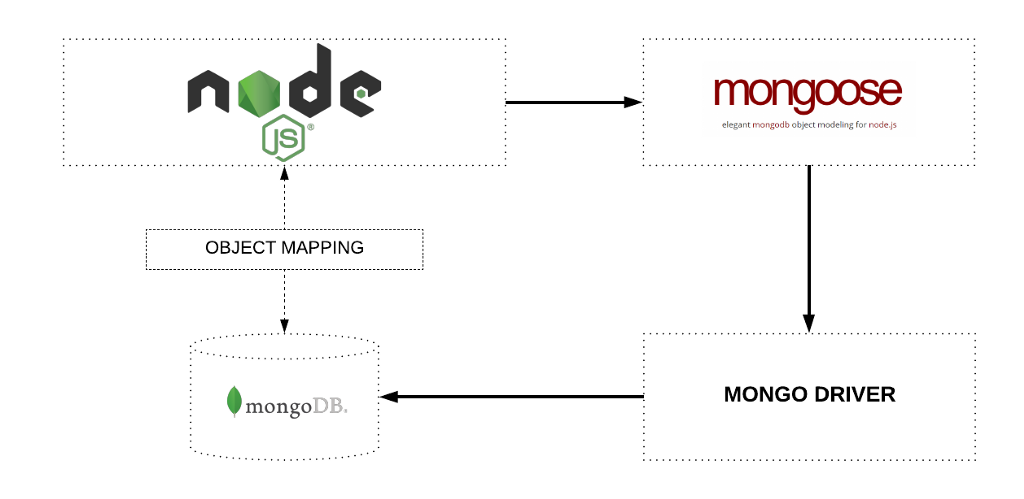
**MODULE MONGOOSE**

I – CÀI ĐẶT & CẤU HÌNH

**1. Mongoose là gì ?**

Mongoose là một thư viện mô hình hóa đối tượng (Object Data Model - ODM) cho MongoDB và Node.js. Nó quản lý mối quan hệ giữa dữ liệu, cung cấp sự xác nhận giản đồ và được sử dụng để dịch giữa các đối tượng trong mã và biểu diễn các đối tượng trong MongoDB.



**2. Cài đặt Module Mongoose**

|  |
| --- |
| npm install mongoose --save |

II – MODEL TRONG MONGOOSE

**1. Model là gì**

Model nói chung, là nơi cung cấp cho chúng ta môi trường, hoặc một giao diện cho CSDL để thực hiện các thao tác như: truy vấn, thêm, sửa, xóa dữ liệu

**2. Mongoose Model**

Mongoose Model cũng mô phỏng lại chính xác giao diện cho CSDL như Model

***Chú ý:***

* *Model là một khái niệm hết sức trừu tượng và vô hình, chủ yếu chúng ta tạo ra một tổ chức code để thao tác với CSDL trong đó thì gọi là Model, còn mỗi một hệ thống khác nhau, mỗi một mô hình khác nhau thì cách làm việc với Model cũng sẽ linh hoạt và khác nhau*

III – MONGOOSE SCHEMA

**1. Mongoose Schema là gì**

*Mongoose schema được sử dụng để xác định cấu trúc dữ liệu của một Collection.*

*Trong thực tế, mỗi một Model sẽ được sử dụng để thao tác dữ liệu với một đối tượng dữ liệu cụ thể (Collection), ví dụ: Model User quản lý Collection User, Model Product quản lý Collection Product,… và Schema sinh ra để mô tả cấu trúc dữ liệu của Collection mà Model đó đang quản lý*

**2. Schema Type**

Schema Type chính là các kiểu dữ liệu được sử dụng trong Mongose để mô tả cho dữ liệu được lưu trữ

* String: chuỗi
* Number: số
* Date: ngày-tháng-năm
* Buffer
* Boolean: logic
* Mixed
* ObjectId: đối tượng ID
* Array: mảng
* Decimal128
* Map

**3. Khai báo một Schema**

**Cú pháp:**

|  |
| --- |
| const mongoose = require("../../../common/database")()  const collectionSchema = new mongoose.Schema({  Property: Type of Value,  Property: Type of Value,  ...    Property: Type of Value  })  const ModelName = mongoose.model("SchemaName", collectionSchema, "CollectionName")  module.exports = ModelName |

**Trong đó:**

* Property: Thuộc tính của Collection
* Type of Value: Kiểu dữ liệu để lưu trữ giá trị tương ứng cho thuộc tính trên
* Phương thức Schema() được sử dụng để quản lý Collection
* Phương thức model() nhận vào 3 tham số
  + Tham số thứ nhất: Tên được đặt cho Schema mới tạo, giống như một tham chiếu tới Schema khi Schema được gọi tới sử dụng (tên này đặt tùy ý)
  + Tham số thứ hai: Đối tượng Schema mới được khởi tạo
  + Tham số thứ ba: Tên Collection mà Schema quản lý

IV – SỬ DỤNG MONGOOSE TRONG NODEJS

1. Mô hình và luồng dữ liệu

**2. Các bước sử dụng Mongoose Model cho NodeJS**

**Bước 1:** Kết nối

Tạo file database.js trong thư mục common, file này sẽ lưu quá trình kết nối tới Mongoose để sử dụng chung cho các file cần sử dụng

|  |
| --- |
| const mongoose = require("mongoose")  module.exports = ()=>{    mongoose.connect('mongodb://127.0.0.1:27017/vietpro\_mobile\_shop', {useNewUrlParser: true})  return mongoose  } |

**Bước 2:** Tạo Model Product

Tạo file product.model.js Model theo đường dẫn apps/models/admin/ product.model.js để tạo ra giao diện Model giúp xử lý truy vấn, thêm, sửa, xóa dữ liệu sau này

**Bước 3:** Quản lý Product Model bằng Schema

Trong file Model product.model.js chúng ta xây dựng cấu trúc Collection Product bằng cách sử dụng đối tượng Schema để quản lý Collection này

|  |
| --- |
| const mongoose = require("../../../common/database")()  const productSchema = new mongoose.Schema({  cat\_id: mongoose.Schema.ObjectId,  prd\_name: String,  prd\_image: String,  prd\_price: String,  prd\_warranty: String,  prd\_accessories: String,  prd\_new: String,  prd\_promotion: String,  prd\_status: Number,  prd\_featured: Number,  prd\_details: String  })  const Product = mongoose.model("Product", productSchema, "Product")  module.exports = Product |

**Trong đó:**

* Phương thức Schema() được sử dụng để quản lý Collection
* Kiểu dữ liệu ObjectId là kiểu dữ liệu đặc biệt phải được lấy ra từ đối tượng Schema của mongoose
* Phương thức model() nhận vào 3 tham số
  + Tham số thứ nhất: Tên được đặt cho Schema mới tạo, giống như một tham chiếu tới Schema khi Schema được gọi tới sử dụng (tên này đặt tùy ý)
  + Tham số thứ hai: Đối tượng Schema mới được khởi tạo
  + Tham số thứ ba: Tên Collection mà Schema quản lý

**Bước 4:** Sử dụng Model

Sử dụng Product Controller để gọi tới Product Model

|  |
| --- |
| const ProductModel = require("../../models/admin/product.model") |

IV – QUERIES

**1. Truy vấn lọc dữ liệu**

1. find()

find() cho phép lọc ra tất cả các Document trong một Collection

Cú pháp :

|  |
| --- |
| ProductModel.find({}, (err, docs)=>{  console.log(docs)  }) |

Trong đó :

* ProductModel: Đối tượng được lấy từ Model qua Controller để thao tác dữ liệu.
* err: Tham số lưu trữ thông báo lỗi mặc định của module Mongoose cho trường hợp truy vấn dữ liệu chính xác nhưng không có dữ liệu trả về
* docs : Tham số lưu trữ data trả về dạng mảng, mỗi phần tử của mảng sẽ lưu trữ một Document (nếu chỉ có duy nhất 1 Document trả về thì data sẽ ở dạng đối tượng lưu trữ duy nhất Document đó)

1. findById()

findById() cho phép lọc ra một Document theo \_id

**Cú pháp:**

|  |
| --- |
| ProductModel.findById(\_id, (err, docs)=>{  console.log(docs)  }) |

**Trong đó:**

* \_id: Chính là ObjectID, tuy nhiên Mongoose đã tự động chuyển đổi từ Object sang String, nên chỉ cần truyền vào ID dạng chuỗi

**2. Thêm mới dữ liệu**

**Cú pháp :**

|  |
| --- |
| ProductInsert = new ProductModel({  cat\_id: "5db7f9faa9da0856c7a4c631",  prd\_name: "iPhone 11",  prd\_image: "iphone-11.png",  prd\_price: "20000000",  prd\_warranty: "12 Tháng",  prd\_accessories: "Sách, sạc, tai nghe",  prd\_new: "Mới 100%",  prd\_promotion: "Tấm dán màn hình 4D",  prd\_status: 1,  prd\_featured: 1,  prd\_details: " iPhone 11 chính hãng"  }, {versionKey: false})  ProductInsert.save() |

**Trong đó :**

* versionKey: false : loại bỏ thông tin \_\_v: 0 mỗi khi thêm mới một Document (đây là một tính năng mặc định trong Mongoose)

3. Sửa dữ liệu

1. updateOne()
2. updateMany()

4. Xóa dữ liệu

1. deleteOne()
2. deleteMany()

V – POPULATE

1. Join