# **📘 Trabajo Práctico: MongoDB con Docker y Mongoose**

## **🎯 Objetivo general**

Que los estudiantes comprendan los conceptos fundamentales de MongoDB y Mongoose, y sean capaces de utilizar Docker para levantar una base de datos MongoDB y conectarse a ella desde una aplicación Node.js utilizando Mongoose.

## **🧠 Parte Teórica (Responder con sus propias palabras)**

**1. ¿Qué es MongoDB? ¿En qué se diferencia de una base de datos relacional como MySQL o PostgreSQL?**

**2. ¿Qué es una colección en MongoDB? ¿Y qué es un documento? Dibuja un ejemplo básico.**

**3. ¿Qué es Mongoose? ¿Para qué se utiliza en una app de Node.js?**

**4. ¿Qué es un esquema en Mongoose? ¿Y qué es un modelo? ¿Cuál es la relación entre ambos?**

**5. ¿Qué métodos de Mongoose se pueden usar para:**

* Insertar un nuevo documento
* Buscar documentos
* Actualizar un documento
* Eliminar un documento

## **🧪 Parte Práctica (Completar)**

### **Consigna**

Vas a crear una pequeña aplicación en Node.js que se conecta a una base de datos MongoDB corriendo en Docker. Solo te damos una base para que completes el código y pongas a prueba lo aprendido.

### **Paso 1: Levantar MongoDB con Docker**

1. Crear un archivo docker-compose.yml y completarlo para que:
   * Se use la imagen oficial de MongoDB
   * Exponga el puerto 27017
   * Mantenga los datos persistentes usando volúmenes

version: "3.8"

services:

mongodb:

image: mongo:latest

restart: always

environment:

MONGO\_INITDB\_ROOT\_USERNAME: ${MONGO\_USER}

MONGO\_INITDB\_ROOT\_PASSWORD: ${MONGO\_PASS}

volumes:

- ./mongo:/data/db

ports:

- \*\_\_\_\_\_\_\* completar

1. Levantar el contenedor con el comando correspondiente.
2. Verificar en Docker desktop que este corriendo la bbdd

### **Paso 2: Crear la app en Node.js**

1. Crear un nuevo proyecto e instalar las siguientes dependencias:

* npm init -y
* npm i nodemon --save -dev
* npm i express mongoose body-parser dotenv

1. **Requiere los módulos necesarios**

* Importá los módulos express y mongoose.
* Inicializá una instancia de Express.
* Agregá el middleware para interpretar JSON.

1. **Conectate a MongoDB**

* Usá mongoose.connect() para conectarse a la base de datos.
* Agregá un .then() y .catch() para mostrar mensajes de conexión exitosa o error.

1. **Define el modelo de datos**

* Usá mongoose.Schema para definir el esquema de un **usuario** con los campos:  
  + nombre (tipo String)
  + edad (tipo Number)
  + email (tipo String)
* Usá mongoose.model() para crear el modelo.

1. **Crea la ruta GET /usuarios**

* Usá app.get() para definir una ruta que devuelva todos los usuarios almacenados en la base de datos.
* Usá Usuario.find() para obtener todos los documentos.
* Devolvé los resultados como JSON.

1. **Crea la ruta POST /usuarios**

* Usá app.post() para definir una ruta que reciba un objeto JSON con datos de un nuevo usuario.
* Usá el modelo para crear y guardar ese nuevo usuario.
* Devolvé como respuesta el objeto guardado.

1. **Iniciá el servidor**

* Usá app.listen() para levantar el servidor en el puerto 3000.
* Mostrá un mensaje por consola indicando que el servidor está corriendo.