WhiteHat Jr
Live online Coding for Kids

Topic	STAR RATING	
Class Description	**Students will also learn to add star rating functionality in the AR Menu Card scene. Students will also learn to update the dish rating and review it in the firestore database.**	
Class	**C173**	
Class time	**45 mins**	
Goal	● Learn to use Bootstrap to add star rating and review for a particular dish. ● Learn to update the database to add the dish rating and review.	
Resources Required	● Teacher Resources: ○ Visual Studio Code Editor ○ laptop with internet connectivity ○ smartphone ○ earphones with mic ○ notebook and pen ● Student Resources: ○ Visual Studio Code Editor ○ laptop with internet connectivity ○ smartphone ○ earphones with mic ○ notebook and pen	
Class structure	**Warm-Up** **Teacher-led Activity** **Student-led Activity** **Wrap-Up**	**5 mins** **15 mins** **20 mins** **5 mins**

- **WARM UP SESSION - 5 mins**

CONTEXT

- **Connecting A-Frame AR and firebase database.**

Teacher Starts Slideshow
Slide 1 to 4
Refer to speaker notes and follow the instructions on each slide.

Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? **Following are the WARM-UP session deliverables:** ● Greet the student. ● Revision of previous class activities. ● Quizzes.	**ESR**: Hi, thanks! Yes I am excited about it! Click on the slide show tab and present the slides

WARM-UP QUIZ
Click on In-Class Quiz

Continue WARM-UP Session
Slide 5 to 11

Following are the session deliverables:

- Appreciate the student.
- Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.

Class Steps	**Teacher Action**	**Student Action**
Step 1: **Warm-Up** **(5 mins)**	We have added a few functionalities to the augmented reality menu card app that a restaurant can use. Can you summarize what we learned in the previous classes while making this app that restaurants can use?	**ESR**: ● We used pattern markers to scan and show the details of the dishes available in that particular restaurant.

	Superb! In short, to eat in a restaurant we can say that: ● We go out to a restaurant. ● Sit at the table. ● Order is taken by the service. ● We eat the food. ● We can rate the food if we want to or write a review in their feedback manual (optional). ● Then the bill arrives (with the order summary) and we pay the bill. Well, this AR app for menu cards at restaurants can give a very great experience to the people coming at the restaurants to eat food. The physical menus cards can be replaced by these augmented reality cards. The dishes can actually be seen out on your table in augmented reality.	● The order was taken for a particular dish and was added to the database, based on the table number. ● The order summary for that particular order is shown for the final payment.

<table>
<tr><td></td><td>That’s really interesting!
In today’s class, we will be winding up this AR menu card app for the restaurants by adding rating and review functionality for a particular dish.
Are you excited?</td><td><b>ESR:</b> Yes.</td></tr>
<tr><td></td><td>Let's get started then.</td><td></td></tr>
<tr><td colspan="3"><b>Teacher Ends Slideshow</b></td></tr>
<tr><td colspan="3"><b>TEACHER-LED ACTIVITY - 15 mins</b></td></tr>
<tr><td colspan="3"><b>Teacher Initiates Screen Share</b></td></tr>
<tr><td colspan="3"><b><u>CHALLENGE</u></b>
● <b>Add star rating using Bootstrap.</b>
● <b>Write into Firebase Database through augmented reality.</b></td></tr>
<tr><td><b>Step 2:
Teacher-led Activity
(15 mins)</b></td><td><i>&lt;The teacher clones the code from Teacher Activity 1.&gt;</i>
<b><i><u>[Teacher Activity 1]</u></i></b>
<i><b>Note</b>: The 3D Food Model might or might not vary(only in the look and feel) in the output image/video reference given in the document to follow, but this would not affect any of the steps mentioned to complete the functionalities of the application. This</i></td><td></td></tr>
</table>

is because the models are taken from the sketchfab.com which keeps updating(adding new models or deleting the older ones) the models on the website. The models will be provided as per the availability on the website, and to make sure the size of the model does not affect the functionality of the app, the models will be updated based on the availability. Teachers are requested to explore models on their own if required.

Star rating can be added using the Bootstrap library.

There are a few options available here that we can use in our AR app.

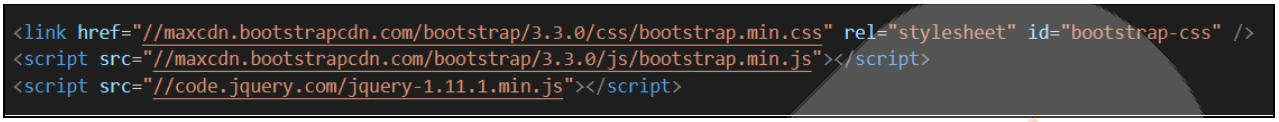
We can use any example from this one.

Note: *The JavaScript (stars.js), HTML, CSS files required to add the star rating on the page are already added/updated in the boilerplate code.*

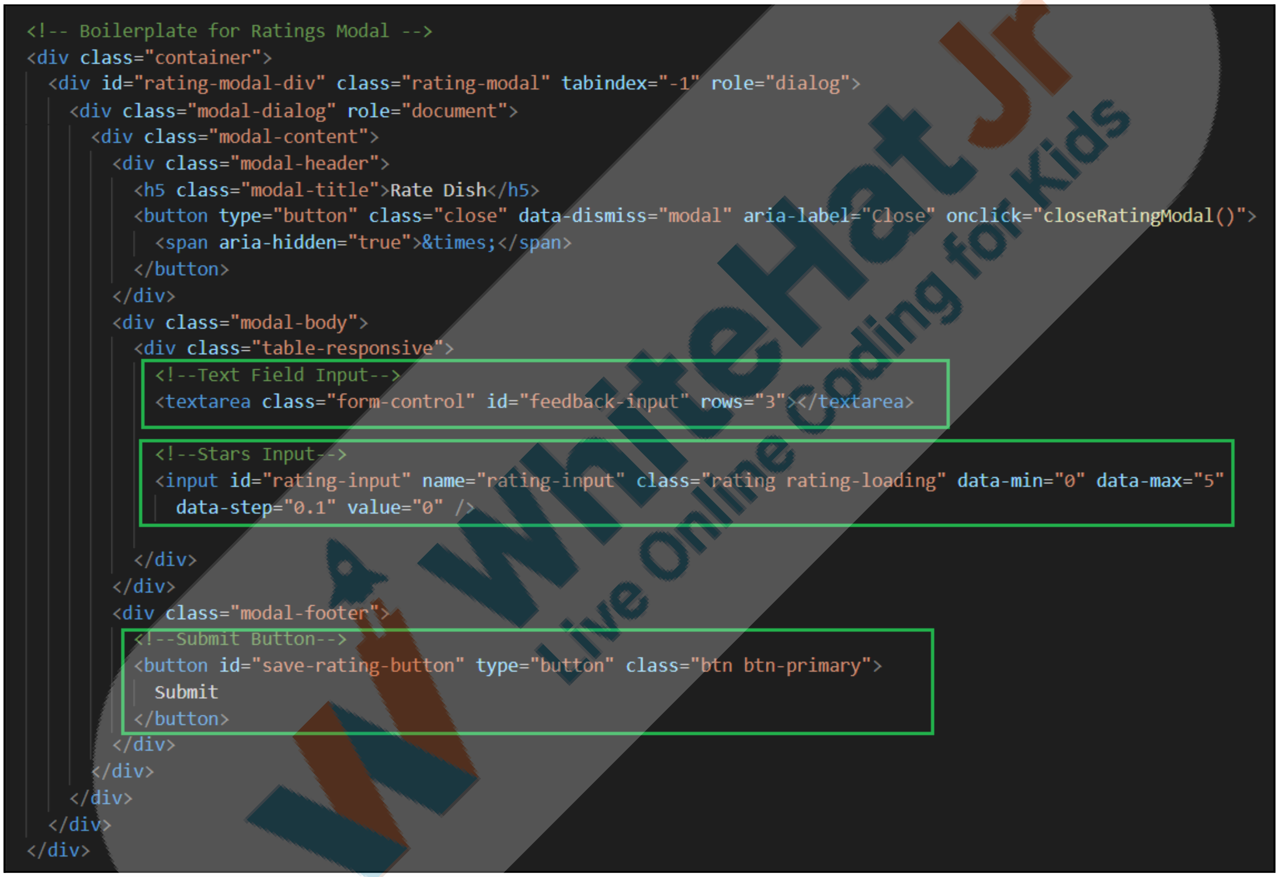
<The teacher opens the index.html and explains the HTML code to the student.>

In the HTML, we are using the Modal Box to show the pop-up to add a text field where you can write a review, a rating field to select stars and the submit button to save the ratings and reviews in the database on click.

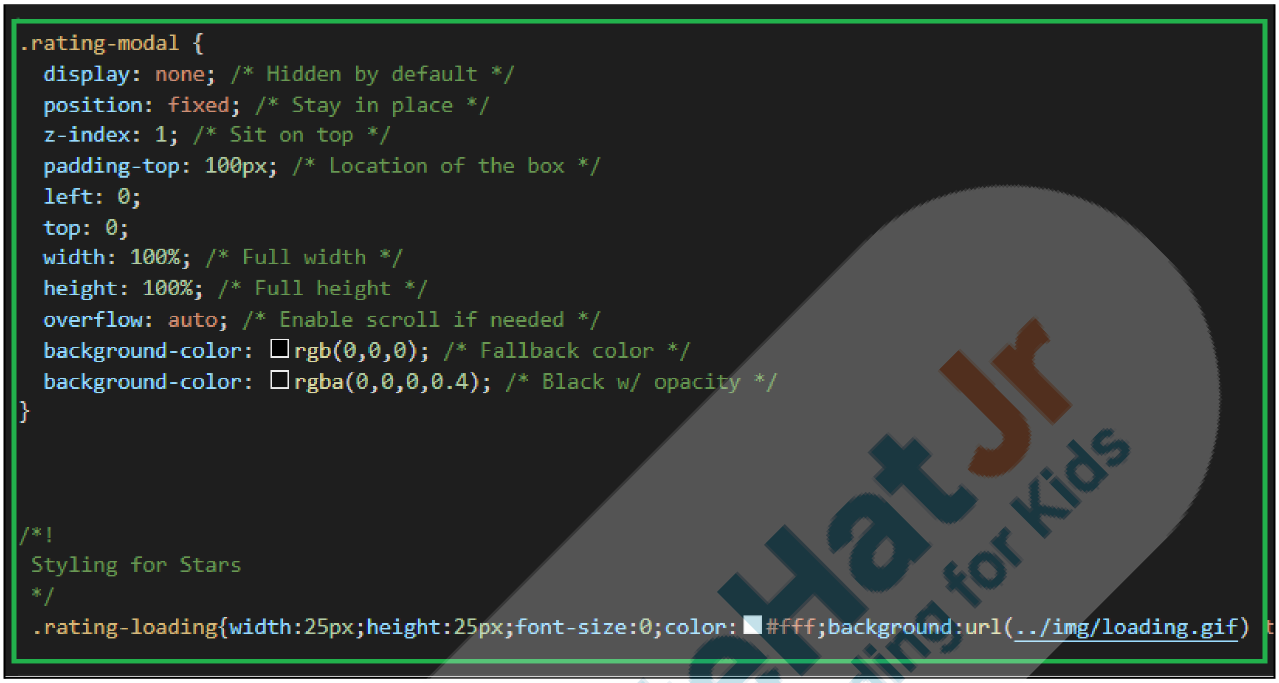
	***Note**: The initial display property is set to none for the rating-modal class in style.css.*	



```
<link href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/css/bootstrap.min.css" rel="stylesheet" id="bootstrap-css" />
<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/js/bootstrap.min.js"></script>
<script src="//code.jquery.com/jquery-1.11.1.min.js"></script>
```



```
<!-- Boilerplate for Ratings Modal -->
<div class="container">
  <div id="rating-modal-div" class="rating-modal" tabindex="-1" role="dialog">
    <div class="modal-dialog" role="document">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title">Rate Dish</h5>
          <button type="button" class="close" data-dismiss="modal" aria-label="Close" onclick="closeRatingModal()">
            <span aria-hidden="true">&times;</span>
          </button>
        </div>
        <div class="modal-body">
          <div class="table-responsive">
            <!--Text Field Input-->
            <textarea class="form-control" id="feedback-input" rows="3"></textarea>

            <!--Stars Input-->
            <input id="rating-input" name="rating-input" class="rating rating-loading" data-min="0" data-max="5"
              data-step="0.1" value="0" />

          </div>
        </div>
        <div class="modal-footer">
          <!--Submit Button-->
          <button id="save-rating-button" type="button" class="btn btn-primary">
            Submit
          </button>
        </div>
      </div>
    </div>
  </div>
</div>
```

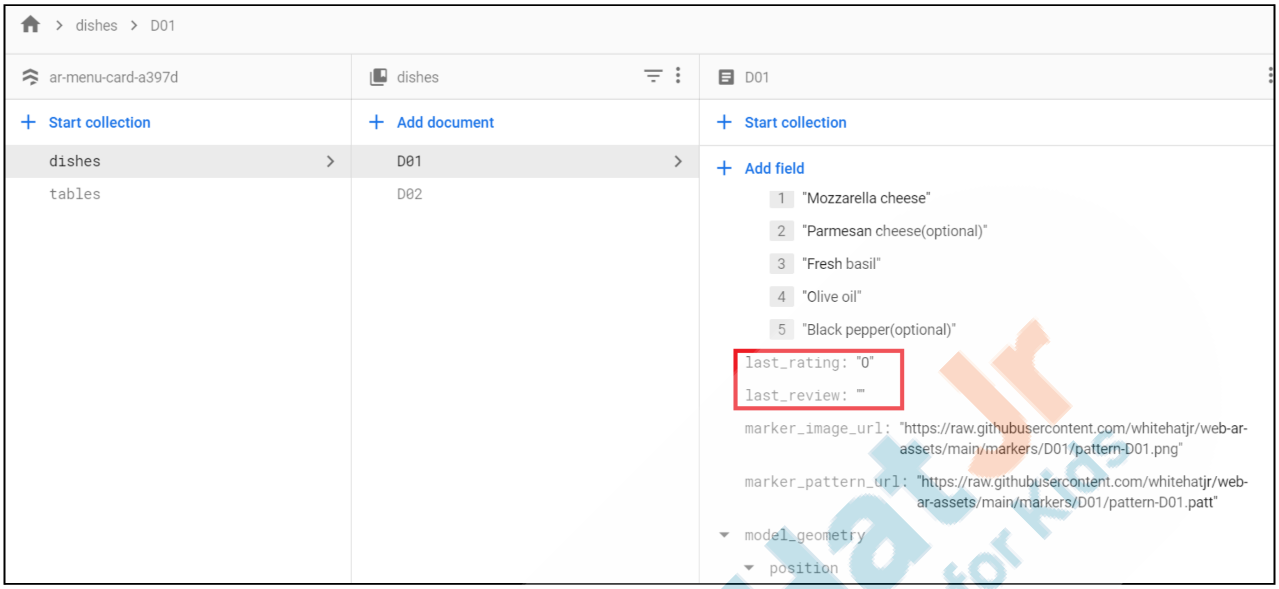


```
.rating-modal {
  display: none; /* Hidden by default */
  position: fixed; /* Stay in place */
  z-index: 1; /* Sit on top */
  padding-top: 100px; /* Location of the box */
  left: 0;
  top: 0;
  width: 100%; /* Full width */
  height: 100%; /* Full height */
  overflow: auto; /* Enable scroll if needed */
  background-color: rgb(0,0,0); /* Fallback color */
  background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
}



/*!
 Styling for Stars
 */
 .rating-loading{width:25px;height:25px;font-size:0;color:#fff;background:url(../img/loading.gif) t
```

Now we can use the id of these 3 elements to update the details in the database.

Let’s add the fields required for this in the database first.

Since rating and review will not be for complete orders in one go, we will have a separate rating and review field for each dish.



dishes > D01

ar-menu-card-a397d | dishes | D01

Start collection | Add document | Start collection

dishes | D01 | Add field

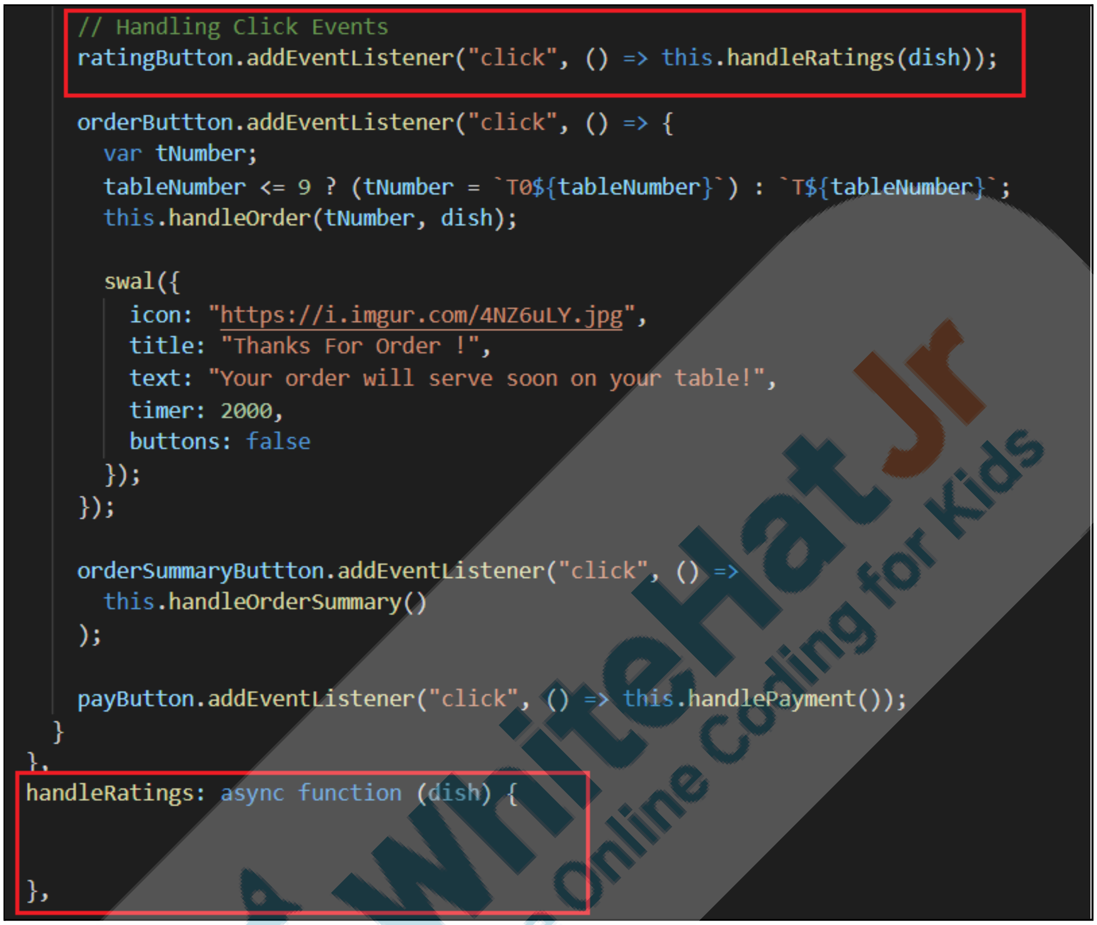
tables | D02

1 "Mozzarella cheese"
2 "Parmesan cheese(optional)"
3 "Fresh basil"
4 "Olive oil"
5 "Black pepper(optional)"

last_rating: "0"
last_review: ""
marker_image_url: "https://raw.githubusercontent.com/whitehatjr/web-ar-assets/main/markers/D01/pattern-D01.png"
marker_pattern_url: "https://raw.githubusercontent.com/whitehatjr/web-ar-assets/main/markers/D01/pattern-D01.patt"
model_geometry
position

Now we will write a function, **handleRatings()** to the values in the database.

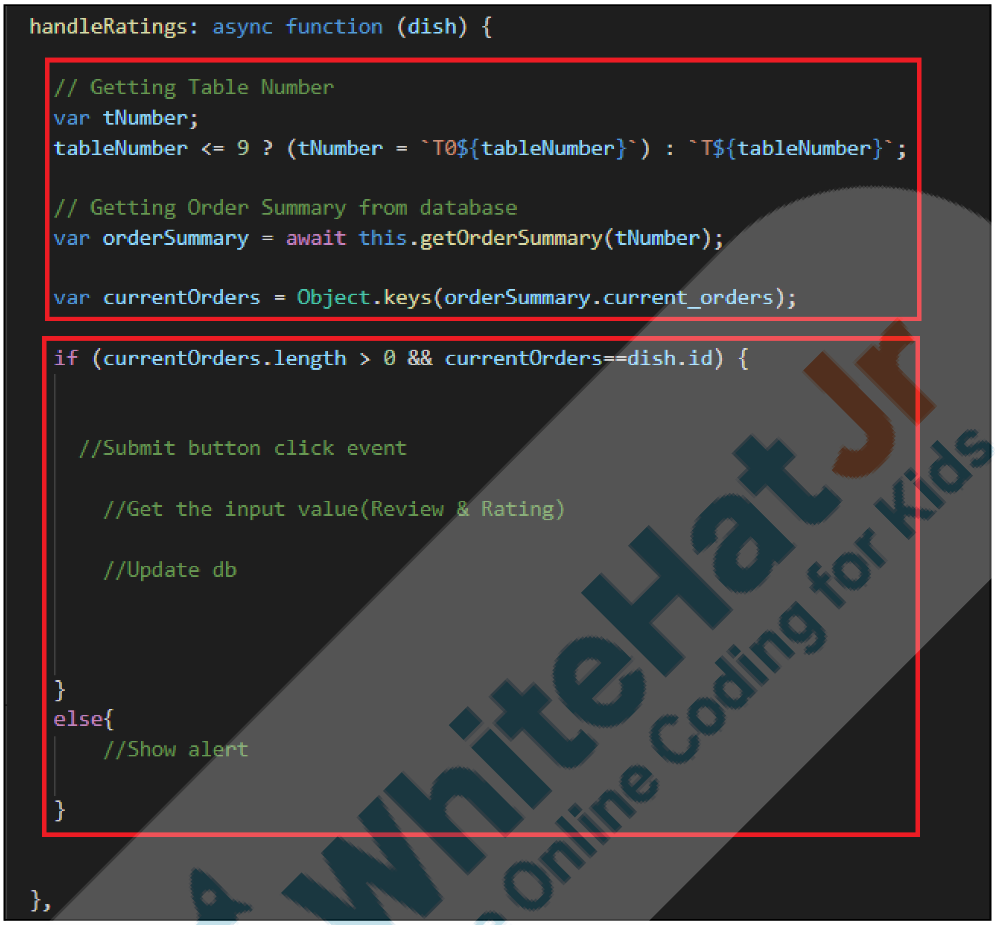
We already have a button “RATE DISH”; for this button, we will update the click event listener to call the **handleRatings()** function.



```
    // Handling Click Events
    ratingButton.addEventListener("click", () => this.handleRatings(dish));

    orderButtton.addEventListener("click", () => {
      var tNumber;
      tableNumber <= 9 ? (tNumber = `T0${tableNumber}`) : `T${tableNumber}`;
      this.handleOrder(tNumber, dish);

      swal({
        icon: "https://i.imgur.com/4NZ6uLY.jpg",
        title: "Thanks For Order !",
        text: "Your order will serve soon on your table!",
        timer: 2000,
        buttons: false
      });
    });

    orderSummaryButtton.addEventListener("click", () =>
      this.handleOrderSummary()
    );

    payButton.addEventListener("click", () => this.handlePayment());
  }
},
handleRatings: async function (dish) {


},
```

	First we need to check whether the dishes exist in the **current_orders** (field in the database) for that particular table. Can you tell me why? Yes, exactly!	**ESR**: Because once the pay now button is provided, the current_orders list for that table is gone.

	The dish can be rated when the customer is sitting at the table before making a payment, if he/she wants to! For this we will: ● Get the table number value. ● Use table number to get the order summary. ● Get the current_orders list. ● Use the if condition: ○ to check if the list is empty using the length property, and ○ if the dish scanned through the marker is present in the current orders list. ● Else show the alert that the dish is not present.	

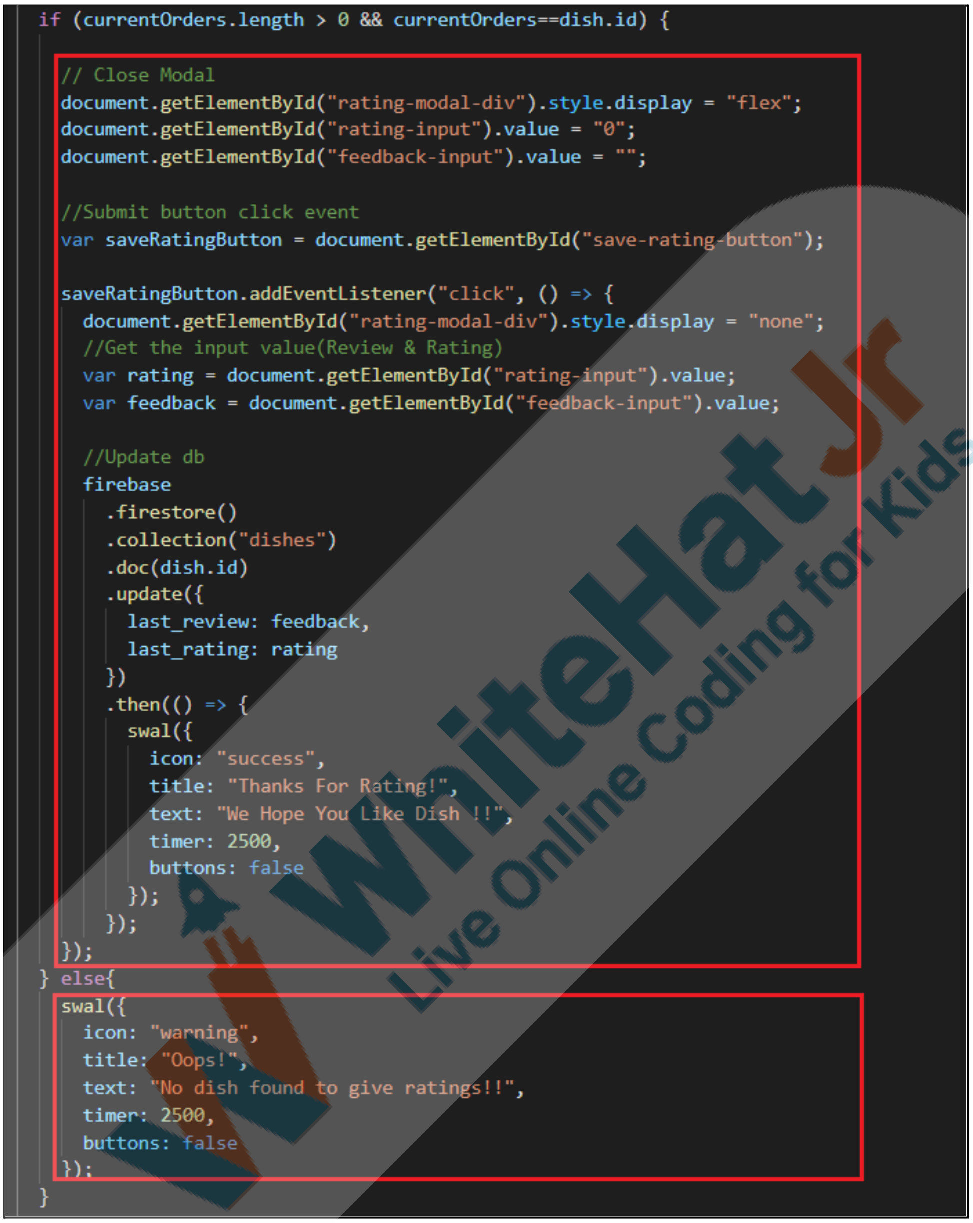


```
handleRatings: async function (dish) {
  // Getting Table Number
  var tNumber;
  tableNumber <= 9 ? (tNumber = `T0${tableNumber}`) : `T${tableNumber}`;

  // Getting Order Summary from database
  var orderSummary = await this.getOrderSummary(tNumber);

  var currentOrders = Object.keys(orderSummary.current_orders);
  if (currentOrders.length > 0 && currentOrders==dish.id) {

    //Submit button click event

      //Get the input value(Review & Rating)

      //Update db



  }
  else{
      //Show alert

  }

},
```

If the list of **current_orders** length is greater than 0 then:

- We can set the display visibility of the modal to flex.
- Get the **rating-input** (id of the element) element using **getElementById()** and initialize the element value with an empty string.
- Get the **feedback-input** (id of the element) element using

	getElementById() and initialize the element's value with 0. - Get the **save-rating-button** (id of the element) using **getElementById()** and add a click event listener to it. - Update the **rating-input** and **feedback-input** in the database. Else show the alert warning using swal() function.	



```
if (currentOrders.length > 0 && currentOrders==dish.id) {

  // Close Modal
  document.getElementById("rating-modal-div").style.display = "flex";
  document.getElementById("rating-input").value = "0";
  document.getElementById("feedback-input").value = "";

  //Submit button click event
  var saveRatingButton = document.getElementById("save-rating-button");

  saveRatingButton.addEventListener("click", () => {
    document.getElementById("rating-modal-div").style.display = "none";
    //Get the input value(Review & Rating)
    var rating = document.getElementById("rating-input").value;
    var feedback = document.getElementById("feedback-input").value;

    //Update db
    firebase
      .firestore()
      .collection("dishes")
      .doc(dish.id)
      .update({
        last_review: feedback,
        last_rating: rating
      })
      .then(() => {
        swal({
          icon: "success",
          title: "Thanks For Rating!",
          text: "We Hope You Like Dish !!",
          timer: 2500,
          buttons: false
        });
      });
  });
} else{
  swal({
    icon: "warning",
    title: "Oops!",
    text: "No dish found to give ratings!!",
    timer: 2500,
    buttons: false
  });
}
```

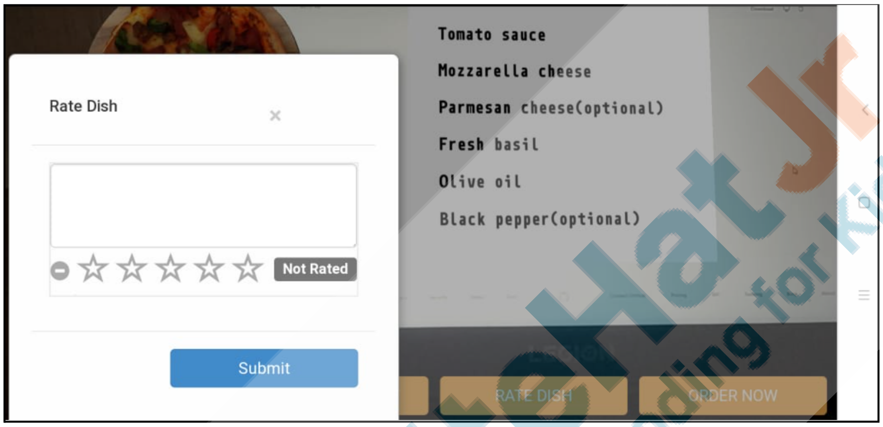
Now we can test the output using ngrok:

- Scan the dish marker.
- Click on “RATE DISH”.
- Write a review.
- Rate.

<table>
<tr>
<td></td>
<td>

- Click on submit.
- Verify the update in the database.
- Test the output for the dish which is not added in the current_order.

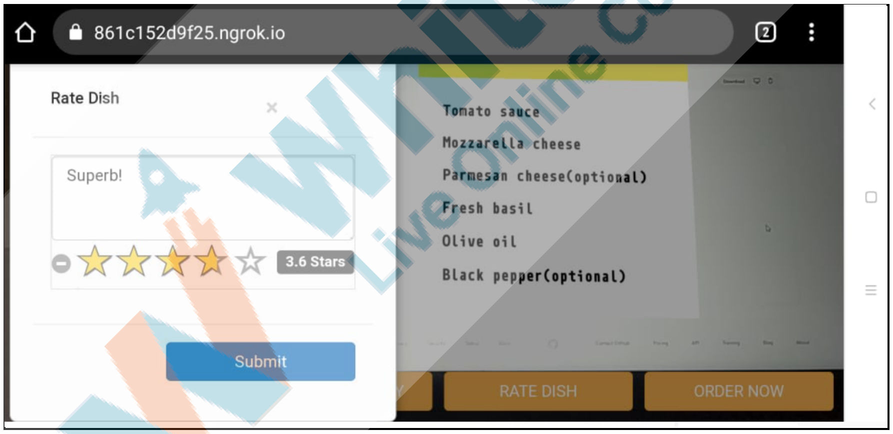
</td>
<td></td>
</tr>
<tr>
<td colspan="3">



Rate Dish

Not Rated

Tomato sauce
Mozzarella cheese
Parmesan cheese(optional)
Fresh basil
Olive oil
Black pepper(optional)



861c152d9f25.ngrok.io

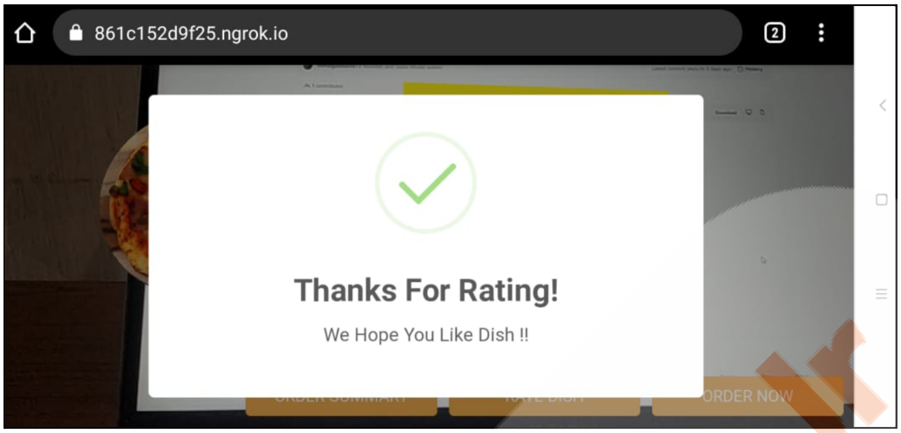
Rate Dish

Superb!

3.6 Stars

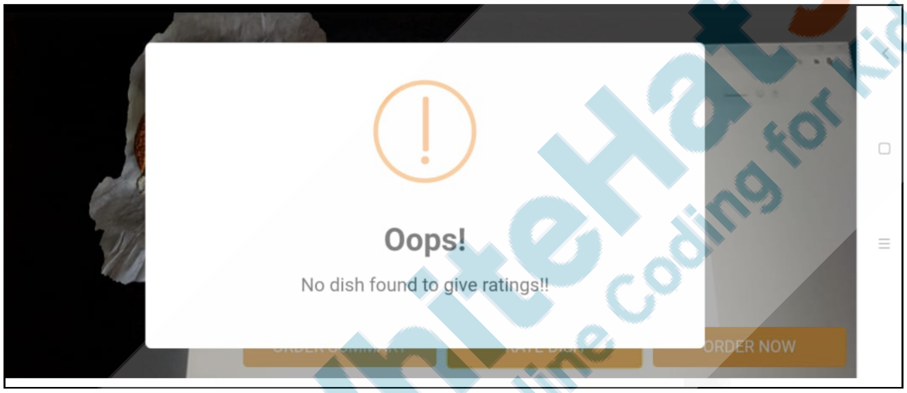
Tomato sauce
Mozzarella cheese
Parmesan cheese(optional)
Fresh basil
Olive oil
Black pepper(optional)

</td>
</tr>
</table>



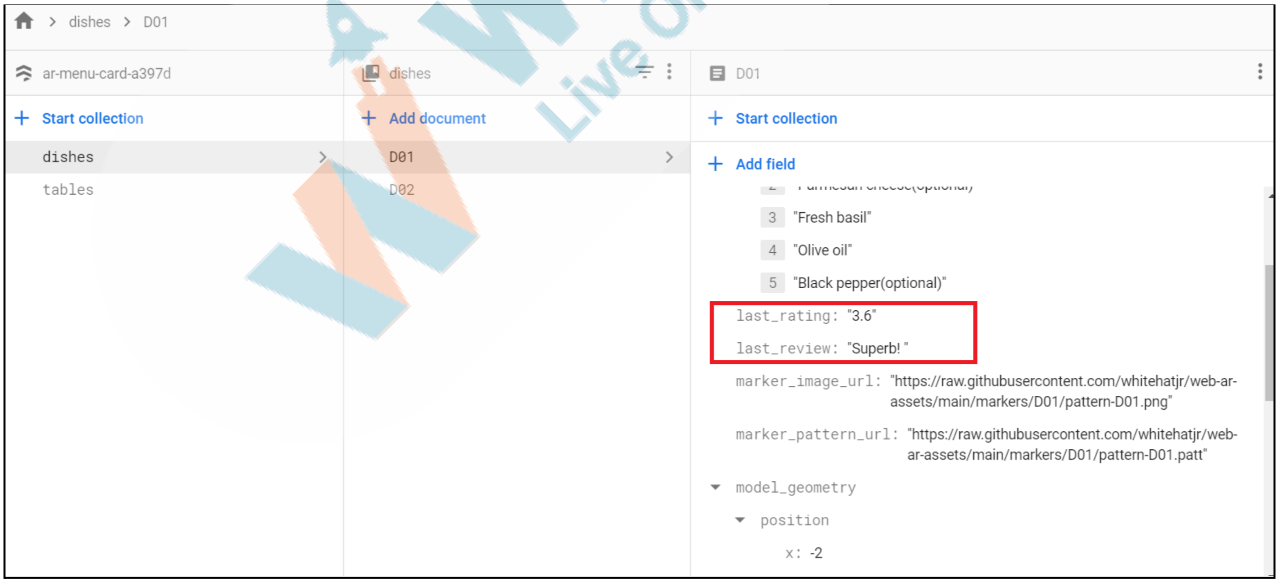
Thanks For Rating!

We Hope You Like Dish !!



Oops!

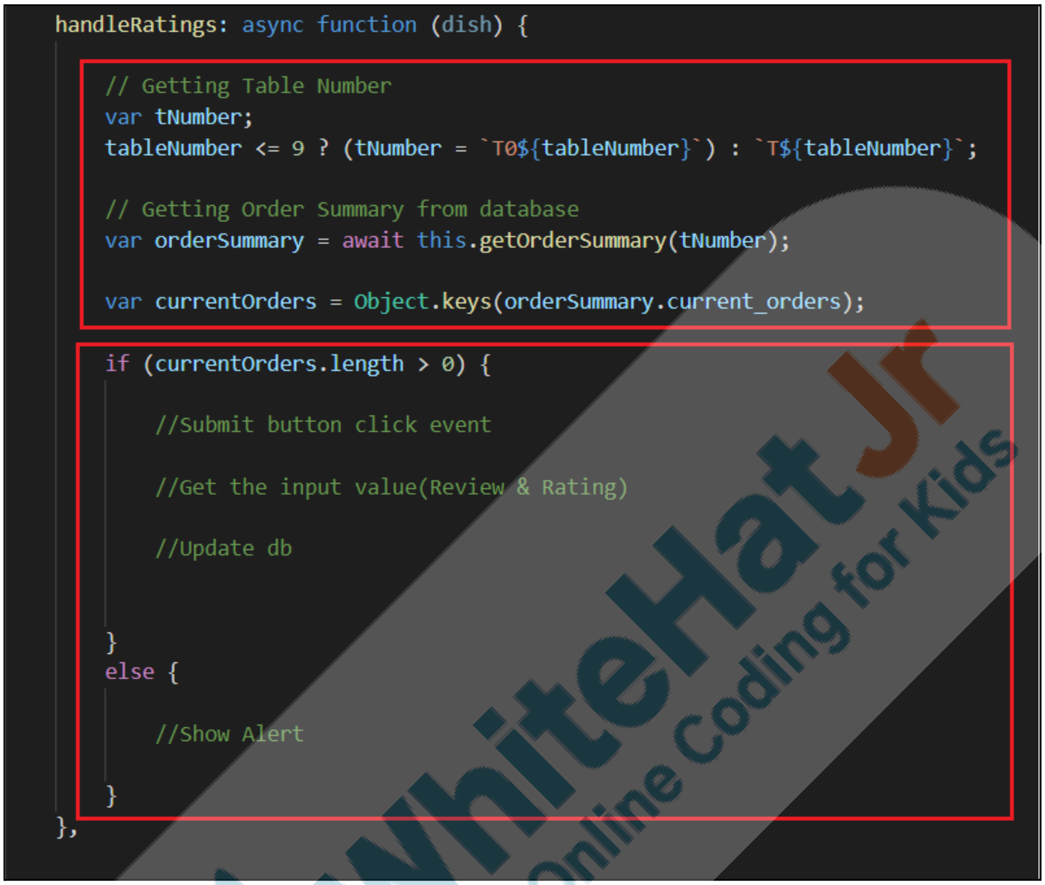
No dish found to give ratings!!



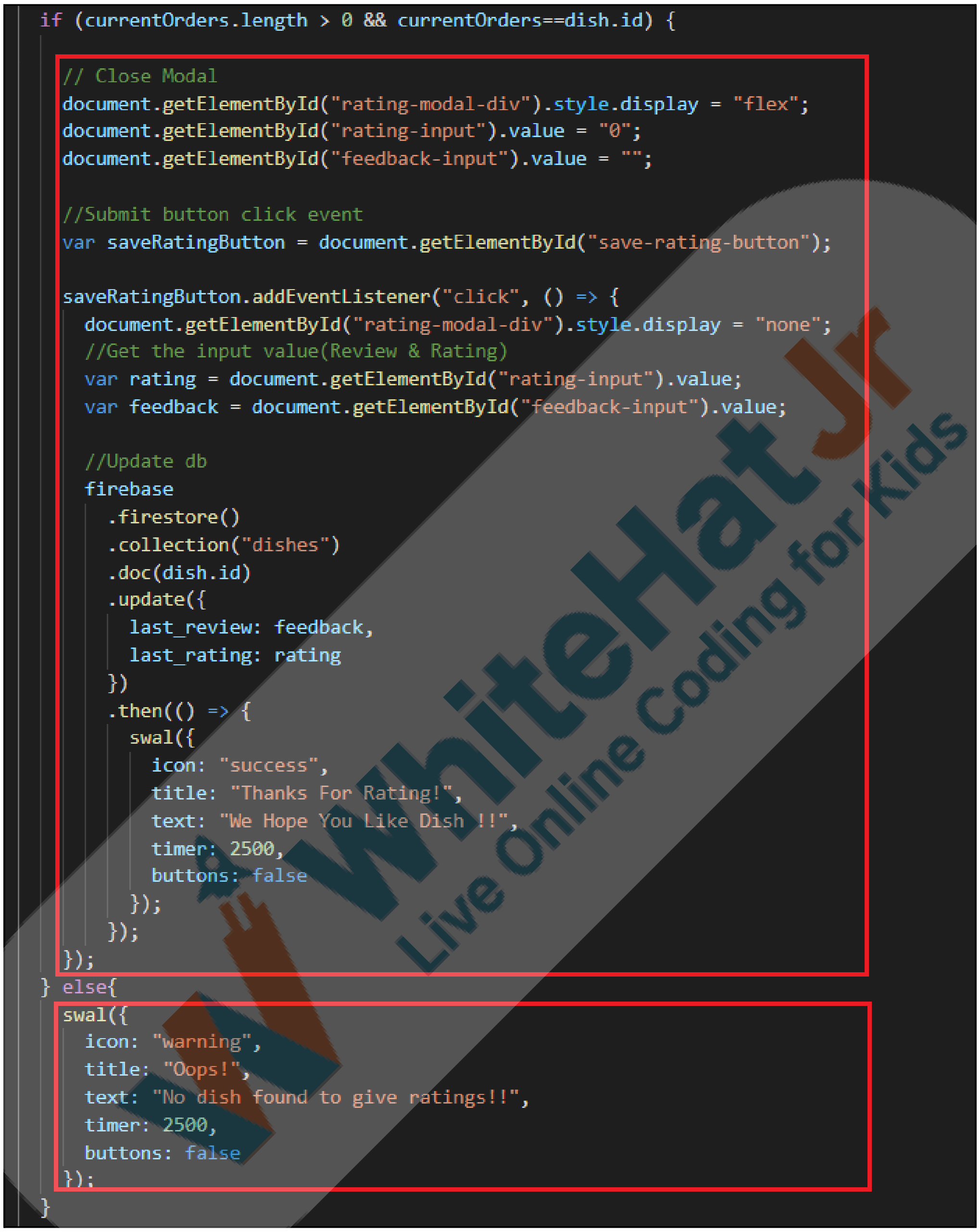
```
2 "Parmesan cheese(optional)"
3 "Fresh basil"
4 "Olive oil"
5 "Black pepper(optional)"
last_rating: "3.6"
last_review: "Superb! "
marker_image_url: "https://raw.githubusercontent.com/whitehatjr/web-ar-assets/main/markers/D01/pattern-D01.png"
marker_pattern_url: "https://raw.githubusercontent.com/whitehatjr/web-ar-assets/main/markers/D01/pattern-D01.patt"
▾ model_geometry
  ▾ position
      x: -2
```

	That's really nice! We added a star rating in Augmented Reality! We can do so much in the web AR. Now you will try adding the submit button functionality to add star rating and update the database. You will also add the functionality to clear the order details of that particular table on a button click. Are you excited?	**ESR**: Yes!
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	
Teacher Starts Slideshow **Slide 12 to 14** Refer to speaker notes and follow the instructions on each slide.		
We have one more class challenge for you. Can you solve it? Let's try. I will guide you through it.		
Teacher Ends Slideshow		
STUDENT-LED ACTIVITY - 20 mins		

<table>
<tr><td colspan="3"><ul><li><b>Ask the student to press the ESC key to come back to the panel.</b></li><li><b>Guide the student to start screen share.</b></li><li><b>Teacher gets into fullscreen.</b></li></ul></td></tr>
<tr><td colspan="3"><b><u>ACTIVITY</u></b><ul><li><b>Write a function to add star rating and review functionality to rate/review a particular dish and update the values in the db.</b></li><li><b>Show the review and ratings in the AR scene UI.</b></li></ul></td></tr>
<tr><td><b>Step 3:
Student-led Activity
(20 mins)</b></td><td><i>The teacher guides the student to clone the code from Student Activity 1.</i>
<i>[Student Activity 1]</i>
<i><b>Note</b>: The student will repeat some of the teacher activity and will continue to add new functionality after teacher activity.</i></td><td></td></tr>
<tr><td></td><td><i>Guide the student to write the <b>handleRatings()</b> function.</i></td><td></td></tr>
</table>



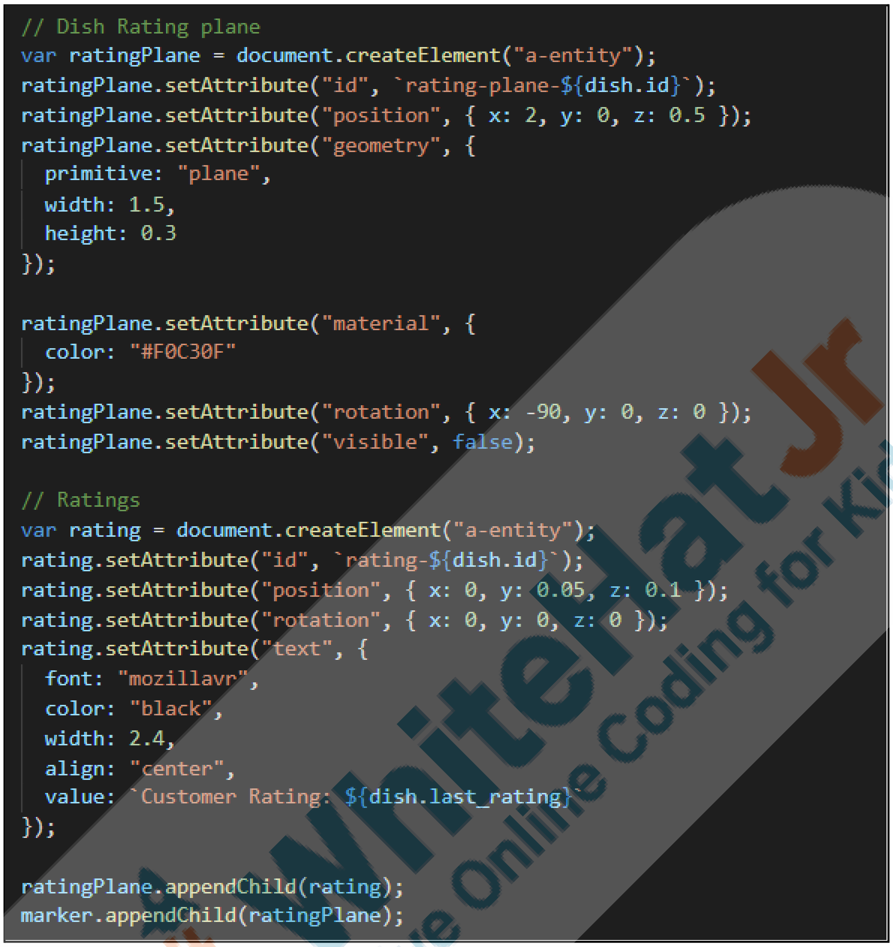
```
handleRatings: async function (dish) {
    // Getting Table Number
    var tNumber;
    tableNumber <= 9 ? (tNumber = `T0${tableNumber}`) : `T${tableNumber}`;

    // Getting Order Summary from database
    var orderSummary = await this.getOrderSummary(tNumber);

    var currentOrders = Object.keys(orderSummary.current_orders);

    if (currentOrders.length > 0) {

        //Submit button click event

        //Get the input value(Review & Rating)

        //Update db


    }
    else {

        //Show Alert

    }
},
```



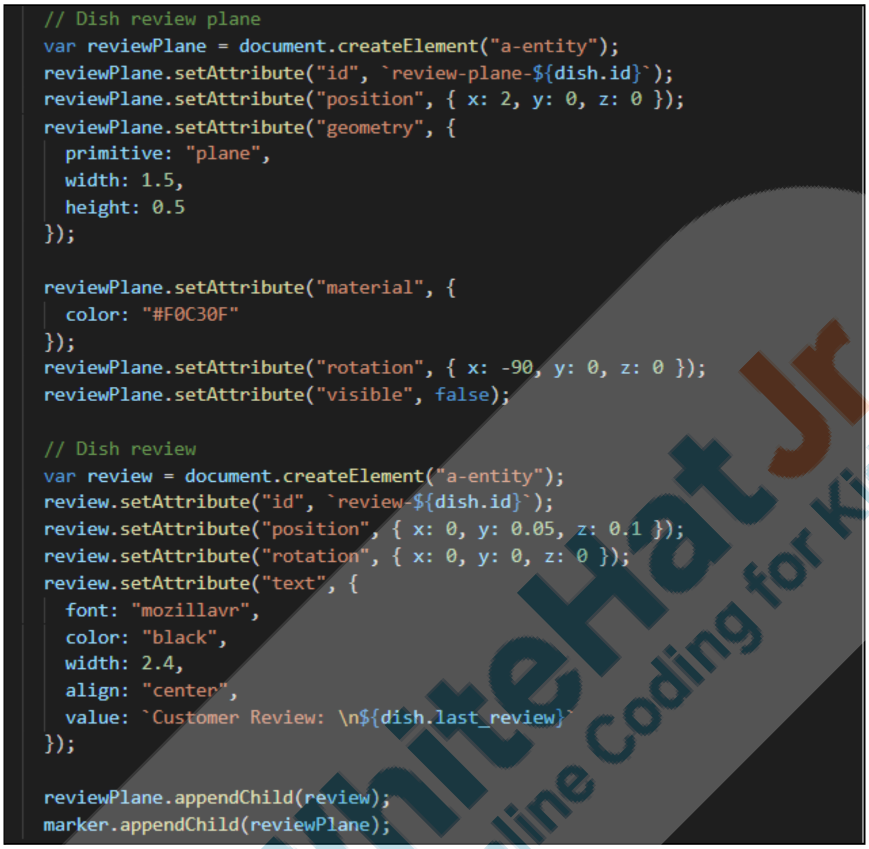
```
if (currentOrders.length > 0 && currentOrders==dish.id) {
  // Close Modal
  document.getElementById("rating-modal-div").style.display = "flex";
  document.getElementById("rating-input").value = "0";
  document.getElementById("feedback-input").value = "";

  //Submit button click event
  var saveRatingButton = document.getElementById("save-rating-button");

  saveRatingButton.addEventListener("click", () => {
    document.getElementById("rating-modal-div").style.display = "none";
    //Get the input value(Review & Rating)
    var rating = document.getElementById("rating-input").value;
    var feedback = document.getElementById("feedback-input").value;

    //Update db
    firebase
      .firestore()
      .collection("dishes")
      .doc(dish.id)
      .update({
        last_review: feedback,
        last_rating: rating
      })
      .then(() => {
        swal({
          icon: "success",
          title: "Thanks For Rating!",
          text: "We Hope You Like Dish !!",
          timer: 2500,
          buttons: false
        });
      });
  });
} else{
  swal({
    icon: "warning",
    title: "Oops!",
    text: "No dish found to give ratings!!",
    timer: 2500,
    buttons: false
  });
}
```

Now we will add the plane entity to show the rating and review as text in the AR scene.

	We will use the value of the **last_rating** and **last_review** from the database. To add the **plane** entity: - Create an 'a-entity' element using **document.createElement()**. - Set the **id**(from the db), **position**, **rotation**, **width**, **height** attribute using **.setAttribute()**. - Append the plane entity to the marker using **.appendChild()**. To add the **text**(review and ratings) entity: - Create an 'a-entity' element using **document.createElement()**. - Set the **id**(from the db), **position**, **rotation**, **width**, **text** attribute using **.setAttribute()**. - Append the text entity to the price plane using **.appendChild()**. *Guide the student to add the plane entity and the text entity in the AR scene.*	

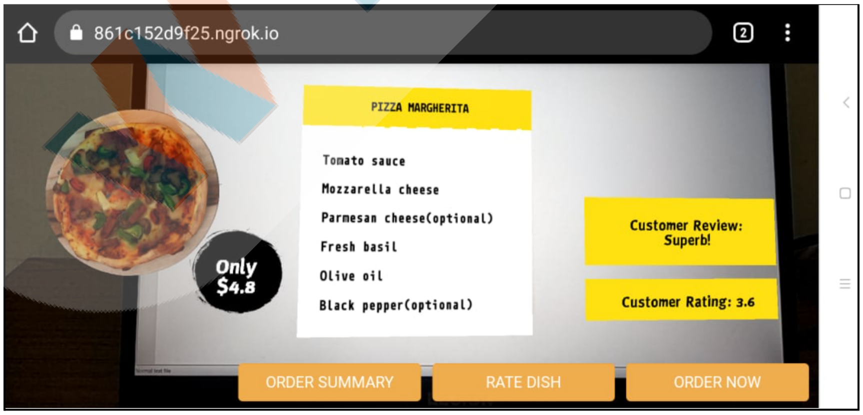


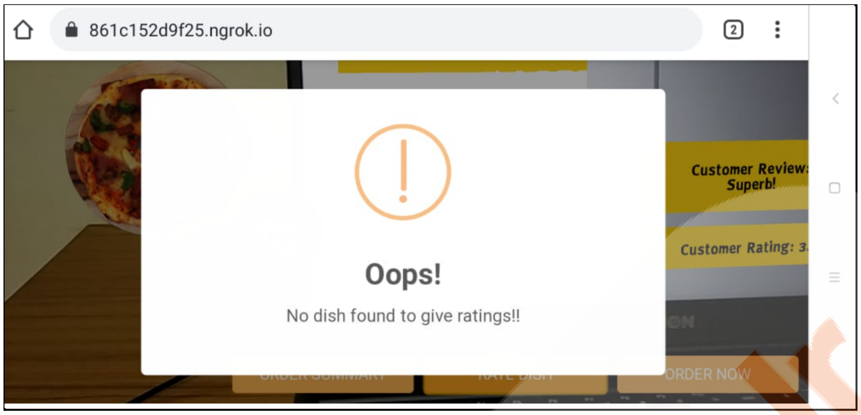
```
// Dish Rating plane
var ratingPlane = document.createElement("a-entity");
ratingPlane.setAttribute("id", `rating-plane-${dish.id}`);
ratingPlane.setAttribute("position", { x: 2, y: 0, z: 0.5 });
ratingPlane.setAttribute("geometry", {
  primitive: "plane",
  width: 1.5,
  height: 0.3
});

ratingPlane.setAttribute("material", {
  color: "#F0C30F"
});
ratingPlane.setAttribute("rotation", { x: -90, y: 0, z: 0 });
ratingPlane.setAttribute("visible", false);

// Ratings
var rating = document.createElement("a-entity");
rating.setAttribute("id", `rating-${dish.id}`);
rating.setAttribute("position", { x: 0, y: 0.05, z: 0.1 });
rating.setAttribute("rotation", { x: 0, y: 0, z: 0 });
rating.setAttribute("text", {
  font: "mozillavr",
  color: "black",
  width: 2.4,
  align: "center",
  value: `Customer Rating: ${dish.last_rating}`
});

ratingPlane.appendChild(rating);
marker.appendChild(ratingPlane);
```



```
// Dish review plane
var reviewPlane = document.createElement("a-entity");
reviewPlane.setAttribute("id", `review-plane-${dish.id}`);
reviewPlane.setAttribute("position", { x: 2, y: 0, z: 0 });
reviewPlane.setAttribute("geometry", {
  primitive: "plane",
  width: 1.5,
  height: 0.5
});

reviewPlane.setAttribute("material", {
  color: "#F0C30F"
});
reviewPlane.setAttribute("rotation", { x: -90, y: 0, z: 0 });
reviewPlane.setAttribute("visible", false);

// Dish review
var review = document.createElement("a-entity");
review.setAttribute("id", `review-${dish.id}`);
review.setAttribute("position", { x: 0, y: 0.05, z: 0.1 });
review.setAttribute("rotation", { x: 0, y: 0, z: 0 });
review.setAttribute("text", {
  font: "mozillavr",
  color: "black",
  width: 2.4,
  align: "center",
  value: `Customer Review: \n${dish.last_review}`
});

reviewPlane.appendChild(review);
marker.appendChild(reviewPlane);
```

Guide the student to test the output using ngrok.





Oops!

No dish found to give ratings!!

Teacher Guides Student to Stop Screen Share

WRAP UP SESSION - 5 mins

Teacher Starts Slideshow
Slide 15 to 18

Activity details
Following are the WRAP-UP session deliverables:

- Appreciate the student.
- Revise the current class activities.
- Discuss the quizzes.

WRAP-UP QUIZ
Click on In-Class Quiz

Continue WRAP-UP Session
Slide 19 to 24

Activity Details

Following are the session deliverables:

- Explain the facts and trivia
- Next class challenge
- Project for the day

● Additional Activity (Optional)		
FEEDBACK ● **Compliment the student for her/his effort in the class.** ● **Encourage the student to think and come up with their own solutions.**		
You get a “hats-off”. Alright. See you in the next class.		*Make sure you have given at least 2 Hats Off during the class for:* Creatively Solved Activities +10 Great Question +10 Strong Concentration +10
PROJECT OVERVIEW DISCUSSION Refer the document below in Activity Links Sections		
Teacher Clicks ✖ End Class		
Additional Activities	*Encourage the student to write reflection notes in their reflection journal using markdown.* Use these as guiding questions: ● What happened today? ○ Describe what happened. ○ The code I wrote. ● How did I feel after the class? ● What have I learned about programming and developing games?	*The student uses the markdown editor to write their reflections in a reflection journal.*

	- What aspects of the class helped me? What did I find difficult?	

Activity	Activity Name	Links
Teacher Activity 1	Boilerplate Code	https://github.com/whitehatjr/PRO-C173-Boilerplate
Teacher Activity 2	Bootstrap Rating Examples	https://bootsnipp.com/tags/rating
Teacher Activity 3	Teacher Reference Code	https://github.com/whitehatjr/PRO-C173
Student Activity 1	Boilerplate Code	https://github.com/whitehatjr/PRO-C173-Boilerplate
Teacher Reference 1	Ngrok Updates	https://docs.google.com/document/d/1dlMry188llEJl6rHEc3AkBashQSOwGQ40HQft29S8vQ/edit?usp=sharing
Teacher Reference 2	Project Document	https://s3-whjr-curriculum-uploads.whjr.online/5134adfd-7332-4a76-9b1b-b802beb91824.pdf
Teacher Reference 3	Project Solution	https://github.com/whitehatjr/PRO-C173-AR
Teacher Reference 4	Visual-Aid	https://s3-whjr-curriculum-uploads.whjr.online/06461881-5347-4dcd-a2f4-5f2ebcebe030.html
Teacher Reference 5	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/02bd37a3-5148-4b0b-a370-0c8a49fb1a0c.pdf