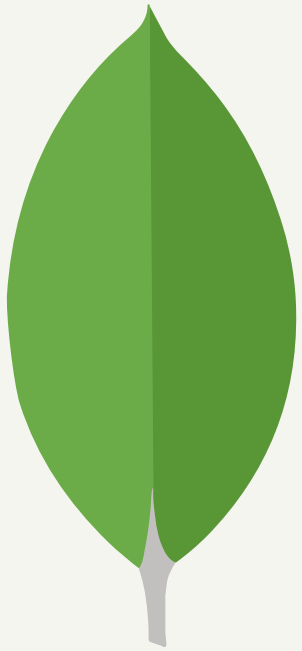
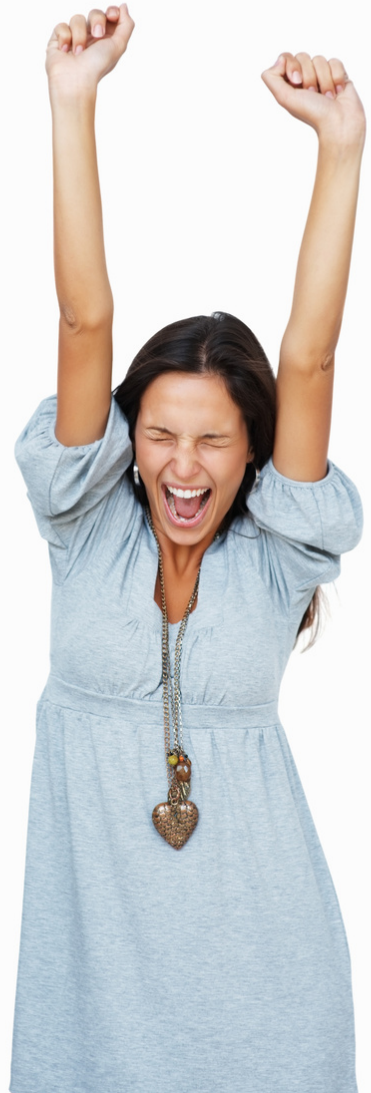


mongoDB

WEB DEVELOPER BOOTCAMP



WHAT IS MONGO?

OUR FIRST DATABASE!

According to Mongo's homepage, it is "the most popular database for modern applications". It is commonly used in combination with Node.

Mongo is a **document database**, which we can use to store and retrieve complex data from.

WHY USE A DATABASE?

INSTEAD OF JUST SAVING TO A FILE?

- Databases can handle large amounts of data efficiently and store it compactly
- They provide tools for easy insertion, querying, and updating of data
- They generally offer security features and control over access to data
- They (generally) scale well.

SQL VS. NOSQL

SQL DATABASES

Structured Query Language databases are relational databases. We pre-define a schema of tables before we insert anything.

NO-SQL DATABASES

NoSQL databases do not use SQL There are many types of no-sql databases, including document, key-value, and graph stores.

POPULAR DATABASES

SQL DATABASES

- MySQL
- Postgres
- SQLite
- Oracle
- Microsoft SQL Server

NO-SQL DATABASES

- MongoDB
- Couch DB
- Neo4j
- Cassandra
- Redis

WHY ARE WE LEARNING MONGO?

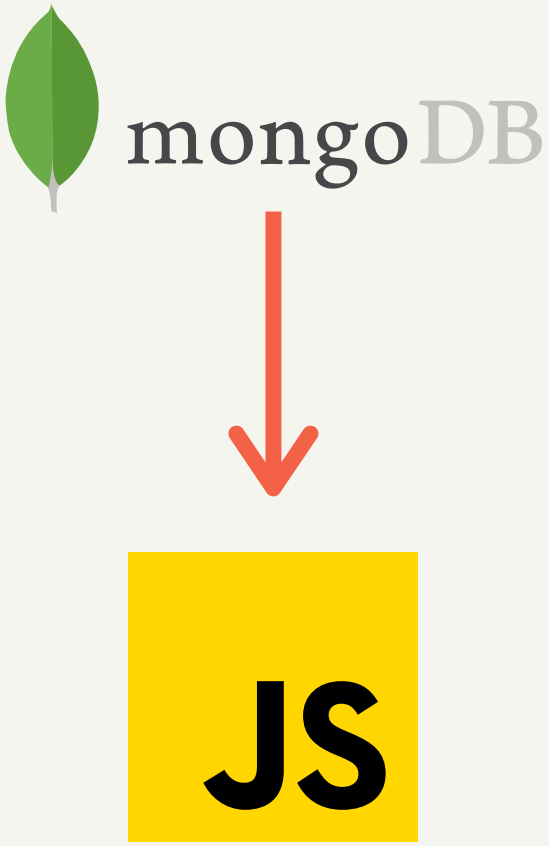
- Mongo is very commonly used with Node and Express (MEAN & MERN stacks)
- It's easy to get started with (though it can be tricky to truly master)
- It plays particularly well with JavaScript
- Its popularity also means there is a strong community of developers using Mongo.

INSTALLING MONGO DB


Mongoose

Express + Mongoose

WEB DEVELOPER BOOTCAMP



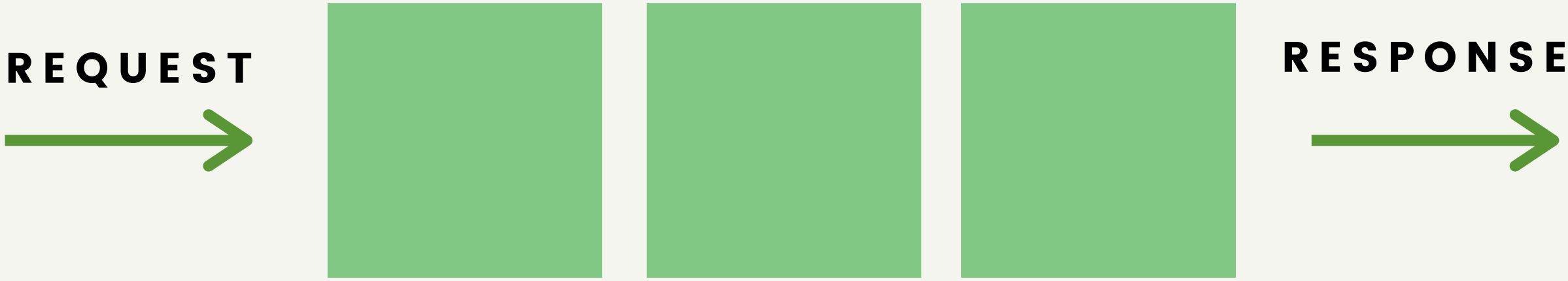
ODM

OBJECT DATA MAPPER?
OBJECT DOCUMENT MAPPER?

ODMs like Mongoose map documents coming from a database into usable JavaScript objects.

Mongoose provides ways for us to model out our application data and define a schema. It offers easy ways to validate data and build complex queries from the comfort of JS.

MIDDLEWARE



Express middleware are functions that run during the request/response lifecycle.

MIDDLEWARE

REQUEST

- Middleware are just functions
- Each middleware has access to the request and response objects
- Middleware can end the HTTP request by sending back a response with methods like res.send()
- OR middleware can be chained together, one after another by calling next()

RESPONSE

ERRORS!

How do we handle errors in an Express application?