

HOMEWORK ASSIGNMENT 8

CSCI 571 – Fall 2022

[Abstract](#)

Ajax, JSON, Angular, Bootstrap, RWD, Node.js, and Yelp API

This content is protected and may not be shared, uploaded, or distributed.

Prof. Marco Papa

papa@usc.edu

Homework 8: Ajax, JSON, Angular, Bootstrap, RWD, Node.js, and Yelp API

1. Objectives

1. Get experience with creating backend applications using JavaScript/Node.js on the server side with Express framework.
2. Get experience with using Angular, TypeScript, and Bootstrap on the client side and creating responsive front-end.
3. Get experience with using HttpClientModule of Angular for AJAX.
4. Get experience with Yelp API.
5. Get experience with Cloud Platform (GCP, AWS, Azure).

2. Homework Description Resources

1. Homework Description Document (This document)
2. Grading Guidelines
3. Web Reference Video: [YouTube video](#)
4. Mobile Reference Video: [YouTube video](#)
5. Piazza

3. General Directions

1. The backend of this homework must be implemented in JavaScript using the Node.js Express framework. Refer to [Node.js website](#) for installing Node.js and learning how to use it. Have a look at “Getting started” guides in the [Express website](#) to learn how to create backend applications using Express. Also, refer to the [Node.js Express framework](#) tutorial to learn more about it. The [Axios library](#) can be useful to make requests from your Node.js backend to Yelp servers. **Implementing the backend in anything other than Node.js will result in a major point reduction.**
2. The frontend of this homework must be implemented using the Angular framework. Refer to the [Angular setup docs](#) for installing Angular and creating Angular projects. The [Angular “Tour of Heroes” app tutorial](#) is a very good tutorial to see different Angular concepts in action. **Implementing the frontend in anything other than Angular will result in a major point reduction.**
3. You are expected to create a responsive website. For that reason, we require you to use Bootstrap, a CSS framework for responsive, mobile-first web development. Bootstrap will save you from the burden of dealing with CSS peculiarities and the website you create will be responsive automatically if you develop within the framework provided by Bootstrap. Please refer to the [Bootstrap docs](#) for reference (especially look at the “Layout” section, and the components that you want to use). Refer to [this post](#) to learn

how to add Bootstrap to Angular projects. **Not using Bootstrap will result in a major point reduction.**

4. The backend of this homework must be deployed to the cloud. The backend should serve the frontend as well as other endpoints you may define. Please refer to Homework 7 for deploying Node.js applications to GCP/Azure/AWS.

5. You must refer to the homework description document (this document), grading guidelines, the reference videos and instructions on Piazza while developing this homework. All discussions and explanations in Piazza related to this homework are part of the homework description and grading guidelines. Therefore, please review all Piazza threads before submitting the assignment. If there is a conflict between Piazza and this description and/or the grading guidelines, answers in Piazza are valid. Piazza posts / responses by the instructors will not increase the functionality of the assignment, and only provide additional specifics on items in this document.

6. The assignment will be graded using the latest version of the Google Chrome browser. Developing this assignment using the latest version of Google Chrome is recommended.

4. System Overview

The system contains three components: 1) browser (frontend), 2) Node.js application (backend) and 3) Yelp servers. You will implement the frontend and the backend. Your backend will include two functionalities: serving the frontend static files to the browser and responding to the frontend's AJAX requests by fetching data from Yelp servers. You will not directly call the Yelp APIs from the frontend as it requires disclosing a secret API key to the public. The data flow diagram after an AJAX call is shown below in **Figure 1**.

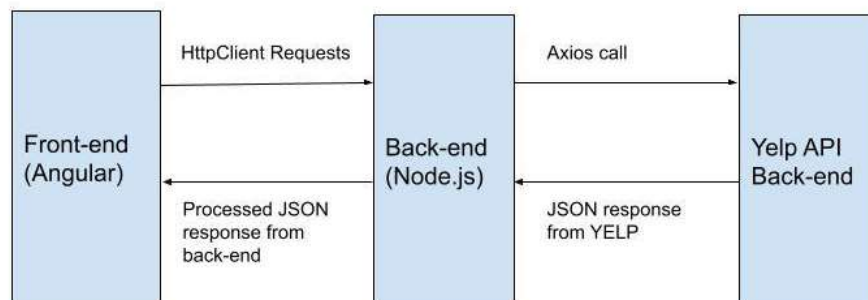


Figure 1: System design overview

NOTE: Setup authorization is required to call Yelp API endpoints and is using request headers with the API key for the app. Refer to the Additional Hints section to see how to add authentication headers. You can re-use the API Key obtained and used in Assignment 6.

Please do not directly call the Yelp API endpoints from your frontend. Calls to other APIs can be made from the frontend.

All requests from frontend to backend must be implemented using the HTTP GET method, as you will not be able to send us sample backend endpoint links as a part of your submission if you use HTTP POST.

5. Description

In this exercise, you are asked to create a web application that allows you to search for business information using the [Yelp API](#), and the results will be displayed in a card and tabular format. It also allows users to make reservations and see the list of their reservations. Also, users can share a post on Facebook and tweet on Twitter about the business.

All implementation details and requirements will be explained in the following sections.

There are 2 front-end routes/pages for this application:

- a) Search Route ['/search'] – It is a default route of this application which is used to search for businesses and see business details
- b) Bookings Route ['/bookings'] – It displays the list of user reservations

Hint: Refer [Angular routing tutorial](#) for implementing routes

5.1 Navbar component

The Navigation bar must be present on top of all the routes of the application as shown in **Figure 2** below. You can use Bootstrap to create a navbar. It consists of following menu options:

- 1. Search
- 2. My Bookings

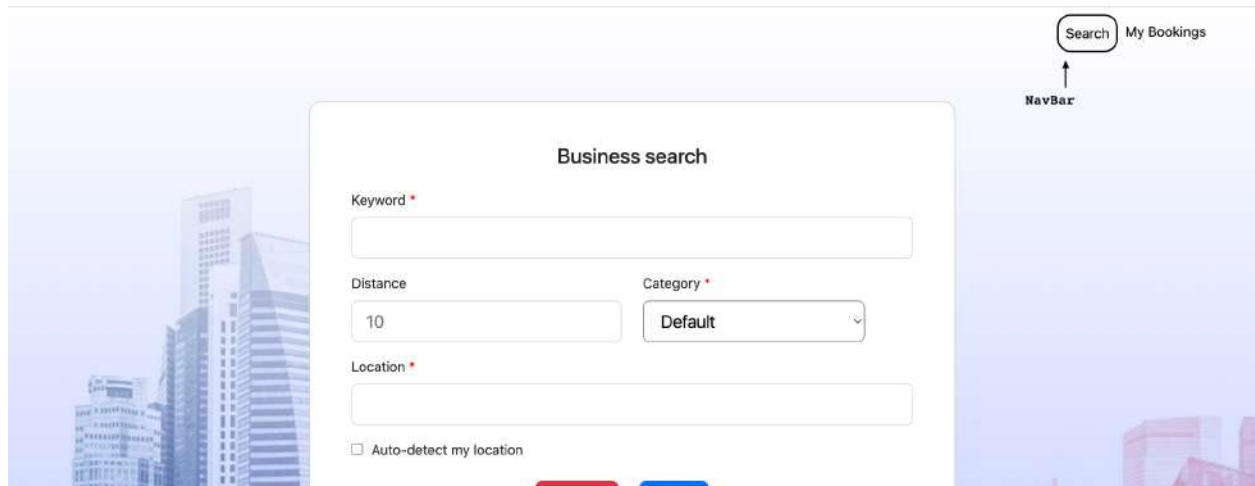


Figure 2: Navbar

5.2 Search Route

The Search route consists of various components:

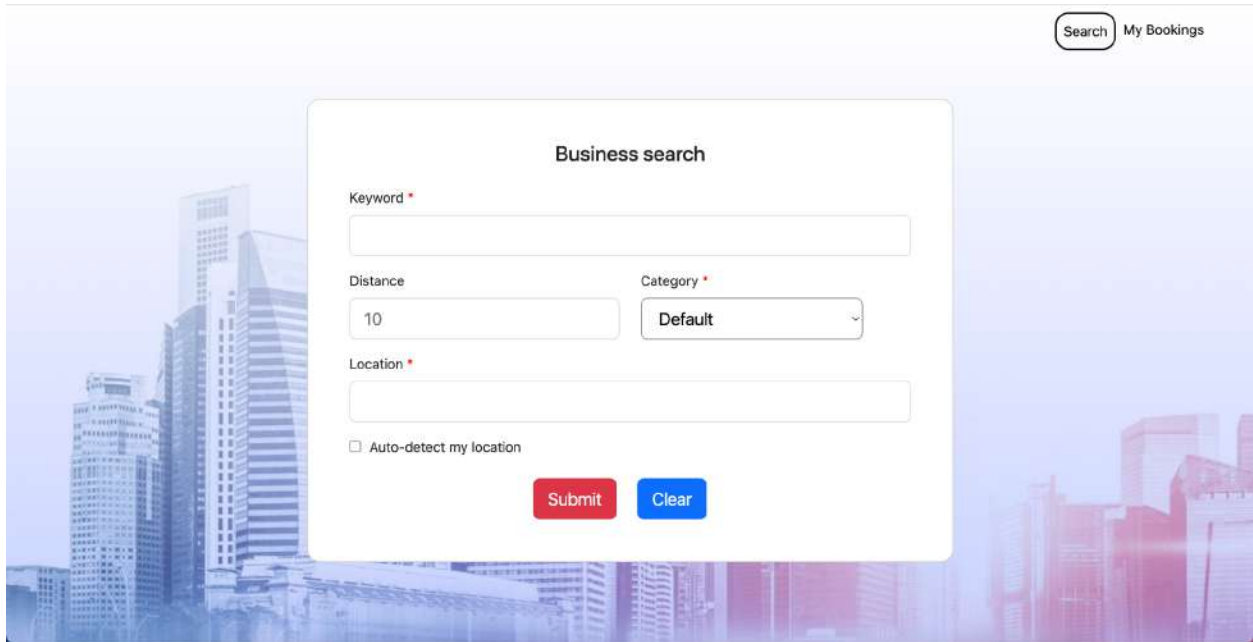
- Search Form
- Results Table
- Details card
- Reservation Form

5.2.1 Search Form

The Search form has 5 fields: Keyword, Distance (miles), Category, Location, and a checkbox to auto-detect location as you can see in **Figure 3**. You should use the ipinfo.io API (as you did in Assignment 6) to fetch the user's geolocation if the location checkbox is checked. Otherwise, the user must enter an address location to search. Keyword, Distance and Location should be implemented as text boxes while Category should be implemented as a dropdown.

These are the categories to include in the dropdown:

- Default
- Arts & Entertainment
- Health & Medical
- Hotels & Travel
- Food
- Professional Services

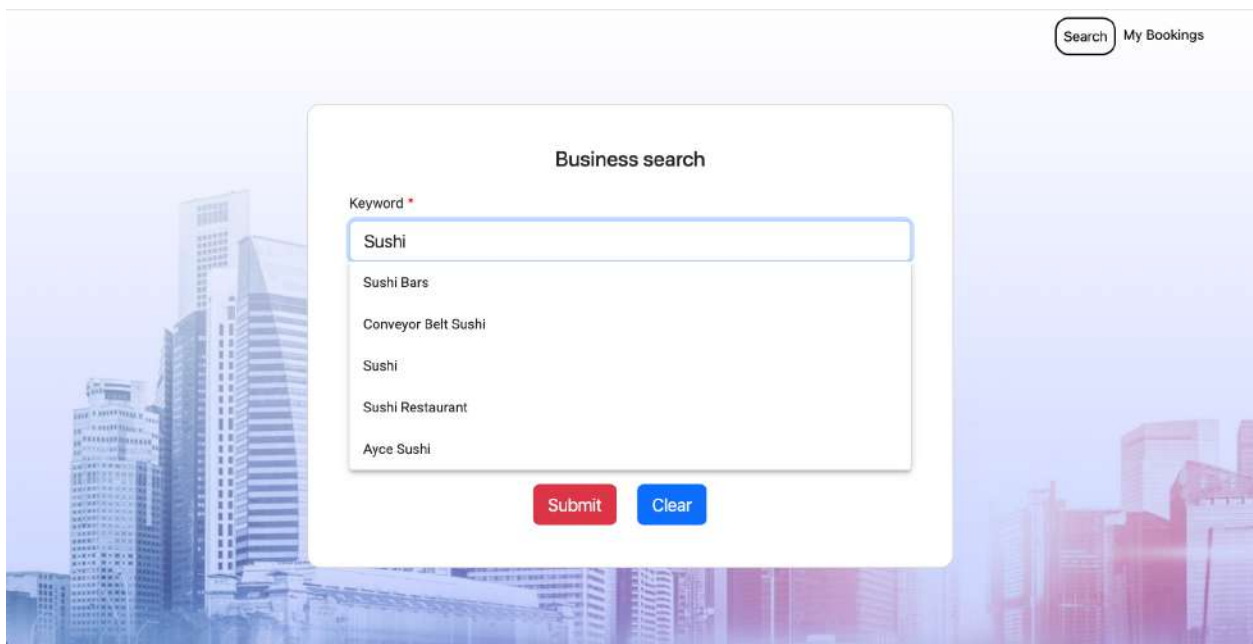
A screenshot of a web application's business search form. The form is titled "Business search" and is set against a background image of a city skyline. In the top right corner, there are two links: "Search" and "My Bookings". The form contains the following fields and controls:

- Keyword ***: A text input field.
- Distance**: A text input field containing the value "10".
- Category ***: A dropdown menu with "Default" selected.
- Location ***: A text input field.
- ☐ **Auto-detect my location**: A checkbox.
- Submit**: A red button.
- Clear**: A blue button.

Figure 3: Business search form

5.2.1.1 Autocomplete

The Keyword Input field allows the user to enter a keyword to retrieve results. Based on the user input, the text box should display a list of all the suggestions fetched using an autocomplete API, as seen in **Figure 4**.

A screenshot of the same business search form as in Figure 3, but with the "Keyword" field populated with the text "Sushi". Below the input field, a list of suggestions is displayed:

- Sushi Bars
- Conveyor Belt Sushi
- Sushi
- Sushi Restaurant
- Ayce Sushi

The "Submit" and "Clear" buttons remain at the bottom of the form.

Figure 4: Example of Autocomplete

This is the API endpoint for “autocomplete”:

GET <https://api.yelp.com/v3/autocomplete>

Please refer to this documentation for the Yelp autocomplete API:

<https://www.yelp.com/developers/documentation/v3/autocomplete>

This is a sample API call:

[https://api.yelp.com/v3/autocomplete?text=\[KEYWORD\]](https://api.yelp.com/v3/autocomplete?text=[KEYWORD])

Table 1 shows relevant information to be fetched from the above mentioned API endpoint’s response.

Name	Used for
categories[x].title	Title of a category for display purpose.
terms[x].text	The text content of the term autocomplete suggestion.

Table 1: Autocomplete Endpoint

The autocomplete function should be implemented using **Angular Material Autocomplete**. Please refer to the [Angular Material Autocomplete](#) tutorial, for the details.

5.2.1.2 Location Field

If the **Auto-detect checkbox** is checked then the location field should reset the **Location** textbox to blank and disable the field. When the **Auto-detect checkbox** is not checked, the user needs to enter the location address. Use the Google’s Geocoding API to get latitude and longitude of the location that the user entered and pass that latitude / longitude values in the Yelp’s business search API.

The Google Maps Geocoding API is documented here:

<https://developers.google.com/maps/documentation/geocoding/start>

The Google Maps Geocoding API expects two parameters:

1. **address**: The location that you want to geocode, in the format used by the national postal service of the country concerned. Additional address elements such as business names and unit, suite or floor numbers should be avoided.

2. **key:** Your Google application's API key. This key identifies your application for purposes of quota management.

An example of an HTTP request to the Google Maps Geocoding API, when the location address is “University of Southern California, CA” is shown below:

https://maps.googleapis.com/maps/api/geocode/json?address=University+of+Southern+California+CA&key=YOUR_API_KEY

The response includes the latitude (lat) and longitude (lng) of the address

```
▼ results:
  ▼ 0:
    ▶ address_components: [...]
    formatted_address: "Los Angeles, CA 90007, USA"
    ▼ geometry:
      ▼ location:
        lat: 34.0223519
        lng: -118.285117
        location_type: "GEOMETRIC_CENTER"
      ▼ viewport:
        ▼ northeast:
          lat: 34.0237008802915
          lng: -118.2837680197085
        ▼ southwest:
          lat: 34.0210029197085
          lng: -118.2864659802915
      place_id: "ChIJ7aVxn0THwoARxKIntFtakKo"
    ▼ types:
      0: "establishment"
      1: "point_of_interest"
      2: "university"
    status: "OK"
```

Figure 5: Example of JSON object from Google Maps Geocoding

Figure 5 shows an example of the JSON object returned by the Google Maps Geocoding API web service response.

The latitude (lat) and longitude (lng) of the location are used when constructing a RESTful web service URL to retrieve matching search results.

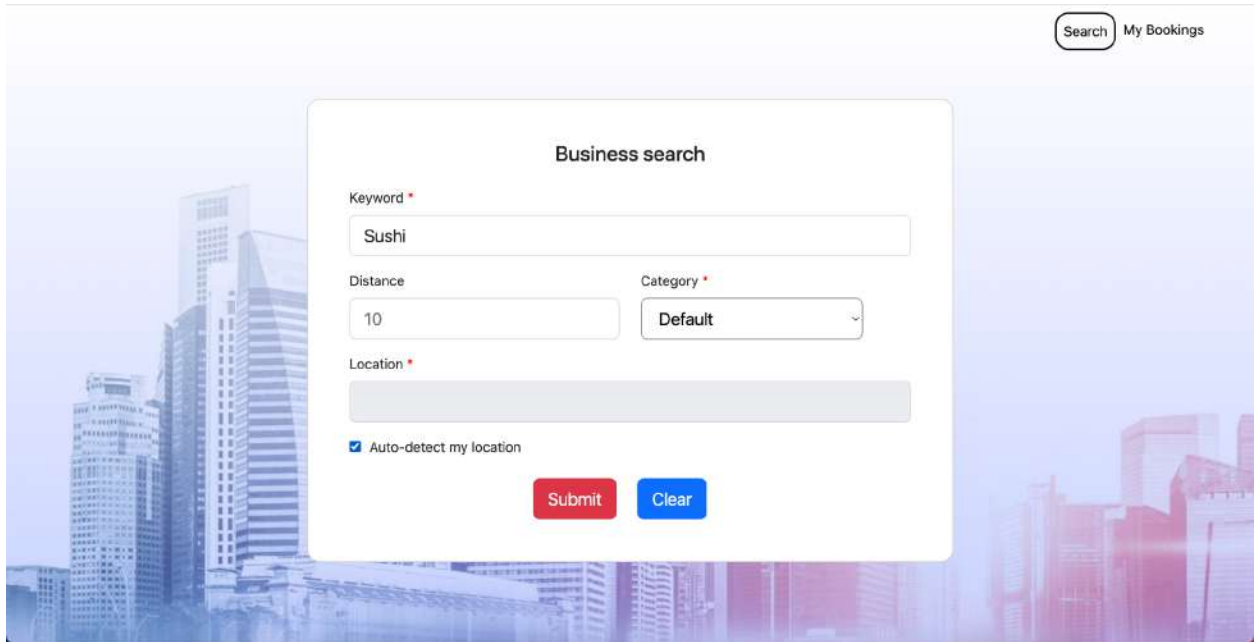


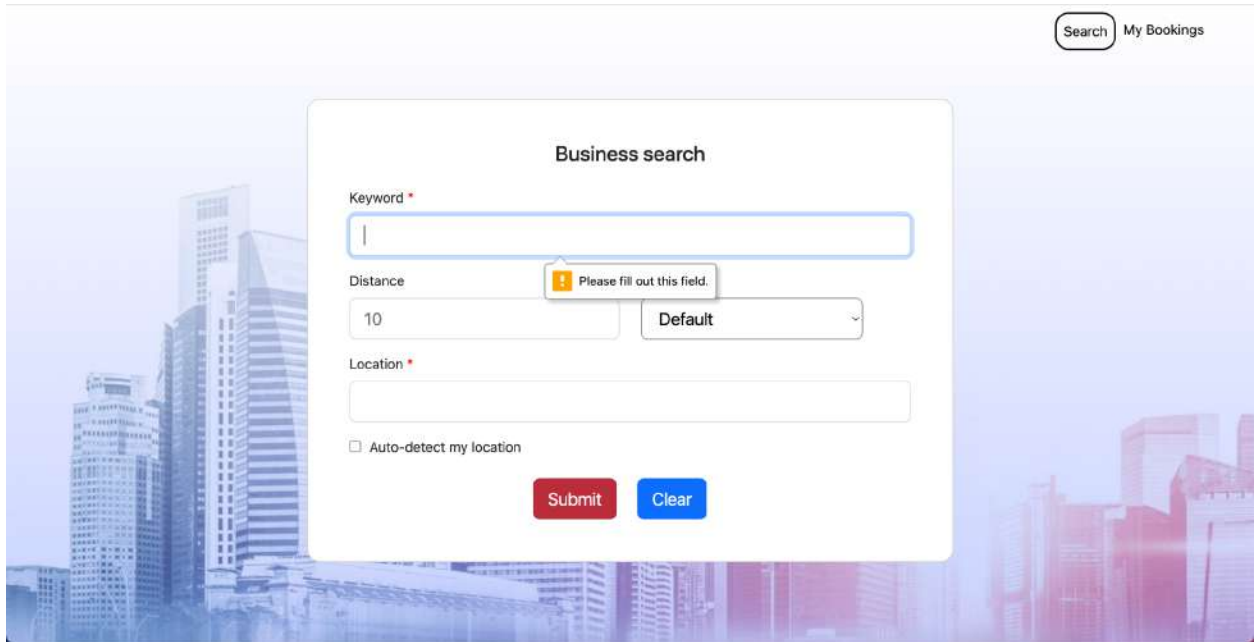
Figure 6: Example of auto-detect location

5.2.1.3 Submit button

Selecting the **Submit button** performs a search using the given business *keyword*, the *distance* filter from the given location and the *category* of businesses. An example of valid input is shown in **Figure 8**. Once the user has provided valid input, your front-end should send a request to your back-end NodeJS server with the form inputs. You must use GET to transfer the form data to your web server (**do not use POST**, as you would be unable to provide a sample link to your cloud services). The back-end code will extract the form inputs and make an `axios()` call using them to invoke the *YELP API* business information service . You need to use the back-end to make all the YELP API calls. You can use other libraries in lieu of `axios()`.

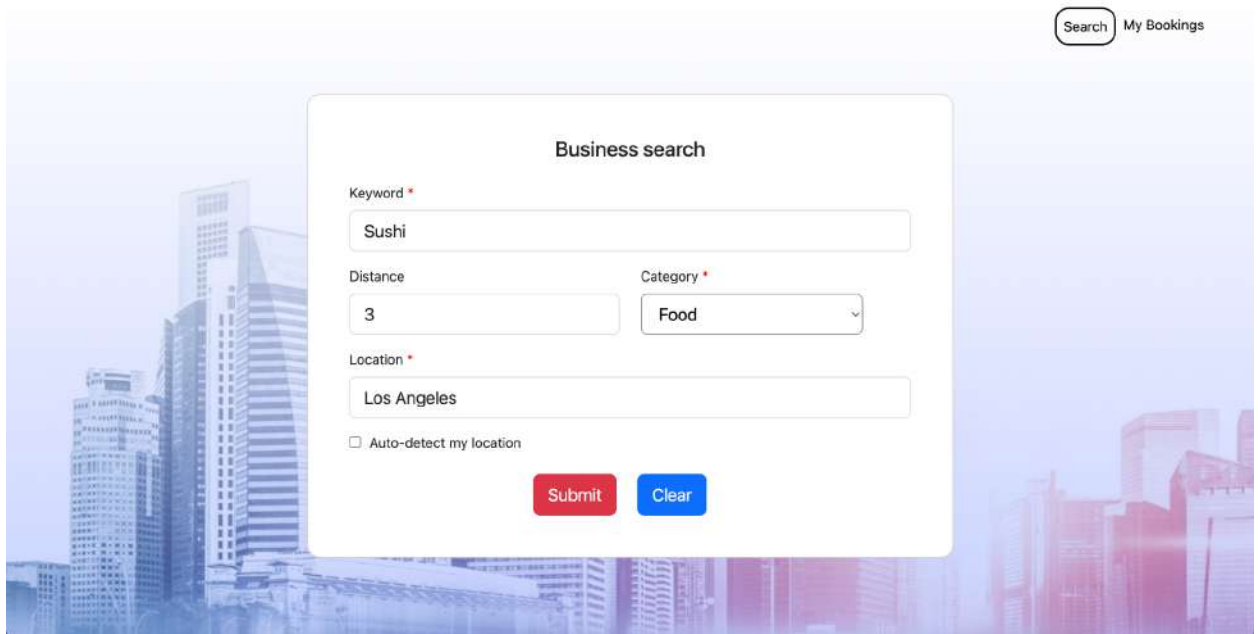
If the user clicks on the Submit button without providing a value in the “Keyword”, “Category” or “Location” fields or checking the location checkbox, you should show an error “tooltip” that indicates which field is missing. An example is shown in **Figure 7**.

Using HttpClient Requests for anything other than calling your own “cloud” backend will lead to a penalty, except for IpInfo calls. Do not call the Yelp API or Google Geocoding API directly from the front-end. You may use HttpClient Request to call the *Ipinfo API* service directly from the front-end.



The image shows a web interface for a business search. At the top right, there are links for "Search" and "My Bookings". The main form is titled "Business search" and contains the following fields: "Keyword" (empty), "Distance" (set to 10), "Location" (empty), and a "Category" dropdown (set to "Default"). A red validation message "Please fill out this field." is displayed above the "Distance" field. At the bottom of the form are "Submit" and "Clear" buttons. The background is a city skyline.

Figure 7: Example of form validation



The image shows the same "Business search" form as in Figure 7, but with valid input. The "Keyword" field contains "Sushi", the "Distance" field contains "3", the "Location" field contains "Los Angeles", and the "Category" dropdown is set to "Food". The "Auto-detect my location" checkbox is unchecked. The "Submit" and "Clear" buttons are at the bottom. The background is a city skyline.

Figure 8: Example of valid input

5.2.1.4 Clear Button

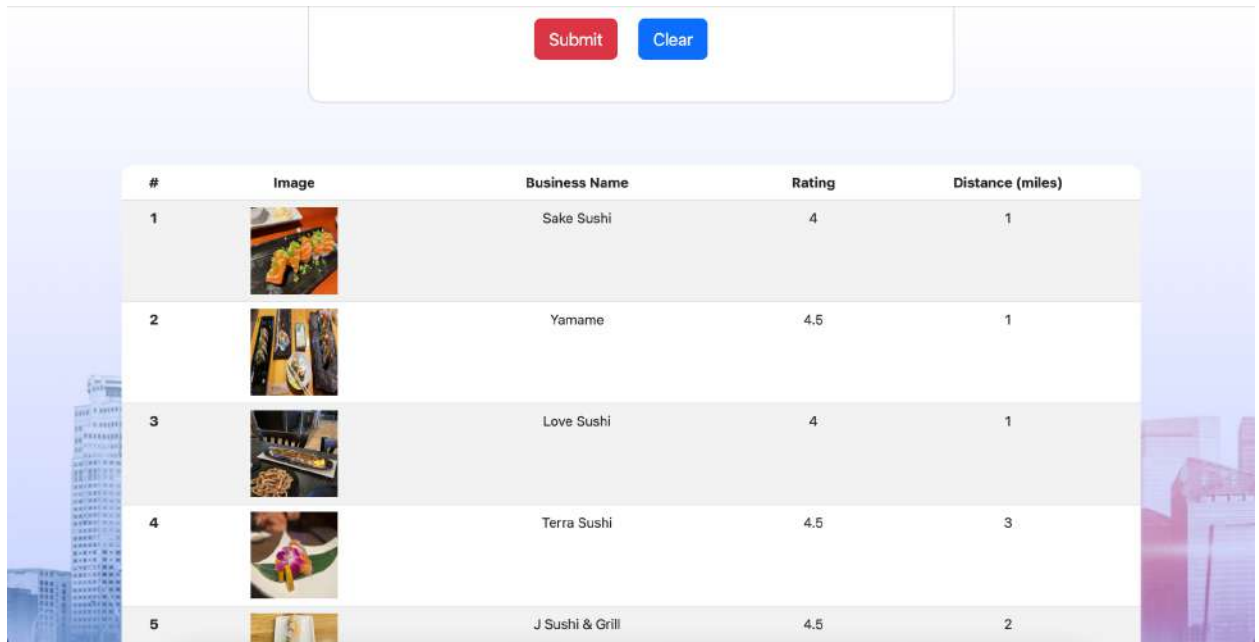
The **Clear button** must clear the result area (below the search area) and set all the form fields to the default values in the search area. The “clear” operation must be done using a JavaScript function.

5.2.2 Results Table

The response received from the backend after clicking the Submit button will be parsed and displayed in a table as shown in **Figure 9** and **Figure 10**. The **results table** consists of 5 columns:

- 1) #
- 2) Image
- 3) Business Name
- 4) Rating
- 5) Distance(miles)

The results table must display **at most 10 results**. Each row of the results table is such that on **clicking anywhere in the row** should display a **card showing more** details about the business, which will be discussed in the next section








#	Image	Business Name	Rating	Distance (miles)
1		Sake Sushi	4	1
2		Yamame	4.5	1
3		Love Sushi	4	1
4		Terra Sushi	4.5	3
5		J Sushi & Grill	4.5	2

Figure 9: Example of results table







5		J Sushi & Grill	4.5	2
6		Happy Oasis	4	0
7		Sashimi	4.5	4
8		Samurai Sushi	4	4
9		Sushi Kizuna	5	9
10		Ahi Ahi Sushi Bar & Grill	4	4

Figure 10: Example of results table (max of 10)

If results are not found, display “No results available”, as shown in **Figure 11**.

Search

My Bookings

Business search

Keyword *

xyzszsszs

Distance

10

Category *

Default

Location *

☒ Auto-detect my location

Submit

Clear

No results available

Figure 11: Example of no search result found

The API endpoint for search:

GET <https://api.yelp.com/v3/businesses/search>

Refer to the search documentation at:

https://www.yelp.com/developers/documentation/v3/business_search

API Sample:

[https://api.yelp.com/v3/businesses/search?term=\[KEYWORD\]&latitude=\[LAT\]&longitude=\[LONG\]&categories=\[CAT\]&radius=\[RAD\]](https://api.yelp.com/v3/businesses/search?term=[KEYWORD]&latitude=[LAT]&longitude=[LONG]&categories=[CAT]&radius=[RAD])

Table 2 shows relevant information to be fetched from the above mentioned API endpoint's response is in the following table

Name	Used for
business[x].id	Business Id
business[x].name	Business Name/Title
business[x].image_url	URL of photo for this business.
business[x].rating	Rating for this business (value ranges from 1, 1.5, ... 4.5, 5).
business[x].distance	Distance in meters from the search location. This returns meters regardless of the locale.

Table 2: Business Search Endpoint

5.2.3 Details Card

The **Details card** consists of back arrow, business name and **three tabs** as follows :

- 1) Business Details
- 2) Map Location
- 3) Reviews

Please refer to the [Angular Material tab](#) tutorial for implementing angular tabs.

API endpoint for search

GET <https://api.yelp.com/v3/businesses/{id}>

Refer: <https://www.yelp.com/developers/documentation/v3/business>

Table 3 shows relevant information to be fetched from the above mentioned API endpoint's response.

Name	Used for
categories[x].title	Title of a category for display purpose.
coordinates	The coordinates of this business.
phone	Phone number of the business.
hours[x].is_open_now	Whether the business is currently open or not.
location	The location of this business, including address, city, state, zip code and country.
id	Unique Yelp ID of this business. Example: '4kMBvIEWPxWkWKFN__8SxQ'
name	Name of this business.
photos	URLs of up to three photos of the business.
price	Price level of the business. Value is one of \$, \$\$, \$\$\$ and \$\$\$\$.
url	URL for business page on Yelp.

Table 3: Business Details Endpoint

Notes:

- Location from **Table 2** is used for address and coordinates can be used to locate the business on the Map location tab
- Display Category using a | separator as shown in the video.

5.2.3.1 Business details tab

The **Business details** tab shows business address, category, phone number, price range, status, business website link, reserve now button, Facebook icon, Twitter icon and business images. These are descriptions of each field:

- Address – The location of this business, including address, city, state, zip code and country.
- Category – Title of a category for display purpose.
- Phone number – Phone number of the business.
- Price range – Price level of the business. Value is one of \$, \$\$, \$\$\$ and \$\$\$\$.

- Status – Open now or closed. If business is open, show it in green color otherwise red color as shown in **Figure 12**.
- Business website link – On clicking the business link, it should open the business website in a new tab.
- **Reserve Now** button – On clicking, it should display a reservation form modal containing business name and other reservation details input fields: Email, Date and Time.
 - Email – Should accept valid email addresses only. Otherwise, an error message must be displayed on clicking the submit button as shown in **Figure 17**.
 - Date – On clicking the calendar icon, it should allow selection from today's date to all the future dates and disabling all the past dates.
 - Time – Time can be selected from drop-down between 10AM to 5 PM. Time slots are available at an interval of 15 mins. More visual details are shown in **Figure 19** and **Figure 20**.

On clicking the **Submit button** in the Reservation form, the input fields are validated and stored in local storage which is explained in **Section 5.3**. See **Figure 16**.

On clicking the **Submit button**, display a 'Reservation created!' alert and change the Reserve Now button to **Cancel Reservation**. Once a reservation is done, if we revisit the details of that restaurant, it should display the Cancel reservation button instead of the Reserve now button, until the reservation is canceled.

On clicking the **Cancel Reservation** button, display a 'Reservation canceled!' alert and change the button to display **Reserve Now**.

7) **Close button** – On clicking the Close button, close the modal and on re-opening the modal, the reservation form should be empty.

- Facebook icon – On clicking the icon, create a post in a new tab containing the business's Yelp link.
- Twitter icon – On clicking the icon, create a tweet in a new tab containing Check <business_name> on Yelp, followed by business's yelp link as shown in **Figure 13**.
- Images – **Carousel** of business images is displayed at the bottom of the card. On clicking the left or right arrow, the carousel should display images in a rotating queue fashion. Once you click on any of the arrows, images should start to auto-rotate with some delay as shown in the video

Please refer to [this](#) tutorial as a bootstrap reference for implementing the Reservation Form modal. Please refer to [this](#) tutorial for the image carousel.

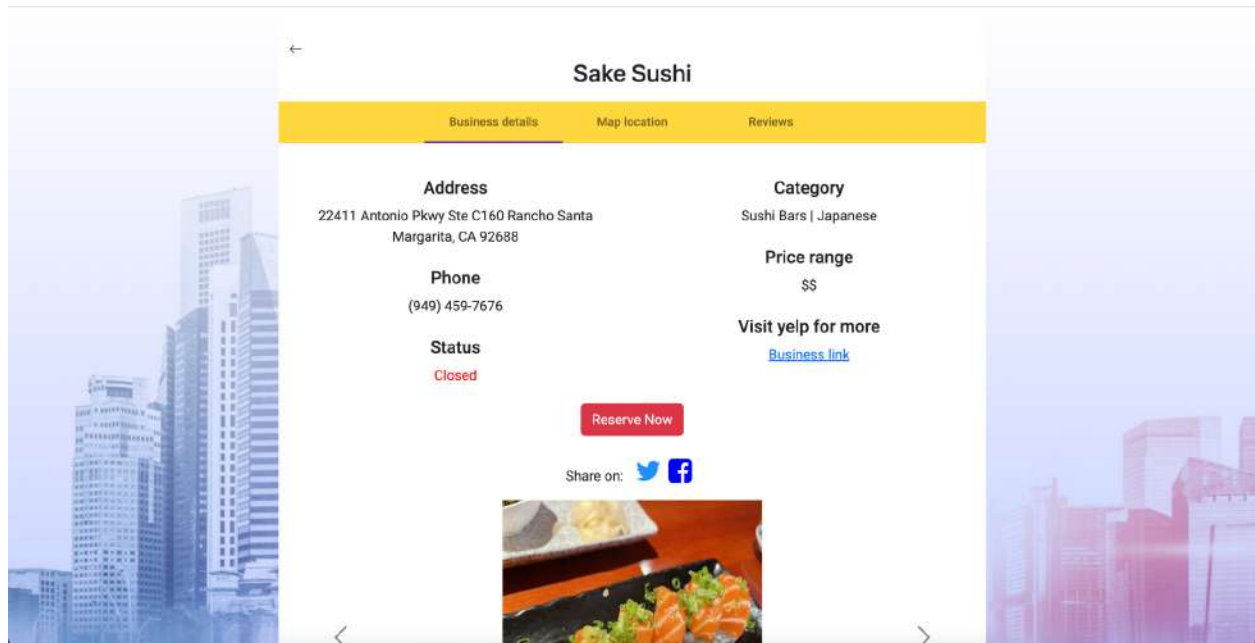


Figure 12: Example of Business Details card

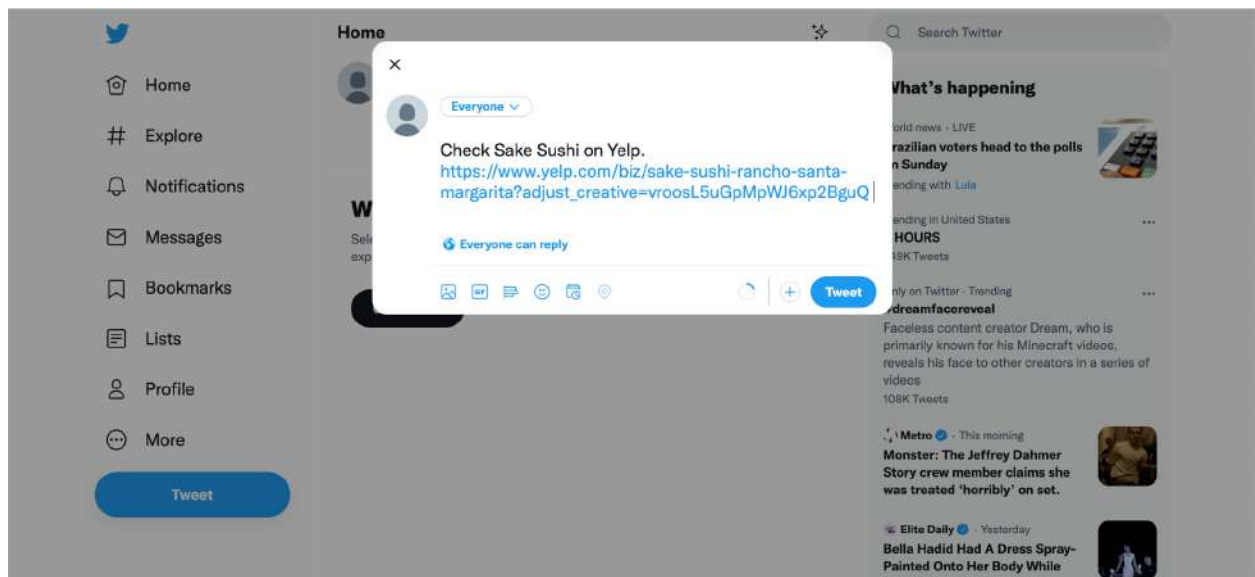


Figure 13: Example of tweet



Figure 14: Example of Facebook post

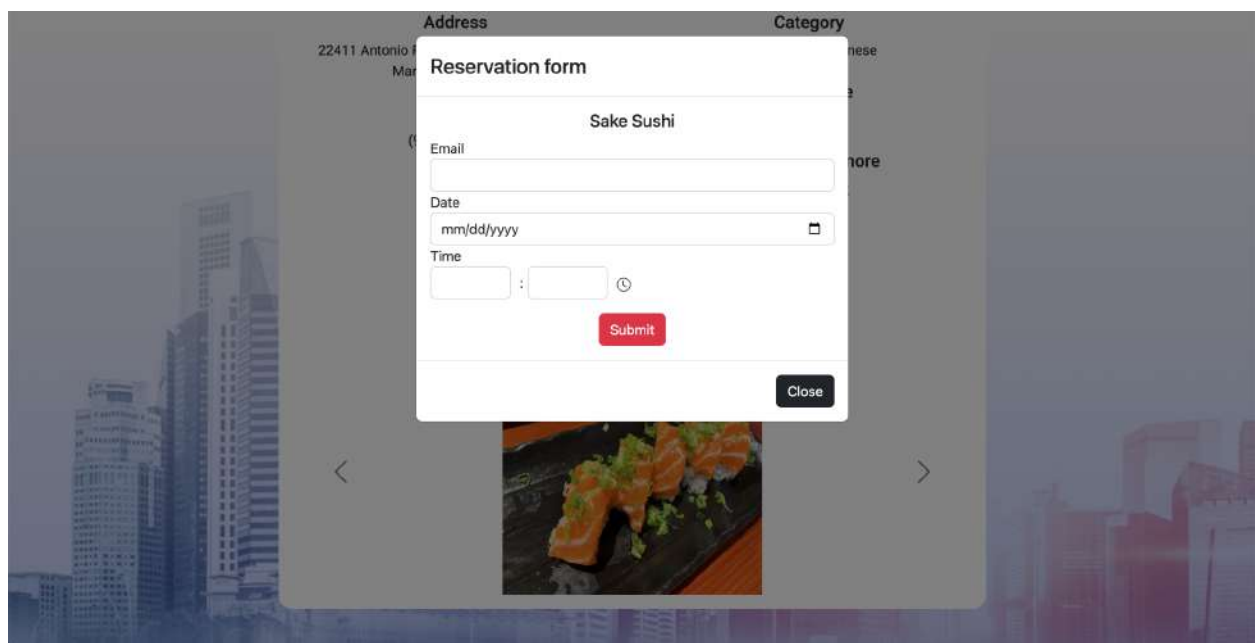


Figure 15: Example of reservation reservation form modal

A screenshot of a mobile application showing a reservation form for "Sake Sushi". The form is titled "Reservation form" and has a "Close" button in the bottom right corner. The form contains the following fields:

- Email:** A text input field with a red border and a red information icon. Below it, the text "Email is required" is displayed in red.
- Date:** A date picker field with a red border and a red information icon. Below it, the text "Date is required" is displayed in red.
- Time:** A time picker field with a red border and a red information icon. Below it, the text "Time is required" is displayed in red.

The form also includes a red "Submit" button. The background of the app shows a city skyline and a restaurant interior.

Figure 16: Example of reservation form validation

A screenshot of a mobile application showing a reservation form for "Board & Brew - RSM". The form is titled "Reservation form" and has a "Close" button in the bottom right corner. The form contains the following fields:

- Email:** A text input field with a red border and a red information icon. The input contains the text "abc". Below it, the text "Email must be a valid email address" is displayed in red.
- Date:** A date picker field with a red border and a red information icon. The input contains the date "10/04/2022".
- Time:** A time picker field with a red border and a red information icon. The input contains the time "12:45".

The form also includes a red "Submit" button. The background of the app shows a city skyline and a restaurant interior.

Figure 17: Example of invalid input

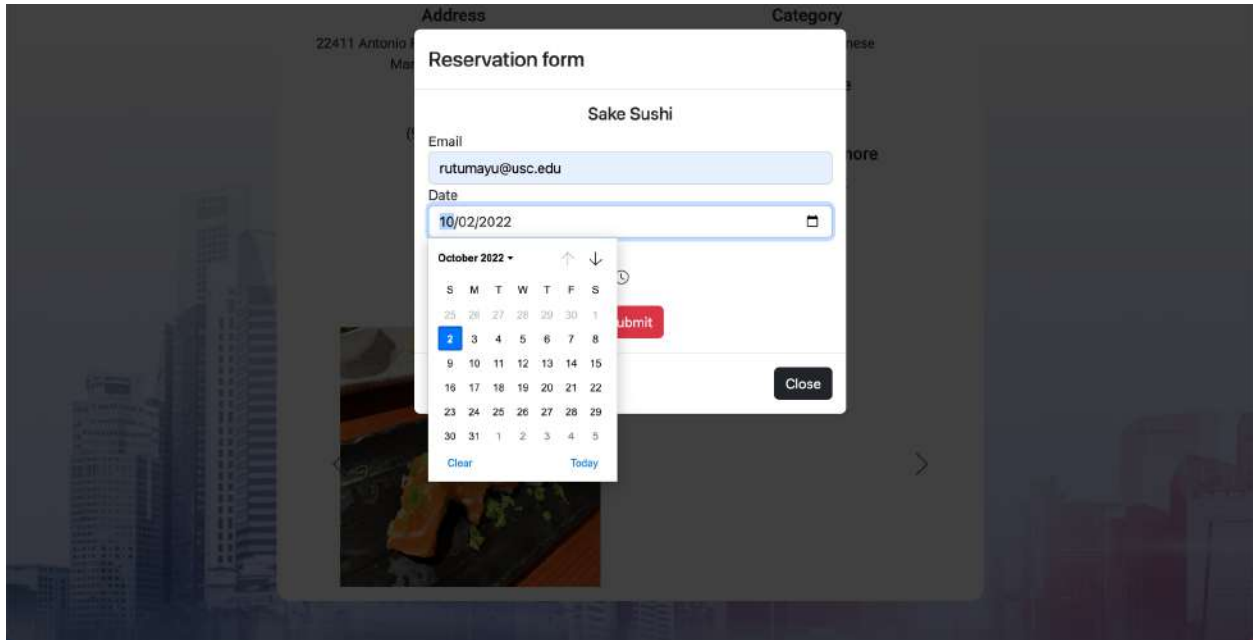


Figure 18: Example of calender display

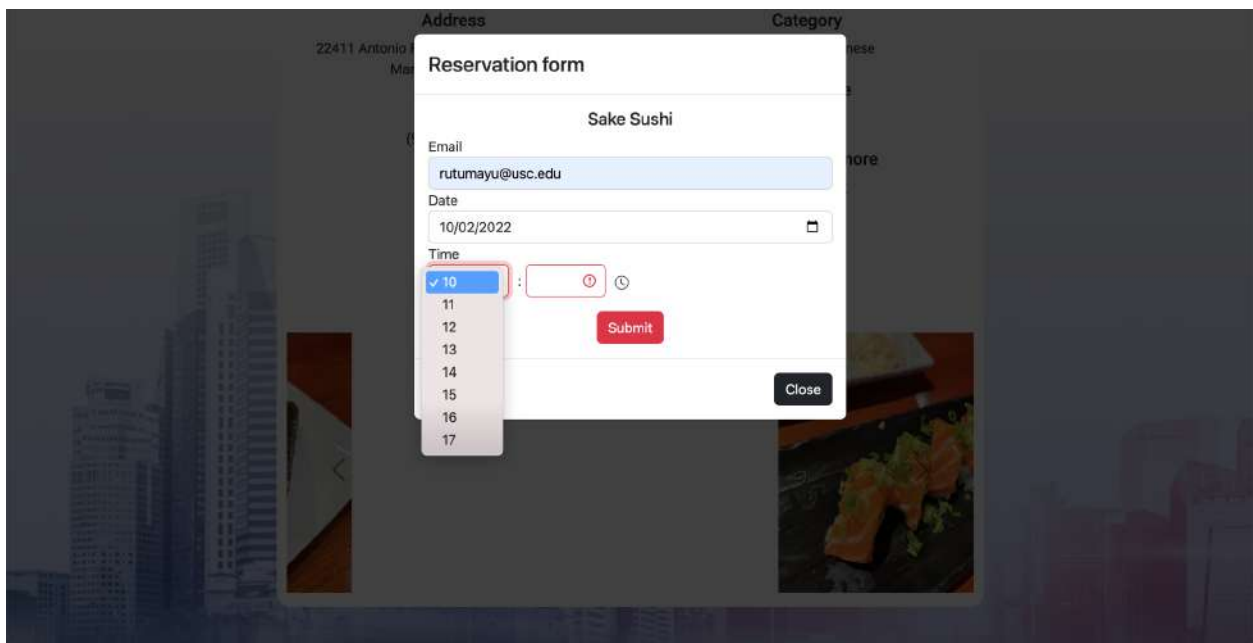


Figure 19: Example of time display

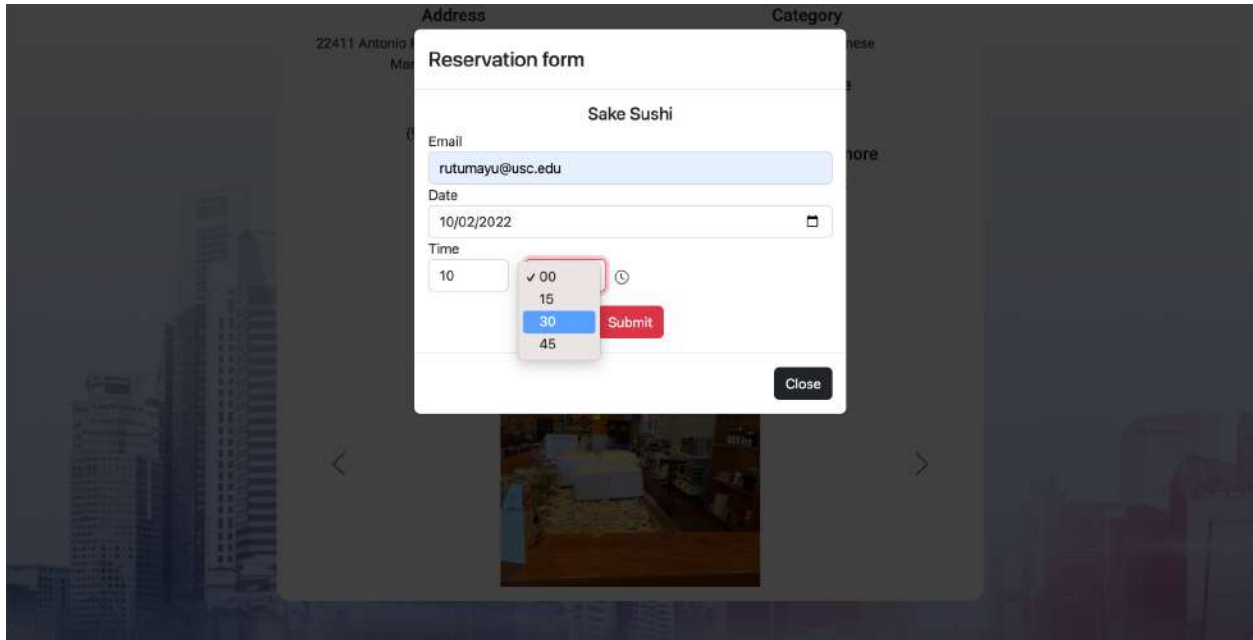


Figure 20: Example of time display

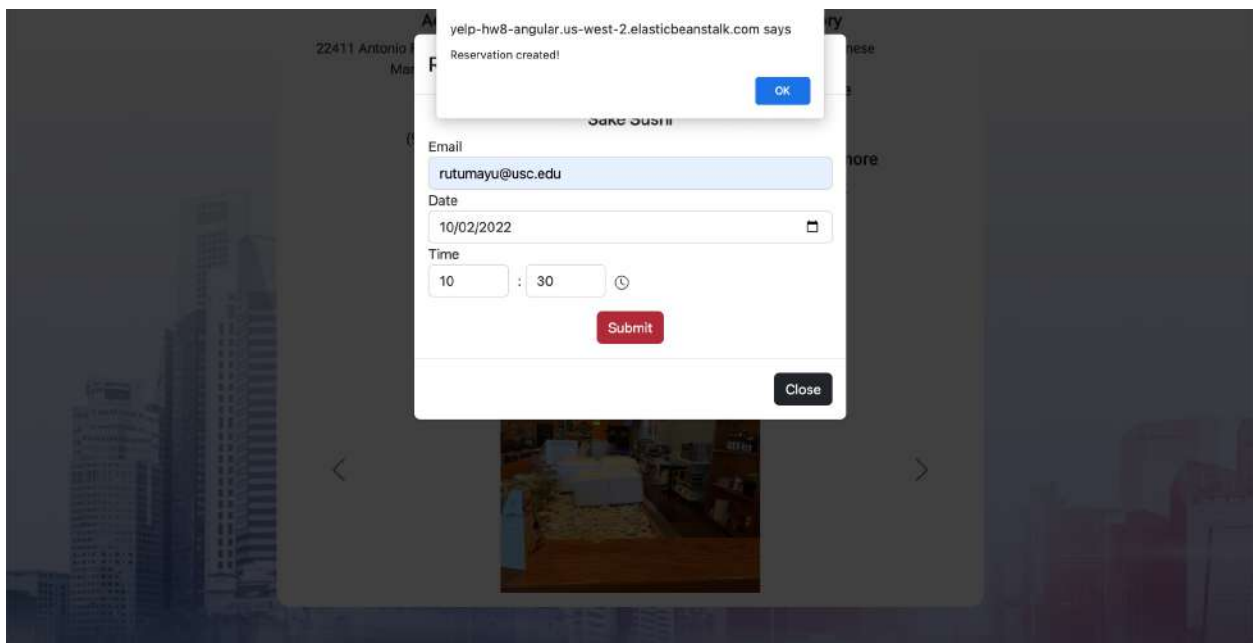


Figure 21: Example of reservation alert

5.2.3.2 Map Location tab

The **Map Location tab** should display a Google map with the business location. You can **only** use the '@angular/google-maps' package to load/display Google Maps.

Please refer to this [tutorial](#) for adding Google Maps to Angular. See **Figure 22**.

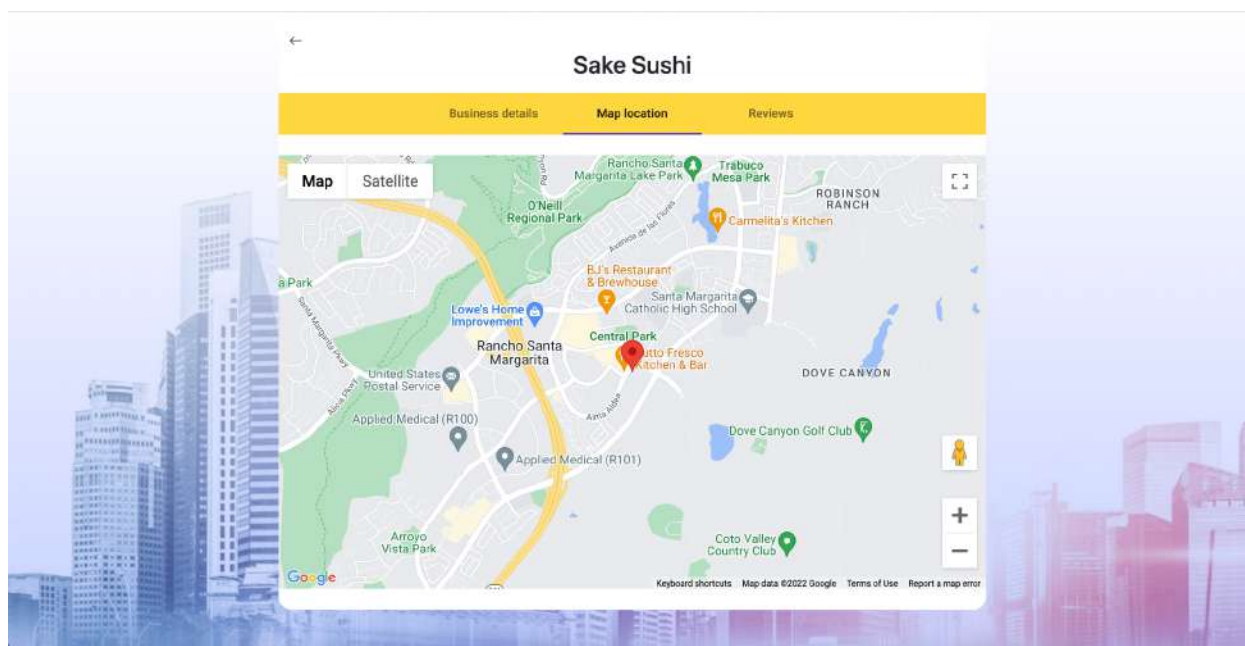


Figure 22: Example of Map location tab

5.2.3.3 Reviews Tab

The **Reviews tab** should display at most 3 reviews for the business as shown in **Figure 23**. Each review should contain Reviewer's Name, Rating, Review and Review's Date. Use the following Yelp API endpoint to fetch results for that business.

API endpoint for reviews:

GET <https://api.yelp.com/v3/businesses/{id}/reviews>

Refer: https://www.yelp.com/developers/documentation/v3/business_reviews

Table 4 shows relevant information to be fetched from the above mentioned API endpoint's response.

Name	Used for
reviews[x].rating	Rating of this review.
reviews[x].user.name	User screen name (first name and first initial of last name).

reviews[x].text	Text excerpt of this review.
reviews[x].time_created	The time that the review was created in PST.

Table 4: Business Reviews Endpoint

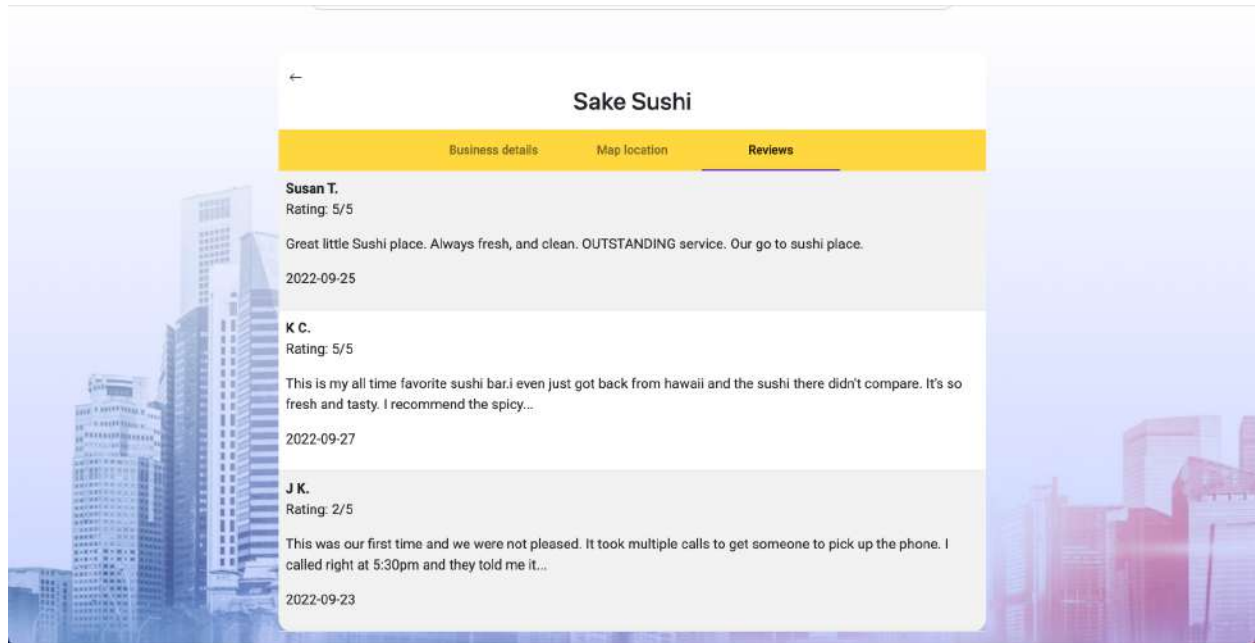


Figure 23: Example of reviews

5.3 Bookings Route

The Bookings route displays a list of bookings (i.e., reservations). It fetches all the reservations from the local storage and displays them in a table of 5 columns. The titles of the 5 columns should be #, **Business Name**, **Date**, **Time** and **E-mail**. See **Figure 24**. For each record, display a Trash Icon. On clicking the Trash icon, Cancel the reservation and display a 'Reservation canceled!' alert. Also remove the canceled reservation from the table of reservations. If no reservations are available, display '**No reservations to show**' message.

Refer to the [HTML Web Storage](#) tutorial for implementing Local Storage.

Note: All the CSS should be matched as shown in Images/Video.

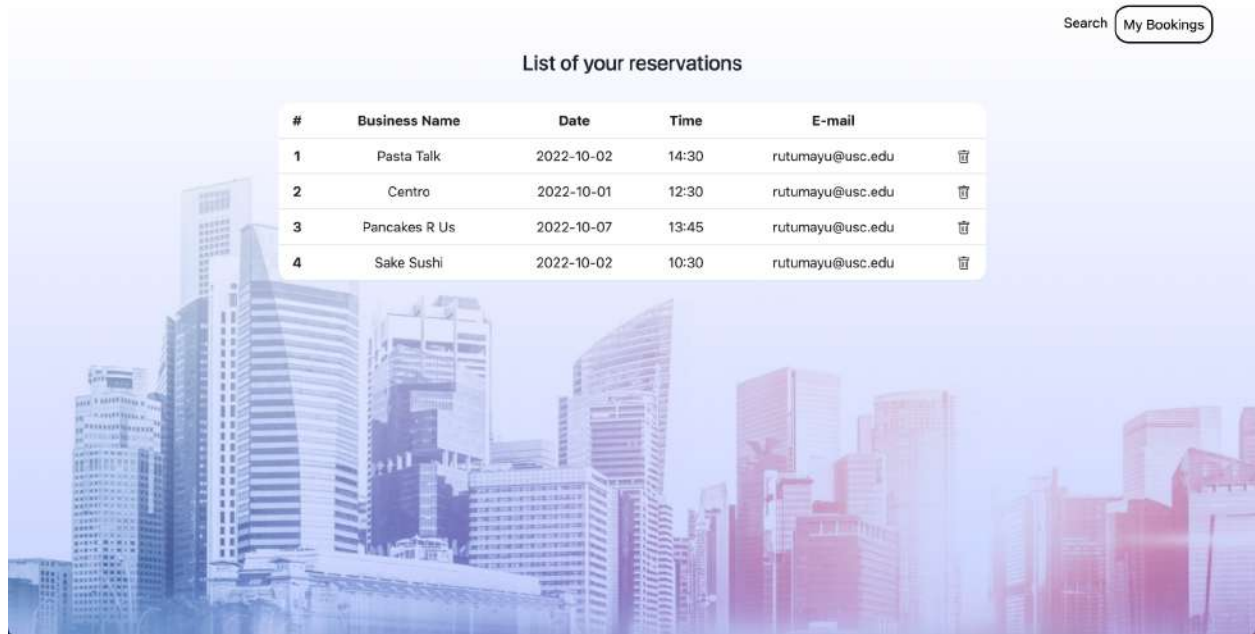


Figure 24: Example of reservation list

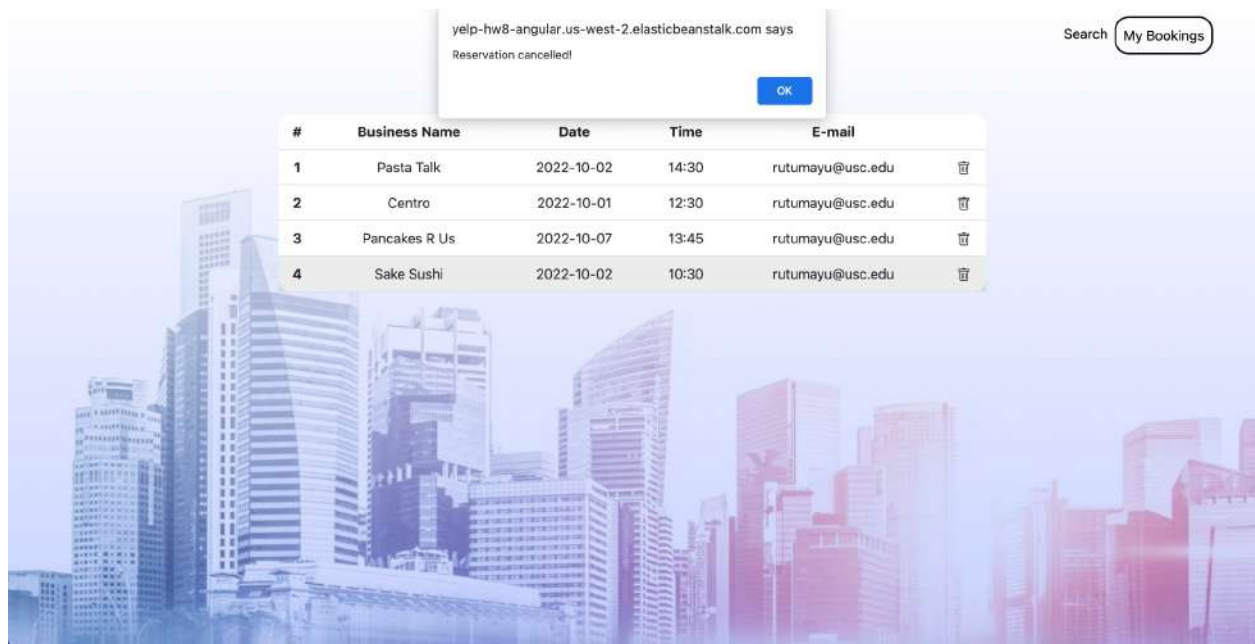


Figure 25: Example of reservation cancel alert

6. Responsive Design

The webpage you develop must be responsive. One easy way to test responsive behavior is to use Google Chrome Responsive Design Mode in Developer Console. Here are some snapshots for iPhone 12 Pro using the Google Chrome Responsive Design Mode. All functions should work on mobile devices.

The image displays three snapshots of a web application interface for business reservations, tested on an iPhone 12 Pro using Google Chrome's Responsive Design Mode. Each snapshot features a top navigation bar with 'Search' and 'My Bookings' links.

Snapshot 1 (Left): Business search form

The 'Business search' form includes the following fields and controls:

- Keyword ***: A text input field.
- Distance**: A text input field containing the value '10'.
- Category ***: A dropdown menu with 'Default' selected.
- Location ***: A text input field.
- ☐ **Auto-detect my location**: A checkbox option.
- Submit** and **Clear** buttons: Red and blue buttons respectively.

Snapshot 2 (Middle): List of your reservations

The 'List of your reservations' section displays a table with the following data:

#	Name	Date	Time	E-mail
1	Pasta Talk	2022-10-02	14:30	rutumayu@usc.ec
2	Centro	2022-10-01	12:30	rutumayu@usc.ec
3	Pancakes R Us	2022-10-07	13:45	rutumayu@usc.ec

Snapshot 3 (Right): Business search form with validation error

This snapshot shows the 'Business search' form with a validation error message: 'Please fill out this field.' displayed next to the 'Distance' input field, which is currently empty.

SearchMy Bookings

Business search

Keyword *

Sushi

Sushi Bars

Conveyor Belt Sushi

Sushi

Sushi Restaurant

Ayce Sushi

☐ Auto-detect my location

SubmitClear

SearchMy Bookings

Business search

Keyword *

Sushi

Distance

10

Category *

Default

Location *

☐ Auto-detect my location

SubmitClear

SearchMy Bookings

Business search

Keyword *

Sushi

Distance

10

Category *











Default

Location *

Los Angeles

☐ Auto-detect my location

SubmitClear

#	Image	Business Name	Rating	Distance (miles)
1		Sushi Gen	4.5	0
2		Sushi Enya - Los Angeles	4.5	0
3		Izakaya Osen - Los Angeles	4.5	3
4		KazuNori The Original Hand Roll Bar	4.5	0
5		Hello Fish	4.5	4
6		Sushi Komasa	4.5	0
7		Sawa	5	0
8		Sushi Takeda	4.5	0
9		OOTORO Little Tokyo	4.5	0
10		Hama Sushi	4	0

SearchMy Bookings

Business search

Keyword *

Sushi

Distance

10

Category *

Default

Location *

☒ Auto-detect my location

SubmitClear

SearchMy Bookings

Business search

Keyword *

Sushi

Distance

10

Category *

Default

Arts and Entertainment

Health and Medical

Hotels and travel

Food

Professional Services

Location *

☒ Auto-detect my location

SubmitClear

#	Image	Business Name	Rating	Distance (miles)
1		J Sushi & Grill	4.5	2
2		Totoya Sushi & Tapas	4.5	10
3		Kyoto Poke & Ramen	4	1
4		California Teriyaki Grill	3.5	3
5		Poke Tiki - Rancho Santa Margarita	4.5	1
6		Poke Bowlz	4	0
7		Pokeworks	4.5	6
8		Poke Wave	4.5	4
9		Wow Bento	4.5	7
10		Maka Poke	4.5	4

←

Wow Bento

< Business details Map location >

Address
10 McLaren Ste A Irvine, CA 92618

Phone
(949) 699-0808


Status
Closed

Category
Japanese | Hawaiian | Poke

Price range
\$

Visit yelp for more
[Business link](#)
[Reserve Now](#)

Share on:



Phone
(949) 699-0808


Status
Closed

Category
Japanese | Hawaiian | Poke

Price range
\$

Visit yelp for more
[Business link](#)
[Reserve Now](#)

Share on:



Phone
(949) 699-0808


Status
Closed

Category
Japanese | Hawaiian | Poke

Price range
\$

Visit yelp for more
[Business link](#)
[Reserve Now](#)

Share on:



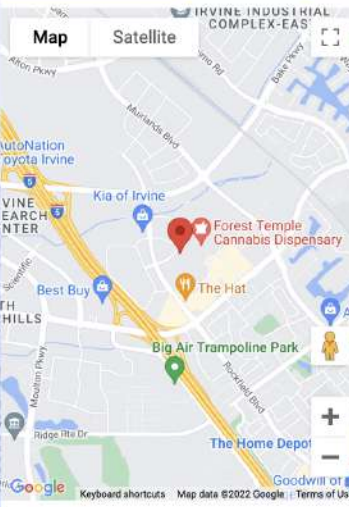
Submit Clear

←

Wow Bento

< Business details Map location >

Map Satellite



Google Keyboard shortcuts Map data ©2022 Google Terms of Use

☒ Auto-detect my location

Submit Clear

←

Wow Bento

< details Map location Reviews >

Joselyn H.
Rating: 5/5
i worked around the area and this was hands down the best poke place i've ever been to! i'm from orange and i even took a bowl to my husband and we still...
2022-07-21

Anita C.
Rating: 4/5
I admit - I didn't have high hopes for this place. But - it actually wasn't too bad! Price point won't break your bank and the amount of food they gave was...
2022-07-20

Vinny R.
Rating: 4/5
I switch between this location and the Oke Poke nearby. Overall they make a great poke bowl and I come here for the monster bowl (5 scoops of fish) when i'm...
2022-05-10

Reservation form

Wow Bento

Email

Date

Time
 : ⌚

Price range
 \$

Visit yelp for more
[Business link](#)

Share on:

Reservation form

Wow Bento

Email

Email is required

Date

Date is required

Time
 : ⌚

Price range
 \$

Visit yelp for more
[Business link](#)

Share on:

Business details **Map location**

Address
 10 McLaren Ste A Irvine, CA 92618

Phone
 (949) 699-0808

Status
 Closed

Category
 Japanese | Hawaiian | Poke

Price range
 \$

Visit yelp for more
[Business link](#)

Share on:

yelp-hw8-angular.us-west-2.elasticbeanstalk.com says
 Reservation cancelled!

Business details **Map location**

Address
 10 McLaren Ste A Irvine, CA 92618

Phone
 (949) 699-0808

Status
 Closed

Category
 Japanese | Hawaiian | Poke

Price range
 \$

Visit yelp for more
[Business link](#)

Share on:

yelp-hw8-angular.us-west-2.elasticbeanstalk.com says
 Reservation created!

Wow Bento

Email

Date

Time
 : ⌚

Price range
 \$

Visit yelp for more
[Business link](#)

Share on:

7. Assignment Submission

In your course Table of Assignments page (on GitHub Pages), update the Homework 8 link to refer to your deployed website. Also, provide an additional link to one of your cloud query entry points, below the homepage link. Your project must be hosted on GCP, AWS or Azure. Graders will verify that these links are indeed pointing to one of the listed cloud platforms. Refer to Homework 7 to deploy node.js applications on [GCP/AWS/AZURE](#).

Also, submit your source code files to DEN D2L as a ZIP archive. Include your frontend and backend source code, plus any additional files needed to build your app (e.g., yaml file). The timestamp of your last submission will be used to verify if you used any grace days. Make sure you don't upload the `node_modules` in the zip file and also make sure `package.json` is added in the zip file.

8. Notes

- You can use jQuery for Homework 8, but it is not needed.
- You can use **React** in lieu of **Angular**, but the look of the app must be the same, or penalties will be assessed. Also note that there will be no support on Piazza on React.
- You must use Bootstrap for Assignment 8. If Bootstrap is not used, there will be a point penalty assessed.
- The appearance of the webpage should be like the reference videos/screenshots as much as possible.

9. Additional References

- Conditional rendering in Angular: <https://angular.io/api/common/NgIf>
- Types of form handling in Angular:
<https://blog.angular-university.io/introduction-to-angular-2-forms-template-driven-vs-model-driven/>
- Add Authentication headers documentation can be found at:
<https://www.yelp.com/developers/documentation/v3/authentication>
- Use Authentication Headers: put the API key inside of the request header as “**Authorization: Bearer <YOUR API KEY>**” and make requests against the API.
- <https://chanchad.medium.com/how-to-setup-angular-10-and-express-js-application-for-developing-it-simultaneously-ca42e009c1ea>
- <https://youtube.com/playlist?list=PLHPSxQDpPvUmhqzw7K6R0zBXW6YJkV5zz>