INTRODUCCIÓN AL DESARROLLO BACKEND

CON NODEJS

2023

ANDÉN Centro de Innovación y Emprendimientos Tecnológicos

SECRETARÍA DE EXTENSIÓN UNIVERSITARIA UTN - FRC SEU

UTN Facultad Regional Córdoba

Agencia CÓRDOBA JOVEN

CÓRDOBA entre todos

Bases de datos con MongoDB

- Introducción a las bases de datos y su uso
  - Elementos principales
  - Uso de las bases de datos
  - Bases de datos SQL
  - Bases de datos NoSQL
- MongoDB
  - Extension VSCode
  - ORM. Mongoose.
  - API con MongoDB

Elementos de una base de datos

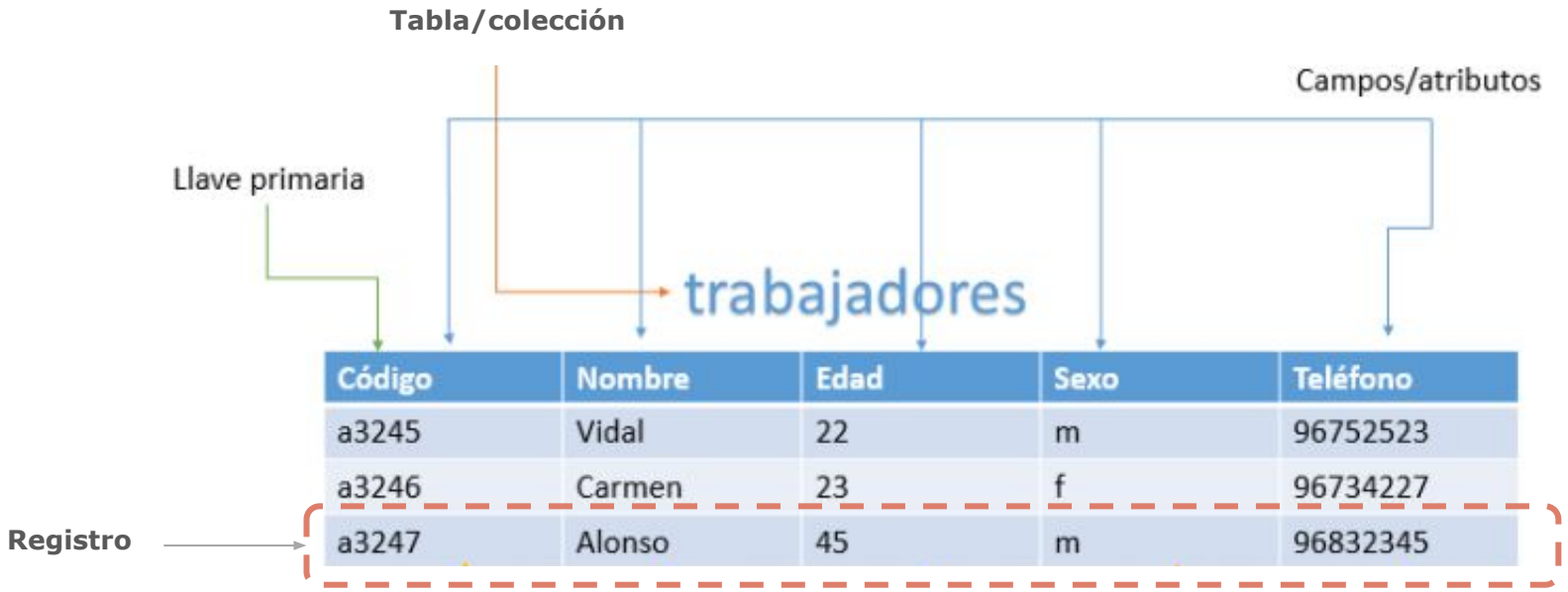
- **Base de datos:** es un sistema organizado para almacenar, gestionar y recuperar información de manera estructurada.
- **Elementos fundamentales** de una base de datos:
  - Motor de base de datos
  - base de datos
  - tablas,
  - registros y
  - campos.
- **Tipos de bases de datos:**
  - Relacionales (SQL)
  - NoSQL.

Motor de base de datos

- Un motor de base de datos es un software responsable de **gestionar y organizar los datos en una base de datos.**
- Un motor de base de datos puede contener 1 o muchas bases de datos.
- Funciones principales de un motor de base de datos:
  - **Almacenamiento y recuperación:** El motor de base de datos es responsable de almacenar los datos de manera persistente en el disco y recuperarlos cuando sea necesario.
  - **Gestión de la concurrencia:** Permite el acceso simultáneo a los datos por parte de múltiples usuarios o aplicaciones, asegurando la consistencia y evitando conflictos.
  - **Consultas y optimización:** El motor de base de datos ejecuta consultas realizadas por las aplicaciones y optimiza la forma en que se accede y se recuperan los datos para mejorar el rendimiento.
  - **Mantenimiento de la integridad:** Asegura que los datos almacenados en la base de datos cumplan con las restricciones y reglas definidas, manteniendo la integridad de los datos.
- Ejemplos populares de motores de base de datos:
  - MySQL, PostgreSQL, Oracle y Microsoft SQL Server son ejemplos de motores de base de datos relacionales.
  - MongoDB, Cassandra y Redis son ejemplos de motores de base de datos NoSQL.

Elementos de una base de datos

- Una base de datos puede contener 0 a muchas tablas.
- Una tabla/colección tiene:
  - Columnas
  - Registros



Código	Nombre	Edad	Sexo	Teléfono
a3245	Vidal	22	m	96752523
a3246	Carmen	23	f	96734227
a3247	Alonso	45	m	96832345

Uso de las bases de datos

Las bases de datos son fundamentales en una amplia gama de aplicaciones y sistemas informáticos. Algunos casos de uso comunes incluyen:

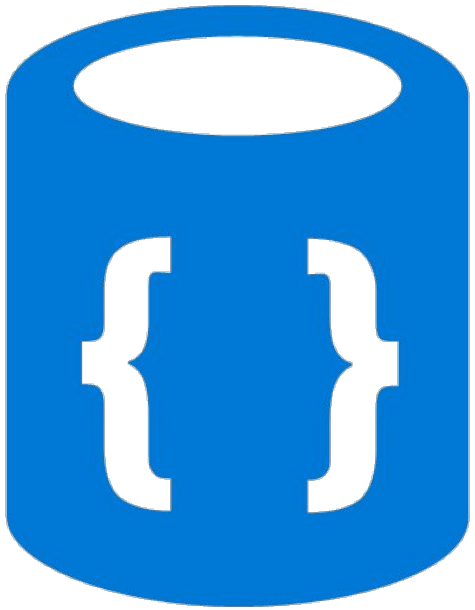
- **Sistemas de gestión de contenido:** Las bases de datos permiten almacenar y recuperar contenido como artículos, imágenes y videos en aplicaciones web y CMS (Content Management Systems).
- **Sistemas de información empresarial:** Las bases de datos son utilizadas para almacenar y gestionar datos empresariales, como información de clientes, inventarios y registros de transacciones.
- **Aplicaciones bancarias:** Las bases de datos aseguran la integridad y la consistencia de los datos financieros en aplicaciones de banca en línea, incluyendo transacciones, cuentas y registros de clientes.
- **Sistemas de reservas en línea:** Las bases de datos permiten el almacenamiento y acceso rápido a información de reservas de vuelos, hoteles, eventos, entre otros.
- **Aplicaciones de comercio electrónico:** Las bases de datos son esenciales para gestionar catálogos de productos, carritos de compras, información de clientes y pedidos en aplicaciones de venta en línea.

Introducción a las bases de datos SQL

- Las bases de datos SQL son sistemas de gestión de bases de datos relacionales que utilizan un **lenguaje de consulta estructurado** (SQL) para interactuar con los datos. Algunos puntos clave a considerar son:
  - **Estructura:** Las bases de datos SQL están organizadas en tablas, donde **cada tabla contiene filas y columnas**. Las columnas representan los atributos o campos de los datos, mientras que las filas representan las entradas individuales en la tabla.
  - **Relaciones:** Las bases de datos SQL utilizan relaciones para establecer conexiones entre tablas a través de claves primarias y claves externas, lo que permite modelar y gestionar datos relacionados.
  - **Ejemplos populares:** Algunos ejemplos populares de bases de datos SQL incluyen MySQL, PostgreSQL, Oracle, Microsoft SQL Server y SQLite.

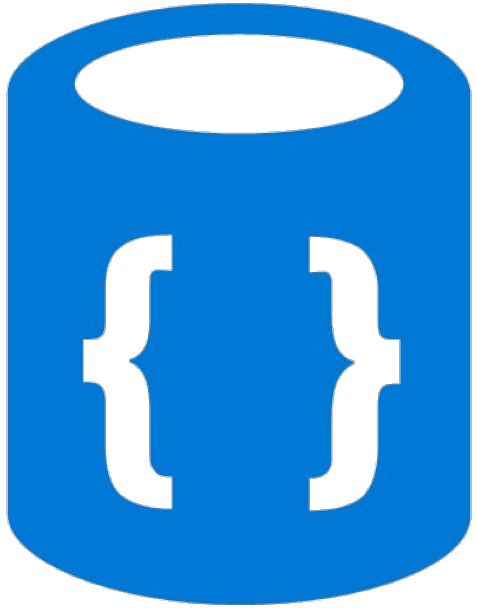
Ejemplo Base de datos SQL

Introducción a las bases de datos NoSQL



- Las bases de datos NoSQL son sistemas de gestión de bases de datos no relacionales que ofrecen una alternativa flexible y escalable a las bases de datos SQL tradicionales. Algunos puntos clave a considerar son:
  - **Estructura flexible:** Las bases de datos NoSQL permiten almacenar datos en diversos formatos, como documentos, grafos o pares clave-valor, sin la necesidad de seguir un esquema fijo.
  - **Escalabilidad horizontal:** Las bases de datos NoSQL están diseñadas para escalar horizontalmente, lo que significa que pueden manejar grandes volúmenes de datos distribuyendo la carga en múltiples servidores.
  - **Ejemplos populares:** Algunos ejemplos populares de bases de datos NoSQL incluyen MongoDB, Cassandra, Redis y Couchbase.

Ejemplo Base de datos NoSQL



Base de datos NoSQL: MongoDB

- MongoDB es una base de datos NoSQL de código abierto y orientada a documentos.
- Características clave de MongoDB:
  - **Modelo de datos flexible:** MongoDB almacena los datos en documentos flexibles tipo JSON llamados BSON, lo que permite una fácil representación y manipulación de datos complejos y anidados.
  - **Escalabilidad horizontal:** MongoDB está diseñado para escalar horizontalmente, lo que significa que puede manejar grandes volúmenes de datos distribuyendo la carga en múltiples servidores y aprovechando clústeres de réplicas.
  - **Alta disponibilidad:** MongoDB proporciona mecanismos integrados de replicación y recuperación ante fallos, asegurando una alta disponibilidad y tolerancia a fallos.
  - **Consultas potentes:** MongoDB admite consultas flexibles y potentes utilizando su propio lenguaje de consulta (Query Language) basado en JSON, permitiendo realizar consultas complejas y filtrar datos de forma eficiente.

Cuando usamos MongoDB

- **Aplicaciones web y móviles:** MongoDB es popular en el desarrollo de aplicaciones modernas que manejan grandes volúmenes de datos no estructurados.
- **Análisis de datos:** MongoDB se utiliza en casos donde se requiere almacenar, consultar y analizar datos en tiempo real, como en la industria de IoT (Internet de las cosas) y en sistemas de registro de eventos.
- **Gestión de contenido:** MongoDB es adecuado para sistemas de gestión de contenido y CMS (Content Management Systems), donde se manejan documentos y contenido no estructurado.
- **Registro y análisis de registros:** MongoDB puede ser utilizado para almacenar y analizar registros de aplicaciones y servidores, permitiendo un análisis eficiente y rápido de grandes volúmenes de datos de registro.

Instalar MongoDB

- Porque hace falta instalar MongoDB en nuestras computadoras:
  - Acceso inmediato: Los alumnos pueden acceder a MongoDB en sus propias máquinas sin depender de una conexión a Internet.
  - Experimentación y práctica: Permite a los alumnos interactuar directamente con la base de datos, realizar consultas, insertar datos y realizar operaciones CRUD.
  - Desarrollo local: Al instalar MongoDB, los alumnos pueden construir y probar aplicaciones en un entorno local antes de implementarlas en un servidor remoto.
- Instructivo instalación Windows:
  - https://docs.google.com/document/d/1qam4mqHj14EABzf9GPwFR-jdB-ufOemU-XVKWe41wB4/edit?usp=sharing
  - + Info: https://www.mongodb.com/docs/manual/administration/install-community/

Conceptos clave de MongoDB

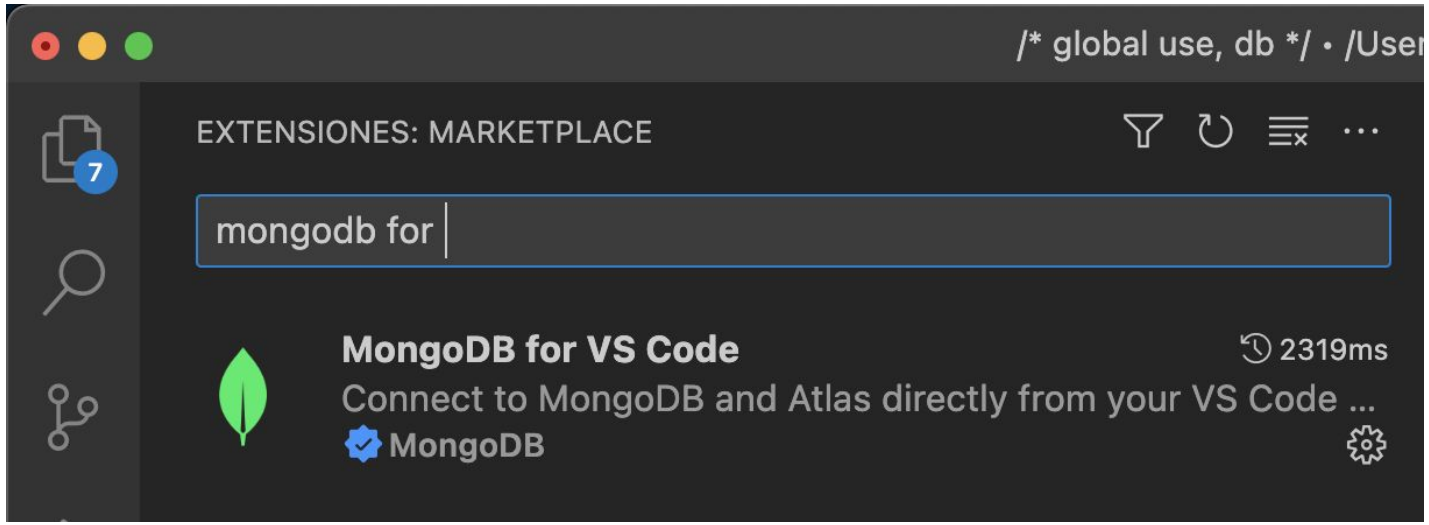
- **Documento:** MongoDB almacena los datos en documentos flexibles tipo JSON llamados BSON (Binary JSON). Un documento es una unidad básica de almacenamiento en MongoDB y puede contener datos estructurados y no estructurados.
- **Colección:** Una colección es un grupo de documentos almacenados en MongoDB. Al igual que una tabla en una base de datos relacional, una colección almacena documentos relacionados entre sí.
- **Base de datos:** Una base de datos en MongoDB es un contenedor físico de colecciones. Puede haber múltiples bases de datos en un solo servidor de MongoDB, y cada base de datos puede contener varias colecciones.
- **ID de documento:** Cada documento en MongoDB tiene un campo único llamado "_id", que actúa como identificador único para ese documento dentro de una colección. Si no se proporciona un "_id" al insertar un documento, MongoDB generará uno automáticamente.
- **Consultas:** MongoDB utiliza un lenguaje de consulta basado en JSON para recuperar datos. Las consultas pueden buscar documentos basándose en criterios de filtrado, operadores lógicos y condiciones complejas.
- **Índices:** MongoDB admite la creación de índices para mejorar el rendimiento de las consultas. Los índices son estructuras que aceleran la búsqueda de datos y pueden ser creados en uno o varios campos de un documento.

mongoDB®

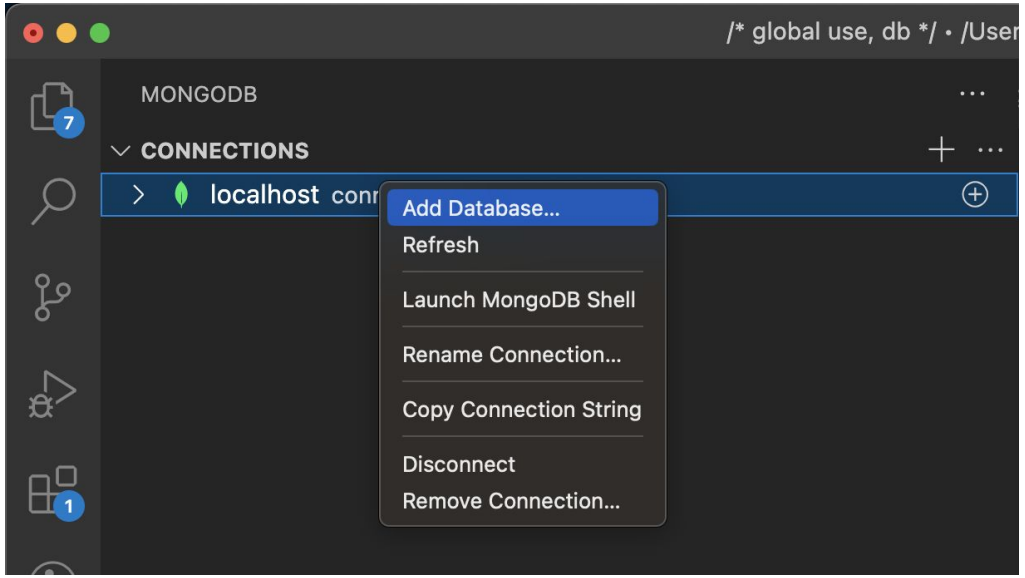
Conceptos clave de MongoDB

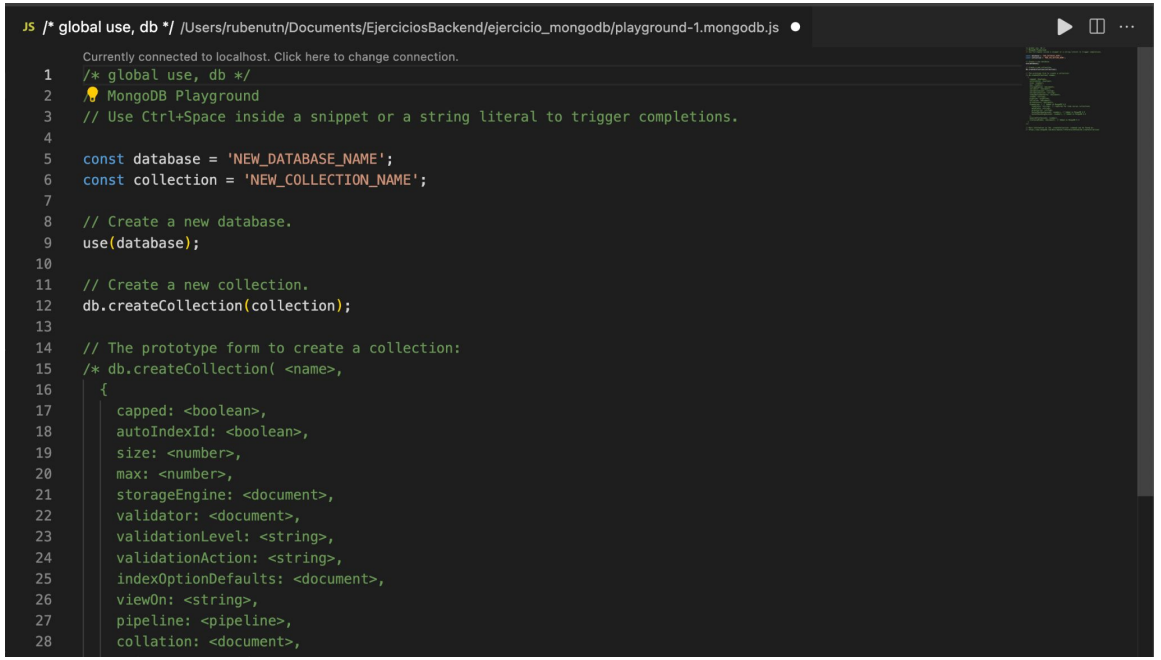
Base de Datos	Tienda	
Colecciones	Usuarios	Pedidos
Documentos	{Nombre: 'Juan', edad: 25}	{...}
	{Nombre: 'Felipe'}	{...}

MongoDB for VS Code



Creando Base de Datos y Colección





Operaciones CRUD en MongoDB

- MongoDB permite realizar las operaciones fundamentales CRUD (Crear, Leer, Actualizar y Eliminar) en la base de datos de manera sencilla:

Crear (Create):

- Insertar documentos en una colección utilizando el método `insertOne()` para insertar un solo documento o `insertMany()` para insertar múltiples documentos a la vez.
- Ejemplo práctico:

```
db.collection('usuarios').insertOne({ nombre: 'Juan', edad: 25 });
```

Actualizar (Update):

- Actualizar documentos utilizando el método `updateOne()` para actualizar un solo documento o `updateMany()` para actualizar múltiples documentos en una colección.
- Ejemplo práctico:

```
db.collection('usuarios').updateOne({ nombre: 'Juan' }, { $set: { edad:
```

Leer (Read):

- Consultar documentos utilizando el método `find()` para buscar documentos en una colección según un criterio de filtrado.
- Ejemplo práctico:

```
db.collection('usuarios').find({ edad: { $gt: 20 } });
```

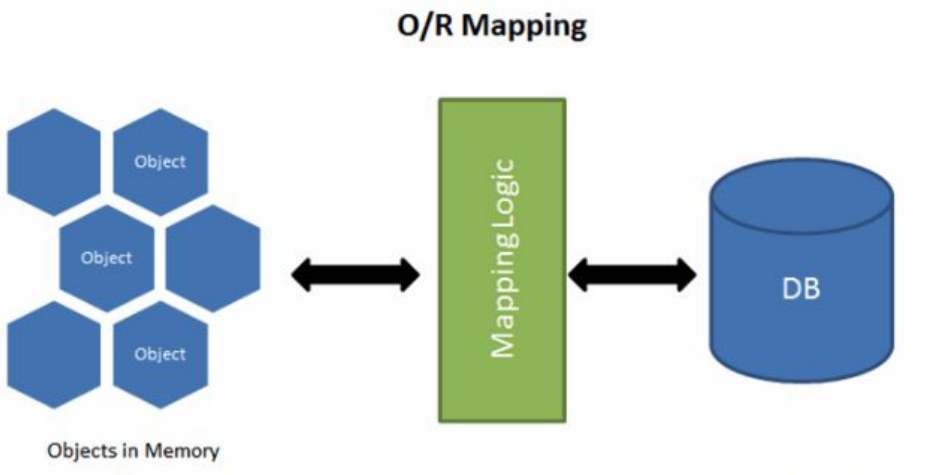
Eliminar (Delete):

- Eliminar documentos utilizando el método `deleteOne()` para eliminar un solo documento o `deleteMany()` para eliminar múltiples documentos de una colección.
- Ejemplo práctico:

```
db.collection('usuarios').deleteOne({ nombre: 'Juan' });
```

Uso de un ORM para trabajar con MongoDB

- Un ORM es una herramienta que permite mapear objetos de una aplicación a documentos en una base de datos.
- Facilita el desarrollo al eliminar la necesidad de escribir consultas y manipulaciones de datos en MongoDB en código de bajo nivel.
- Proporciona una abstracción de la base de datos, permitiendo a los desarrolladores trabajar con objetos y operaciones orientadas a objetos en lugar de manipular directamente documentos BSON.

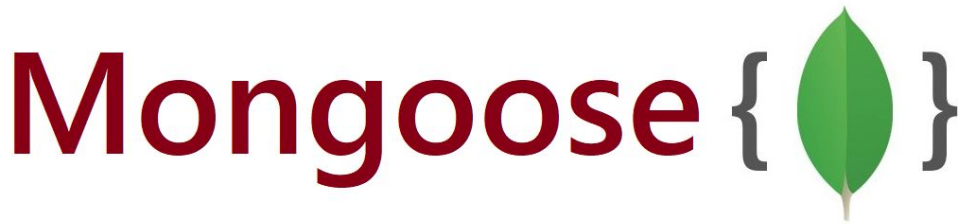


Ventajas de utilizar un ORM en MongoDB

- **Productividad:** Un ORM simplifica la interacción con la base de datos al proporcionar métodos y funciones para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) de forma más sencilla y concisa.
- **Mantenibilidad:** Al utilizar un ORM, el código se vuelve más legible y mantenible, ya que las operaciones y consultas se expresan en términos de objetos y relaciones, en lugar de manipular directamente documentos BSON.
- **Portabilidad:** Utilizar un ORM puede facilitar la portabilidad del código, ya que el ORM puede ser compatible con múltiples bases de datos, lo que permite cambiar de base de datos sin tener que modificar en gran medida el código de la aplicación.

Mongoose: ORM para MongoDB

- Mongoose es una librería ORM popular para MongoDB en Node.js.
- Permite definir modelos y esquemas para mapear objetos JavaScript a documentos en MongoDB.
- Proporciona una capa de abstracción para realizar operaciones CRUD, validaciones y consultas utilizando métodos y funciones sencillos.



Beneficios de usar Mongoose

- **Validaciones de datos:** Mongoose permite definir reglas de validación en los modelos para garantizar la integridad de los datos antes de ser almacenados en MongoDB.
- **Relaciones y referencias:** Mongoose permite establecer relaciones y referencias entre documentos en diferentes colecciones, facilitando el modelado de relaciones complejas.
- **Middleware y hooks:** Mongoose ofrece middleware y hooks que permiten realizar acciones antes o después de ciertas operaciones en la base de datos, cómo validar datos, transformar datos o ejecutar lógica personalizada.

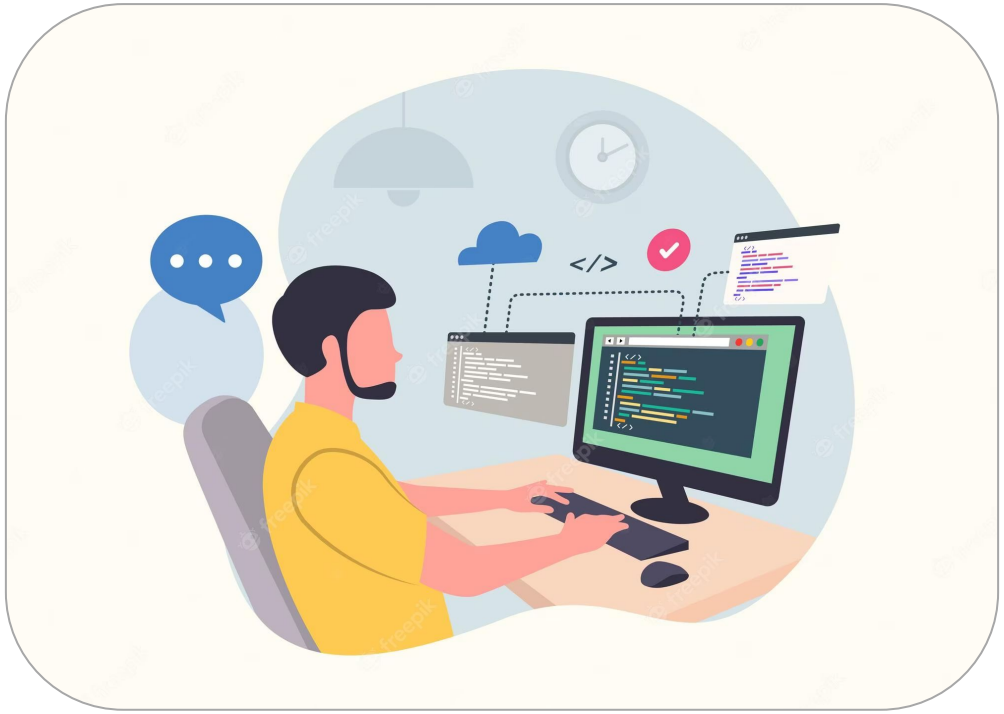
Bueno, Vamo a Codea!!!

1. Mongoose
2. API + Mongoose

Mongoose { }

Actividad 5: Paso a Paso API con MongoDB

- Seguir las instrucciones de la actividad publicada en la UVE.



MUCHAS GRACIAS

ANDÉN Centro de Innovación y Emprendimientos Tecnológicos

SECRETARÍA DE EXTENSIÓN UNIVERSITARIA UTN - FRC SEU

UTN Facultad Regional Córdoba

Agencia CÓRDOBA JOVEN

CÓRDOBA entre todos