# Create Crowe’s Pizza Joint Public Website

## Purpose:

* + Become comfortable with consuming the CRM Web API through JavaScript

## Description:

* + In this lab you’ll be creating the customer facing website for Crowe’s Pizza Joint.  Customers will use this site to place see the menu and place their orders.

## Key Technologies:

* + Microsoft Dynamics CRM Web API, Angular 2.0, TypeScript

|  |  |  |
| --- | --- | --- |
| Initial Project Setup | | |
|  |  | Download the starter project from <https://github.com/ericcahoon/Blitz2016> |
| Installing the Angular 2.0 Resources | | |
|  |  | From the VS menu click ‘Package Manager Console’, which can be found in Tools > NuGet Package Manager |
|  |  | In the ‘Package Manager Console’ console run  cd Blitz2016CrmCapstone |
|  |  | In the ‘Package Manager Console’ console run  *npm install* |
| Debug the Project | | |
|  |  | Press F5 to begin debugging your project. You should see something like the image below open in your default browser |
|  |  | Click the ‘Next Button’ in the bottom left until you’ve completed configuring your order. Then click the ‘’ button. Your screen should now a have a container with a label of ‘My Pizza(s)’ |
|  |  | Click the ‘Place Order’ button to open the ‘Contact Info’ dialog. In this dialog you’re asked to provide your contact info before placing your order. However, this dialog is not wired up; thus, you’ll spend the rest of this lab wiring this dialog up to   * Place your order with ‘Crowe’s Pizza Joint’ CRM API * Display a confirmation dialog to the user that shows them their order number |
| Wiring up The Place-Order Dialog to Persist Data to CRM | | |
|  |  | In Visual Studio, open the ‘place-order-ts’ file in the following directory  TypeScript > PizzaBuilder |
|  |  | Add a new method named ‘submitOrder’ and an alert method to it. It might look something like  ***private*** submitOrder **()** **{**  alert**(“**You clicked the submit order button“**);**  **}** |
|  |  | Open the ‘place-order-dialog.htm’ file in the following directory  app > Templates > PizzaBuilder |
|  |  | Locate the OK button. It should look like  <button [disabled]=**"!\_firstName || !\_lastName || !\_phone"**>**OK**</button> |
|  |  | Add a click attribute to that line for your submitOrder method. *Note!!!* Angular 2.0 uses the following format to bind a method to a click event  (click)=”submitOrder()” |
|  |  | Press the F5 button to begin debugging your application again. |
|  |  | Walk through the process of configuring a pizza and placing your order. After clicking the ‘Place Order’ button and providing your contact information, ensure you see the alert message you provide. |
|  |  |  |
| Persisting Data to CRM through the Web API | | |
|  |  | Close your browser and go back to the ‘place-order.ts’ file and find the submitOrder you added. |
|  |  | Add an object variable to represent the order that will be placed. It might look something like  let order = {  }; |
|  |  | To persist the order to CRM, we’re going to make use of the Create method provided by the Crm.WebApi, which is an angular service I created to abstract away interacting with the CRM Web API. However, before using it we need to make it available to our class. To do that, add the following an import reference to the service at the top of the ‘place-order.ts’ file  ***import*** **{** Crm **}** from "../Services/Crm.WebApi"**;** |
|  |  | We now need to make use of DI to inject the service into our class. To do that we define a parameter in the constructor that is a Crm.WebApi object. It’ll look like the following  constructor**(*private*** \_client**:** Crm.WebApi**)** **{}** |
|  |  | The service is now available to our class, so let’s now refactor our submitOrder method to make use of our service. To do that locate the submitOrder method in the ‘place-order.ts’ file and remove the alert method you previously added and then add the following.  ***this*.**\_service**.**Create**(**“ch\_order”**,** order**);** |
|  |  | And you’re done, well almost. We need to make a few more changes to handle the response returned by the API and display the order number to the user. Since the Create method returns a promise we can make use of the ‘then’ method handle the response from the CRM Web API. To do that refactor the submitOrder method in the ‘place-order.ts’ file to look like this  ***this*.**\_client**.**Create**(**"ch\_order"**,** order**)**  **.**then**((**recordId**)** **=>** **{**  ***this*.**\_isSubmitted **=** ***true*;**  ***this*.**\_isSaving **=** ***false*;**  **})**  **.*catch*((**ex**)** **=>** ***this*.**handleException**(**ex**));** |
|  |  |  |
|  |  | r |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | Press the F5 button to begin debugging your application again. |
|  |  | Walk through the process of configuring a pizza and placing your order. After clicking the ‘Place Order’ button and providing your contact information, ensure you see a congratulations dialog that includes your order number. |
|  |  | End |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | Your project should now look something like this |