ISSN 1870-4069

# Web Service for Inventory Control with Machine Learning and Interactive Maps with Leaflet on the Google Cloud Platform

Jaime A. García-Pulido[1], José M. Salomón-Montaño[1], Uziel H. López-Meneses[2], Pablo Yamamoto-Magaña[1], Raúl Morales-Salcedo[2], Yoel Ledo-Mezquita[1], Elioth Macias-Frotto[1]

[1] Tecnológico de Monterrey, School of Engineering and Sciences, Mexico

[2] NDS Cognitive Labs., Mexico

{a01652094, a01252989, a01733922, a01022382, yledo, a00837025}@tec.mx, raulms@ndscognitivelabs.com

**Abstract.** This article will show and detail the process that was followed for the development of a web page capable of tracking delivery vehicle routes using interactive maps using Leaflet technology, and an inventory control optimized with Machine Learning for the calculation of replenishment of inventory in distribution centers. Copyright © 2022 TEC de Monterrey y NDS Cognitive Labs.

**Keywords.** Machine Learning, Leaflet, React, Node.js, Javascript, HTML, CSS, Highcharts.

## 1 Introduction

Due to the growth of establishments and companies responsible for delivering products to these places for restocking, and due to the large population growth, there has arisen the need to create a system that companies can use to keep a record of their shipments, both past and present, to ensure that the products sent are arriving on time and in good condition to their respective destinations.

In this article, we aim to describe the procedure followed to create two screens of a web application that allow us to keep track of orders and their frequency with an interactive map; and a screen capable of displaying inventory information from each distribution center optimized with Machine Learning to calculate when a new order should be created to supply the distribution centers responsible for these products.

We also seek to develop these two screens in the Visual Studio text editor with the Node.js and React tools along with the JavaScript programming language. All these concepts will be explained in the next section.

## 2 Definitions and Acronyms

Before starting with the process and steps that were followed to achieve our results, it is essential to mention the different concepts, definitions, and technologies that will be used throughout the entire article. This is to make reading this article easier to understand by managing these concepts.

### 2.1 Frontend

Refers to the visual part of the website. It is what displays when showing all the visible elements the website has. Users can not only view these elements but also interact with them, whether by clicking on them, moving with the tab key, or using the mouse wheel to scroll through the page.

### 2.2 Backend

The backend refers to everything that happens behind the page. All the processing and/or logic that may take place. This can be seen, for example, when entering an account and its respective password, the backend takes care of registering that information and verifying if the provided data is correct to then redirect the user to the main page or their personal account.

### 2.3 Database

All the information that can be obtained/retrieved from the website is stored in a database, from the username, passwords, or the settings of a specific user. Another example would be a bank account and how much money are in a particular account and to whom it belongs.

### 2.4 Visual Studio

This is the text editor that helps us develop the frontend and backend of our website. It was the text editor chosen for this project.

### 2.5 API

An environment that allows two software elements to communicate with each other for the application to function correctly. An example of this would be the weather application found on a phone. It communicates with a database that contains all the meteorological data, and the API allows us to have those updates on our phone. In this way, this information can be reinterpreted so that the phone can understand and update it [1].

### 2.6 HTML and CSS

The language is responsible for the distribution/skeleton of the various elements that make up the website. For instance, this language helps us create buttons, menus, whether navigation or side menus, etc. On the other hand, the CSS language helps us style the website, from the background color, the type and size of the font, border color, structure of components, etc. [2].

### 2.7 Javascript

This programming language helps us give logic to the website; it helps the backend function. It can also act as a bridge between HTML and CSS when using an API that can generate these 2 languages from Javascript to facilitate our application's development.

### 2.8 Node.js and React Node

The programming environment that allows us to run all our Javascript programs in Visual Studio. Within these programs, we will be using React, an API that helps us generate HTML and CSS codes directly from Javascript and allows us to create components for our website that can be reused in different parts of it [3].

### 2.9 Leaflet

An open-source visual tool that helps us embed maps on our website. It works similarly to Google Maps, but we chose Leaflet because it is free, unlike the tool Google offers. In the table below, we can observe the most significant differences between the two tools, considering the features deemed most important for this project's development.

Even though Google Maps has more features and functions than Leaflet, as seen in Table 1, the decisive factor was the price, as this way, the project's budget is safeguarded.

### 2.10 Highcharts

For the parts of the page that need a graph to display our inventory system data, we used the Highcharts library. This library allows us to modify our graph's title, the data we want to show, the axis names, etc. This library also lets us export the graphs to different file formats, such as image formats, .CSV file extensions, or even print the graph if desired.

### 2.11 Machine Learning

This term refers to one of the branches of artificial intelligence. This concept allows us to program computers to 'learn' to classify data and scenarios based on past experiences or information collected over time [4].

**Table 1.** Comparison between Google Maps and Leaflet.

| **Differences between APIs** | | |
|---|---|---|
| | Google Maps | Leaflet |
| Price | MXN $416 | Free |
| Markers | Yes | Yes. |
| Route | Yes | Yes, with an additional library. |
| Heat maps | Yes | Yes, with an additional component. |
| Documentation | Extensive | Moderate |

### 2.12 Google Cloud Platform

Finally, we have Google's cloud platform. This platform makes it easier for us to place our website on a secure domain provided by Google. Also, if we need any of its various APIs, Google's platform allows us easy access, but for now, we will only use it to host our website.

## 3 Development

At the beginning of the project, the first thing decided was the interactive map where the user can filter the cities or establishments they want to see on the map at any time. Due to the vast number of places in our data, from Boston to Rio de Janeiro, it is quite challenging to discern one establishment from another when having a global view on the map.

To solve this, we decided to use a heat map on the entire map to visualize the frequency with which each establishment appears in our data. In the heat map, a warm color, for example, red, indicates a more frequent presence of locations on the map, while a cooler color, like blue, indicates that there isn't as much frequency of that particular establishment, or it only appears once in our data.

All of the above was created using Leaflet as a tool for the interactive map, while for the development environment, Visual Studio was used as a text editor, and Node.js and React as APIs for the entire page and its components.

For this, 4 drop-down menus were created where the type of delivery that each establishment had, the type of vehicle used for each delivery, the product of each delivery for each establishment, and the establishment itself, filtering by their respective names, can be filtered. In this way, only what the user wants can be displayed on the map. This works best with the markers developed for this part of the project, so

we have two ways to view the map. One would be with the heat map to visualize the frequency, and the other would be with individual markers to show every establishment.

Having accomplished the above, another screen was developed where inventory calculation is performed. This part of the website is responsible for keeping a record of the total inventory that the distribution centers have.

For this page, a dropdown menu was developed that allows the selection of a product from a particular inventory, this being the inventory of each distribution center. Hence, the person in charge of the inventory can only view the inventory assigned to them and not the inventory of another distribution center.

Therefore, only the current inventory being viewed appears on the page and cannot be modified. Once the product is selected, the user can view one of four charts available on the page. The default chart is the EOQ chart, which shows how much of a particular product remains in inventory. The other three charts display the quantity delivered with each restocking order, the price of each order over time, and the frequency of the orders. These charts can be seen in Figs. 1 to 4.

To change the desired chart for analysis, there are four buttons on the left side of the screen that display which chart is shown when clicking on one of these buttons. Additionally, if the user wishes to download the chart for further analysis, they can download the chart in image format or data tables to use them in another application or analysis.

There is also a button at the bottom of the screen for the user to place an order to restock the distribution center when the inventory chart indicates that an order is needed. For this part, the backend is equipped with Machine Learning to predict when is the best time to order more inventory.

In this case, the variables to be used for this prediction would be the time when things should be ordered, in other words, when the inventory is lower than a certain limit, and how long it takes the supplier to deliver the necessary products so that the inventory exceeds the previously mentioned limit.

## 4 Implementation

As previously mentioned, the text editor used for the development of the website was Visual Studio, and Node.js was used to utilize the React API [5]. For the map part, the dropdown menus were first developed, where the user can select the filters, they want to apply to the map. To get the different order options for the filters, data was obtained through a JSON file containing the geolocation data of each store, the type of order and product received, as well as the vehicle used for the delivery. Once this was achieved, it was verified that the JSON could be accessed correctly to proceed with the implementation of the interactive map.

### 4.1 Interactive Map with Leaflet

As previously mentioned, Leaflet was used as a tool for the map. To insert the map on the website, the component was first developed separately to verify the functionality of the tool and understand what features we could implement. In this case, markers and heat maps offered by the tool were decided to be used.

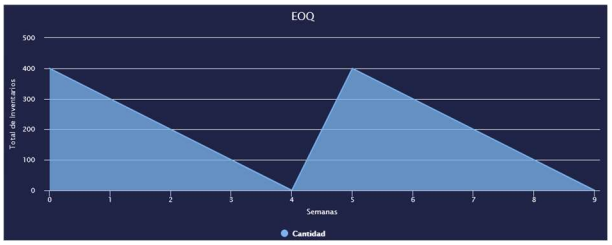


**Fig. 1.** EOQ Chart with Highcharts.

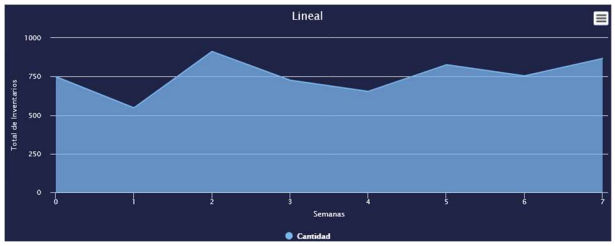


**Fig. 2.** Time series chart of inventories with Highcharts.

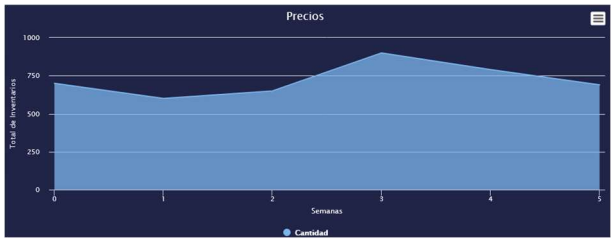


**Fig. 3.** Time series chart of prices with Highcharts.

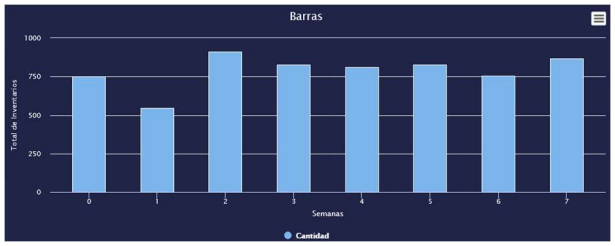


**Fig. 4.** Bar chart of inventories with Highcharts.

When implementing the map, the initial coordinates at which the map will position, when the map loads must first be provided to the tool, along with a number from 0 to 20 indicating the zoom level the map will have over the initial coordinates.

Then, for the implementation of the markers, it was only necessary to provide the longitude and latitude coordinates of each establishment, and if desired, a text could be added indicating the name of the establishment and the vehicle used in that delivery.

On the other hand, for the heat maps, it was necessary to provide the coordinates of the establishments, in the same way as the markers, and add them to the map using

Leaflet's heat Layer function and the leaflet. Heat component, which are responsible for loading these geographical points on the map to create heat maps. In this way, both functionalities in Leaflet can be implemented.

To apply the necessary filters to the map, it was only necessary to filter by name or type of delivery within the JSON and then display the location points that match the applied filters. If there's a match, it's displayed on the map; otherwise, the map remains empty.

### 4.2 Inventory Control with Machine Learning

Finally, for inventory control, the dropdown menu was first developed to select the product whose inventory is to be viewed at the moment. Then, the chart was implemented where the different inventory data can be viewed, for example, with an EOQ chart. Next, the buttons were created with which the type of chart to be viewed in the previously developed chart section can be selected. Lastly, a button was added that, as mentioned in section 3 of this document, will allow an order to be placed to restock the distribution center with the desired product.

For the chart part, the Highcharts library was used [6], which allows us to easily modify many of its functions. In this case, area charts were used for the EOQ chart, a time series chart for costs, a bar and time series chart indicating the quantity delivered with each order, and another time series chart showing how often orders have been made.

Thanks to the Highcharts library, it is easy to modify specific parts of each chart according to the needs of what is to be shown. In this case, mainly, the titles of the charts were modified, as well as the names of the x-axis and the y-axis, the names of our data, etc. This library also allows us to export any chart we want, either to an image, export it to a PDF file, or it can also be exported as a .CSV file, which allows us to reuse these data elsewhere, for example, for the analysis of orders, prices, among other things.

Finally, to modify any of these chart properties, an object must be created that has all the specifications and features for our charts.Once that is done, we simply have to specify which feature object we are going to use for each chart when creating the Highcharts component, in this way we can reuse code and decide when to show each of the charts.

When integrating this screen with the rest of the website, a request must be made to a file that contains the code responsible for the Machine Learning analysis to then show the request results on this screen so that the user can decide whether to place the order based on the results of the charts.

## 5 Results

Once the above development was achieved, it was possible to create two responsive screens depending on what the user desires. It was also possible to create each of the components described in this document with the tools mentioned in the previous sections. The following images show some sections of the screens that were created to create the inventory system website.


**Fig. 5.** Final screen of the interactive map with Leaflet.


**Fig. 6.** Final screen of the inventory system.

As shown in Figs. 1 and 2, you can see how the screens of both applications run, taking into account the CSS styles assigned to each of the screens. In Fig. 1, we can observe the interactive map with Leaflet using an example of a heat map showing the frequency with which some of the establishments are reached for deliveries. In Fig. 2, you can see the buttons and the dropdown menu implemented to change the charts you want to view.

## 6 Conclusions

Thanks to the procedure that was followed, it was possible to develop the two screens and their functionalities correctly. Initially, it was a challenge and confusing to learn a new API and a completely new development environment. Fortunately, there is a lot of documentation regarding these two technologies, and in the end, if there is any doubt about how to make certain implementations or functions in an application, there is a solid documentation base where a possible solution can be sought.

Regarding the development of the screens, thanks to the technologies learned to generate these applications, it was possible to implement the necessary functionalities for the screens to function correctly.

For the interactive map part, it was a challenge to embed the map on the page and have everything work correctly on the user's side, this being the logical part of that page. Plus, it was a challenge to learn how the Leaflet tool worked to take full advantage of its potential. Finally, for the inventory control part, the High charts tool was learned

to implement each of the necessary charts so that the user can interpret the data displayed on the screen, and so that the user can download or export those charts and can handle this data more easily on their own. Fortunately, thanks to all the previous learning acquired when developing the interactive map, it was beneficial to develop this part of the web application.

The choice of such tools (leaflet and High charts) allows for efficiency and effectiveness, enabling companies to remain competitive. This study lays the groundwork for future research and developments on the integration of these technologies into logistical and inventory systems.

## References

1. How to create a public API with AWS. https://aws.amazon.com/es/what-is/api/
2. NA: Hypertext Tag Language | MDN. https://developer.mozilla.org/es/docs/Web/HTML (2022)
3. Leaflet: An open-source JavaScript library for interactive maps. https://leafletjs.com/ (2022)
4. Alameda, T.: Machine Learning: What is it and how does it work? BBVA NEWS, https://www.bbva.com/es/machine-learning-que-es-y-como-funciona/ (2022)
5. Jones, M.: Creating a Node.js and React application - Visual Studio (Windows). https://learn.microsoft.com/es-es/visualstudio/javascript/tutorial-nodejs-with react-and-jsx?view=vs-2022 (2022)
6. Jhenrryalvaro: HighCharts: Library for creating charts. Accessed: Available: https://enboliviacom.wordpress.com/2013/03/01/highcharts-libreria-para-creacion-de-graficos/ (2022)