ngx-admin Backend Bundle Java instruction

Intro

This is readme and instructions how start using backend bundle from Akveo. Backend bundle is integrated solution of Backend Code and Angular Frontend code. Backend code plays mostly API role, giving data to the client side as REST API.

Running Instruction

1. install Java 8 here (https://www.oracle.com/technetwork/java/javase/downloads/index.html) and NodeJs here (https://nodejs.org/en/download/)
2. in folder `java` run spring boot with following command `mvn spring-boot:run`
3. in folder `angular` run commands

```
npm install
npm start
```

4. run `http://localhost:4200`
5. create new user or login with default credential using interface and start working with app

In-memory database

For demo purpose, we use an in-memory database - H2. You don't need to do any preparations to setup it. It will run automatically when you start the application. You can connect to H2 via a browser. Do the following steps:

1. open `<api_url>/h2-console`
2. fill field `JDBC URL` with `spring.datasource.url` from application/properties. By default in our properties, it's `jdbc:h2:mem:testdb`
3. fill credentials `spring.datasource.username` and `spring.datasource.password`. By default in our properties, it's `sa` and `password`
4. click `connect` and you can use web interface for H2 database

Every time you rerun the application, the database will be recreated and filled with data from `data.sql`. For production purposes, please remove this file and use real Database

Test User / Password

You can use these test users for application testing:

1. admin@admin.com / password
2. user@user.com / password1

Tech Stack

This Code Bundle has backend and frontend parts.

Backend Part uses following libraries and frameworks:

- Spring 4.0.0
- Spring Boot 2.1.4.RELEASE
- Maven 3.6.0
- Json Web Token 0.9.1
- Model Mapper 2.3.3
- Springfox-swagger2 2.9.2
- Findbugs plugin 3.0.5
- Maven Checkstyle plugin 3.0.0
- H2 database 1.4.197

Frontend Part is basically Angular project with following stack:

- Angular 8.0.0
- RxJs 6.5.2
- Nebular 4.1.2
- Eva-icons 1.1.0
- Typescript 3.4.5
- ...

Frontend part is based on the latest ngx-admin dashboard template, but with edited UI components and service layer for data getting. Bundle UI supports both data from API and mock data, you can switch it inside file `core.module.ts` by editing `NB_CORE_PROVIDERS` collection.

API Documentation

You can check API documentation by running api and accessing <api_url>/swagger-ui.html link. Default url is `http://localhost:8080`

To use swagger with token authentication please follow these steps:

- open swagger link `<api_url>/swagger-ui.html` 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 (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
    - angular // this floder contains all UI code
    - java // server side java code
        - .mvn

        - wrapper // allow to build project without having to install Maven first
    - src
        - main
            - java
                - bundlejava
                    - address
                    - authentification
                    - config
                    - exception
                    - role
                    - user
            - resources
                - application.properties // properties, which are using for configure app
                - data.sql // uses for filling database with default values. Just for testing purpose
        - test
            - java
    - checkstyle.xml // rules for checkstyle plugin
    - findbugs-excludes.xml // uses for static code analysis

Findbugs and Checkstyle plugins

The Checkstyle Plugin generates a report regarding the code style used by the developers. Help developers follow the same rules and keeping the clean code.
For more information please check https://maven.apache.org/plugins/maven-checkstyle-plugin/index.html

FindBugs looks for bugs in Java programs. FindBugs uses static analysis to inspect Java bytecode for occurrences of bug patterns
For more information please check https://gleclaire.github.io/findbugs-maven-plugin/

Model Mapper

Usually, application consists of similar object models such as Entity and DTO. Every time to convert one into another it's necessary to write many converters. Model mapping makes it easy to convert one model to another. You just need to pass entity and class in which it should be converted. Sometimes models have inners entities and for mapping that entities you can add some additional configuration. Few examples you can find in `Config.java`.
For more information please check http://modelmapper.org/

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)