Logotipo

Descripción generada automáticamente

2022



Iteración 1: Plan de mejora individual

Nicolás ignacio cortez manzano

docente: Mauricio figueroa

Contenido

[1. Introducción 2](#_Toc113227769)

[2. Bases de datos NoSQL y MongoDB 2](#_Toc113227770)

[2.1 Descripción y comparación con SQL 2](#_Toc113227771)

[2.2 Bases de Datos de Documentos 3](#_Toc113227772)

[2.3 Gestores de Bases de datos NoSQL 4](#_Toc113227773)

[2.4 Ventajas y desventajas 4](#_Toc113227774)

[2.4 Introducción a MongoDB 5](#_Toc113227775)

[2.5 MongoDB Cheat Sheet 6](#_Toc113227776)

[2.6 Conclusiones MongoDB 7](#_Toc113227777)

[3. Flask 8](#_Toc113227778)

[3.1 Descripción 8](#_Toc113227779)

[3.2 Ventajas 8](#_Toc113227780)

[3.3 Instalaciones 8](#_Toc113227781)

[3.4 Primeros Pasos 9](#_Toc113227782)

[3.5 Aplicación Flask 10](#_Toc113227783)

[3.5 Endpoints API 13](#_Toc113227784)

[4. Aplicación Vuejs 14](#_Toc113227785)

[4.1 Introducción 14](#_Toc113227786)

[4.2 Instalaciones 14](#_Toc113227787)

[4.3 Codificando 15](#_Toc113227788)

[4.5 Vistas 16](#_Toc113227789)

[5. Conclusiones 17](#_Toc113227790)

[6. Referencias 18](#_Toc113227791)

# 1. Introducción

El propósito de este documento es mostrar el avance realizado en la primera iteración respecto a las debilidades presentadas junto a los respectivos planes de mejoras agregados. En este documento se evidenciará el aprendizaje e incursión en un STACK tecnológico que DuocUC no cubre en la malla académica actual. Se mostrarán las evidencias y se adjuntarán el repositorio de trabajo para este avance. Los temas y tecnologías para desarrollar son: Mongo Compass, Flask y Vue.

# 2. Bases de datos NoSQL y MongoDB

## 2.1 Descripción y comparación con SQL

* Son estructuras de almacenamiento de información que no cumplen con el esquema entidad-relación.
* Tampoco utilizan una estructura de datos en forma de tablas.
* No utilizan el lenguaje SQL como lenguaje de consultas.
* No realizan operaciones tipo Join.

A continuación, se muestra una tabla comparativa de conceptos y sus similares frente a datos relacionales.

|  |  |
| --- | --- |
| Relacional | NoSQL |
| Database | Database |
| Table | Collection |
| Row | Document |
| Index | Index |
| Join | $lookup |
| Foreing Key | Reference |

## 2.2 Bases de Datos de Documentos

Las bases de datos de Documentos (NoSQL) almacenan los datos en formato JSON, esto está basado en un sistema tipo Key:Value. Se considera que es la forma más natural de pensar en los datos,siendo más expresivo y poderoso que el esquema tradicional de filas y columnas.

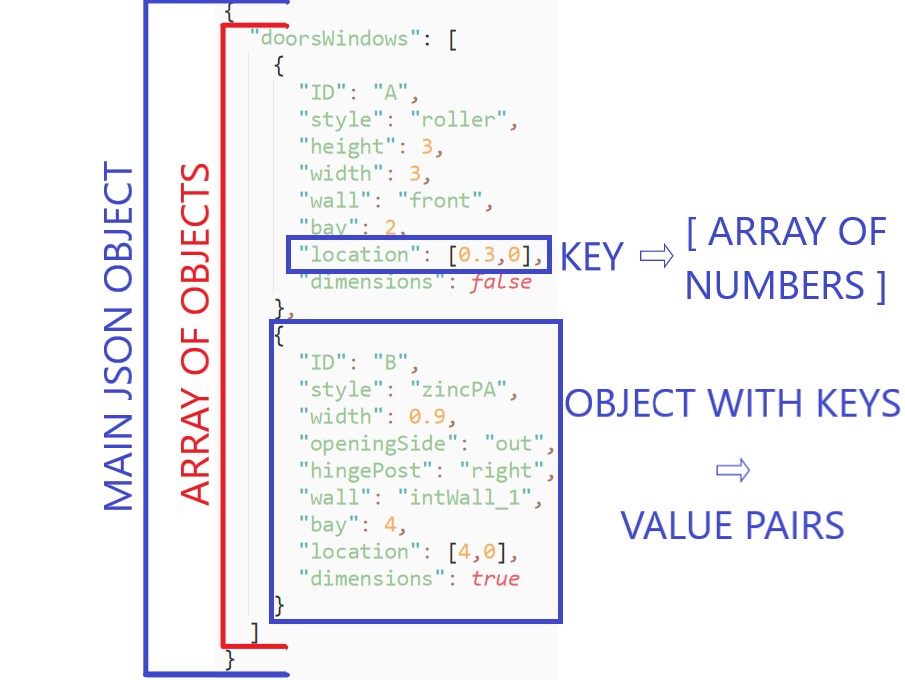


Ilustración : Ejemplo JSON

En la Ilustración 1, podemos ver que tenemos un documento el cual tiene un título “doorsWindows”, el cual tiene 2 objetos dentro, uno con el ID=”A” y otro con ID=”B”, ambos objetos tienen distintos atributos llave/valor. Se distinguen 3 tipos de datos en este ejemplo, por ejemplo, el valor de la Key “ID” es un String, también, la altura (height) es un integer. A su vez, también se puede asignar como value un objeto anidado o un Array, como es el caso del atributo Location, que es un array de números.

## 2.3 Gestores de Bases de datos NoSQL

Algunos de los más famosos sistemas de gestión de bases de datos NoSQL son los siguientes:

Para efectos de este plan de mejora, el gestor de bases de datos que usaré será MongoDB, con su capa y servicio online llamado Mongo Atlas.

Ilustración :SGBD NoSQL más usadas

## 2.4 Ventajas y desventajas

|  |  |
| --- | --- |
| Ventajas | Desventajas |
| * Alto rendimiento y baja exigencia: En general se pueden ejecutar en máquinas con recursos modestos. A diferencia de los sistemas SQL, la inversión que se necesita es significativamente más baja. * Escalabilidad horizontal: Para mejorar el rendimiento de estos sistemas se puede incrementar nodos de forma horizontal para aumentar el rendimiento. * Volumen de datos no es problema: Se pueden manejar gran cantidad de datos sin generar cuellos de botella ni ralentizando el sistema. * Flexibilidad: Las estructuras de datos son extremadamente flexibles y se pueden ajustar a casi todo lo que se necesita. | * Funciones de Fiabilidad: Las bases de datos NoSQL no admiten funciones de fiabilidad como ForeignKeys o constrains. * Aplicabilidad: La falta de inclusión de funciones de fiabilidad limita la aplicabilidad de estas BBDD. * Incompatibilidad en consultas SQL: En la mayoría de los casos las bases de datos NoSql son incompatibles a consultas SQL, lo que agrega una mayor dificultad para realizar procesos que requieran más complejidad. |

## 2.4 Introducción a MongoDB

Ahora Identificando lo más general de NoSQL, empieza la incursión en MongoDB, para esto utilizaré Mongo Atlas, el cual es un ambiente Cloud de mongo el cual nos pide instalar solamente MongoDB Compass para su ejecución. Así también en algunas ocasiones utilizaré WSL ya que usaré un ambiente de desarrollo sobre esta plataforma.

Para esto utilizaré un curso el cual pude conseguir gratis gracias a un cupón del autor. [MongoDB: Aprende desde cero a experto | Udemy](https://www.udemy.com/course/mongodb-aprende-desde-cero/)

**Drivers MongoDB**

**Interfaz de usuario gráfica

Descripción generada automáticamente**

Más adelante en los siguientes puntos especificaré que trabajaré MongoDB con Python.

## 2.5 MongoDB Cheat Sheet

|  |  |
| --- | --- |
| Conceptos | |
| Database | Un contenedor para las colecciones. Se refiere a lo mismo que una Database en SQL y usualmente cada proyecto tiene su propia database con sus distintas colecciones. |
| Collection | Una colección es un grupo de documentos de una database. Es similar a lo que es una tabla en SQL. |
| Document | Document es un registro dentro de una colección. Es similar a lo que vendría siendo una fila dentro de SQL. |
| Field | Field es par llave-valor dentro de un documento. Es similar a lo que es una columna dentro de SQL. |

|  |  |
| --- | --- |
| Comandos Básicos | |
| mongosh | Abre una conección con una instancia local de MongoDB. |
| show dbs | Muestra todas las Databases que hay en la instancia actual. |
| use <dbname> | Cambia a la database ingresada |
| db | Muestra el nombre de la Database actual |
| cls | Limpia la terminal |
| show collections | Muestra las colecciones dentro de la database |
| db.dropDatabase() | Elimina la dabase actual |
| exit | Cierra la sesión de Mongosh |

|  |  |
| --- | --- |
| Crear | |
| db.<collectionName>.insertOne | Crea un nuevo documento dentro de una colección específica |
| db.<collectionName>.insertMany | Crea múltiples nuevos documentos dentro de una colección específica |

|  |  |
| --- | --- |
| Leer |  |
| db.users.find() | Obtener todos los documentos |
| find(<filterObject>) | Obtiene todos los documentos que tengan el par llave-valor ingresado. |
| Find(<filterObject>,<selectObject>) | Obtiene todos los documentos que coincidan con el filtro pero retorna solo los campos seleccionados. |
| findOne | Lo mismo que Find, solamente que retorna el primer documento. |
| countDocuments | Retorna la cantidad de documentos que coincidan con el filtro que se le envía como parámetro. |

|  |  |
| --- | --- |
| Modificar |  |
| updateOne | Modifica el primer documento que coincida con el filtro y lo reemplaza con el segundo parámetro que entra dentro de un $set |
| updateMany | Modifica todos los documentos que coincidan con el filtro y les cambia el valor con el segundo parámetro que entra dentro de un $set |
| ReplaceOne | Reemplaza el primer valor que coincida con el entregado por el filtro como parámetro y el valor asignado es el segundo parámetro. |

## 2.6 Conclusiones MongoDB

En este punto tengo el entendimiento básico de las bases de datos NoSQL y las funciones y operaciones básicas de MongoDB, aunque el curso aún no logro terminarlo, esto me abre la posibilidad de trabajar con el Framework Flask. Para poder generar un backend y conectarlo con la base de datos en Mongo.

# 3. Flask

## 3.1 Descripción

La palabra "micro" no significa que sea un proyecto pequeño, o que nos permita hacer pequeñas páginas web, sino que al instalar Flask tenemos las herramientas que necesitamos para crear aplicaciones web funcionales, y si se llegase a necesitar nuevas funcionalidades en algún momento hay excelentes extensiones (complementos) que se pueden instalar para proporcionarle funcionalidad.

En este caso Flask lo usaré para crear una REST API para luego consumirla desde un Frontend hecho con la herramienta Vuejs.

## 3.2 Ventajas

* Proporciona una estructura del proyecto, es decir, todas las Apps que estén construidas con Flask van a tener los mismos elementos y ficheros.
* Facilita la colaboración.
* Es fácil encontrar bibliotecas adaptadas al Framework.

## 3.3 Instalaciones

Para tener Flask, lo único que se necesita es tener instalado Python en nuestro sistema, yo personalmente ocupare mi ambiente de trabajo en WSL. El comando para instalar flask es “pip install flask”.

Texto

Descripción generada automáticamente

## 3.4 Primeros Pasos

Texto

Descripción generada automáticamente

En esta imagen podemos ver una app sencilla, la cual se conecta a mi conexión que creé en MongoDB Atlas (Aunque por ahora no guarda ningún registro). Creé una ruta “/users”, la cual retorna un mensaje e imprime en pantalla el json del request enviado por la petición POST. Para probar esto utilizaré la herramienta Postman.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Como se ve en la imagen se muestra el mensaje que se designó en la ruta.

## 3.5 Aplicación Flask

Luego de estudiar y leer documentación empecé con la creación de una API Básica en Flask, la cual gestiona datos sobre mangas, los Fields son el id, nombre, autor, y volúmenes. En esta imagen podemos ver la primera ruta, la cual recibe peticiones de tipo POST, esta petición recibe un objeto JSON, y luego lo inserta en la base de datos MongoDB. Para esto tuve que configurar el “MONGO\_URI”, el string de conexión es el que obtuve desde MongoDB Atlas.

Texto

Descripción generada automáticamente

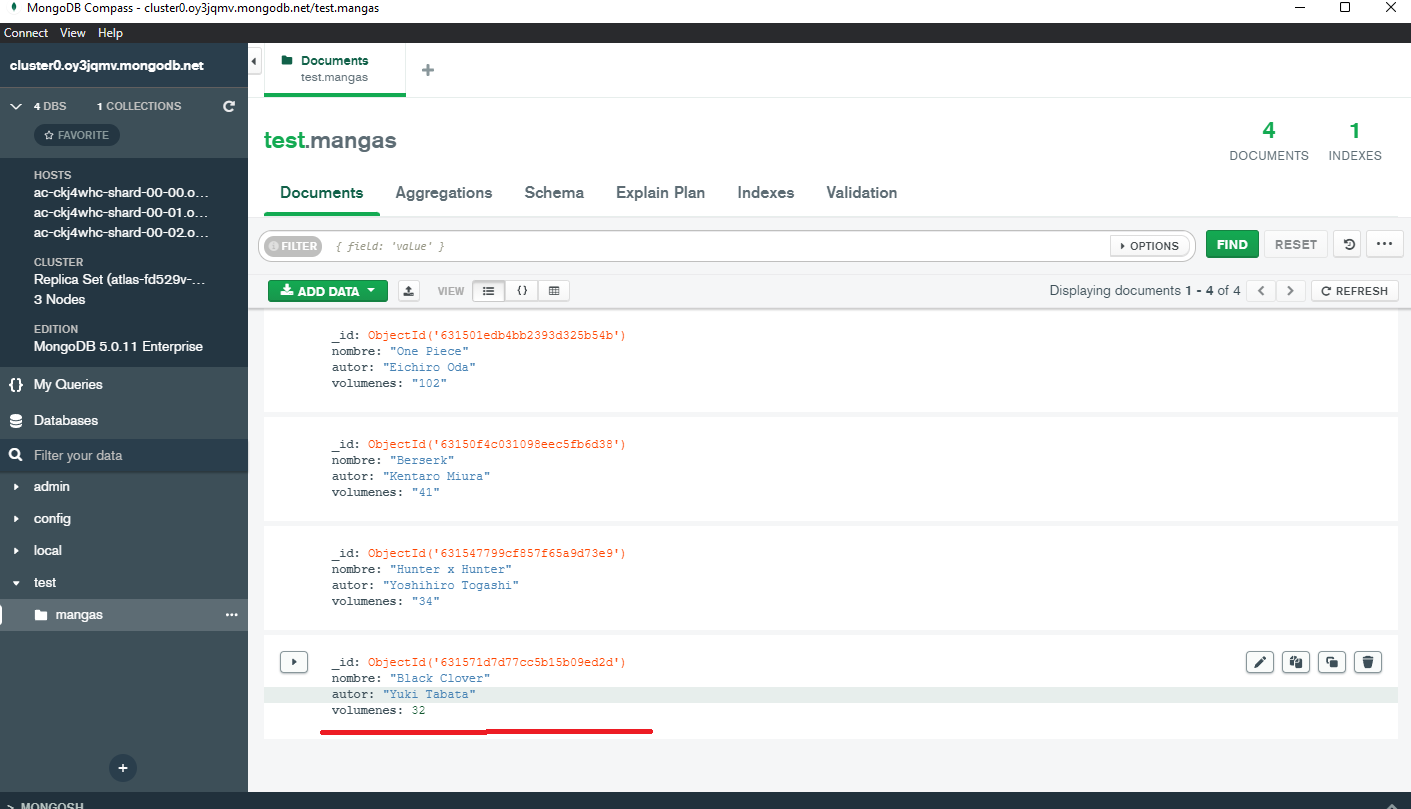
Para probar este método usaré Postman de nuevo. Como se vió en la imagen anterior cuando el objeto efectivamente es insertado en la base de datos retorna un JSON con una copia de los datos que fueron insertados.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Como se ve en la imagen al insertarse el registro, este fue retornado como Json a una respuesta para mostrar donde se desee.

Mientras tanto, con la herramienta MongoDB Compass, podemos ver que efectivamente se guardó el registro. Además de mostrar los que había agregado como prueba anteriormente.



Además de esta ruta también fui agregando otros *Endpoints* a esta *API* para poder consumirlos desde el *Frontend.*

## 3.5 Endpoints API

A continuación, muestro todos los Endpoints que construí en la aplicación:

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre Endpoint | Descripción | Ruta | Método |
| create\_mangas() | Recibe un objeto JSON y lo inserta en la base de datos MongoDB, a la vez, retorna un objeto JSON con los datos insertados. | ‘/mangas’ | POST |
| get\_mangas() | Retorna un objeto JSON con todos los documentos de la colección. | ‘/mangas’ | GET |
| get\_manga(id) | Retorna el documento que tenga el id que se le entrega por parámetro. | ‘/mangas/<id>’ | GET |
| delete\_manga(id) | Elimina el registro que tenga el id asociado, retorna un mensaje de confirmación | ‘/mangas/<id>’ | DELETE |
| update\_manga(\_id) | Modifica un registro que tenga el id asociado, retorna un mensaje de confirmación. | ‘/mangas/<\_id>’ | PUT |

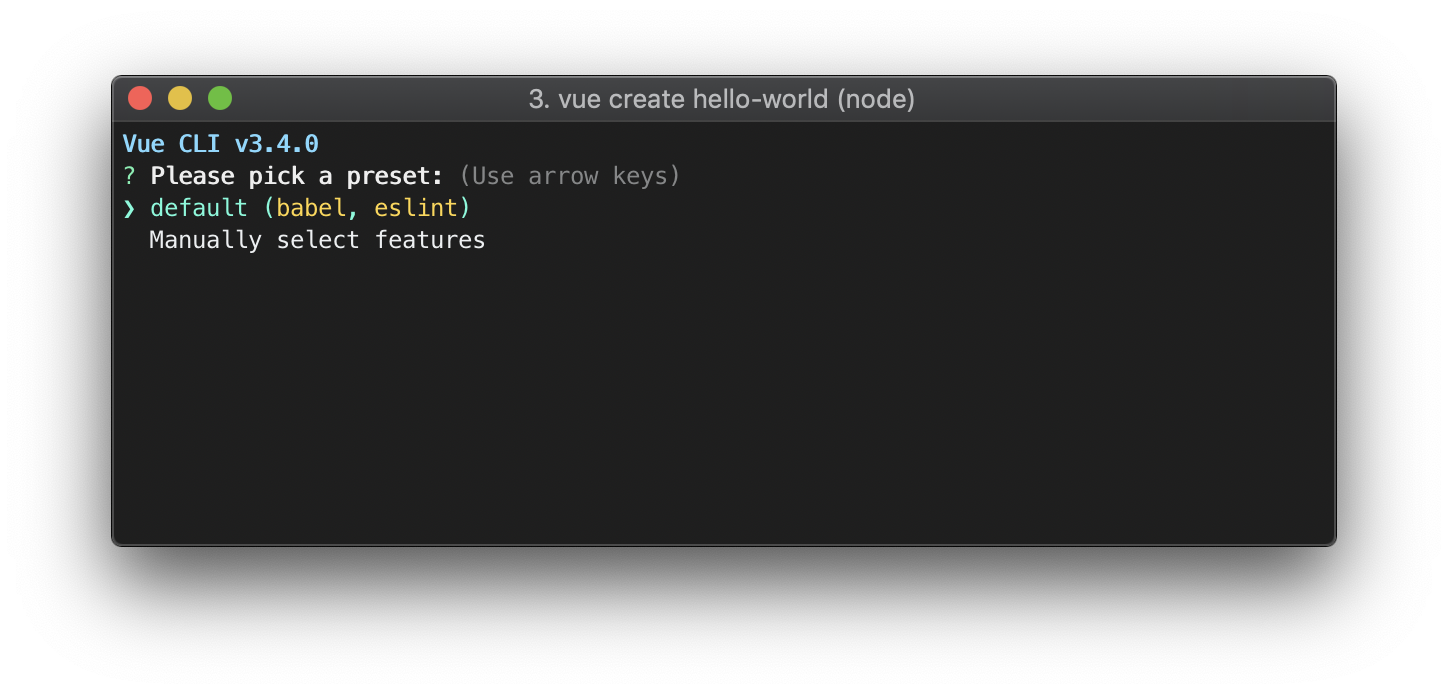
# 4. Aplicación Vuejs

## 4.1 Introducción

Esta aplicación será creada para poder consumir algunos de los Endpoints creado con Flask. En esta iteración pretendo adquirir conocimientos básicos de este Framework basado en Javascript, por lo que no tendrá una alta complejidad en el código.

## 4.2 Instalaciones

Para poder crear una aplicación en Vuejs, lo primero que debemos tener instalado es NPM, que es un gestor de paquetes de Node. Luego se ingresa el comando *“npm i vue”.* Luego para crear el proyecto necesitamos ingresar el siguiente comando: *“vue create <nombre proyecto>”.* Una vez creado el proyecto nos situamos con una terminal en el directorio del proyecto, para luego escribir y ejecutar el comando “npm install” y posteriormente “npm run serve”.



Debería mostrarnos una interfaz parecida a la anterior.

## 4.3 Codificando

Primero partimos modificando el archivo “App.vue”, el cual es básicamente la App principal de vue, junto al main.js.

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamenteVue facilita mucho la utilización de “trozos de código” también llamado componentes, los cuales son independientes unos de otros, pero a la vez se pueden comunicar entre ellos si es que lo necesitamos.

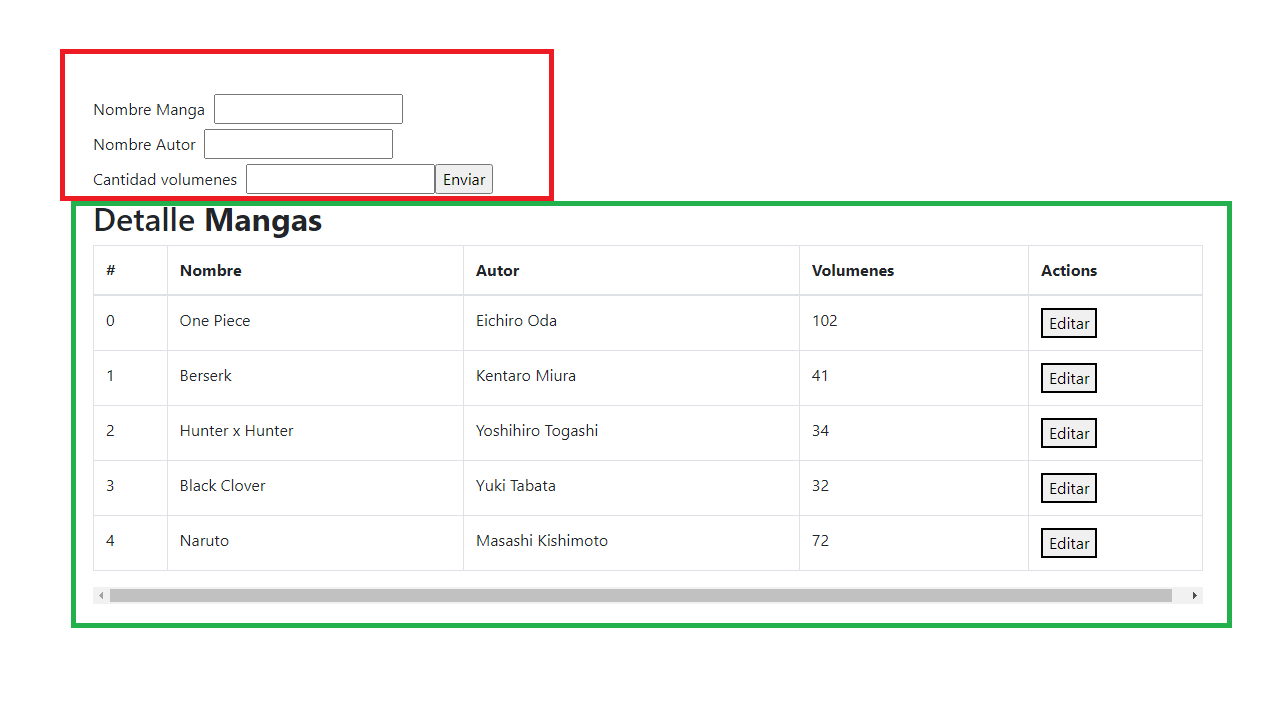
Los componentes creados en esta iteración son los ListComponent y Save component.

ListComponent, básicamente es el componente que me ayuda a mostrar el listado de todos los mangas utilizando el endpoint get\_mangas()

SaveComponent, es un componente el cual me ayuda tanto a guardar como modificar mangas. Utilizando 2 enpoints create\_manga() y update\_manga(\_id).

## 4.5 Vistas

La aplicación que estoy desarrollando será una SPA, lo cual significa Single Page Aplication. Esta aplicación tendrá los distintos componentes en una misma pagina para hacer todo más cómodo y minimalista.



En la imagen se pueden ver los 2 componentes que están dentro de esta vista, el que está marcado en rojo es el componente para guardar y modificar, y el segundo componente es el que lista todos los mangas.

# 5. Conclusiones

Dentro de esta iteración y este proceso de mejora pude comprender muchas de las cosas que en su tiempo no pude entender sobre como funcionaban las API, además de que yo mismo pude crear una para luego consumirla con un Framework totalmente nuevo para mí. Desde el estudio leyendo documentaciones hasta tutoriales y cursos que fui viendo a lo largo de todo el proceso, pude obtener las nociones básicas de cómo desarrollar aplicaciones con Backend y Frontend separados (se comunican a través de la API).

Espero seguir estudiando más sobre Flask y Vue, ya que considero que aún me falta muchísimo y no hice nada más que empezar con este Framework. Espero poder entregar una aplicación más robusta para las siguientes iteraciones.

Repositorio Github Plan de mejora: <https://github.com/nicortezm/plan-de-mejora>

# 6. Referencias

<https://www.udemy.com/course/mongodb-aprende-desde-cero/> - Curso Udemy MongoDB

<https://www.youtube.com/playlist?list=PLB5SYDBi8M3GfbkVFBO5YE6feVH9RntF4> – Guía consumo de REST API

<https://vuejs.org/guide/introduction.html> - Documentación Vuejs

<https://www.mongodb.com/docs/> - Documentación MongoDB

<https://flask.palletsprojects.com/en/2.2.x/> - Documentación Flask