectively when everything is in the same location. When we update something, our GitHub lead will update it in GitHub so everyone can see the new file. GitHub has a convenient desktop application that we are using called GitHub Desktop which allows us to open our site in Visual Studio Code and edit it. When finished, we will be able to push it back to GitHub using GitHub Desktop.
- Visual Studio Code – This is a coding environment where we are writing the actual website code in our chosen languages. We then integrate this using GitHub. Visual Studio Code provides an easy-to-use interface and a pleasant coding environment where each group member can develop their own sections. It allows for an interactive environment where extra tools can be added as the need presents itself. It has a built-in terminal that makes it easy to integrate other parts of our project as well, like the databases. This is easy to use with GitHub as well because GitHub has a built-in feature to allow projects to be quickly opened in Visual Studio Code which allows for easier editing. This will come in handy when we get to our revisions part and begin to edit the site as well.
- JIRA – JIRA is a tool that we will use for collaboration. It will allow us as a group to keep track of what each group member is doing, which will in turn help us to not duplicate tasks. This will help our group to hold each other accountable and to make sure that we are all doing what we need to do. This will also have a big

impact in making sure that we stay on track and on time with what we are developing and submitting. JIRA allows users to create work sprints where we can set for a certain amount of time and have each member work on their own part for as long as a sprint is and then we can all come together when we are done and check in on what has been done. We can assign tasks with JIRA and mark them as complete when they are finished. It will act as a motivational to-do list for our group members.

- MySQL – This will be used to store our information in the databases and to run our queries through. This is used because our group members are familiar with how it works and it will be the easiest platform to integrate our databases into. It is a trusted platform that will allow us to have data security which is one of our non-functional requirements that we have chosen to include. We believe that MySQL is a trustworthy software tool that we can store our data in and effectively manipulate it.

The languages and systems we have chosen to use are as follows:

- Typescript/Javascript – Typescript allows us to implement functionality into our code, it also helps us modularize our project by creating reusable components. Typescript has an advantage over JavaScript where we can add types to our codebase, which is one of the downsides of using plain JavaScript.
- Bootstrap – Bootstrap is a tool that allows developers to use pre-made website templates to speed along the development process. This is important to us because there is no sense in wasting time recreating something that already exists, so we will use a Bootstrap template to get us started with the website framework.

- HTML5 – This stands for Hyper Text Markup Language and will be used for the structure of our website. It will allow us to strategically place content and is a language that we are all familiar with, so it will make it easier for our group to use to create our platform.
- CSS – This is a design language that we will use to make our website pleasant to navigate and include useful graphics to help customers find their way around. Our group members are familiar with this language which makes it the best to use.
- SQL – This is the database language we will use. The group members who will be working with the databases are familiar with it.

The browsers / operating systems that will support our site are as follows:

- Microsoft Edge
- Chrome
- Safari
- FireFox
- Windows
- OSX
- IOS Mobile Browsers
- Android Mobile Browsers

This diversity of operating systems and browsers that our website will be compatible with is important because it will be available to a wider range of customers. Since accessibility is a key factor of our site, the more applications that it can be used on, the more accessible our site will be.

Database Organization

1. See below UML Diagram for the high level coding architecture.
2. We are using four tables: login, menu, orders, and reservations. Login is used for the employee when they go to login to the restaurant to view the information that the customers have put in, such as reservations made and orders placed. Menu is used to store all of the menu item information. In this way, the customer is able to view this database on the menu page and the employee is able to add / delete menu items. Orders is used to show all the customer orders that have been placed and this is then reflected for the employees on the orders placed page. Reservations is used to store all of the reservations available, and it will mark the reservations that have been taken as well. Here are the tables' structure:

Login

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**username**	varchar(100)	utf8mb4_0900_ai_ci		No	*None*			Change Drop Primary Unique Index Spatial Fulltext Distinct values
2	**password**	varchar(100)	utf8mb4_0900_ai_ci		No	*None*			Change Drop Primary Unique Index Spatial Fulltext Distinct values

Menu

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**id**	varchar(255)	utf8mb4_unicode_ci		No	*None*			Change Drop Primary Unique Index Spatial Fulltext Distinct values
2	**name**	varchar(100)	utf8mb4_unicode_ci		Yes	*NULL*			Change Drop Primary Unique Index Spatial Fulltext Distinct values
3	**price**	int			Yes	*NULL*			Change Drop Primary Unique Index Spatial Fulltext Distinct values

Orders

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	**id**	varchar(255)	utf8mb4_unicode_ci		No	*None*			Change Drop
2	**card_number**	varchar(255)	utf8mb4_unicode_ci		Yes	*NULL*			Change Drop
3	**created_at**	timestamp			Yes	*NULL*			Change Drop
4	**name**	varchar(100)	utf8mb4_unicode_ci		Yes	*NULL*			Change Drop
5	**card_cvv**	int			No	*None*			Change Drop
6	**card_exp**	varchar(100)	utf8mb4_unicode_ci		No	*None*			Change Drop
7	**total**	int			No	*None*			Change Drop

Reservations

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	**resdate** 🔑	date			No	*None*			Change Drop Primary
☐	2	**nopeople** 🔑	varchar(5)	utf8mb4_0900_ai_ci		No	*None*			Change Drop Primary
☐	3	**restime** 🔑	time			No	*None*			Change Drop Primary
☐	4	**name**	varchar(255)	utf8mb4_0900_ai_ci		Yes	*NULL*			Change Drop Primary

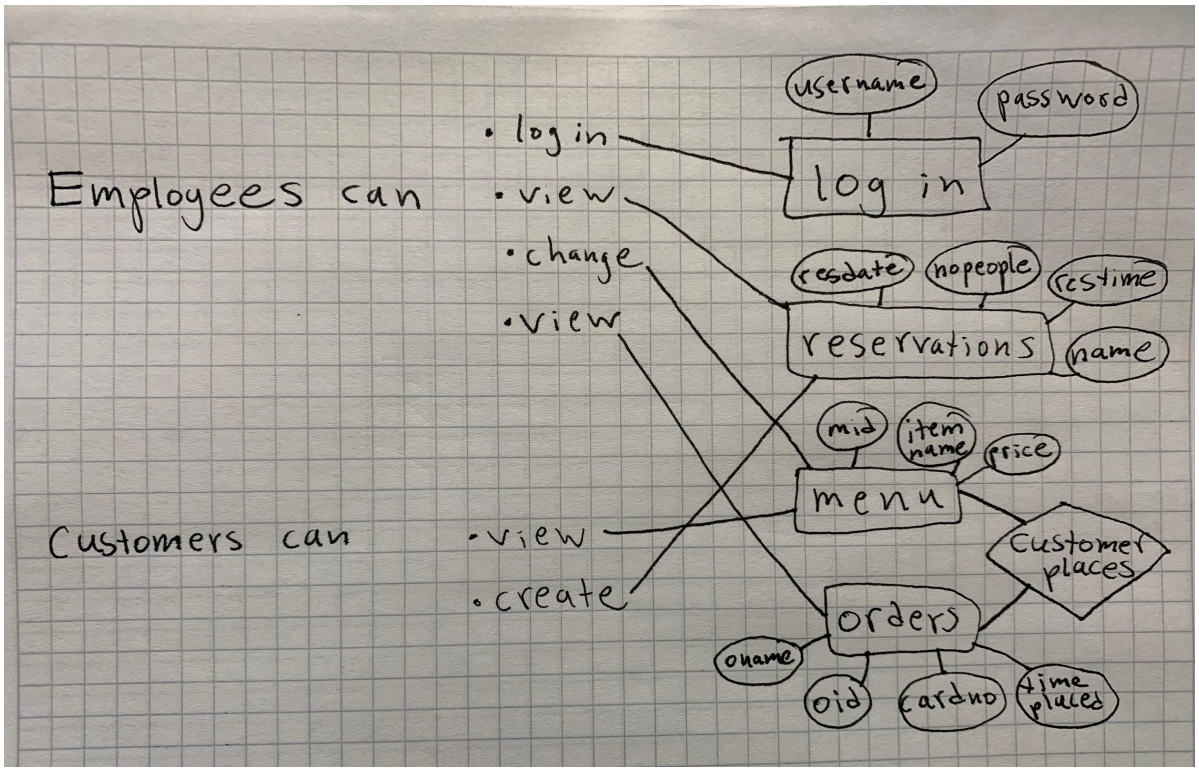
ItemsonOrders

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	**orderId** 🔑🔑	varchar(191)	utf8mb4_unicode_ci		No	*None*			Change
☐	2	**assignedAt**	datetime(3)			No	CURRENT_TIMESTAMP(3)		DEFAULT_GENERATED	Change
☐	3	**assignedBy**	varchar(191)	utf8mb4_unicode_ci		No	*None*			Change
☐	4	**itemId** 🔑	varchar(191)	utf8mb4_unicode_ci		No	*None*			Change

Items

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	**id** 🔑	varchar(255)	utf8mb4_unicode_ci		No	*None*			Change Drop
☐	2	**name**	varchar(100)	utf8mb4_unicode_ci		No	*None*			Change Drop
☐	3	**price**	int			No	*None*			Change Drop
☐	4	**quantity**	int			No	*None*			Change Drop

Here is a ER-style diagram that is showing how all of these tables are connected:



This diagram shows that the only intrinsically linked tables here are the menu and the

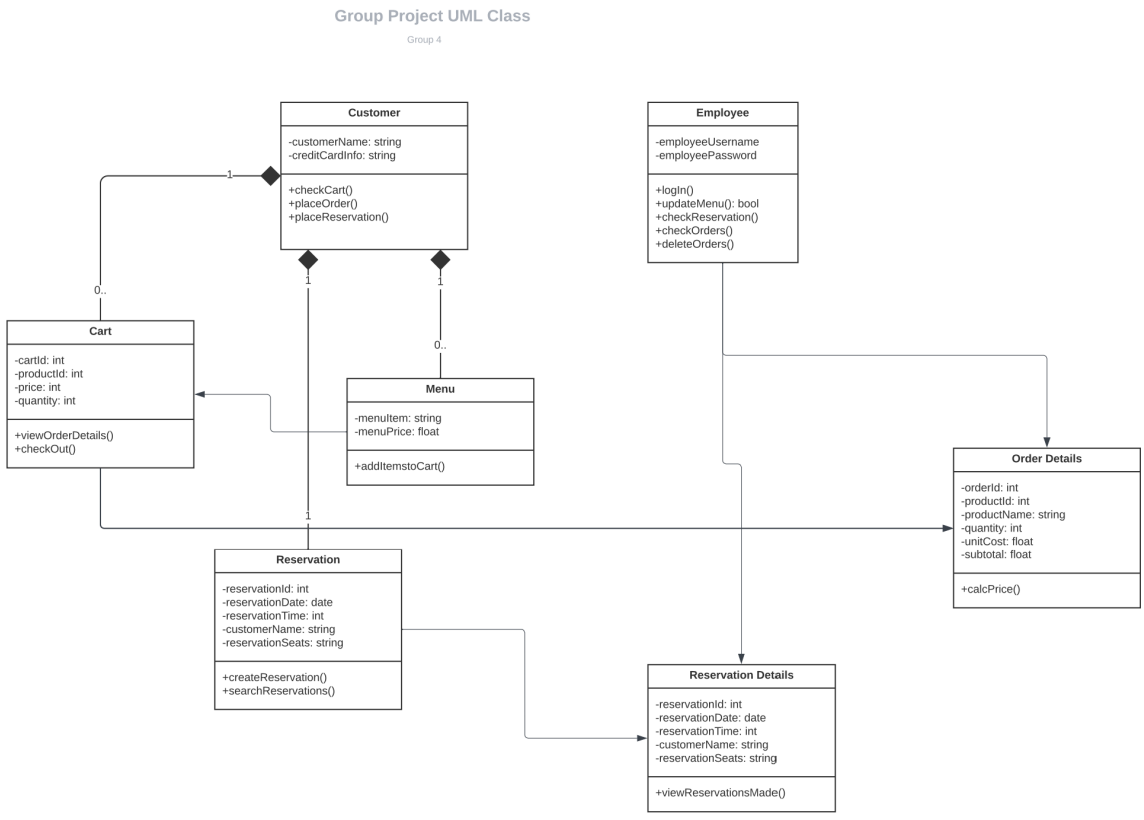
orders. A customer can view the menu and from there they can place an order.

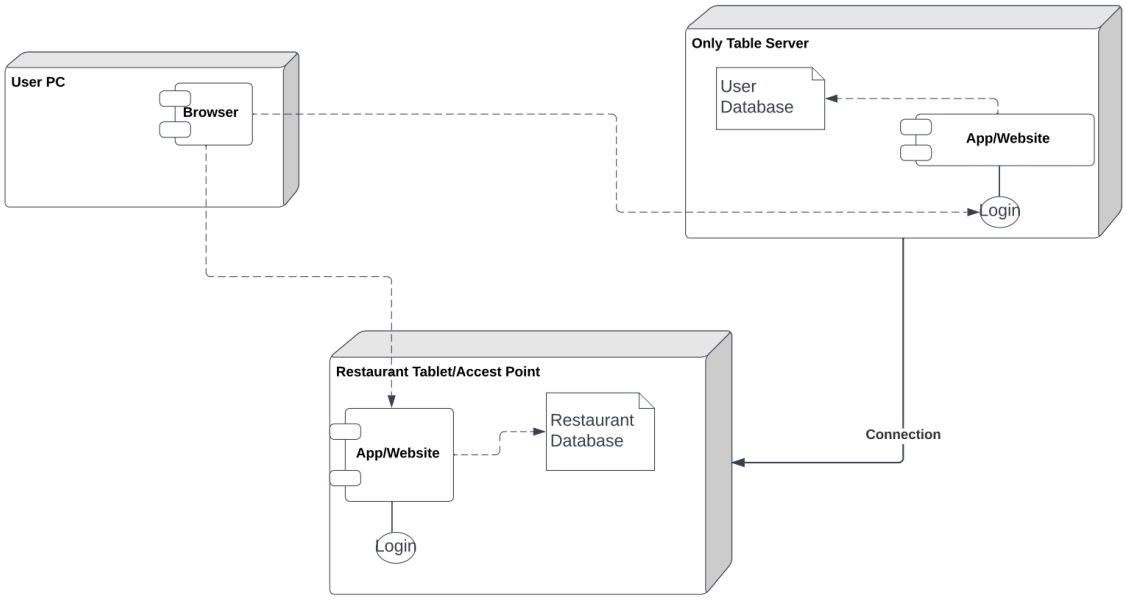
ER Diagram:

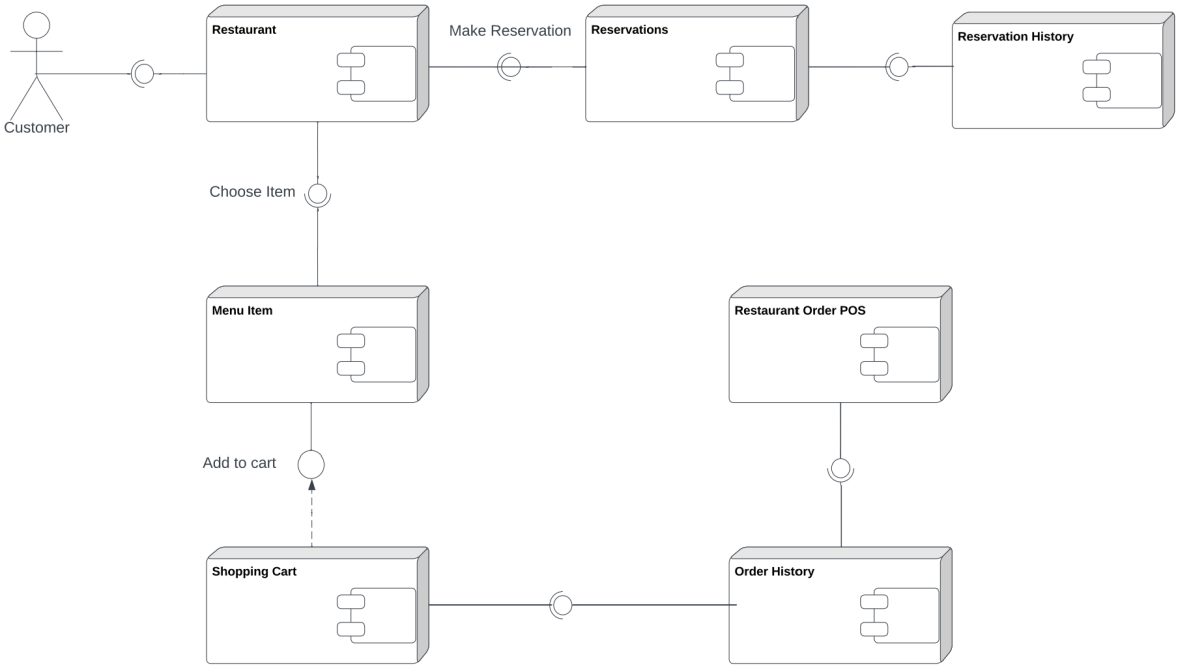
3. Images will be held in the file systems. We do not anticipate having many images, but we will not be putting them in our database if we do.
4. The algorithm for searching will involve calling the search function -> filter using the terms specified -> outputting those results. The database terms that will be searched are the reservations in the reservations table to show the customer which reservations are available for the number of guests that they have to accommodate. We will also have to search the orders placed to show the employee who placed what order, as well as the reservations made page to do the same. The update menu items page will have to be searched as well so that items can be added and removed.

   In the DB, these will all be coded using SQL and it will be organized by the term that is being searched.
5. For our project, our API is built with Node.js using the Express backend framework. We are using Next.js as our overall project framework, which provides API routes and middleware support for us to build on top of. For our authentication solution, we are using passport for local authentication and @hapi/iron for session serialization / deserialization. We decided to use MySQL for our database as our collections have a lot of relations. We are using freesqldatabase to host and serve our database / collections. Using the mysql packing via node package manager (npm) we are able to connect and query the database. On top of that, we are using Prisma to model our data and work with the sql database in an object oriented manner.
6. We will have an algorithm so that when items are added to the cart from the menu page, they will be reflected in the cart.

High-Level UML Diagrams






Customer
Restaurant
Make Reservation
Reservations
Reservation History
Choose Item
Menu Item
Restaurant Order POS
Add to cart
Shopping Cart
Order History

Team

Team Name: Avalanche

- Scrum Master: Steven Luongo
- Product Owner: Sadie Shank
- Development Team: Jaden Badal-Campbell, Isabela Costa, and Brian Rudowitz

Final Roles:

- Steven Luongo - Host site, Back-End Development
- Sadie Shank – Create UI, Write Documents
- Jaden Badal-Campbell – Databases, Set up + SQL
- Isabela Costa – Add More Design to UI, Documents check
- Brian Rudowitz – UML Diagrams

These are the final roles for our group. We have created these after figuring out what each member's strengths are. These final roles might vary a bit as we proceed onto Milestones 4 and 5.

Collaboration

We have decided to include collaboration as its own section because it is an important part of any software project, and it is important for the customer to know that we will collaborate and work efficiently as a team. In order for our team to work efficiently, we have to work together. We cannot have some team members working on parts without reporting their progress to the rest of the team. This would be a problem because when we come to a project deadline, if we have not been reporting our individual progress, then a major development piece might not be ready to launch. This would cause us to miss deadlines and it would be a problem for everyone else because some pieces might build off other pieces so it will have a ripple effect, and nothing will be turned in on time.
For these reasons, collaboration is essential. We must have open communication with each other to ensure that project deliverables are in on time. To provide for collaboration and communication, we have taken the following measures.

First, our group has created a private chat using the Discord application. This tool is useful to us because all group members are familiar with it and check it regularly. Secondly, we are using JIRA to see tasks that are coming up and to assign individual tasks to each group member. They can then see what they are responsible for contributing to and know when they must finish it by. They can mark their tasks as finished so the next person can begin working on their part.

If a group member falls behind due to a lack of knowledge on some part of the application, other group members will come together to help move the project forward in that particular area. If a group member falls behind simply because they are not doing their work on time, then the group will have to communicate together to come to a resolution to ensure everyone is equitably doing their work.

Identify Actual Key Risks for your Project at This Time

1) Skills Risks –

- Problem: Lack of knowledge on connecting website to database. Solution: The most skilled team member – Steven Luongo – who has some experience with it will work on filling in that knowledge gap to get a vertical prototype up. He will then explain to the rest of the members how it works so we can all contribute with the final product.
- Problem: We all lack web development and full stack development skills. Solution: We all try to improve our skills to get our project up and running. For Milestone 4, we work together to fill in each other's knowledge gap.

2) Schedule Risks –

- Problem: With our lack of skills, getting our project finished by the deadline is difficult. Solution: We focus our efforts solely on this project and help each other when needed. We plan more meetings and check in on each other's progress so we don't hold each other up.

3) Technical Risks –

- Problem: We don't know how to host our database. Solution: A group member does research on it to figure out how best to host it. They then report back to the rest of us who can use that information.

4) Teamwork Risks -

- Problem: Team members might not know what their job is supposed to be or there is a lack of communication. Solution: We tell each member from the

beginning what their job is so that everyone knows.

5) Legal / Content Risks –

- Problem: We found out that a similar site exists that allows customers to book reservations and place online orders. Solution: We will ensure that our website is nothing like the one that exists.