MERN PART-TIME ONLINE

MongoDB

MongoDB

Mongoose

Express + Mongoose

Mongoose Commands

Validations

Quiz I

Jokes API

Nested Documents

Chapter Survey

Nested Documents

Learning Objectives

- Understand the basics of Nested Documents
- Understand how to determine use cases for nesting documents
- Understand the alternative to nesting documents

Because MongoDB stores data as JSON objects, we have the ability to store any data type (including arrays and nested objects) we want inside of one single object. This helps make nesting documents so simple! A nested document is a mongoose schema that contains a field with the value being a completely separate mongoose schema. For example:

A super simple example of a User schema that can have many friends might look like this:

```
const UserSchema = new mongoose.Schema({
  fName: String,
  lName: String,
  friends: [UserSchema]
});
```

Nesting documents can look super simple but we need to take caution because it can become real easy to store duplicate data in our database.

Let's run through a little more complex example. Say we are creating a Bank account application. Our User schema would look like something like this:

```
const UserSchema = new mongoose.Schema(
  {
    fName: String,
    lName: String,
    email: String,
    password: String,
    bankAccounts: [BankAccountSchema]
  },
  { timestamps: true }
);
```

./user.model.js

Since one user can have many bank accounts, we determine that relationship by nesting the `BankAccountSchema` into an array. This basically means the field `bankAccounts` will be an array of nothing but BankAccount objects.

Since each bank account can have many transactions, we can nest a TransactionSchema into our BankAccountSchema:

```
const TransactionSchema = new mongoose.Schema(
  {
    amount: { type: Number, required: true },
    vender: { type: String, required: true }
  },
  { timestamps: { createdAt: true } }
);

const BankAccountSchema = new mongoose.Schema(
  {
    accountType: { type: String, required: true },
    balance: { type: Number, default: 0 },
    transactions: [TransactionSchema]
  },
  { timestamps: true }
);
```

./bank.model.js

Previous

Next

Privacy Policy