Devin Capaldi

Alexander Karmazin

Justice Koradi

Kevin Jayasekera

Ludovic Provost

GNG 1103

DELIVERABLE K

User Manual

University of Ottawa

3rd of December, 2020

Abstract

At the beginning of the semester project group A1 was asked by JAMZ to design a front-end user interface for the JAMZ delivery service. The following paper is a culmination of the work done by project group A1 and aims to explain to the reader how the final prototype for this application was made, as well as to explain to the reader the research behind the prototype.

Table of Contents

Introduction

A website was created for the accommodation and processing of orders in the fast food industry as part of the JAMZ business model. Having been assigned corresponding user requirements by JAMZ and staff running the GNG1103 course, Group A1 followed the engineering design process taught in class to conceptualize website elements, choose a method for website design and prototyping (in our case the WIX web-builder platform) and present our final prototype, according to the group's evaluation prepared to be put into production. The following is a user guide intended to familiarize a user with the product and it's design history, to ensure a user is capable of performing modifications for the improvement of overall product performance, as well as reproducing all work performed by the group in the website's design. The design history overviews the team's Problem Statement, Benchmarking, Design Criteria, decision to choose the Wix web design platform, work with the Google Maps API, Content Manager, Prototypes, Maintenance of Prototypes, and Future Plans/Recommendations.

Problem Statement

We are to design a User Interface for the Jamz delivery service. The interface needs to include tracking, a shopping cart, and have a simple, easy to use modern look.

Benchmarking

Skip the Dishes: The layout of this app was received highly by our clients and the user reviews seemed to praise the layout of this app. However, the users disliked the fact that tracking wasn't accurate and orders took a very long time to reach the user. They also experienced inconsistent service with some drivers passing off their orders to other drivers, leading to the long wait times.

Uber Eats: There is a growing frustration among users that orders are not getting to their house or are damaged by the time they get there. There were also complaints about overpricing of certain products. However the simple design is well received by the user base, as well as the tracking feature.

Doordash: Many complaints about inconsistent experiences with drivers. There were many complaints about the tracking on the app being incorrect. Furthermore, there are also issues with credit cards being incorrectly charged. There are a few positive reviews about having an itemized receipt so you can see if any mistakes are made. Customers noted that the interface easily allowed them to contact doordash when they experienced difficulties with order deliveries, thus improving the overall quality of customer service.

Design Criteria

	Needs	Design Criterias
1	The interface is easy for the user to operate and understand.	- Simplicity - Navigability - User-Centricity - Type of interface (website and/or app)
2	The app is able to store a restaurant's item(s) inside a shopping cart.	- Direct inventory interfaces - Ability to communicate with other programs
3	The app can track the delivery drone and displays its whereabouts and a progress bar to the user.	- Ability to communicate with other programs - Layout
4	The app shows a list of nearby restaurants, as well as their menu(s).	- Direct inventory interfaces - Ability to communicate with other programs

5	A user can login and register for an account that stores details like their name and their address.	- Authentication of user and server - Compatibility with different devices and browsers - speed/responsiveness
6	The interface has clean and modern looks.	- Layout - Visual style - Colors - Reduce user's cognitive effort
7	The restaurants shown on the app can be sorted in categories.	- Accessibility - User-Centricity

We selected the seven needs based on user and technical benchmarking along with needs that the client specified in the initial client meeting. We determined that the most important need was that it needs to be very easy and quick to use. We determined this using the bounce rates that we found when we were benchmarking. We then determined that the second most important need is that the website has to have a shopping cart to store and view items. This need was high on the short list of the clients needs.

We structured the website around the shop and cart section of the website. We determined that the shopping cart should be visible one each page of the website and it should be very accessible when browsing through various sections and categories. Thirdly, we prioritized the functionality of the tracking over the other remaining needs. The clients also specified these

needs in the first and second client meeting, and they were happy about the functionality of the tracking.

Although a little bit of work needs to be done to implement the back end to the tracking, it is quite functional. The fourth need, the visibility of nearby restaurants, is closely related to the previous need. The back end must be implemented to view restaurants only in your area, meaning in a 10 kilometer radius. The fifth need was to have a members section on the website, as they users like to order the same meal multiple times. We have a completely functional members section with the ability to save locations/address and previous orders.Although this is fully functional, some of the features, like the use of coupons were not available with the free version of wix, thus we are not available to test those features.

Our sixth need ties with the first need but it is a little more specific. Many fellow competitors in the food delivery services industry, have complete modern and clean looking websites, thus we need to be the same. Lastly, we needed to categorize the menus in categories, such as breakfast, lunch, dinner etc. We determined this with the use of benchmarking, as we found that ordering with categories is much easier for the user to find exactly what he or she is looking for. With application of all of these needs, we were highly confident that our website would have a high degree of success.

Wix

To create the first prototype (along with the rest), we decided to use wix. As we didn't have much time to search for other websites designing softwares, and some of us already knew about wix, we agreed to continue with wix. We were already somewhat familiar with it and we knew that it would be able to do enough of what we wanted to do. Whether another software would be better for our project is still unknown, but nonetheless, wix was able to get the job done very well. The format of wix and the design tools are simple to use, yet very flexible with creativity.

With the group's general lack of knowledge/experience on coding, the design tools were found to be very helpful as we were able to quickly implement different types of images, backgrounds, texts, buttons, headers, functions, etc… Also, the requests from the clients after client meetings were able to quickly get met by using the wix features, as it is easy to make changes to.

If more time was allotted for the project, we may have attempted to discover other website designing platforms, however we found wix very efficient for our project and would maybe have been satisfied with it either way. For an added bonus, most features are free and it only begins to cost money once more exclusive features are being implemented (ex; having a custom URL address).

Google Maps API

Many problems were encountered while figuring out the tracking for the website. The first one being the “what’s your location?” input box, where a user is asked to enter his address. Wix does not have any way to suggest addresses depending on what the user has entered and automatically fill in the box for the user, so code was written to deal with that problem. It works by showing a repeater (list of text boxes) and adjusting the text inside each text box to a corresponding address found on Google Maps. Google Cloud was used to obtain an API key so that the website is allowed to access the Google Maps data (which contains all known addresses) and can access the billing information and the API requests from the website. The code also contains a counter that prevents people from spamming the website with API requests. The API’s that were used on the website are the Places API, Geocoding API and Distance Matrix API.

Now that we have access to possible addresses, we can let the user enter his address and it will automatically be saved in the cookies. A part of the code also translates an address into latitude, longitude and a description so that it can be used to calculate the distance and time from the restaurant to the user with the Distance Matrix API. The Google Maps element from wix

(#GoogleMaps1) can read from either the cookies or the user database to find and display an address.

The code has been documented and explains what most of the lines do, to facilitate the implementation of the back-end. It can also be viewed in the appendix below. Furthermore, every element of the page has a name and can be viewed on dev mode. The API key used, as well as the full control of the Google Cloud account can be easily switched to another Google account to provide the team with full control over the information of the website, including the traffic, the number of errors, the median latency and access to many other services like libraries, credentials, domain verification, etc.

Content Manger

Addresses: This content manager system temporarily stores the address of the user in four different fields:

- Title: the user's location as text.
- UserAddress: the full address of the location as an address.
- Latitude: stores the latitude of the location as a number.
- Longitude: stores the longitude of the location as a number.

ContactForm: This stores the messages the users enter on the "Our team" page in six sections:

- Submission Time: stores when the user sent the message/review as a time stamp.
- Name: Last name of the user as a text.
- Message: the message from the user as a text.
- Email: the contact email address of the sender as a text.
- Subject: the subject of the message, as filled in on the "Our team" page as a text file.
- First Name: first name of the sender as a text file.

Coupons: Stores different coupons for money or percent off. The amount off, code name, time it is available, minimum subtotal, limit per customer and start time can be changed directly inside the content manager. Other data can also be filled in like the number of times the code was used.

Badges: Badges can be created and customers can be assigned to badges that will show on their member's area.

PrivateMembersData: This is where everything about the member's data is stored, like their names, addresses, paiement information, etc.

Collections: Contains two columns: Name (text) and Main Media (picture or gallery). This can be used to change the different subsections of the restaurant menus, like "Hamburgers", "desserts", etc.

Orders: This is where the order of the user is stored, like the number of items and the number / SKU of the different items in the cart.

Products: Stores all menu items of all restaurants. Each item has:

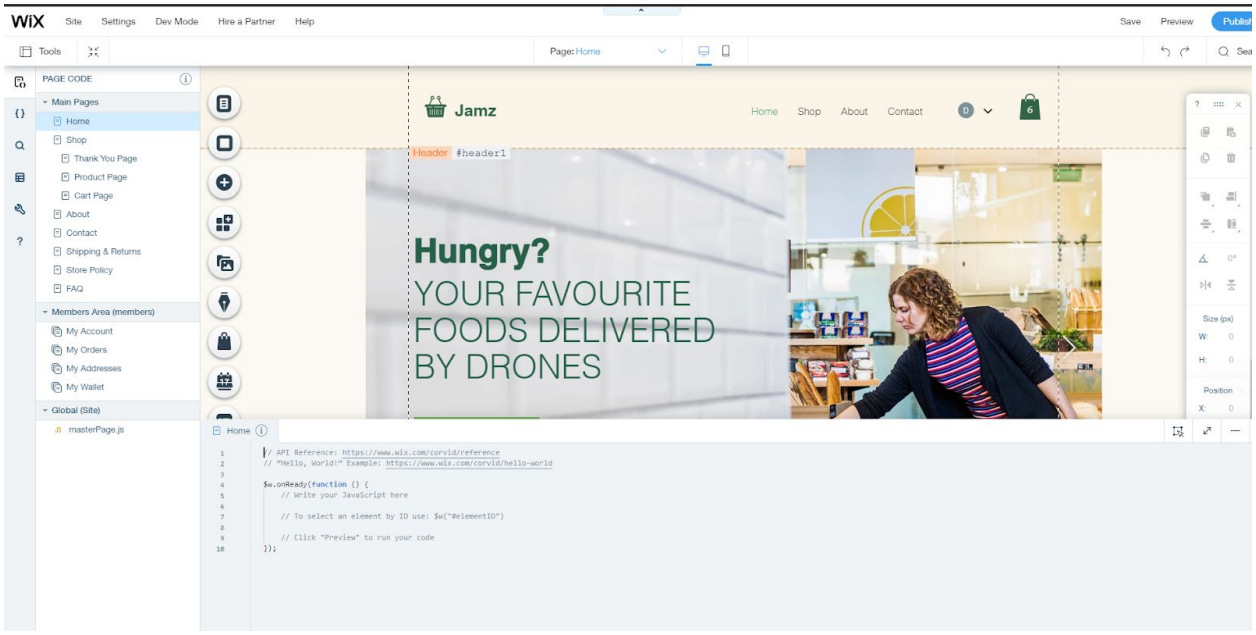
- Name: the name of the item as a text file.
- Description: can include the ingredients and other information about the item. The backend can allow restaurants to modify those themselves. This is stored as a rich text.
- Main Media: a picture of the item, stored as an image.
- Media Items: if the restaurant wants to display more than one image per item. It is stored as a gallery.
- SKU: this is where the restaurant can input their items as SKU. This facilitates the management of products and the count of the inventory for restaurants. It is stored as a

text file so it can also include letters if any restaurant uses both letters and numbers in their SKU.

- Currency: It can be changed depending on the country. It will affect what is shown on the shop page. It is stored as a text.
- Price: The price of the item stored as a number.
- Discounted Price: The discounted price of the item stored as a number. This could be modified by the restaurant directly.
- Quantity In Stock: Quantity in stock for every item. It can be very useful for the restaurants. It is stored as a number.
- Weight: This is where the weight of the item can be stored. It can be useful to calculate the total weight so that the drone knows how to calibrate its motors. It is stored as a number.
- Collections: This allows the item to be moved form a menu to another easily. The field type is Multi-Reference, which means it communicates directly with the “Collections” content manager.

Introduction to Wix Web Editor

Project group A1 used the Wix Web Editor to create each of the prototypes, this section aims to show the reader a brief overview of the website editor and how to work within it.



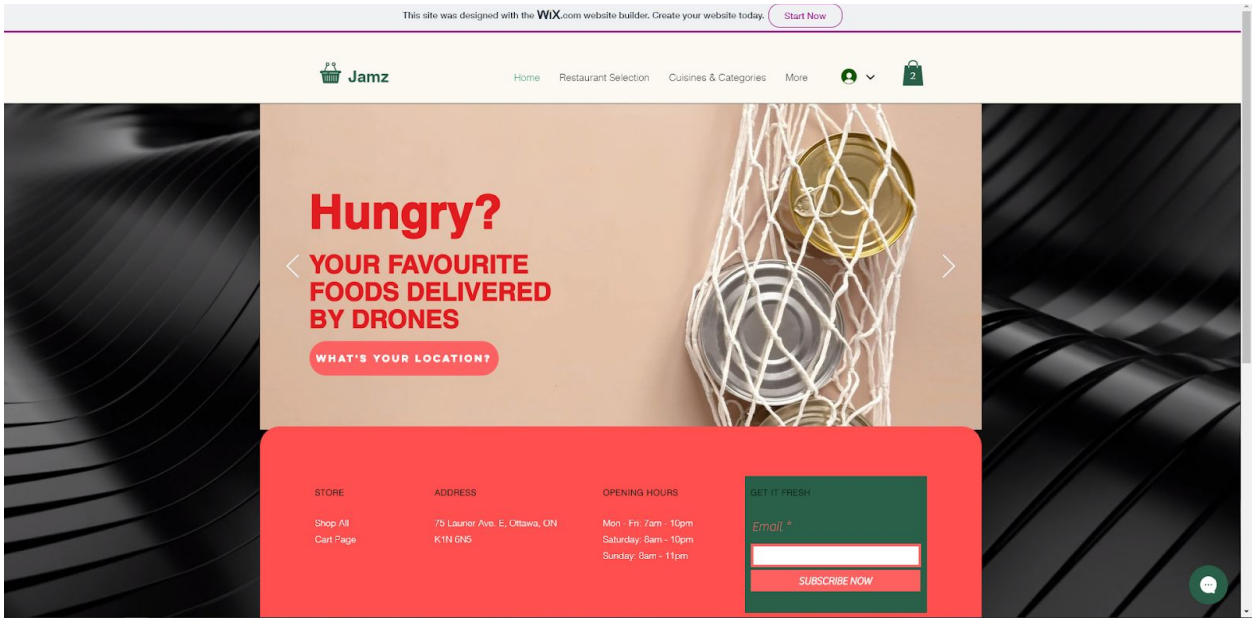
When first starting up the wix website editor this is the page you are greeted with, you are given a toolbar as seen on the right-hand side of the screen that allows you to input numbers or trash pictures as well as add text to the web page etc.

Moving on the left-hand side of the screen you are able to select each page's individual code and input code onto each of those pages. Also seen on the left hand side is each of these little bubbles. By clicking on each of these individual bubbles you are able to access editor features. The top bubble allows you to add or remove pages. The second bubble allows you to

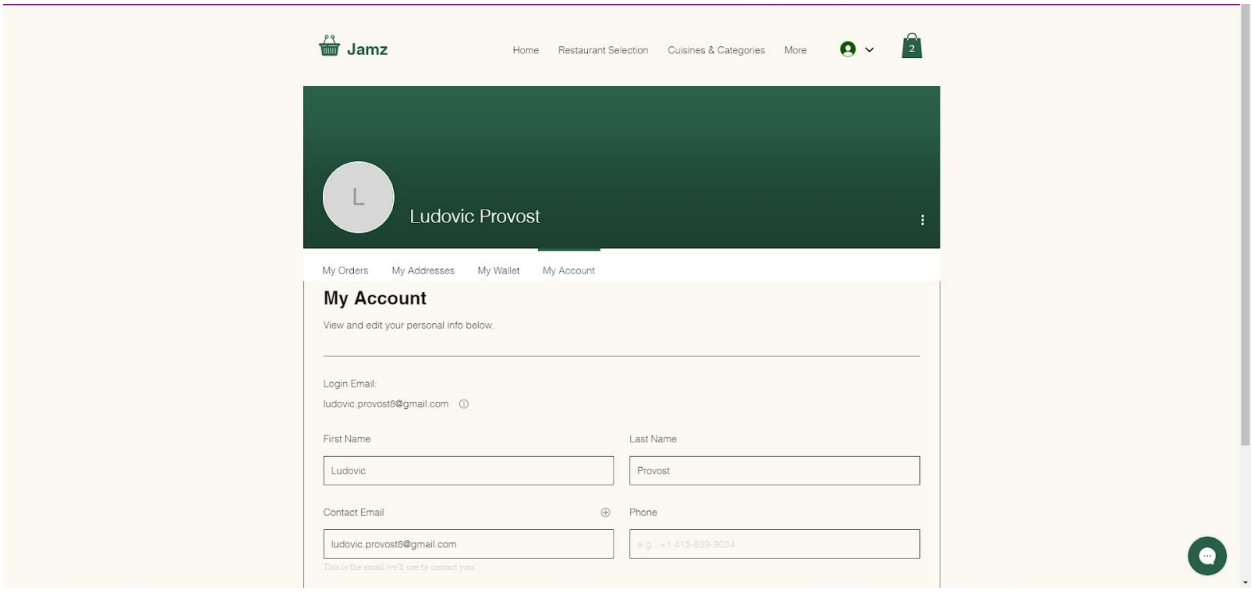
select the background of each individual pages, and the fourth bubble allows you to add Wix apps to your webpage. These bubbles are the only relevant editor features that we're used in the making of each prototype

Prototype I

For our first prototype, an effort was made to learn the ins and outs of the website builder wix, and we created a basic template consisting of 2 sections, the shop and the members area. The first iteration of the home page is seen below, where the colors, text and buttons were not final.

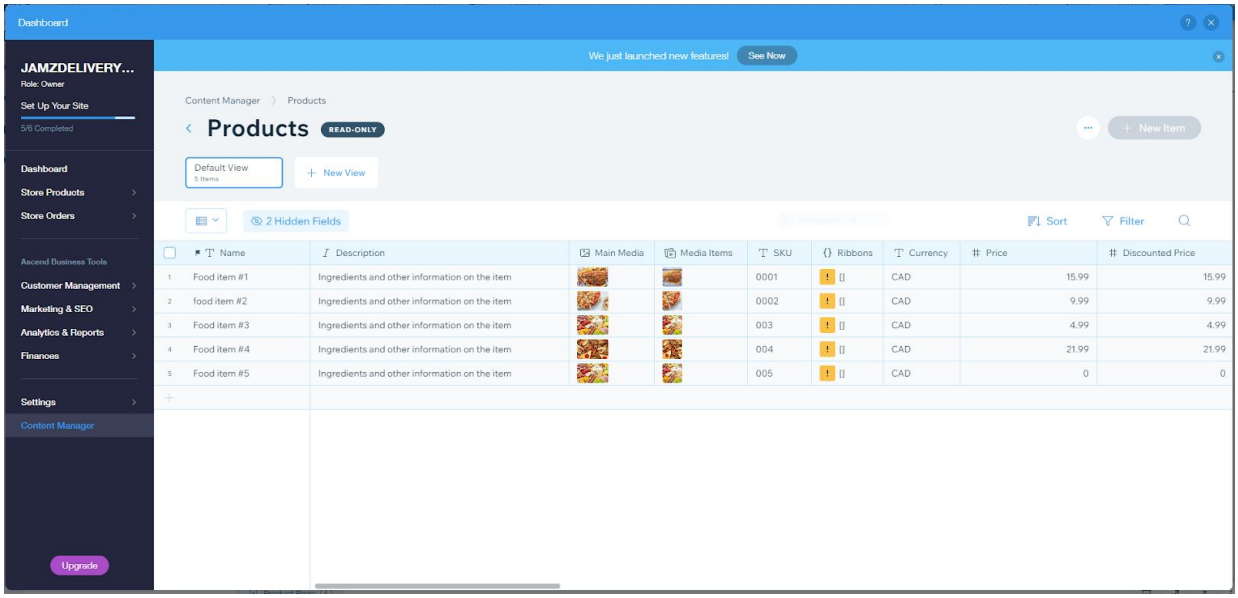


The majority of the work in the first prototypes went into the members area. In the members area, the user can see past orders, store multiple addresses, decide the payment method, add or remove cards and change his information, for instance; the name, contact info, and main address. Note, this was only a preliminary iteration of what we were initially striving for.

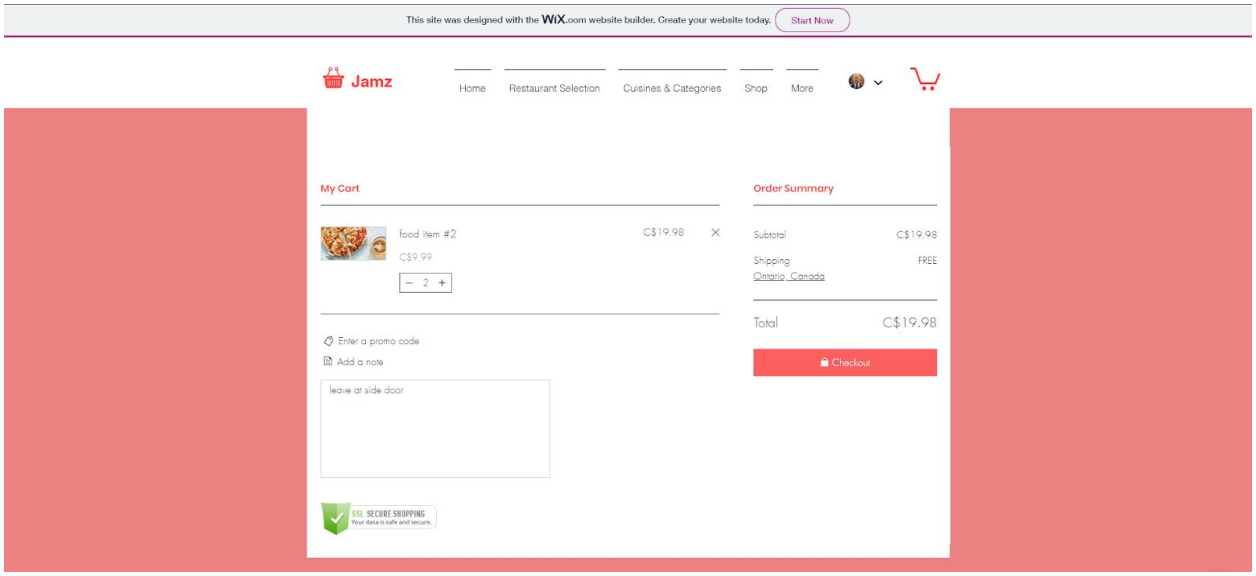


Prototype II

For prototype 2 primary focus was on making the restaurant menu functional. Thus, we created a content manager that can store products and their information such as their name, description, a price, etc. to later display them on the shop page. The context manager can easily communicate with the backend and create or delete products. This allows us to have a sole shop page for all of the restaurants, and depending on which restaurant is selected, the product gallery displayed on the shop page will show the corresponding menu items for that specific restaurant. The backend can also be used to allow restaurants to edit the prices, discounts and descriptions.

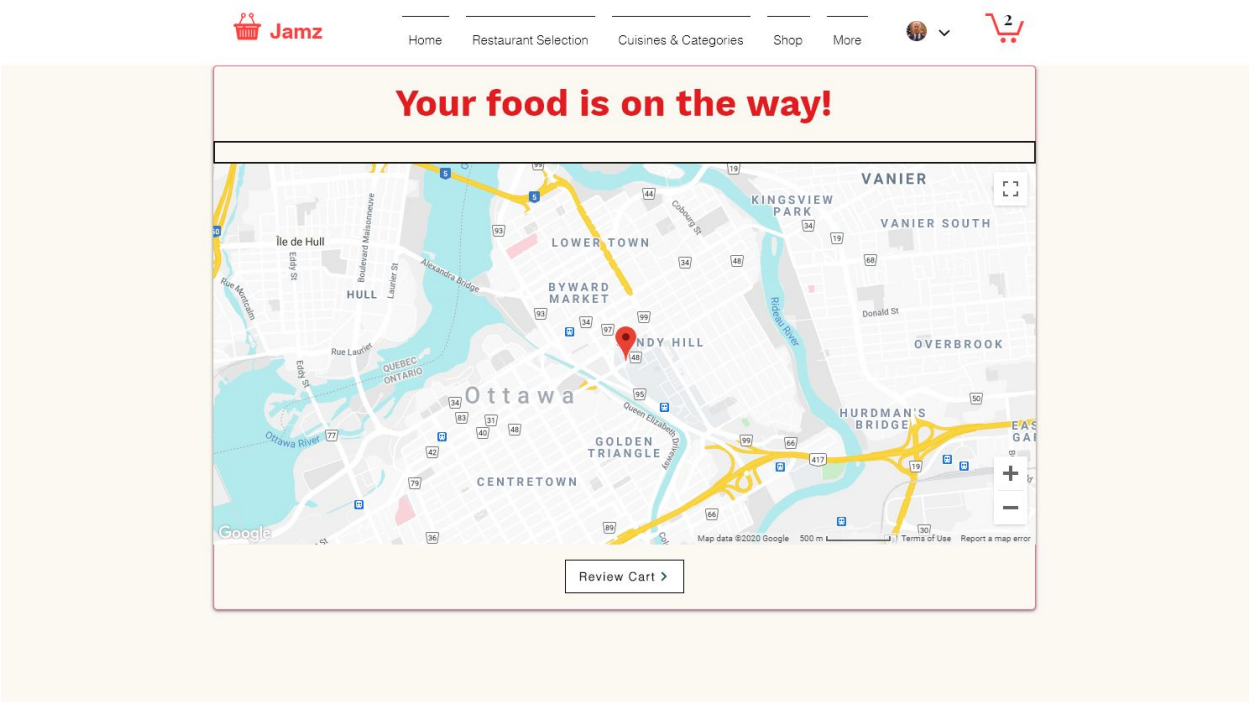


We also implemented a shopping cart which can be accessed at all times. Inside the cart, the user can modify items he pleases, use a promo code, add a note to the restaurant, and finally checkout. The subtotal, total can also be viewed and seen below. Once the user has paid, it will send a confirmation email, which can also include a QR code or a bar code. To complete what we had initially planned, our only remaining task was to implement the tracking feature.

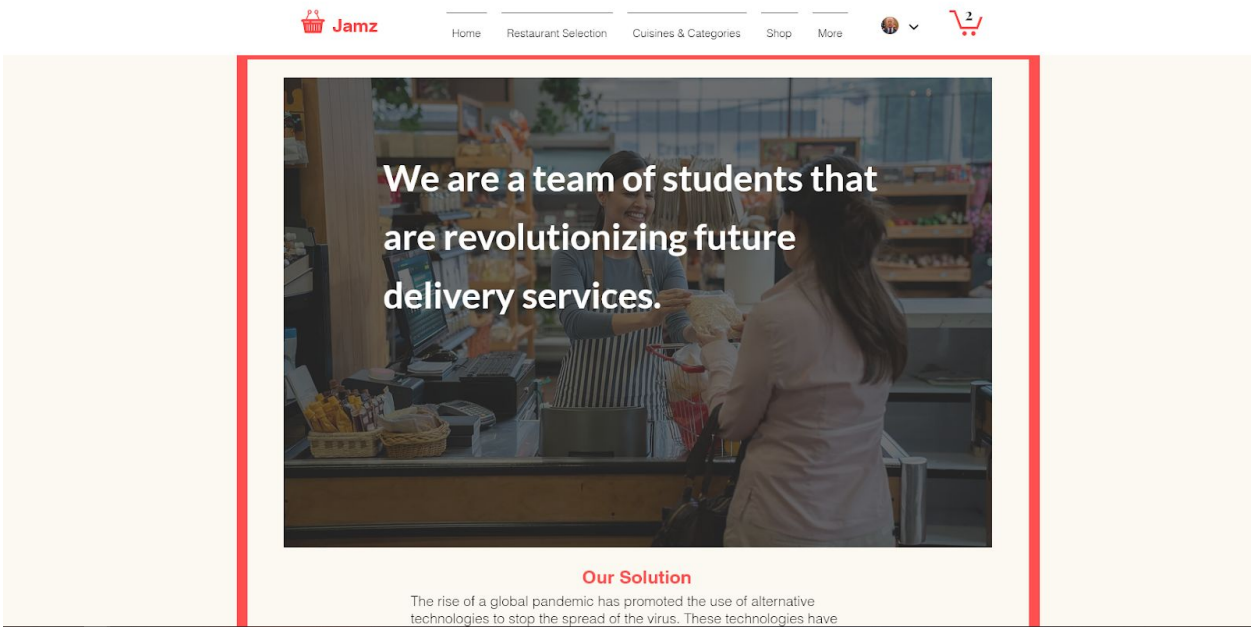


Prototype III

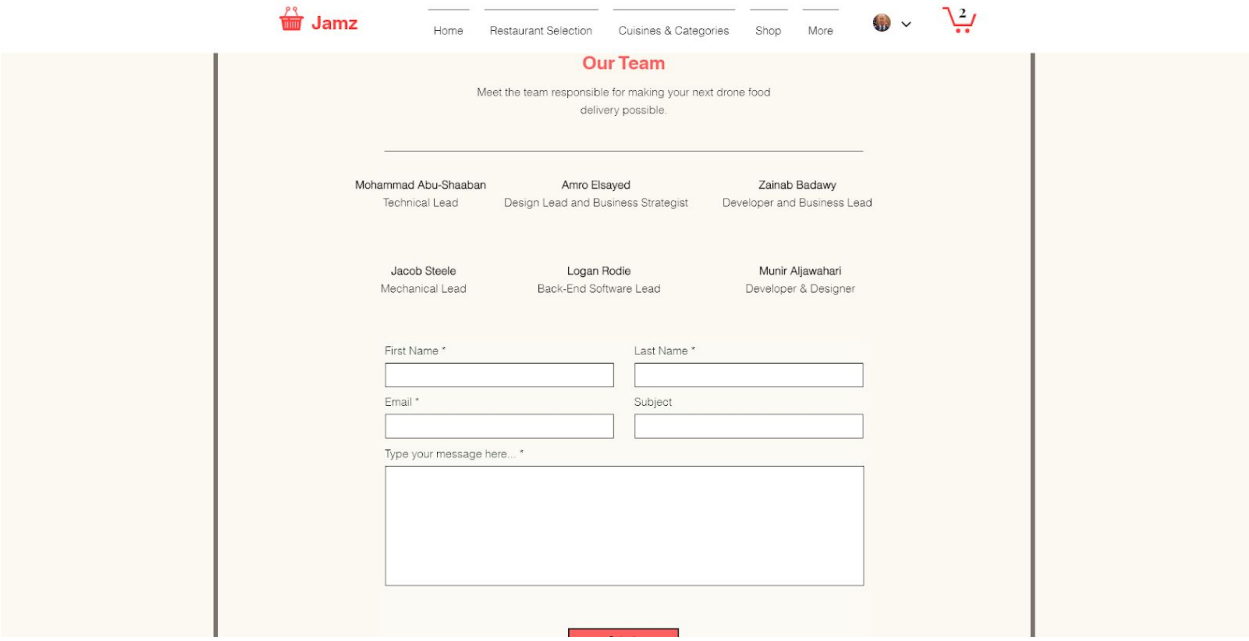
On the home page, the user is asked to enter his address or he may choose to skip that step if he has previously saved that information. The address is later used to show a map of the area using an API key and a google maps element from wix. Since the location is stored, it can be used later to determine how long the delivery will take as well as the distance. The black rectangle seen below is a progress bar that can be easily manipulated with the back end to fill up when certain actions are completed, like when the food is leaving the restaurant and once it is at its desired location.



The final touches include the addition of an “about us” page and “our team” page.



Users can also contact the team on this page.

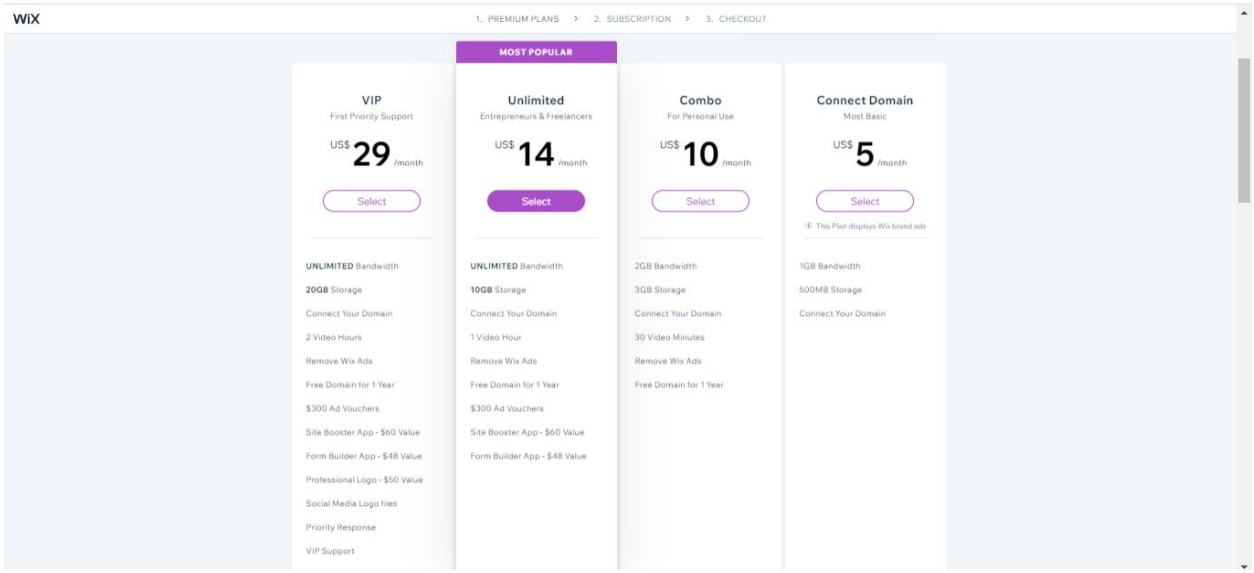


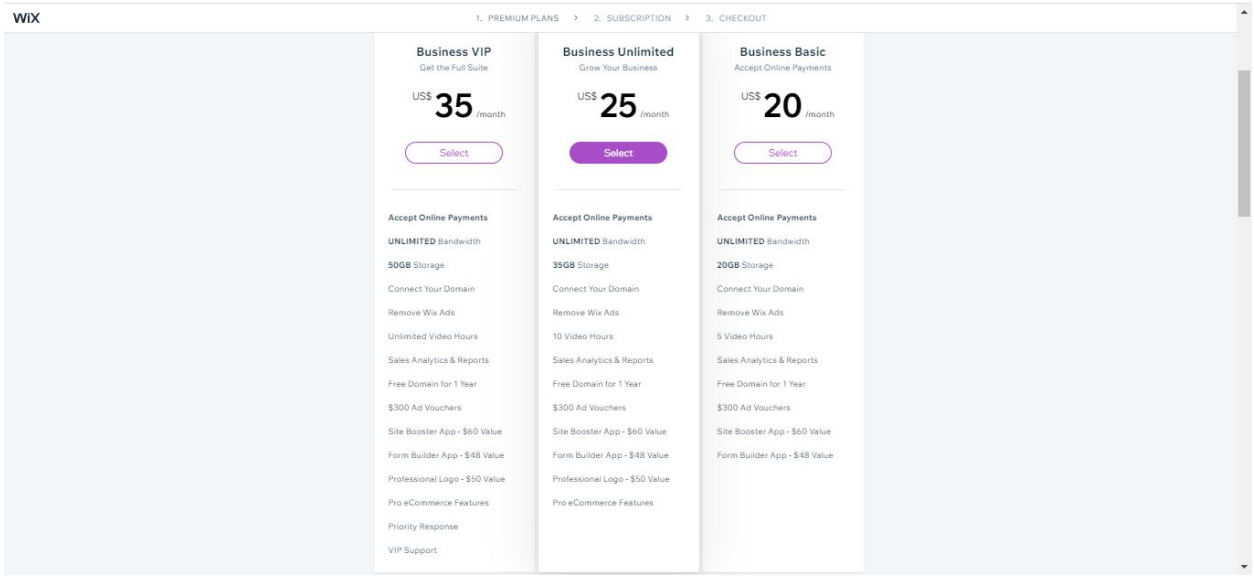
Prototype III Use

When first entering the website, a user is greeted with the home screen where he/she is prompted to enter a delivery address. Once the address has been entered or a user has logged in, they then have the option to proceed to either the “Cuisines and Categories” or the “Restaurant Selection” sections/tabs, formatted to offer two different methods of browsing and selection of desired products. This model was implemented for greater flexibility of selection methods. After a user selects the cuisine/category they desire they select items they wish to purchase, and are then taken to the shop page, where they select how much of each item is desired, accompanied by the corresponding cost having selected these values the user clicks the “View Cart” button. Subsequently, item quantity and cost totals are displayed in the “My Cart” page, where the user reviews their order and pays for it. The user then clicks the “View Tracking” button, where they can see updates on their order’s location as it is being shipped to their location. They also have the option to return to their cart.

Maintenance of Prototype

Although we were able to complete the whole project just by using Wix’s free features, Wix actually has premium features. Any user of the Wix application can pay to access premium, exclusive features. There are many options available with different available features, different durations, and different costs. Furthermore, these options are categorized into two categories; website and business & ecommerce. In order to enact the ability to purchase items on the website a subscription is required. A subscription is also required to change the website URL.


WiX
1. PREMIUM PLANS > 2. SUBSCRIPTION > 3. CHECKOUT
MOST POPULAR
VIP
First Priority Support
US$ 29 /month
Select
UNLIMITED Bandwidth
20GB Storage
Connect Your Domain
2 Video Hours
Remove Wix Ads
Free Domain for 1 Year
$300 Ad Vouchers
Site Booster App - $60 Value
Form Builder App - $48 Value
Professional Logo - $50 Value
Social Media Logo files
Priority Response
VIP Support
Unlimited
Entrepreneurs & Freelancers
US$ 14 /month
Select
UNLIMITED Bandwidth
10GB Storage
Connect Your Domain
1 Video Hour
Remove Wix Ads
Free Domain for 1 Year
$300 Ad Vouchers
Site Booster App - $60 Value
Form Builder App - $48 Value
Combo
For Personal Use
US$ 10 /month
Select
2GB Bandwidth
3GB Storage
Connect Your Domain
30 Video Minutes
Remove Wix Ads
Free Domain for 1 Year
Connect Domain
Most Basic
US$ 5 /month
Select
This Plan displays Wix brand ads
1GB Bandwidth
500MB Storage
Connect Your Domain


WiX
1. PREMIUM PLANS > 2. SUBSCRIPTION > 3. CHECKOUT
Business VIP
Get the Full Suite
US$ 35 /month
Select
Accept Online Payments
UNLIMITED Bandwidth
50GB Storage
Connect Your Domain
Remove Wix Ads
Unlimited Video Hours
Sales Analytics & Reports
Free Domain for 1 Year
$300 Ad Vouchers
Site Booster App - $60 Value
Form Builder App - $48 Value
Professional Logo - $50 Value
Pro eCommerce Features
Priority Response
VIP Support
Business Unlimited
Grow Your Business
US$ 25 /month
Select
Accept Online Payments
UNLIMITED Bandwidth
35GB Storage
Connect Your Domain
Remove Wix Ads
10 Video Hours
Sales Analytics & Reports
Free Domain for 1 Year
$300 Ad Vouchers
Site Booster App - $60 Value
Form Builder App - $48 Value
Professional Logo - $50 Value
Pro eCommerce Features
Business Basic
Accept Online Payments
US$ 20 /month
Select
Accept Online Payments
UNLIMITED Bandwidth
20GB Storage
Connect Your Domain
Remove Wix Ads
5 Video Hours
Sales Analytics & Reports
Free Domain for 1 Year
$300 Ad Vouchers
Site Booster App - $60 Value
Form Builder App - $48 Value

Future Plans/ Recommendations

Due to time constraints there were many things in this project that were planned but were unable to be acted upon. For example with more time, development of the tracking feature would've been more advanced, changing the pin from a simple red pin to a more representative pin. There were also talks, in the future about buying a subscription to Wix, and having the website indexed, as well as a custom web url. We also had plans to release a darkmode version of the website. The last thing that was planned for the future, is to make this website responsive to mobile phones. As Wix doesn't automatically convert to mobile mode, but rather needs specific development to be made.

If this project is to be done again we recommend that it's developed in tandem with the backend application, as a lot of features dependent on the backend would be able to enter into further development faster, as well as have certainty that it would be fully functional. We also recommend if this project is to be done again that a Wix subscription is bought upfront so that money processing features can be developed sooner.

Conclusion

In conclusion, we have learned a lot from this project. We feel that this project was completed to a satisfactory degree with the time constraints. We have learned some difficult lessons and if we were to do this project again we would really emphasize proper planning and making better estimations of the duration of building certain features. However, we have a fully functioning prototype, that if JAMZ wanted to be able to use it, it would be ready in a week. What separates us from other groups, is the simplicity of our design and in the malleability of the Wix Web editor platform, we're able to adapt quickly to new situations, as well as due to our simple design being able to give very good user experience. We're thankful to JAMZ for the opportunity to design for a real client and we're proud of what we have done.

Bibliography

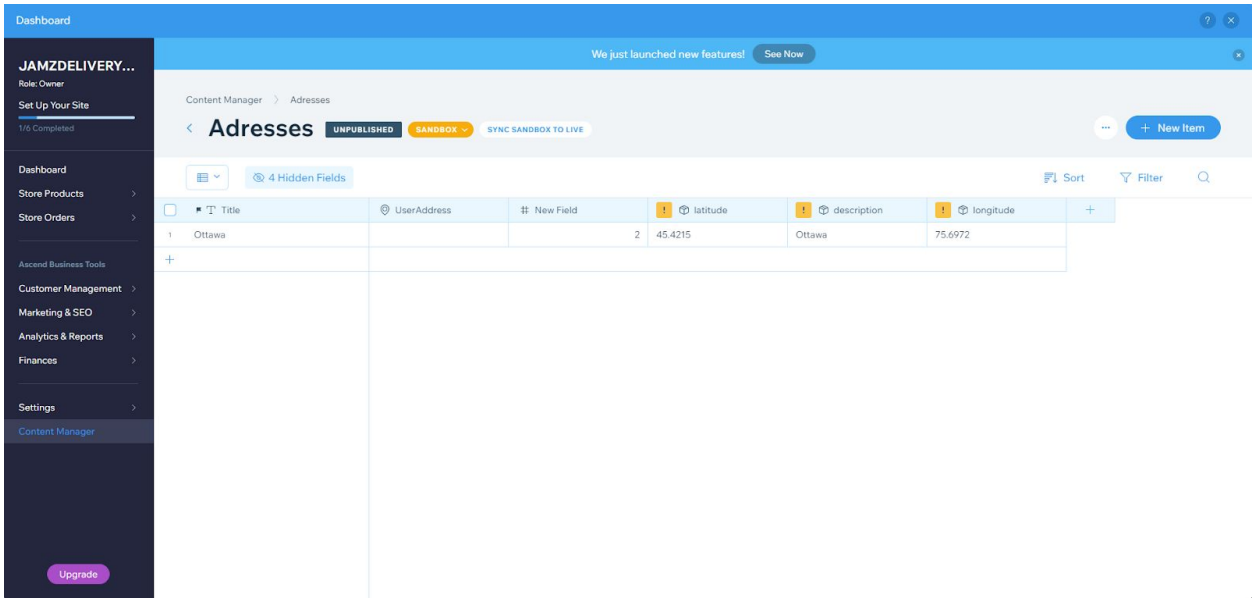
N. Babich, J. Morales, and M. Rae, “Web Page Design: A Comprehensive Guide: Adobe XD Ideas,” *Ideas*, 24-Nov-2020. [Online]. Available: https://xd.adobe.com/ideas/principles/web-design/web-page-design/.

Barraclough, Dan. “How to Use Wix: An Easy Step-by-Step Guide (2020).” Website Builder Expert, 1 June 2020, www.websitebuilderexpert.com/website-builders/wix/how-to-use-wix/.

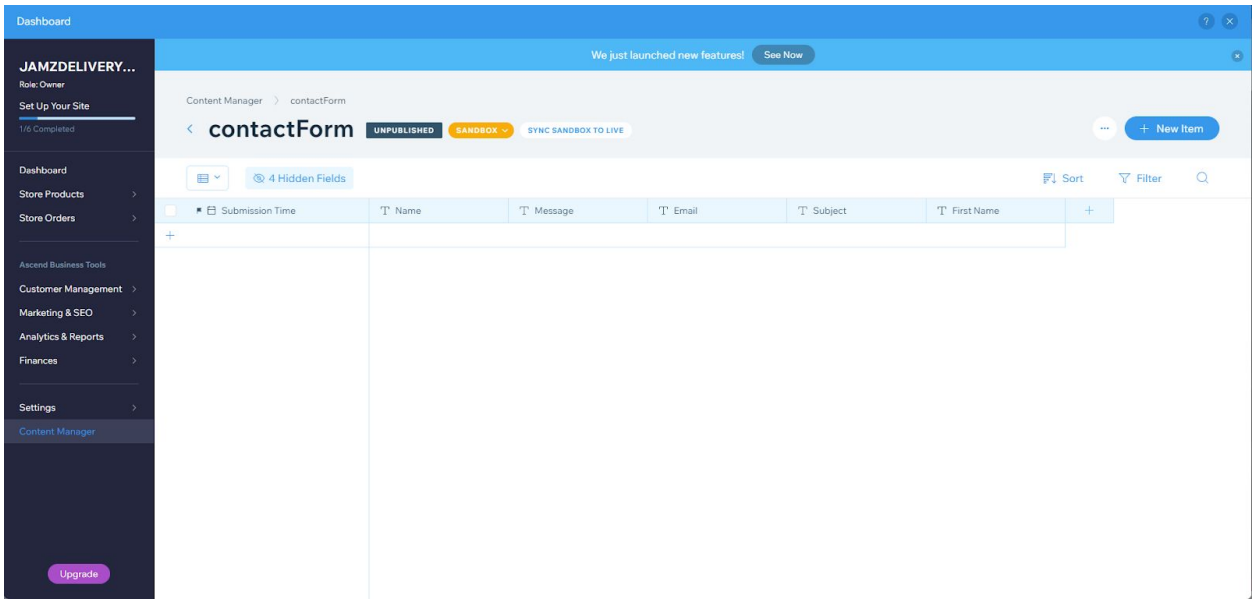
Touitou, Margot. “How to Make a Website: GUide for Creating a Website in 2020.” How to-Wix.com, 12 Jan. 2020, www.wix.com/how-to/make-a-website.

Appendix

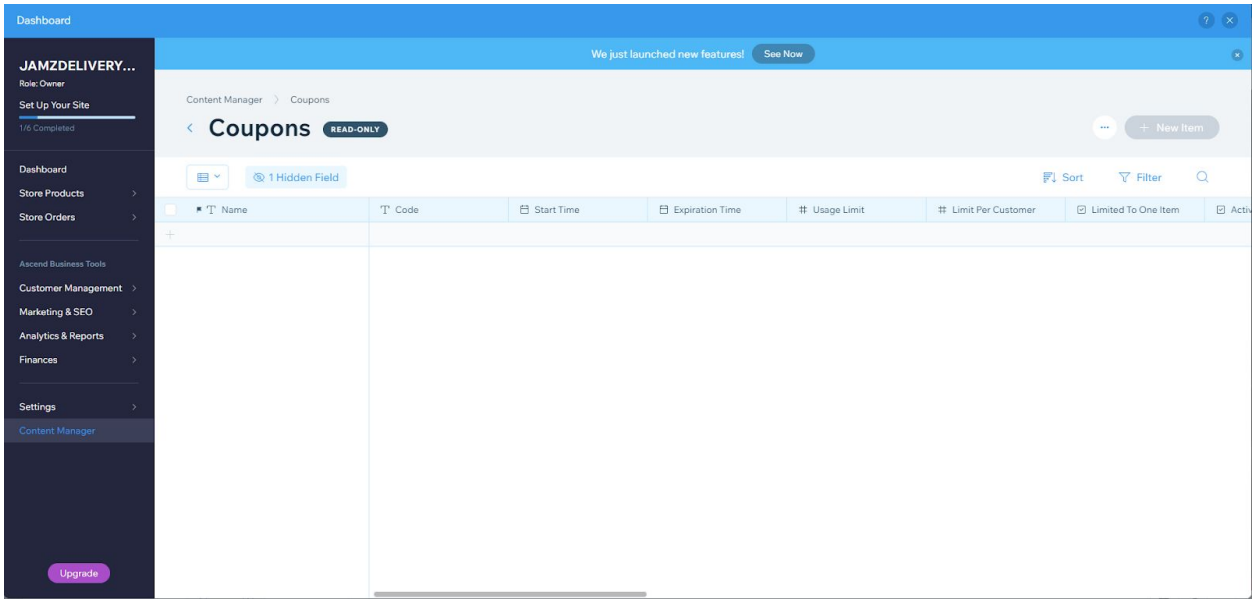
Addresses Content Manager :

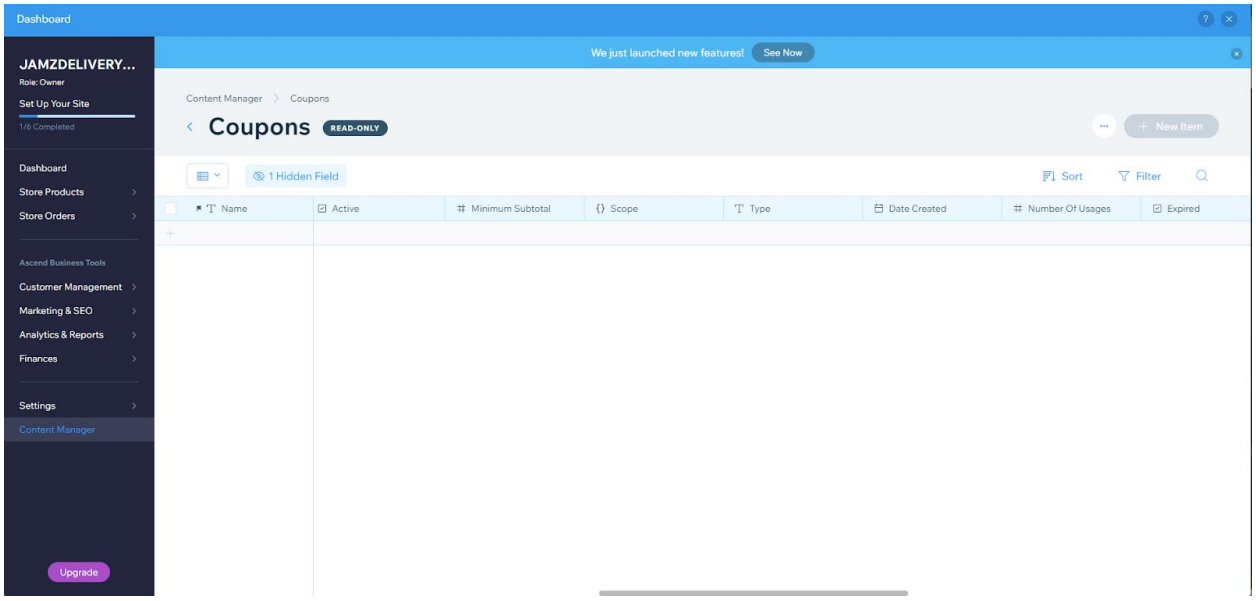


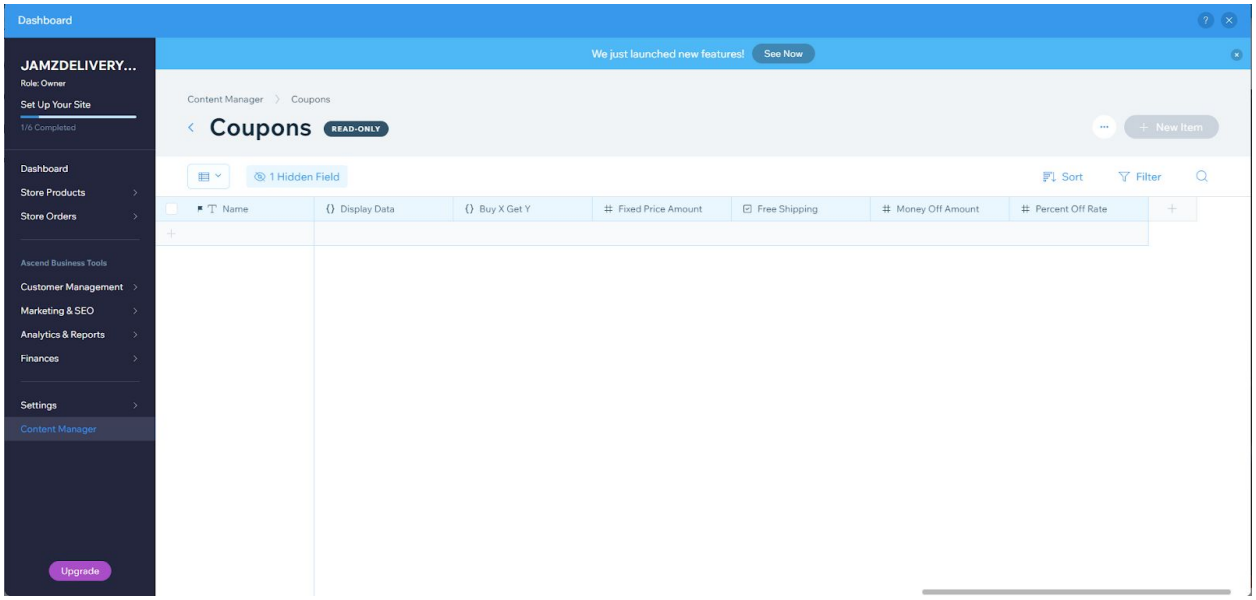
ContactForm Content Manager :



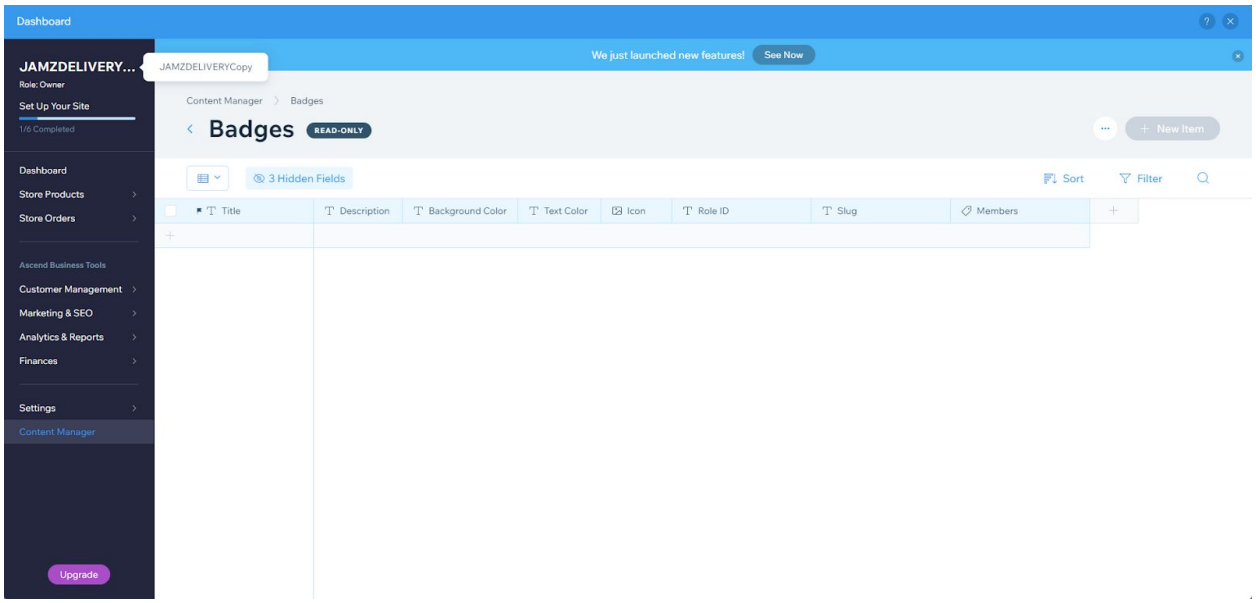
Coupons Content Manager :



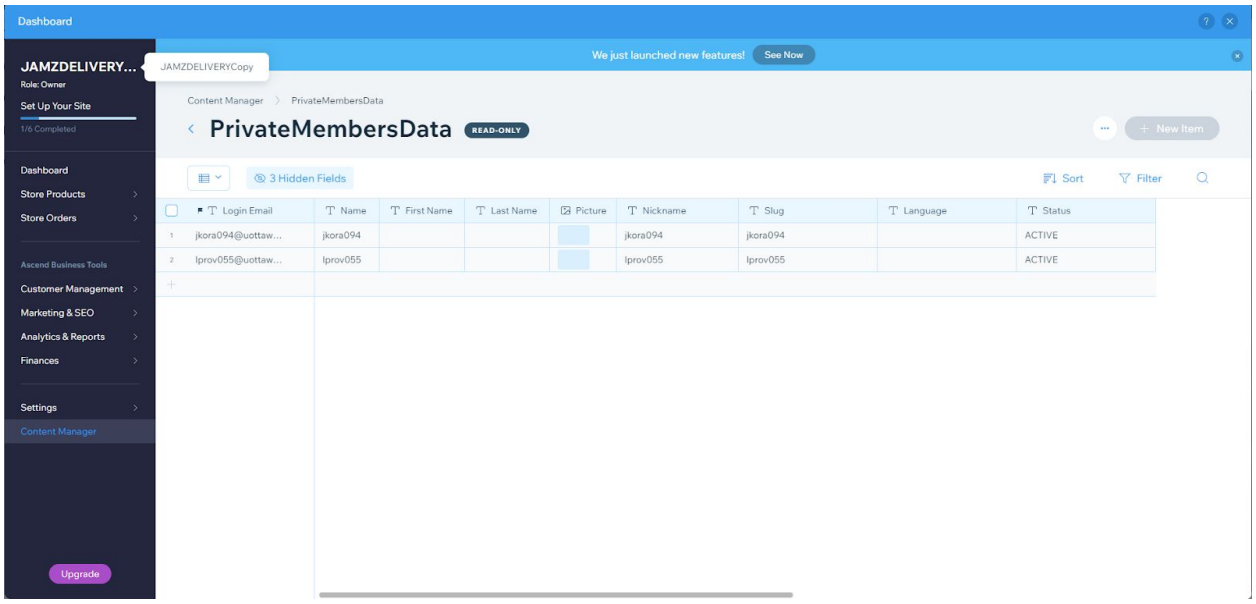




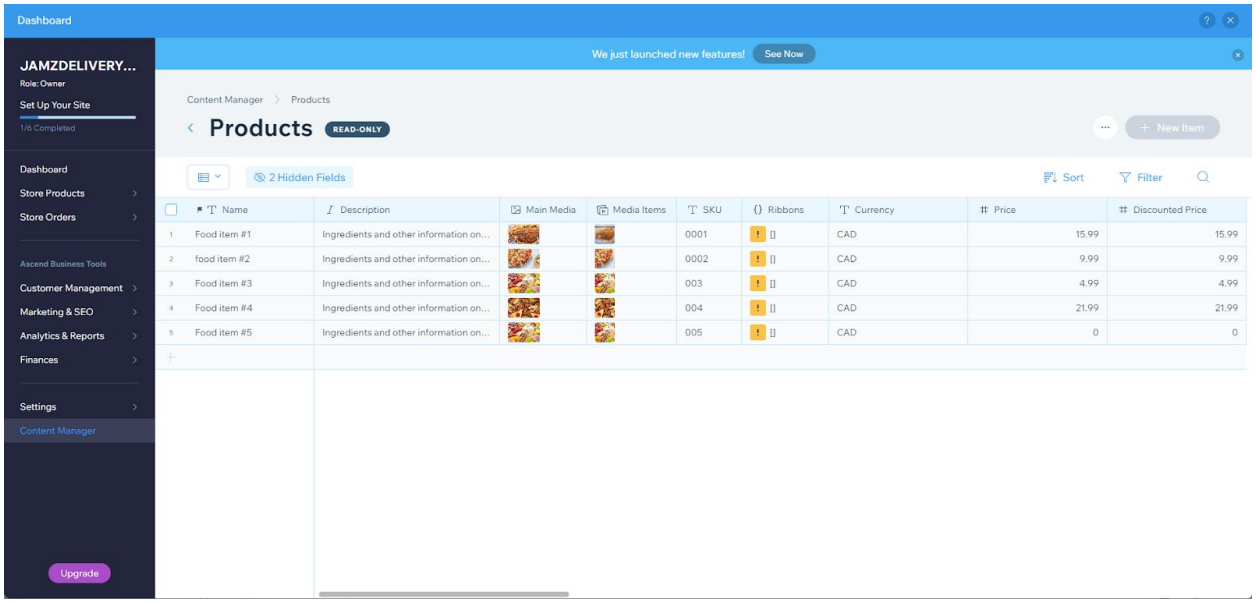
Badges Content Manager :



PrivateMembersArea Content Manager :



Products Content Manager :



Code for the home page :

```
import wixWindow from 'wix-window';
import {local} from 'wix-storage';
import {autocomplete} from 'backend/gapi';
import {details} from 'backend/gapi';
import {reverse} from 'backend/gapi';
let lastQueryTime = new Date();


$w.onReady(function () {
    // handle each suggestion repeater item
    $w("#repeater1").onItemReady( ($w, itemData, index) => {
        const text1 = $w("#text26");
        text1.text = itemData.text1;
        const text2 = $w("#text27");
        text2.text = itemData.text2;
        text2.hide();
    });
    $w("#repeater1").collapse(); // hidden on page load
    // retrieve saved location (if exists) from local storage
    let id = local.getItem("place_id");
    if(id === undefined || id === null || id.length === 0) {
        // if no location saved, find the IP-based geographic location
```

```
        geoloc();
    }
    else {
        // if a location was saved in local storage, get the details
        set_details(id);
    }
    $w('#input4').onKeyPress((event) => {
        if(event.key === "Enter") {
            local.setItem("address", event.target.value); //stores the entered user address in their
cookies
            return;
        }
        input4_keyPress(event,$w);
    });
});


function set_details(val) {
    details(val).then(function(resp) {
        // find the city (locality) and country of the location
        let place = resp.result;
        var filtered_array = place.address_components.filter(function(address_component){
            return address_component.types.includes("country");
```

```
    });
    var country = filtered_array.length ? filtered_array[0].long_name: "";
    filtered_array = place.address_components.filter(function(address_component){
      return address_component.types.includes("locality");
    });
    var locality = filtered_array.length ? filtered_array[0].long_name: "";
console.log("details: " + locality);

    let name = place.formatted_address;
    let id = place.place_id;
    let utc = place.utc_offset;
    let lat = place.geometry.location.lat;
    let lng = place.geometry.location.lng;

    // save the details of the location with wix-storage
    local.setItem("place_city", name);
    local.setItem("place_lat", lat);
    local.setItem("place_lng", lng);
    local.setItem("place_utc", utc);
    local.setItem("place_id", id);
    $w("#input4").value = name; // set input field to location
```

```
// array of location detail items for the repeater
let detailsList =
[
    {
        "_id": "1",
        "text3": "place name",
        "text4": name
    },
    {
        "_id": "2",
        "text3": "latitude",
        "text4": "" + lat
    },
    {
        "_id": "3",
        "text3": "longitude",
        "text4": "" + lng
    },
    {
        "_id": "4",
        "text3": "utc",
        "text4": "" + utc
```

```
        },
        {
          "_id": "5",
          "text3": "place id",
          "text4": id
        }
      ];
  });
}

export function geoloc() {
  wixWindow.getCurrentGeolocation()
  .then( (obj) => {
    let lat = obj.coords.latitude;
    let lng = obj.coords.longitude;
    reverse(lat, lng).then(function(resp) {
      let status = resp.status;
      // removed the pittsburg example
      let results = resp.results;
      var country = null, city = null, place_id = null;
      var c, lc, component;
      for (var r = 0, rl = results.length; r < rl; r += 1) {
```

```
    let result = results[r];
    // look for city (locality) and country
    if (!city && result.types[0] === 'locality') {
    for (c = 0, lc = result.address_components.length; c < lc; c += 1) {
        component = result.address_components[c];
        if (component.types[0] === 'locality') {
            city = component.long_name;
            continue;
        }
        if (component.types[0] === 'country') {
            country = component.long_name;
            if(city && country)
                break;
        }
    }
}
else {
    continue;
}
if (city && country) {
    place_id = result.place_id;
    // set_details(place_id);
```

```
                break;
            }
        }
    });
} )
    .catch( (error) => {
        let errorMsg = error;
        console.log(errorMsg);
    });
}

function input4_keyPress(event, $w1) {
    let current = new Date();
    if((current - lastQueryTime) >= 3000) { //change the 3000 to change number of milliseconds
        lastQueryTime = current; // timer
    }
    else {
        return;
    }
    setTimeout(() => {
        // use the current value to get a list of location suggestions
        // we call the autocomplete() web module from the backend
```

```
        let val = event.target.value;
        if(val.length === 0)
            return; // ignore if empty

        autocomplete(val).then(function (resp) {
            // create an array of suggestions for the repeater
            let predictions = resp.predictions;
            let suggestions = [];
            let i = 0;
            predictions.forEach(function (prediction) {
                let item = { "_id": i.toString(), "text1": prediction.description, "text2":
prediction.place_id };
                i++;
                suggestions.push(item);
            });
            console.log(suggestions);
            $w("#repeater1").data = suggestions; // add the suggestions to the repeater
            $w("#repeater1").expand();  // we have data so we can expand the repeater
        });
    }, 10);
}
```

Code for the tracking page:

```
import wixWindow from 'wix-window';
import {local} from 'wix-storage';
import {details} from 'backend/gapi';
import {distance} from 'backend/gapi';


$w.onReady( function () {
  $w("placeholder").onReady( ()  => { //replace placeholder by place_id so that #googleMaps1 can use it as the location
        let currentItem = local.getItem("placeholder"); //replace placeholder by place_id so that #googleMaps1 can use it as the location
      $w("#googleMaps1").location = {
          "latitude": currentItem.latitude,
          "longitude": currentItem.longitude,
          "description": currentItem.title,
      };
  } );
} );


$w.onReady( function () {
  distance().then(function(resp) {
    console.log(resp);
```

```
    });


});
```

Code for the back-end:

```
import {fetch} from 'wix-fetch';


const key = "AIzaSyDAULCaG_C1zfK2-rvmz7lMHLggTybvKGI";


const apart1 = "https://maps.googleapis.com/maps/api/place/autocomplete/json?";
const apart2 = "&types=address&components=country:ca&key=";
export function autocomplete(string) {
    let input = "input=" + string;
    let url = apart1 + input + apart2 + key;
    return fetch (url, {method: 'get'}).then( (httpResponse) => {
        if (httpResponse.ok) {
            return httpResponse.json();
        }
    });
}


const dpart1 = "https://maps.googleapis.com/maps/api/place/details/json?";
```

```
const dpart2 = "&key=";
export function details(id) {
    let placeid = "placeid=" + id;
    let url = dpart1 + placeid + dpart2 + key;
    return fetch (url, {method: 'get'}).then( (httpResponse) => {
        if (httpResponse.ok) {
            return httpResponse.json();
        }
    });
}

const rpart1 = "https://maps.googleapis.com/maps/api/geocode/json?";
const rpart2 = "&key=";
export function reverse(lat, lng) {
    let latlng = "latlng=" + lat + "," + lng;
    let url = rpart1 + latlng + rpart2 + key;
    return fetch (url, {method: 'get'}).then( (httpResponse) => {
        if (httpResponse.ok) {
            return httpResponse.json();
        }
    });
}
```

```
export function distance() { // uses google's distance matrix api to find the distance between 2
set points
   const url =
"https://maps.googleapis.com/maps/api/distancematrix/json?origins=Vancouver+BC|Seattle&des
tinations=San+Francisco|Victoria+BC&" + key;
   return fetch (url, {method: 'get'}).then( (httpResponse) => {
      if (httpResponse.ok) {
         console.log("OK");
         return httpResponse.json();
      }
   })
}
```