BE NodeJS

Tugas: Mongoose

Leonardho R. Sitanggang

Soal :

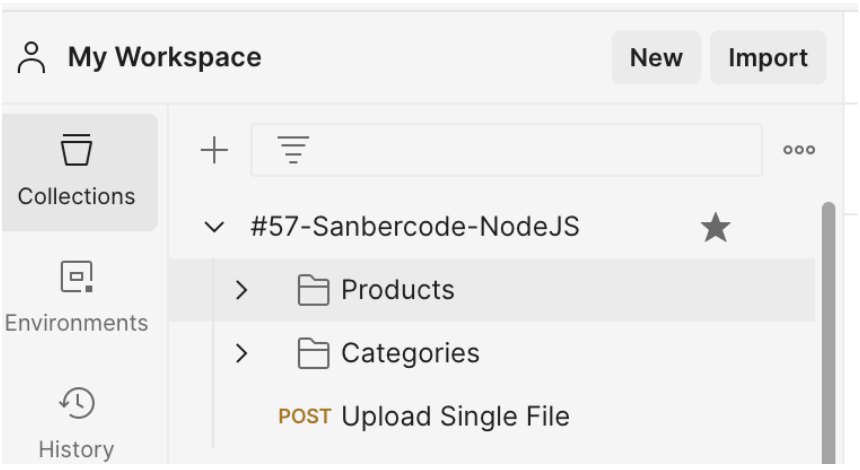
Gunakan source code materi 7,

Buatlah file di folder models, dengan nama **categories.model.ts** , kemudian isi dengan kode berikut ini untuk membuat schema categories

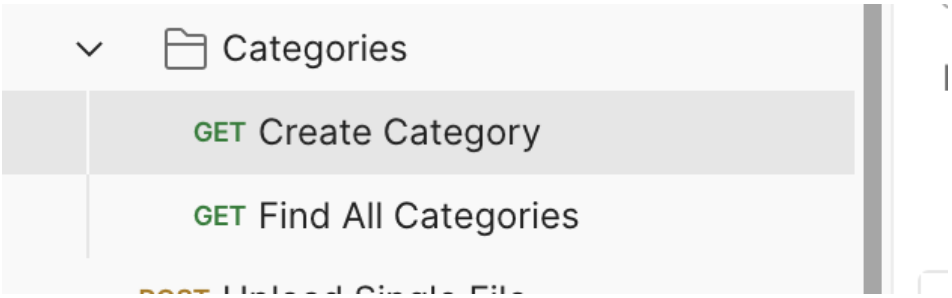
Pastikan sudah import categories.controller di bagian atas file routes.ts

Jalankan server dengan perintah `npm run dev`

Buka postman, kemudian buatlah struktur collection seperti ini



Di dalam folder categories buatlah beberapa request contohnya seperti ini

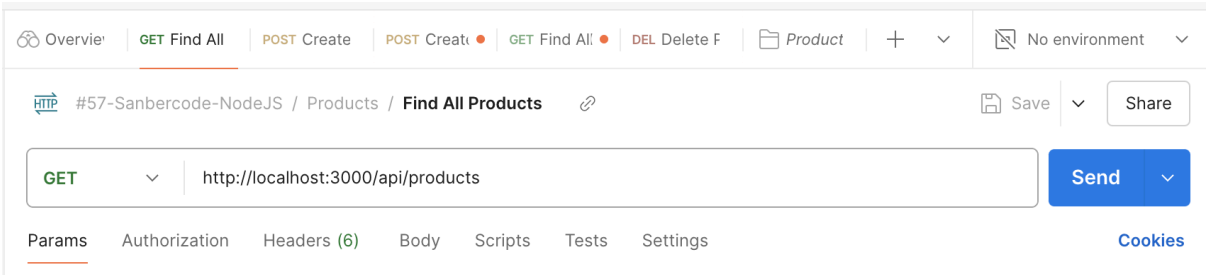


Lakukan proses CRUD pada data Category

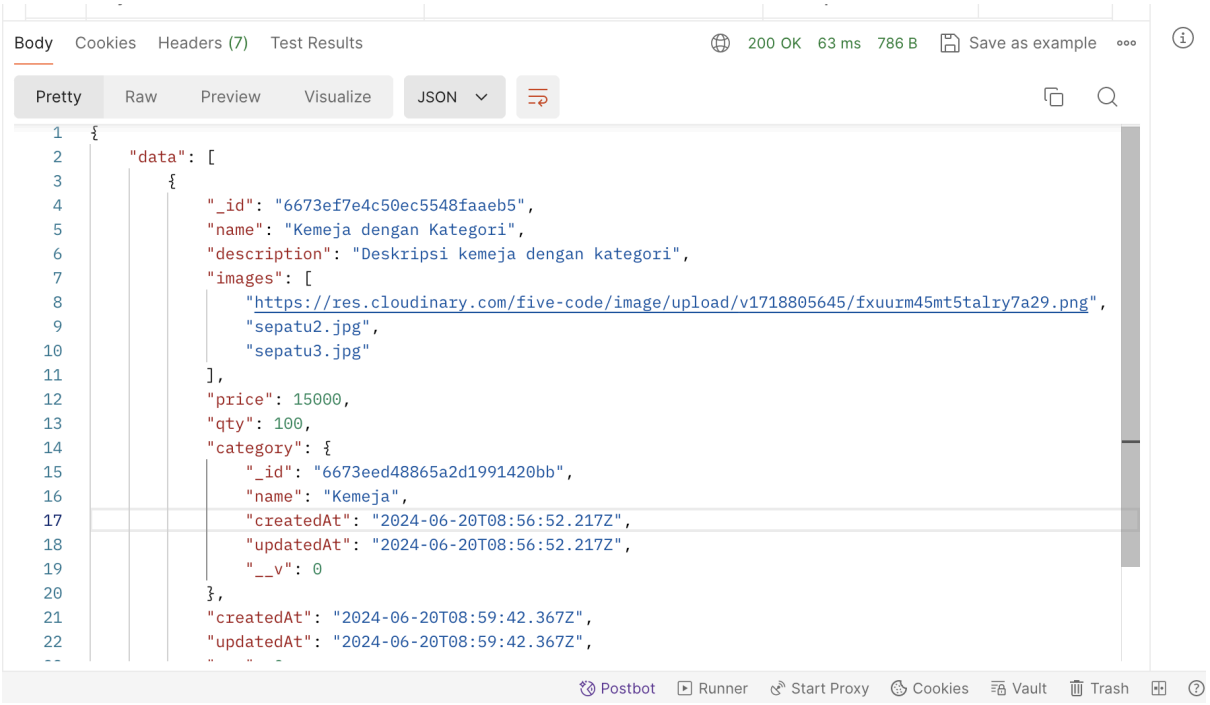
Kemudian, buatlah data **Product** dengan JSON seperti ini :

```
{
    "name": "Kemeja dengan Kategori",
    "description": "Deskripsi kemeja dengan kategori",
    "images":
["https://res.cloudinary.com/five-code/image/upload/v1718805645/fxuurm
45mt5talry7a29.png", "sepatu2.jpg", "sepatu3.jpg"],
    "price": 15000,
    "qty": 100,
    "category": "-- isi dengan _id dari salah satu category --"
}
```

Gunakan endpoint **Find All Products** untuk menampikan list data Products



Pastikan hasil dari endpoint ini menampilkan data seperti pada contoh di bawah ini:



di dalam data product dapat menampilkan isi dari **category**.

Jawab :

Source Code

```
16  router.get("/categories", categoriesController.findAll);
17  router.post("/categories", categoriesController.create);
18  router.get("/categories/:id", categoriesController.findOne);
19  router.put("/categories/:id", categoriesController.update);
20  router.delete("/categories/:id", categoriesController.delete);
```

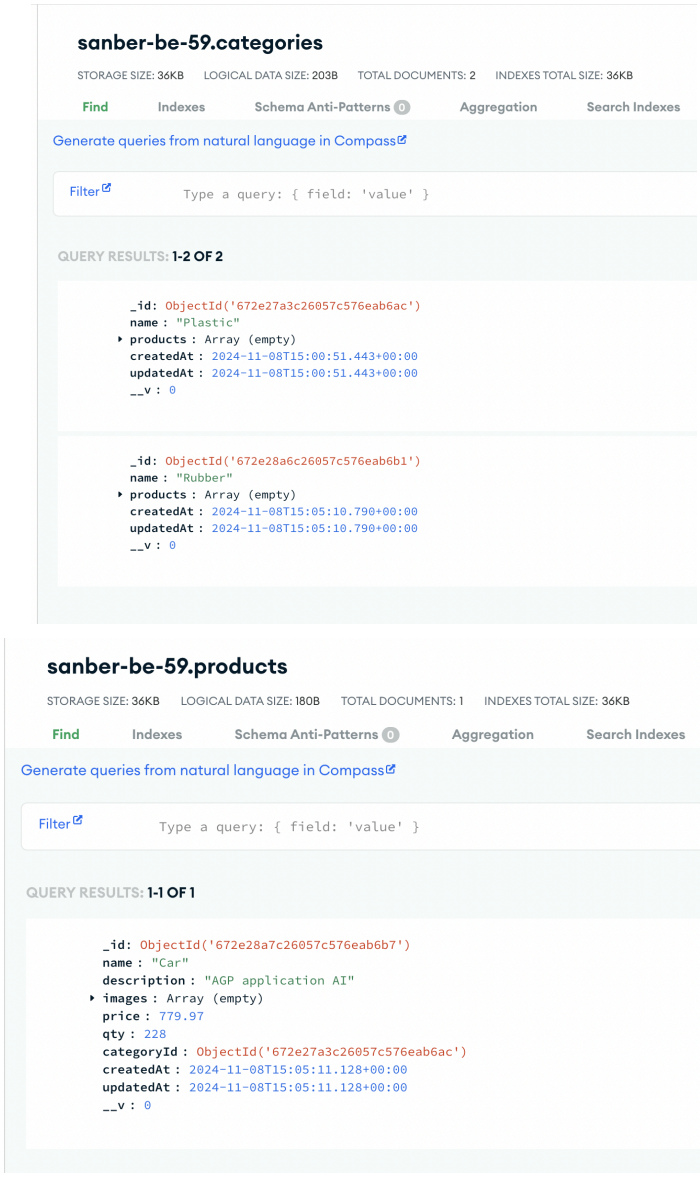
Gambar 1.1 Route Categories

```
export default {
    async create(req: Request, res: Response) {
        try {
            const result = await CategoriesModel.create(req.body);
            res.status(201).json({
                data: result,
                message: "Success create product",
            });
        } catch (error) {
            const err = error as Error;
            res.status(500).json({
                data: err.message,
                message: "Failed create product",
            });
        }
    },
    async findAll(req: Request, res: Response) {
        try {
            const result = await CategoriesModel.find();
            res.status(200).json({
                data: result,
                message: "Success get all products",
            });
        } catch (error) {
            const err = error as Error;
            res.status(500).json({
                data: err.message,
                message: "Failed get all products",
            });
        }
    },
    async findOne(req: Request, res: Response) {
        try {
            const result = await CategoriesModel.findOne({
                _id: req.params.id,
            });
            res.status(200).json({
                data: result,
                message: "Success get one product",
            });
        } catch (error) {
            const err = error as Error;
            res.status(500).json({
                data: err.message,
                message: "Failed get one product",
            });
        }
    },
```

Gambar 1.2 Controller Create, Read

```
52      async update(req: Request, res: Response) {
53          try {
54              const result = await CategoriesModel.findOneAndUpdate(
55                  { _id: req.params.id },
56                  req.body,
57                  {
58                  new: true,
59                  }
60              );
61
62              res.status(200).json({
63                  data: result,
64                  message: "Success update product",
65              });
66          } catch (error) {
67              const err = error as Error;
68              res.status(500).json({
69                  data: err.message,
70                  message: "Failed update product",
71              });
72          }
73      },
74      async delete(req: Request, res: Response) {
75          try {
76              const result = await CategoriesModel.findOneAndDelete({
77                  _id: req.params.id,
78              });
79
80              res.status(200).json({
81                  data: result,
82                  message: "Success delete product",
83              });
84          } catch (error) {
85              const err = error as Error;
86              res.status(500).json({
87                  data: err.message,
88                  message: "Failed delete product",
89              });
90          }
91      },
```

Gambar 1.3 Controller Update, Delete

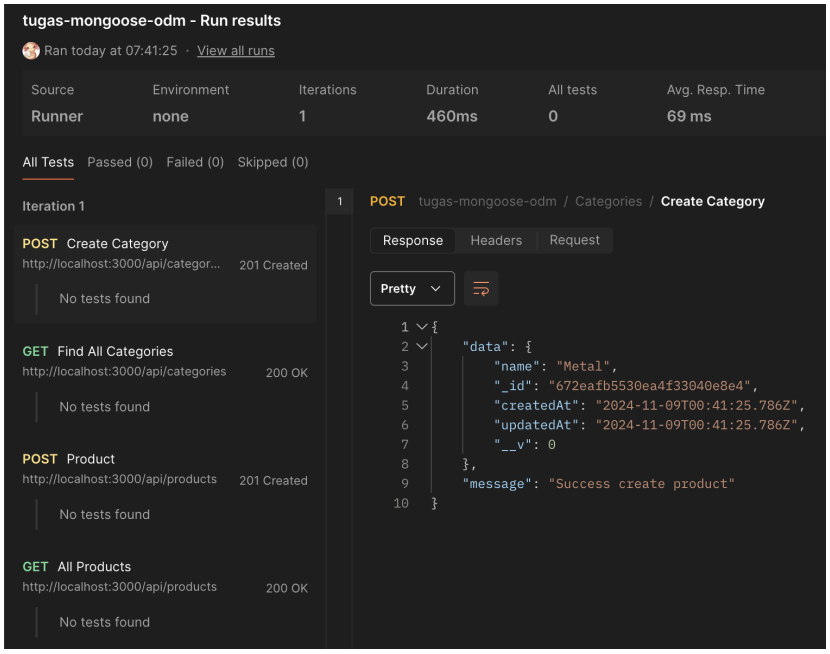


Gambar 2.1 Evidence Database MongoDB

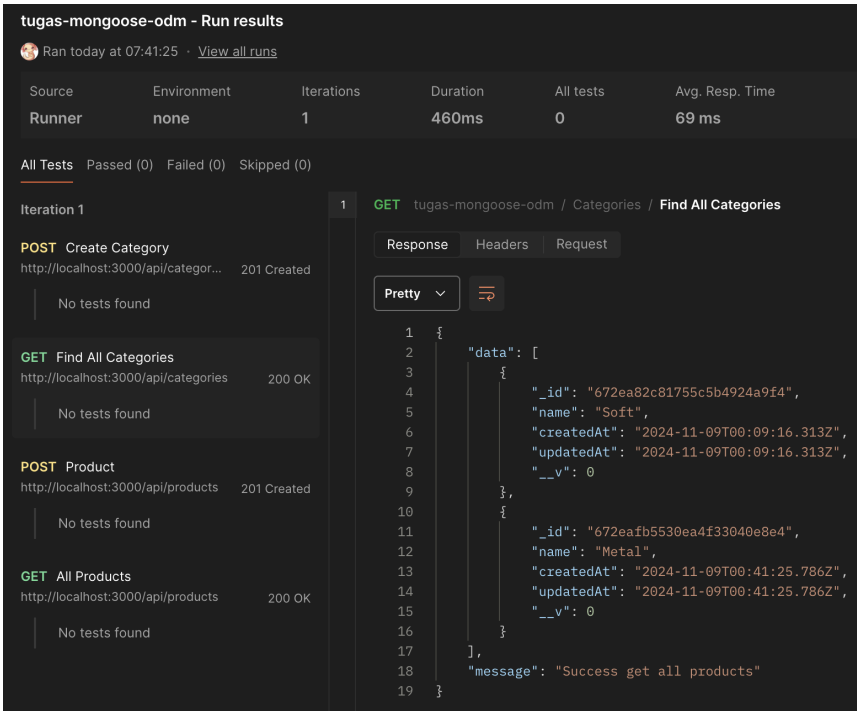
```
const ProductsSchema = new Schema(
  {
    name: {
      type: String,
      required: true,
    },
    description: {
      type: String,
      required: true,
    },
    images: {
      type: [String],
      required: true,
    },
    price: {
      type: Number,
      required: true,
    },
    qty: {
      type: Number,
      required: true,
      min: [1, "Quantity can not be less than 1"],
    },
    category: {
      type: mongoose.Schema.Types.ObjectId,
      ref: "Categories",
    },
  },
  {
    timestamps: true,
  }
);

const ProductsModel = mongoose.model("Products", ProductsSchema);
```

```
const CategoriesSchema = new Schema(
    {
        name: {
            type: String,
            required: true,
        },
    },
    {
        timestamps: true,
    }
);

const CategoriesModel = mongoose.model("Categories", CategoriesSchema);

export default CategoriesModel;
```

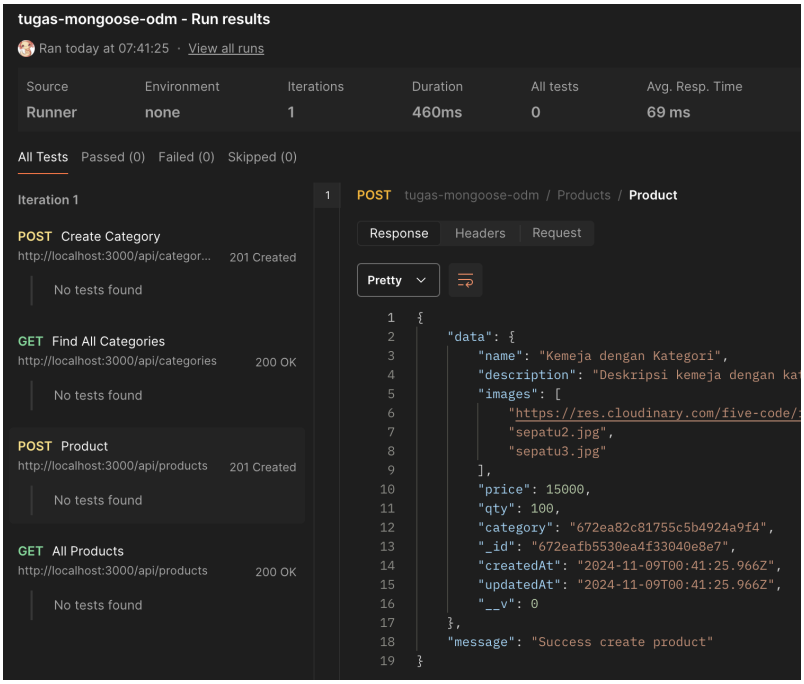
Gambar 2.2 Model Products & Categories



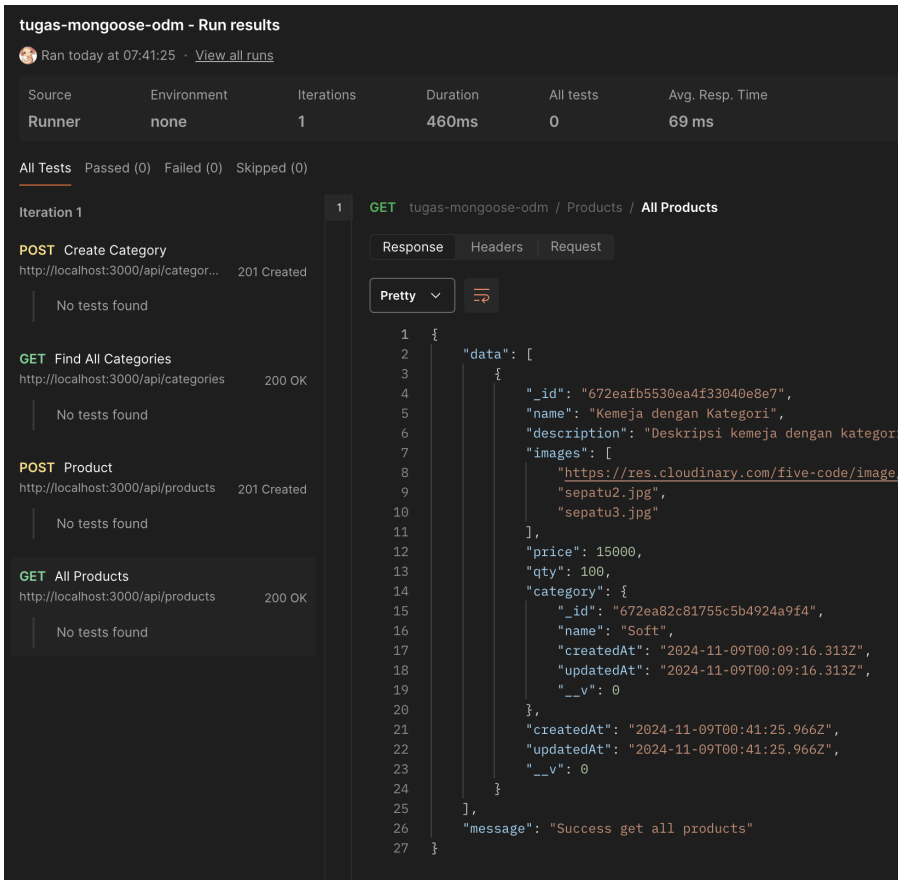
Gambar 3.1 Test Result Post Category



Gambar 3.2 Test Result Get All Categories



Gambar 3.3 Test Result Post Product with Category Relation



Gambar 3.4 Test Result Get All Product with Category Relation

Repository :

- *Github https://github.com/FlazeFy/sanbercode-nodejs-xpress/tree/main/tugas-mongoose-odm*
- *Postman Documentation https://documenter.getpostman.com/view/18882129/2sAY52bz5m*

Refrensi :

- *Materi Pekan 2.1 Framework Express JS https://sanbercode.com/bootcamp/class/684/materi/2-1-framework-expressjs/dc0618b7-fd52-4eec-a578-5ed88c4d5b3c*
- *Materi Pekan 2.2 Routing Rest API https://sanbercode.com/bootcamp/class/684/materi/routing-rest-api/de70a7be-9056-40e0-bed0-c280c6bc1de0*
- *Materi Pekan 2.3 Middleware Express https://sanbercode.com/bootcamp/class/684/materi/2-3-middleware-expressjs/c33672de-a33d-4916-b07c-9221b9e836b5*
- *Materi Pekan 2.4 Database MongoDB https://sanbercode.com/bootcamp/class/684/materi/2-4-database-mongodb/ae319daf-6cf4-4282-8fe2-0c21c4da0d13*
- *Materi Pekan 2.5 Mongoose https://sanbercode.com/bootcamp/class/684/materi/2-6-video-mongoose/f919e6e2-ec3e-4888-aa76-1fb85c552df2*