# Notas

Explicación de ABC: <https://qualitymant.com/como-se-realiza-un-analisis-abc/>

En la empresa se requiere un módulo de ventas para clientes exclusivos, para ello se debe desarrollar un sistema para trabajar procesos de ventas para clientes.

Para la parte de clientes se necesita un ABC para el mantenimiento de clientes, considere los **campos para cliente**. Para ventas que se puedan seleccionar de 1 a n productos, se calcule su precio de venta total. Se debe de mostrar el número de venta al completarse la misma. Como regla de negocio se tiene que seleccionar cliente para la venta.

Manejar de 10 **registros** para clientes y **productos**, para ventas que sean 3 mínimo. (Carga de datos)

Para esto debe de trabajarse de la siguiente manera:

- Realizar análisis de base de datos y presentar diagrama entidad/relación.

- Implementar base de datos en MySQL o SQL Server.

- Utilizar procedimientos almacenados o funciones en base de datos para insert y update.

- Desarrollar Backend API Restful (json) en estas posibles opciones NodeJS, Java o C# para la integración en base de datos y frontend.

- Desarrollar Frontend como Aplicación Web en estas posibles opciones Angular, React o ASP.NET.

Al terminar el examen deberá explicar el desarrollo trabajado.

Vistas

Port 3000

Request

Response

HTTP

User

Request

Response

API

Port 8000

React

Fetch

SERVER

Node

Oracle

CRUD sobre Ventas, productos y clientes

HOME Page

Ver usuarios

Crear Tarea

Filter: | byDate | byUser | byPriority. Search

TODO LIST APP

Tareas

Tarea1

Tarea2

Tarea3

….

API

ENDPOINT: <http://localhost:8000/>unicomerAPI/

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Methodo | urlPath | | | | |  |
| cliente/ | cliente/:idUser | producto/ | producto/:idproducto | venta/ | Venta/:IdVenta |
| GET | **Devuleve todos los clientes** | **Devuelve el cliente del id** | **Devuleve todos los productos** | **Devuelve un producto** | **Devuelve todas las ventas** | **Devuelva una venta, del id** |
| POST | **Añade un cliente nuevo** | -------- | **Añade un producto** | --------- | Añade una venta | ------------- |
| PATCH | -------- | **Actualiza al cliente con la información enviada** | -------- | **Actualiza un producto** | ---------------- | Actualiza una venta |
| DELETE | ~~Elimina todos los clientes~~ | ~~Eliminar al cliente~~ | Elimina todos los productos | Elimina un producto | Elimina todas las ventas | Elimina la venta específica |

Response structure:

{

httpStatus: statusCode,

message: “ServerMessage”,

data: [

{“dataRequested1”},

{“dataRequested2”}, …

{“dataRequested3”},

]

}

statusCode definitions

200 = Todo está bien

300 = bad request | request erróneo

400 = server error

Tabla de Usuario

CREATE TABLE clientes(

**id** INT AUTO\_INCREMENT PRIMARY KEY,

**nombres** VARCHAR(50) NOT NULL,

**apellidos** VARCHAR(50) NOT NULL,

**direccion** VARCHAR(100),

**telefono** VARCHAR(15),

**email** VARCHAR(100)

);

CREATE TABLE productos (

**id** INT AUTO\_INCREMENT PRIMARY KEY,

**nombre** VARCHAR(100) NOT NULL,

**descripcion** TEXT,

**precio** DECIMAL(10, 2) NOT NULL

);

CREATE TABLE ventas (

**id** INT AUTO\_INCREMENT PRIMARY KEY,

**fecha** DATE NOT NULL,

**cliente\_id** INT,

);

CREATE TABLE ventasProductos (

**id** INT AUTO\_INCREMENT PRIMARY KEY,

**venta\_id** FOREIGN KEY REFERENCES ventas,

**producto\_id** FOREIGN KEY REFERENCES producto,

);