// Clase N°5:

Una Aplicación consta de dos módulos, un módulo Frontend y un módulo Backend.
● El módulo FrontEnd es lo que visualiza el usuario, pantallas, formularios, etc.
● El módulo BackEnd es lo que realiza la acción o la funcionalidad propiamente del sistema.

REQUEST: petición a una API.
RESPONSE: respuesta de esa API.

Qué es un ORM y como se utiliza?
Significa Object Relational Mapping.
Nos permite comunicarnos con nuestra base de datos evitando escribir sentencias de SQL, seria como una interfaz más amigable para poder trabajar con un motor de base de datos.
Permite almacenar y leer objetos de tu BD fácilmente.

Object: estos se definen usualmente con un lenguaje de programación, en nuestro caso se define por JS en Node, se crea un archivo que se conoce como el modelo y en él se definen la forma que van a tener nuestros datos.
Es decir, en el ObjectRM es donde le das la forma al objeto de clientes, iD,
con nombre, apellido, tel, etc. y se definen con un lenguaje de programación.
Relational: es la BD, donde se almacenan los objetos.
Mapping: es la unión de ambos, esto me permite que con código de lenguaje de programación, pueda realizar las consultas a la BD, unir los objetos con el Relational y utilizar esos datos y mostrarlos en mi aplicación.

VENTAJAS DEL ORM
Primero, evitemos la repetición de código, ya que el modelo se define en un lugar y solo con importar el modelo al controlador, tendrás acceso a todos los métodos del ORM.
Aceleran mucho el proceso de desarrollo ya que hay una gran cantidad de métodos para el CRUD (?.
No es necesario escribir código SQL.
Seguridad: los ORM cuentan con sentencias preparadas para evitar la inyección de datos no esperados y con una gran cantidad de medidas de seguridad que hacen que no tengas que preocuparte por esto en tus aplicaciones.
Escrito en el lenguaje de tu aplicación web.

DESVENTAJAS DEL ORM
Siempre hay que aprender una sintaxis nueva, eso puede ser una desventaja.
Cuando alguien comienza a utilizar un ORM, comienza a utilizar código SQL, y si es muy fácil la consulta, tiran un SELECT, y si es muy complicada la consulta, deciden utilizar lenguaje ORM, y eso no se hace.
Siempre hay que utilizar los métodos del ORM, buscar y encontrar las funciones que hacen lo que necesitamos.
Instalarlo a veces no es tan sencillo, e instalarlo localmente es un problema, y en un servidor, mucho más aún. (aunque NODE lo hace muy sencillo gracias a NPM)
En cuanto a performance es un poco más lento que las consultas SQL nativas.
Un ORM es una capa de abstracción sobre la BD, si utilizar un método para crear un registro, el ORM se encarga de escribir el código SQL, pero tú estás escribiendo código en tu lenguaje de programación. Es decir en lo que se interpreta tu lenguaje, más lo que se ejecuta, puede llegar a ser un poco más lento que si estuvieras escribiendo las consultas nativas SQL. Nada es más rápido que consulta nativa, el ORM puede ser un poquito más lento, pero te salva de aprender código SQL si no lo sabes.

Un ORM no es específico de NODE. Muchos lenguajes de programación tienen un ORM:

PHP: Propel y Doctrine.
JAVA: Hibernate.
Python: SQL Alchemy.
C#: Entity Framework.

Y en muchos lenguajes vas a encontrar un ORM. Es una herramienta muy común que ahorra mucho tiempo.
En cuanto a NODE, existen dos opciones muy claras:
SEQUELIZE soporta MySQL, PostgreSQL, SQL Server y SQLite.
Y otro es MONGOOSE que permite conectar MongoDB con tus aplicaciones NodeJS.

Muchas veces MODELO y ORM están muy relacionados y puede ser
posible que sea difícil de comprender qué es lo que hace cada uno.
El modelo es un lugar único donde se describe la forma de los objetos y esta forma hace que sea fácil agregar y quitar campos para la BD.
Este describe la tabla de productos que tiene un ID, un nombre, un código, un precio, que type va a ser, todo esto se define en el modelo.

Mientras que el ORM tiene los métodos para la BD.
Este cuenta con todos los métodos, para crear, para eliminar, para actualizar, para relacionar productos con una categoría, productos con un cliente, etc.

Para ejemplificar citamos una imagen:

La clara diferencia acá, es que el modelo da la estructura de todos los datos,
mientras que el ORM, nos da los métodos para trabajar o consultar la BD.