Mongoose

Getting started with mongoose:

1) Easily Develop Node.js and MongoDB Apps with Mongoose

This first tutorial is very basic (no client-server involved, just plain database). It does however explain important topics nicely as it proceeds.

https://scotch.io/tutorials/using-mongoosejs-in-node-js-and-mongodb-applications

2) Re-write the plain-mongoDB exercise "Our very first MEAN application" to use mongoose.

a) Clone this project as the start code for this exercise: https://github.com/Lars-m/mongooseExercise.git

b) It provides a simple Express project, meant for a REST-only project (no server side templating).

1. Run npm install
2. Start your local MongoDB database. The project will make a connection (and create, if necessary) to a collection `demoJokes`. See db.mongooseConnect.js + ServerStart.js.
3. Run `npm test`:
   This will start the Server and run five REST-tests, using Mocha, Chai, and the node-fetch package to perform the HTTP-requests.
   This will execute tests up against a test collection `testJokeDB`, and not the "normal" development collection used when you run npm start.
   All five tests will initially fail. Your task is to make them pass.
4. Start the server via `npm start`. It won't do anything until you implement the required REST-API.

c) Design a schema matching this json-object:

```
{
  "joke" : " Reality is an illusion created by a lack of alcohol",
  "category" : ["short", "alcohol", "quote"],
  "reference": { "author" : "Someone", "link" : ""},
  "lastEdited" : new Date()
},
```

Ensure:

- The `joke` property is required, and the minimum accepted length is 5;
- The `lastEdited` property is initial set to the current time.

Add a middleware function to the schema that "automatically" will set the lastEdited value, whenever a document is changed.

d) Implementing the REST-API (in `api/api.js`) that will make the test pass in the order:

- **GET:** `/api/jokes`
- **GET:** `/api/jokes/:id`
- **POST:** `/api/jokes/`
- **PUT:** `/api/jokes/:id`
- **DELETE:** `/api/jokes/:id`

The test cases should provide you with the necessary information about the json to receive and return and expected

status-codes.
e) If not already done in the original exercise, create a simple Angular application using the API. If already done, very it still works with the new backend.

Now you have a full MEAN application, using:

- NodeJS
- Express
- MongoDB
- Mongoose
- AngularJS