Node NestJS Starter Bundle Instructions

Description

This is readme and instructions how start using node backend bundle from Akveo. Backend bundle is integrated solution of Node Backend Code and Angular Frontend code. Backend code plays mostly API role, giving data to the client side as REST API.
Backend part of Node NestJS Starter Bundle. Based on modern typescript Nest (https://github.com/nestjs/nest) framework.
NestJS Documentation (https://docs.nestjs.com/) helps with understanding of NestJS principles.

Running Instruction

1. install mongoDB locally or register mongoDB in some hosting. in case of local node run command: `mongod` in separate terminal

2. setup connection to mongoDB instance in `.env`

```
MONGO_DB_URL = mongodb://localhost:27017/bundle-node-nest
MONGO_DB_NAME = bundle-node-nest
```

3. in folder `backend` run commands

```
npm install
npm run start:dev
```

Test the api by opening `<api_url>` or locally `http://localhost:3001/`
Check the api documentation by opening `<api_url>/swagger` or locally `http://localhost:3001/swagger`

4. in folder `frontend` run commands

```
npm install
npm start
```

5. run `http://localhost:4200`

Test User / Password

You can use these test users for application testing:

1. user@user.user / 12345
2. admin@admin.admin / !2e4S

find more node and NestJS related details in docs or README.md file of node folder.

Issue tracking

You can post issues and see sample materials at GitHub Support Repository (https://github.com/akveo/ngx-admin-bundle-support/issues)

Other Commands

```
# development
$ npm run start

# development with watch mode - code will be rebuild after each change. it runs `nodemon` module to watch over changes and re-run node api automatically.
$ npm run start:dev

# build dist for prod deployment
$ npm run build

# production mode
$ npm run start:prod
```

Test

```
# unit tests
$ npm run test

# e2e tests
$ npm run test:e2e

# test coverage
$ npm run test:cov
```

Style Check

```
# lint
$ npm run lint
```

Features

- NestJS framework, feature based modules
- Express server is used
- Compatible with ngx-admin out of the box
- JWT authentication using Passport module
- MongoDB is used for data storage
- Mongoose is integrated as ORM for data CRUD operations
- Compression setup for API
- Configuration using dotenv module
- Logging to constole and files using Winston module
- API documentation using Swagger for NestJS
- nodemon is used for better experience while develop
- 6 months free updates

API Documentation

You can check API documentation by running api and accessing `http://localhost:3001/swagger` link.
To use swagger with token authentication please follow these steps:

To use swagger with token authentication please follow these steps:

- open swagger link `http://localhost:3001/swagger` while running api
- expand **Auth** controller and open `POST /auth/login` action
- click `try out` and put correct user info into loginDto field (there is sample in swagger). Click `execute`
- when received response with token, copy token `access_token` (ctrl+c)
- click `Authorize` button. Paste there token in format: `Bearer <token>` and click `Authorize`
- after UI was refreshed, you can try any requests, token will be added there

Basic Code Structure

Code is organized in following structure

- Main Folder
    - frontend // this folder contains all UI code
    - backend // server side NestJS code
        - .env // all configuration is here. if file doesn't exist, please create one using `.env.dev` sample
        - src
            - main.ts // api starting point
            - config.ts // configuration reading code is here
            - auth // auth module and code there
            - core // general module independent code
            - user // user module and code there
    - licenses // build version and licenses

NestJS related architecture hints

- Node API based on NestJS has feature based implementation. Each data entity (like 'User' has own folder, module class, controller, service, schema, ...)
- Each entity has 3 mostly similar code structures like: Schema, describing how entity is stored in MongoDB, Interface - used for Mongoose data manipulation (should necessary be interface) and DTO class for usage in Swagger and for exporting entity info as API call result. DTO contain only public data (no ._id field) and decorators for swagger.

Module Sample - Users

- user.module.ts - module settings, has references to other module parts
- user.controller.ts - code of controller routing and security settings
- user.service.ts - code of CRUD operations with user data
- user.schema.ts - interface and schema of data

Support

Please post issues in [Bundle Support Issue Tracker (https://github.com/akveo/ngx-admin-bundle-support/issues)](https://github.com/akveo/ngx-admin-bundle-support/issues)

License

Node NestJS bundle is provided under Single Application or Multi Application license. Please find details in License files.