Mongoose Seeding

Full Stack Web Development

Motivation for Database Seeding

- Pre-populating the database can enhance developer productivity
- It facilitates simple exploring of various scenarios
- Is particularly useful in establishing normalised data models with inter-document relationships
- Can also be used to pre-configure database for production

Mongoose seeders on NPM

- Variety of modules available
- Most fairly simple

mongoose-seed public

Seed data population for Mongoose

mongoose-seed lets you populate and clear MongoDB documents with all the benefits of Mongoose validation

mongoose-seed-plus public

Seed data population for Mongoose

mongoose-seed-plus lets you populate and clear MongoDB documents with all the benefits of Mongoose validation

https://github.com/ermaxnet/mais-mongoose-seeder

- mongoose-seeder loads from an enhanced JSON file
- Includes special notation for loading relationships between documents

ermaxnet / mais-mongoose-seeder

Watch 2 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights

Seed your MongoDB database easily with mongoose 5.x

74 commits | 1 branch | 0 releases | 2 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

Max Eremin fix bug — Latest commit e0afa1f on 12 Jun 2018

.gitignore	refactoring	9 months ago
CHANGELOG.md	fix bug	8 months ago
LICENSE	Initital commit	9 months ago
README.md	fix some issued bugs	9 months ago
index.js	fix bug	8 months ago
package.json	fix bug	8 months ago

README.md

mais-mongoose-seeder

I made this package for academic purposes and it is based on Sam Verschueren's mongoose-seeder package. It happened so that I have use Sam's package which doesn't work with current mongoose version. So, I decided to refactor it so that it can work with most mongoose versions as users need. And also, I deleted all the dependencies from the package so that no half npm is installed at the same time.

Thanks to Sam for his great job! He wrote one perfect package in 2015.

I do not in any way pretend to be authorship, since what I did it is basically just a refactoring. But I post this package in open access in case there is a need for someone else to use the mongoose-seeder package.

This library offers a nice, clean and elegant solution that will create the dummy data objects from a JSON file.

Install

user.js

```
import Mongoose from "mongoose";
const { Schema } = Mongoose;

const userSchema = new Schema({
  firstName: String,
  lastName: String,
  email: String,
  password: String,
});

export const User = Mongoose.model("User", userSchema);
```

Collection

Schema/Model

Document

seed-data.json

```
export const seedData = {
  users: {
    _model: "User",
    homer: {
      firstName: "Homer",
      lastName: "Simpson",
      email: "homer@simpson.com",
      password: "secret"
    },
    marge: {
      firstName: "Marge",
      lastName: "Simpson",
      email: "marge@simpson.com",
      password: "secret"
    },
    bart: {
      firstName: "Bart",
      lastName: "Simpson",
      email: "bart@simpson.com",
      password: "secret"
    }
  },
};
```

seed function

- On application startup:
  - Delete any existing data in the collections
  - Populate the database on initial connection during startup

```
import { seedData } from "./seed-data.js";
import * as seeder from "mais-mongoose-seeder";

const seedLib = seeder.default;

async function seed() {
  const seeder = seedLib(Mongoose);
  const dbData = await seeder.seed(seedData, { dropDatabase: false, dropCollections: true });
  console.log(dbData);
}
```

seed function

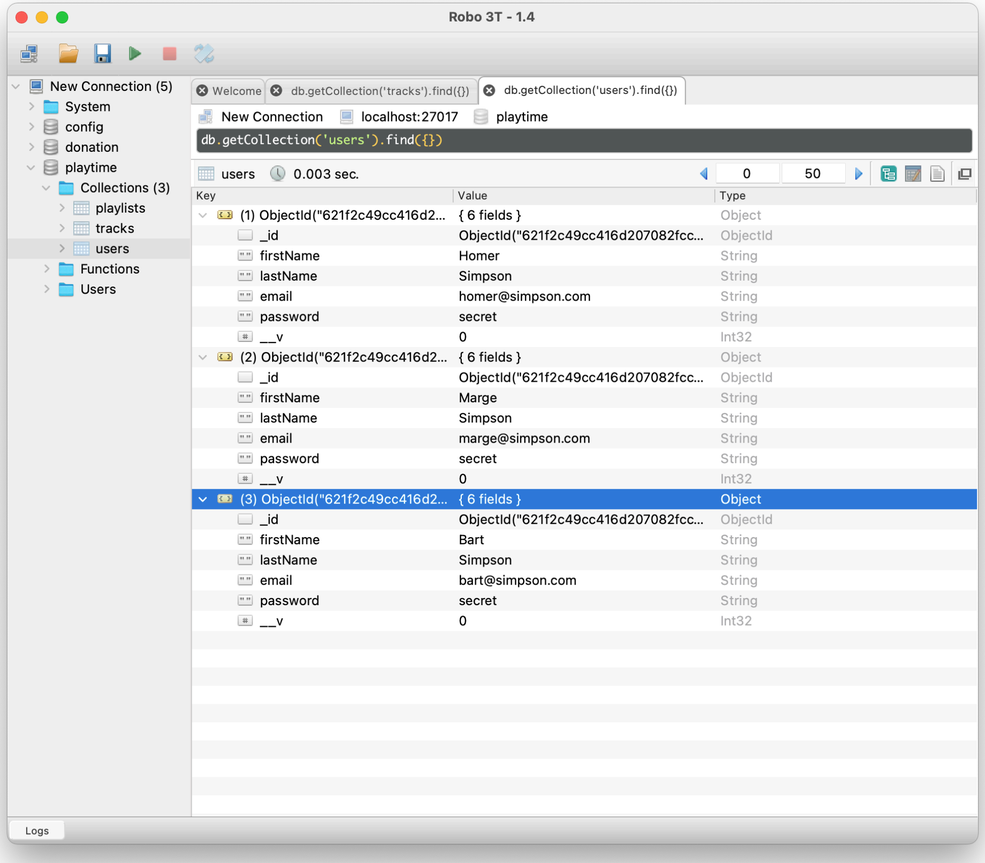
- Call the seed function;
  - pass ‘data’ loaded from JSON file
  - options:
    - keep the database
    - delete contents of all collections

```
const dbData = await seeder.seed(seedData, { dropDatabase: false, dropCollections: true });
```

Call seed on db connect

```
db.once("open", function() {
  console.log(`database connected to ${this.name} on ${this.host}`);
  seed();
});
```

```
export const seedData = {
  users: {
    _model: "User",
    homer: {
      firstName: "Homer",
      lastName: "Simpson",
      email: "homer@simpson.com",
      password: "secret"
    },
    marge: {
      firstName: "Marge",
      lastName: "Simpson",
      email: "marge@simpson.com",
      password: "secret"
    },
    bart: {
      firstName: "Bart",
      lastName: "Simpson",
      email: "bart@simpson.com",
      password: "secret"
    }
  },
};
```




```
{
  users: {
    homer: {
      firstName: 'Homer',
      lastName: 'Simpson',
      email: 'homer@simpson.com',
      password: 'secret',
      _id: new ObjectId("621f2bf90f8832d1b1b3630a"),
      __v: 0
    },
    marge: {
      firstName: 'Marge',
      lastName: 'Simpson',
      email: 'marge@simpson.com',
      password: 'secret',
      _id: new ObjectId("621f2bf90f8832d1b1b3630c"),
      __v: 0
    },
    bart: {
      firstName: 'Bart',
      lastName: 'Simpson',
      email: 'bart@simpson.com',
      password: 'secret',
      _id: new ObjectId("621f2bf90f8832d1b1b3630e"),
      __v: 0
    }
  }
}
```

Object References

```
export const seedData = {
  users: {
    _model: "User",
    homer: {
      firstName: "Homer",
      lastName: "Simpson",
      email: "homer@simpson.com",
      password: "secret"
    },
    marge: {
      firstName: "Marge",
      lastName: "Simpson",
      email: "marge@simpson.com",
      password: "secret"
    },
    bart: {
      firstName: "Bart",
      lastName: "Simpson",
      email: "bart@simpson.com",
      password: "secret"
    }
  },
```

- User Reference
- Playlist Reference

```
  playlists: {
    _model: "Playlist",
    mozart: {
      title: "Mozart Favourites",
      userid: "->users.bart"
    }
  },
  tracks: {
    _model : "Track",
    track_1 : {
      title: "Violin Concerto No. 1",
      artist: "Mozart",
      duration: 15,
      playlistid: "->playlists.mozart"
    },
  }
};
```

Object Log

```
{
  users: {
    homer: {
      firstName: 'Homer',
      lastName: 'Simpson',
      email: 'homer@simpson.com',
      password: 'secret',
      _id: new ObjectId("621f2dd4425b1754cc840a80"),
      __v: 0
    },
    marge: {
      firstName: 'Marge',
      lastName: 'Simpson',
      email: 'marge@simpson.com',
      password: 'secret',
      _id: new ObjectId("621f2dd5425b1754cc840a82"),
      __v: 0
    },
    bart: {
      firstName: 'Bart',
      lastName: 'Simpson',
      email: 'bart@simpson.com',
      password: 'secret',
      _id: new ObjectId("621f2dd5425b1754cc840a84"),
      __v: 0
    }
  },
```

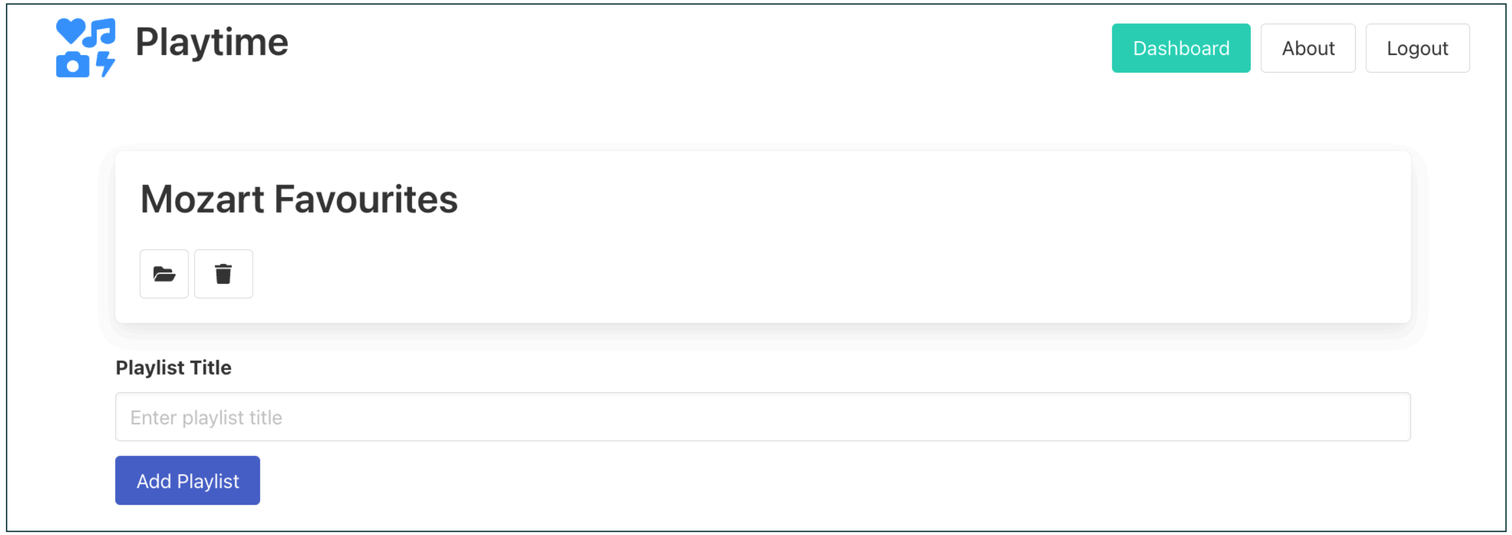
User Reference

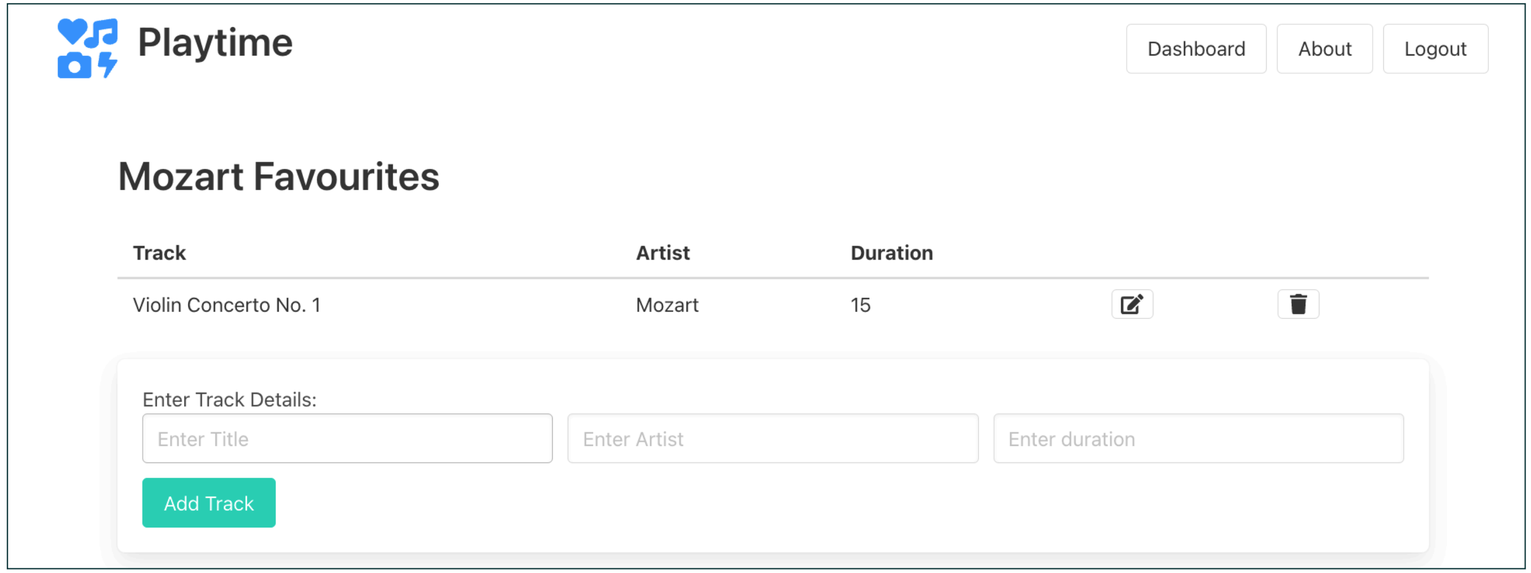
Playlist Reference

```
  playlists: {
    mozart: {
      title: 'Mozart Favourites',
      userid: new ObjectId("621f2dd5425b1754cc840a84"),
      _id: new ObjectId("621f2dd5425b1754cc840a86"),
      __v: 0
    }
  },
  tracks: {
    track_1: {
      title: 'Violin Concerto No. 1',
      artist: 'Mozart',
      duration: 15,
      playlistid: new ObjectId("621f2dd5425b1754cc840a86"),
      _id: new ObjectId("621f2dd5425b1754cc840a88"),
      __v: 0
    }
  }
}
```

```
export const seedData = {
  users: {
    _model: "User",
    homer: {
      firstName: "Homer",
      lastName: "Simpson",
      email: "homer@simpson.com",
      password: "secret"
    },
    marge: {
      firstName: "Marge",
      lastName: "Simpson",
      email: "marge@simpson.com",
      password: "secret"
    },
    bart: {
      firstName: "Bart",
      lastName: "Simpson",
      email: "bart@simpson.com",
      password: "secret"
    }
  },
  playlists: {
    _model: "Playlist",
    mozart: {
      title: "Mozart Favourites",
      userid: "->users.bart"
    }
  },
  tracks: {
    _model : "Track",
    track_1 : {
      title: "Violin Concerto No. 1",
      artist: "Mozart",
      duration: 15,
      playlistid: "->playlists.mozart"
    },
  }
};
```

UX





https://medium.com/@pkosiec/seeding-mongodb-database-the-right-way-32a8a0e75490

Seeding MongoDB database the right way

Paweł Kosiec Nov 3, 2018 · 7 min read

How many projects which used MongoDB have you worked on? How do you test your database queries? What is your way of the initial database seeding?

About year ago, my friend and I started building a basic CRUD app as a side project. On the back-end side there was nothing fancy — a simple GraphQL API server written in TypeScript. When it comes to storing the data, it was a typical use case for NoSQL database, so we've picked the most popular solution out there — MongoDB.

At some point we needed to import some development data to test our database queries. There are many tools for MongoDB data import, including the official one, `mongoimport`. I started to do a research and it turned out that none of them is good enough. Every single solution had three flaws described below.

Mongo Seeding

Mongo Seeding

release v3.6.0 | build passing | License MIT

The ultimate solution for populating your MongoDB database 🚀

Define MongoDB documents in JSON, JavaScript or even TypeScript files. Use JS library, install CLI or run Doc image to import them!

Introduction

Mongo Seeding is a flexible set of tools for importing data into MongoDB database.

It's great for:

- testing database queries, automatically or manually
- preparing ready-to-go development environment for your application
- setting initial state for your application

Mongoose Seeding

Full Stack Web Development