

TECNOLÓGICO DE ESTUDIOS SUPERIORES DE CHALCO

TESCHA

TECNOLÓGICO DE ESTUDIOS SUPERIORES

CHALCO

ALUMNO: AMAYA DELGADO BRYAN

SEMESTRE: NOVENO **GRUPO:** 4953 **CICLO ESCOLAR:** 2022-2

CARRERA: INGENIERIA EN SISTEMAS COMPUTACIONALES

ASIGNATURA: DESARROLLO DE APLICACIONES PARA EL COMERCIO ELECTRÓNICO

PROFESOR: IVAN AZAMAR PALMA

TITULO: PRACTÍCAS DE LABORATORIO.

TEMA: 3

FECHA DE ENTREGA: 30 DE NOVIEMBRE DEL 2022

LIGA DE ACCESO AL REPOSITORIO:

<https://github.com/Bryan-Amaya-Delgado/Practicas-Unidad-3-4953.git>

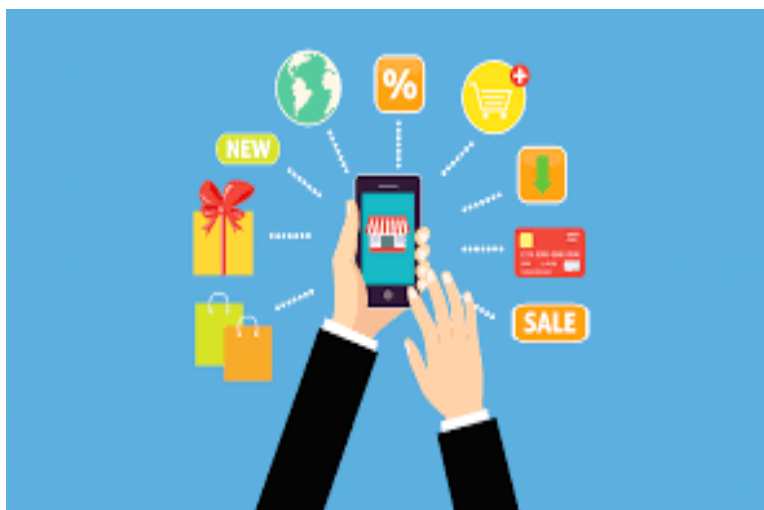


Ilustración 1 Aplicaciones Web

ÍNDICE

MARCO TEÓRICO	3
Definición de librería.....	3
Ventajas De Un Librería.	3
Desventajas De Las Librerías.....	4
¿Qué es una librería?.....	4
Ejemplos de Librerías.	5
Tipos De Librerías En Programación.....	5
DESARROLLO-PRACTICO	5
Practica 2 – Aplicaciones Para El Lado Del Servidor (Api`S Dependencias Y Librerías)	6
Practica 3 – Aplicaciones Para El Lado Del Servidor (Conexión BD y Creando un Servidor).....	7
Practica 4 – Aplicaciones Para El Lado Del Servidor.....	8
Practica 5 – APLICACIONES PARA EL LADO DEL SERVIDOR	9
CONCLUSIONES.....	12
REFERENCIAS BIBLIOGRÁFICAS.....	13

Tabla de ilustraciones

Ilustración 1 Aplicaciones Web	1
Ilustración 2. Diferentes Librerías Y Frameworks.....	3

MARCO TEÓRICO

Definición de librería.



Una Librería es básicamente un componente de software que expone funcionalidades, pero no son ejecutables. En líneas generales, estas funcionalidades son usadas por los programas (ejecutables) que tienen acceso a esa biblioteca.

Pueden ser estáticas o dinámicas, dependiendo de si están compiladas dentro o fuera del ejecutable. En este último caso, las verás como ficheros dentro del file system.

Ilustración 2. Diferentes Librerías Y Frameworks.

Ventajas De Un Librería.

Entre las ventajas de utilizar una librería para el desarrollo de software distinguimos:

- **Estructura y organización del código predeterminada.** Las Librerías proporcionan tanto un esqueleto como una forma de trabajar. Por lo tanto, evitan tener que realizar un análisis sobre dónde situar los diferentes archivos de la aplicación (recursos, controladores, vistas, modelos, etc.).
- **Reutilización del código. Evitar duplicidad de código.** En el desarrollo de una aplicación existen ciertos apartados que suelen repetirse, como la conexión con la base de datos, validación de formularios, páginas de estilos, etc. Con la utilización de una Librería ahorraremos tiempo en desarrollar funcionalidades que ya están cubiertas y podremos enfocarnos en el funcionamiento de la aplicación más que en cómo llevarla a cabo.
- **Agilidad y rapidez en el desarrollo.** Precisamente gracias a la reutilización de código mencionada anteriormente, conseguimos mayor rapidez en el desarrollo, ya que no perderemos tiempo en desarrollar funcionalidades nuevas.
- **Menor coste en el desarrollo.** El coste es un parámetro que está directamente relacionado con la rapidez y agilidad. Acabar antes un proyecto implica que la dedicación es menor y por lo tanto el coste del proyecto también disminuye.
- **Al igual que la rapidez en el desarrollo,** esta ventaja beneficia tanto al cliente como al desarrollador.
- **Buenas prácticas de desarrollo con el uso de patrones.** La mayoría de frameworks y Librerías están basados en patrones de diseños, que nos indican pautas sobre cómo solucionar un problema específico que ya ha ocurrido con anterioridad. El patrón de

diseño más popular es MVC (Modelo-Vista-Controlador), que nos ayuda a separar la capa de datos de la lógica del negocio de la interfaz con el usuario.

- **Minimizar errores y mayor facilidad para solucionarlos.** Como la Librería ya incorpora código implementado por otros programadores, los posibles errores que este pueda tener siempre serán menores que al desarrollarlo desde cero. Además, en caso de que hubiera un error, lo más probable es que ya haya sido solucionado por la comunidad.
- **Facilidad a la hora de encontrar una librería o código que ya cubra funcionalidades de tu desarrollo.** Resulta más fácil encontrar herramientas (utilidades, librerías) adaptadas al framework que para un desarrollo propio.
- **Facilita la colaboración con otros desarrolladores.** Tanto si son compañeros de tu equipo como de GitHub, leer el código desarrollado por otra persona puede resultar complejo. Sin embargo, si ya sabes qué estructura va a seguir el código y cómo se organiza, resultará más fácil comprenderlo y poder aplicarle nuevos cambios. Lo que nos lleva a la siguiente ventaja.
- **Facilita el mantenimiento.** Si todos los miembros de un equipo trabajan de la misma forma, en el momento que haya que actualizar la aplicación o realizar algún evolutivo, tardaremos menos tiempo y el coste será menor.

Desventajas De Las Librerías.

❖ **Tiempo de aprendizaje.** Antes de empezar a utilizar un framework debemos familiarizarnos con él, con cómo se estructuran sus archivos, con la forma en la que se comunican los componentes, etc.

Por lo tanto, tendremos que invertir tiempo en superar la curva de aprendizaje para poder comenzar un nuevo desarrollo utilizando el framework.

❖ **Versiones inestables.** El hecho de que los frameworks sean tan populares provoca que estén en constante actualización para cumplir con las nuevas tecnologías y políticas de seguridad.

Por ello, si en el desarrollo surgen incompatibilidades con otras librerías o se detectan errores de seguridad, la elección de una versión muy reciente del framework podría ralentizarnos.

❖ **Menor rendimiento.** Los frameworks consumen, en general, más recursos que una aplicación creada desde cero y orientada al rendimiento. En aplicaciones muy exigentes, un framework puede resultar poco apropiado.

❖ **Código sin utilizar.** Si la aplicación es pequeña o no requiere mucha funcionalidad, probablemente estaremos desaprovechando

¿Qué es una librería?

Una librería es uno o varios archivos escritos en un lenguaje de programación determinado, que proporcionan diversas funcionalidades.

A diferencia de un framework, una librería no aporta la estructura sobre cómo realizar el desarrollo, sino que proporciona funcionalidades comunes, que ya han sido resueltas previamente por otros programadores y evitan la duplicidad de código. Además, reducen el tiempo de desarrollo y aumentan la calidad de este.

Ejemplos de Librerías.

Estos son algunos de los frameworks más conocidos:

- ✓ **Javascript:** JQuery, Mootools, Moment.js, Anime.js, Ramda, D3.js, Chart.js, Mathjs, Hammer.js, Glimmer.js, Etc.
- ✓ **C++:** Iostream, Cmath, Cstring, Ctime, Algorithm, Etc.
- ✓ **Python:** Matplotlib, Seaborn, Bokeh, Numpy, Scipy, Pandas, Numba, Gensim, Etc.

Tipos De Librerías En Programación

Hay, en términos generales, dos tipos de librerías en programación. Las librerías pueden ser caseras o externas, en función de si están creadas por ti o por otros programadores, pero la principal distinción es entre librerías estáticas y librerías dinámicas.

1.- Librerías Estáticas

Estas se graban en un programa como ejecutables. Sirven exclusivamente para esto; después, podemos borrarlas sin problemas, ya que el programa seguirá funcionando con la función necesaria.

2.- Librerías Dinámicas

Son distintas a las estáticas en tanto en cuanto no se copian en el programa al compilarlas. Las subrutinas son cargadas en tiempo de ejecución, en vez de enlazarse en tiempo de compilación.

3.- Librerías Remotas:

Esta es una biblioteca que utiliza ejecutables separados y por medio de un procedimiento remoto, llevado a cabo por otra computadora. Estas bibliotecas trabajan con un enfoque que maximiza la reutilización del sistema operativo; el código que soporta la biblioteca es el que le da a la aplicación el soporte y seguridad para otro programa.

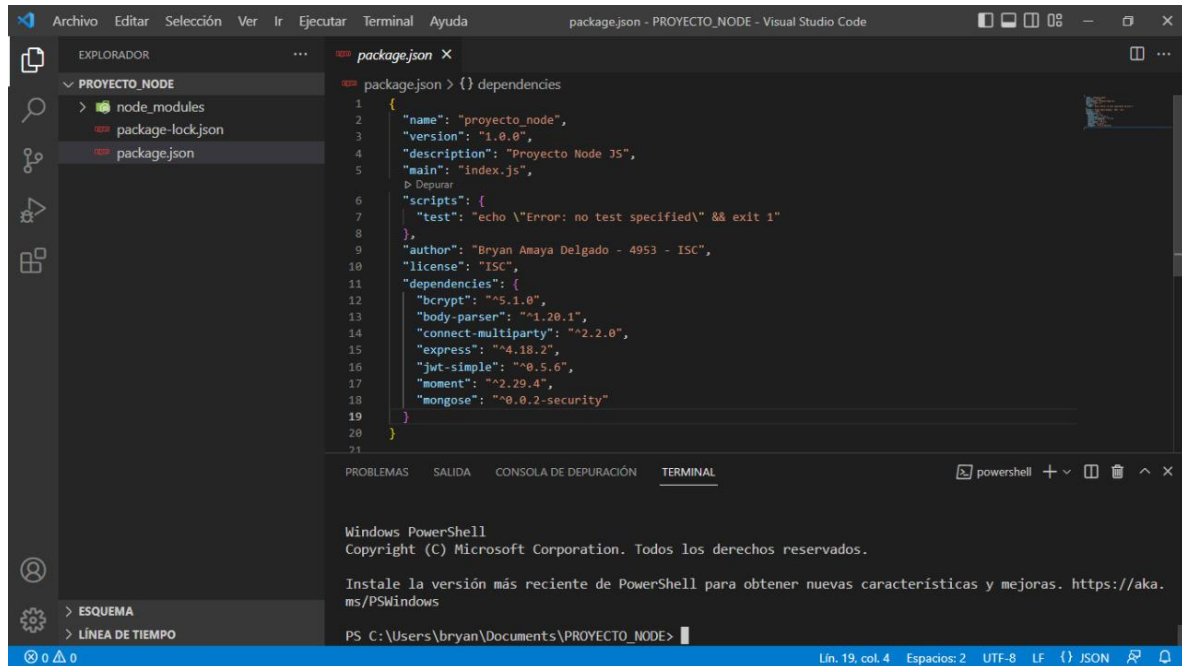
DESARROLLO-PRACTICO

En esta parte solamente se mostrarán los resultados de la práctica.

Practica 2 – Aplicaciones Para El Lado Del Servidor (Api`S Dependencias Y Librerias)

RESULTADOS

1.- Instalación de las librerías.



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows the project structure with 'PROYECTO_NODE' containing 'node_modules', 'package-lock.json', and 'package.json'. The main editor displays the 'package.json' file with the following content:

```
1 {
2   "name": "proyecto_node",
3   "version": "1.0.0",
4   "description": "Proyecto Node JS",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "author": "Bryan Amaya Delgado - 4953 - ISC",
10  "license": "ISC",
11  "dependencies": {
12    "bcrypt": "^5.1.0",
13    "body-parser": "^1.20.1",
14    "connect-multiparty": "^2.2.0",
15    "express": "^4.18.2",
16    "jwt-simple": "^0.5.6",
17    "moment": "^2.29.4",
18    "mongoose": "^0.0.2-security"
19  }
20 }
```

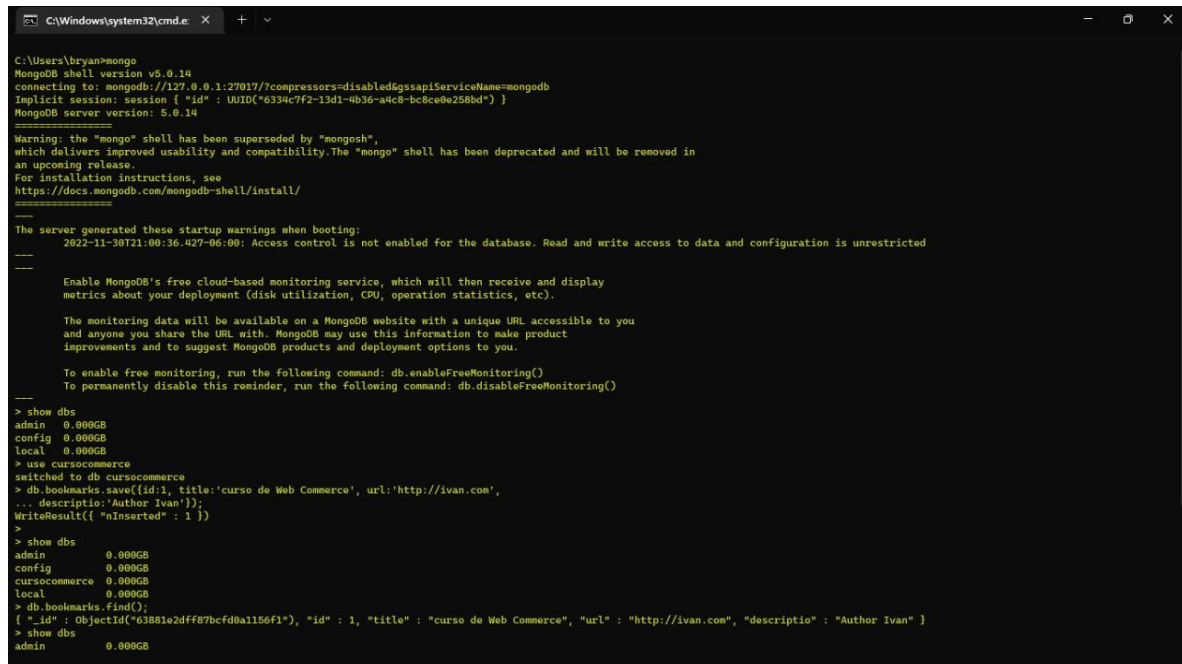
The bottom pane shows the PowerShell terminal with the following output:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\bryan\Documents\PROYECTO_NODE>
```

2.- Conexión a La Base De Datos Local (MongoDB).



The screenshot shows a Windows command prompt window with the following output:

```
C:\Windows\system32\cmd.exe
C:\Users\bryan>mongo
MongoDB shell version v5.0.14
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("6334c7f2-13d1-4b36-a4c8-bc8ce0e258bd") }
MongoDB server version: 5.0.14

Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility. The "mongo" shell has been deprecated and will be removed in
an upcoming release.
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/

-----
The server generated these startup warnings when booting:
  2022-11-30T21:00:36.427-06:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()

-----
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
> use cursocommerce
switched to db cursocommerce
> db.bookmarks.save({id:1, title:'curso de Web Commerce', url:'http://ivan.com',
... descriptio:'Author Ivan'});
WriteResult({ "nInserted" : 1 })
>
> show dbs
admin      0.000GB
config     0.000GB
cursocommerce 0.000GB
local      0.000GB
> db.bookmarks.find();
{ "_id" : ObjectId("63b81e2dff87b9cf0a1156f1"), "id" : 1, "title" : "curso de Web Commerce", "url" : "http://ivan.com", "descriptio" : "Author Ivan" }
> show dbs
admin      0.000GB
```


Practica 4 – Aplicaciones Para El Lado Del Servidor.

RESULTADOS.

1.- Verificamos que el mensaje Se imprima Exitosamente.

The screenshot shows a web browser window with the address bar displaying `localhost:3977/api/probando-controlador`. The page content shows a JSON response: `{\"mensaje\": \"Probando una accion del controlador de usuarios del api REST con node y mongo\"}`.

2.- Envío de Datos.

The screenshot shows a REST client interface with a POST request to `http://localhost:3977/api/registro`. The request body is a JSON object:

```

{
  "usuarios": {
    "_id": "63885198e4e84d8d61764b",
    "nombre": "Bryan",
    "apellido": "Amaya",
    "email": "bryan_ad@tesch.edu.mx",
    "password": "$2b$10$anEm0NH6sxdm.Yv4e36uipNbw8M1KzGpK90vwj.j9Quwaf8jhZS"
  }
}

```


Practica 5 – APLICACIONES PARA EL LADO DEL SERVIDOR

RESULTADOS.

1.- Enviamos los datos por el método post.

POST Metodoget_Autorizac Apis No Environment

Practicas4853 / Metodoget_Autorizacion.php

POST http://localhost:3977/api/registro

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

nombre	Bryan
apellido	Amaya
email	bryan_ad@tesch.edu.mx
password	zeroblazer

Body Cookies Headers (7) Test Results Status: 200 OK Time: 503 ms Size: 467 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "usuarios": {
3     "_id": "638851988e4e84d80d61764b",
4     "nombre": "Bryan",
5     "apellido": "Amaya",
6     "email": "bryan_ad@tesch.edu.mx",
7     "password": "$2b$10$SanEm0NH6sxdm.Yv4e36uipNbm8M1KzGpK90vWj.f9QuWaf8jhZS",
  
```

POST Metodoget_Autorizac Apis No Environment

Practicas4853 / Metodoget_Autorizacion.php

POST http://localhost:3977/api/login

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

nombre	Tatiana
apellido	Amaya Delgado
email	tatiana_sp@tesch.edu.mx
password	Corazon

Body Cookies Headers (7) Test Results Status: 200 OK Time: 222 ms Size: 475 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "user": {
3     "_id": "638851988e4e84d80d61764b",
4     "nombre": "Tatiana",
5     "apellido": "Amaya Delgado",
6     "email": "tatiana_sp@tesch.edu.mx",
7     "password": "$2b$10$8XJatCRBlcoXf0Dzfd.tp0F1MtgwL7slz3dpk9pFG8Qm/PlX8NNRm",
  
```

2.- Ejecución del Método Login.

The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** http://localhost:3977/api/login
- Body Type:** x-www-form-urlencoded
- Parameters:**

nombre	Tatiana
apellido	Amaya Delgado
email	tatiana_sp@tesch.edu.mx
password	Corazon
- Status:** 200 OK, Time: 222 ms, Size: 475 B
- Response Body (JSON):**

```

1 {
2   "user": {
3     "_id": "638851988e4e04d80d61764b",
4     "nombre": "Tatiana",
5     "apellido": "Amaya Delgado",
6     "email": "tatiana_sp@tesch.edu.mx",
7     "password": "$2b$10$8XJatCRBlcoXf0Dzfd.tp0FlMtgeL7slz3dpk9pF68Qm/PlX8NNRm",
8     "rol": "ROLE_USER"
9   }
10 }

```

3.- Autenticación a través de consola.

The screenshot shows a code editor with a JavaScript file named `usuarioControl.js`. The function `registrarUsuario` is defined as follows:

```

11 }
12
13 function registrarUsuario(req, res) {
14   var params = req.body; //recibe todos los datos por el metodo post
15   console.log(params);
16
17   usuario.nombre = params.nombre;
18   // usuario.nombre = params.nombre;
19   usuario.apellido = params.apellido;
20   usuario.email = params.email;
21   usuario.password = params.password;
22   usuario.rol = 'ROLE_USER';
23   usuario.imagen = 'null';
24
25   if (params.password) {

```

The console output shows the following messages:

```

Conexion Exitosa
El servidor api rest de musica escuchando en http://localhost:3977
[Object: null prototype] {
  nombre: 'Bryan',
  apellido: 'Amaya',
  email: 'bryan_ad@tesch.edu.mx',
  password: 'zeroblazer'
}
[Object: null prototype] {
  nombre: 'Tatiana',
  apellido: 'Amaya Delgado',
  email: 'tatiana_sp@tesch.edu.mx',
  password: 'Corazon'
}
Coincide el password

```

4.- Fallo de autenticación.

The screenshot shows a REST client interface with a POST request to `http://localhost:3977/api/login`. The request body is form-urlencoded with the following data:

Field	Value
nombre	Tatiana
apellido	Amaya Delgado
email	tatiana_sp@tesch.edu.mx
password	Corazon

The response status is 404 Not Found. The response body in JSON format is:

```
{  "message": "El usuario no existe"}
```

CONCLUSIONES

En la siguiente actividad perteneciente a la Asignatura de Desarrollo de Aplicaciones para el Comercio Electrónico, fue un trabajo muy interesante debido al gran análisis que se realizó sobre las actividades propuestas Del Lado del Servidor. En la cual se habla sobre cómo ha avanzado el procesamiento de datos a través de la programación de software y su implementación en los servicios de almacenamiento y generación de aplicaciones a través de la implantación de código puro enfocado al área de programación web.

A su vez cabe resaltar La web no es un ente individual como nos gusta pensar, se trata un conjunto de archivos variados entrelazados entre sí a los cuales podemos acceder por medio de una interfaz virtual que conecta dichos archivos con los usuarios.

Por otro punto debemos de saber que, desde la perspectiva del usuario, utilizando el navegador, la persona encuentra una bonita página web de uso intuitivo que le muestra la información ordenada y fácil de leer que más tarde podrá comentar o compartir en sus redes.

Por otra parte, desde la perspectiva del programador el esquema es totalmente distinto y sumamente complejo. Son ellos los que deben organizar a través de lenguajes de programación tales como HTML o CSS un conjunto de reglas de comportamiento que incluyen los más mínimos detalles para que el usuario pueda navegar con total comodidad sin conocer nada de lo que se esconde detrás.

Como pudimos notar en el trabajo que realizamos, nos da una detallada información sobre el uso de frameworks, los cuales nos permitirán darle más funcionalidades a los sistemas web que estemos desarrollando en un determinado momento.

Ya que gracias a esto nos da la pauta para que empecemos a crear y diseñar nuestras propias aplicaciones, los pasos que requerimos para hacerlo y darle una solución correcta al problema que queremos trasladar a la nube y después enlazarlo ala BD, y para hacerlo requerimos de las formas normales, las cuales hacen que nuestra base de datos quede con una buena lógica, ya que, sin eso, sería un caos y no resolvería nada.

Otro punto importante fue el saber que JS nos permite darle “movimiento” y dinamismo a la web. Es un lenguaje de programación del lado del cliente, es decir, se ejecuta en el navegador, no el servidor, lo que nos permite hacer acciones más rápidas.

Gracias a esto es importante decir que es vital para cualquier negocio, pues nos ayuda a encontrar información potencial para usarla para mejorar tu empresa. es un instrumento muy interesante para predecir el ritmo de consumo de los clientes. Si, con nuestro análisis, logramos establecer qué recurrencia tiene el cliente para comprar de nuevo nuestro producto o servicio, podremos ajustar la producción.

REFERENCIAS BIBLIOGRÁFICAS

1. Alondra Lara. (22 de Noviembre de 2013). *Programador Clic*. Obtenido de <https://programmerclick.com/article/29141531099/>
2. Aritnetrics. (s.f.). *Developers Aritmetricks*. Obtenido de <https://www.aritmetrics.com/glosa>
3. Desconocido . (25 de Mayo de 2014). *Thithink*. Obtenido de <https://www.tithink.com/es/2018/08/29/framework-o-librerias-ventajas-y-desventajas/>
4. Desconocido . (28 de Diciembre de 2020). *Girhub*. Obtenido de <https://gist.github.com/BCasal/026e4c7f5c71418485c1>
5. Desconocido. (08 de Marzo de 2014). *Courters - Que es un Yarg*. Obtenido de <https://www.geeksforgeeks.org/node-js-yargs-module/>
6. Desconocido. (03 de Septiembre de 2016). *Mind Web*. Obtenido de https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Statements/async_function#:~:text=Una%20función%20async%20puede%20contener,y%20devuelve%20el%20valor%20resuelto.
7. Desconocido. (s.f.). *Persistencia Web* . Obtenido de https://agpele92.es.tl/2-.4_-PERSISTENCIA.htm