**Read Me First**

Please read the following steps to setup the project in your local machine

**Getting Started**

**Reference Documentation**

For reference, please follow the next steps:

* In Eclipse, import the project as Maven project
* Check the **Build Path** of the project is compiled with Java 8
* Run the class **com.exercise.todoapi.TodoApiApplication** to execute the spring-boot project
* If everything is fine, open the Swagger API documentation of this project located in <http://localhost:8080/swagger-ui.html>

**Guide to probe the API**

The following guides illustrate how to use some features concretely:

* The project has 2 controllers:
* The main one, it's the **to-do-controller** that manages the to-do collections
* The second one, it's the **item-controller** that manages the items for each to-do object
* The authenticator one, it's the **authentication-controller** that manages the login and the logout of the application
* The **/api/v1/login** is the method that generates the **token** to be used to invoke the other API methods

**Guide to login into the API**

The following guides illustrate how to login:

* The project manages 2 roles: **OWNER** that can read and write. And **READER** that can just read the data.
* Also this project has 2 predefined users:
* The **first user** has the following properties: *username:* **super\_user** *password:* **password1** and this one has the **OWNER** role
* The **second user** has the following properties: *username:* **reader\_user** *password:* **password2** and this one has the **READER** role
* In the Swagger documentation, please expand the **authentication-controller** tab. Go to **/api/v1/login** and enter the username and password provided in the previous step.
* Try out this login and you'll receive a **token** with this format: **Bearer XXXXXXXXXXXXX** in the **authorization** header. Please copy it inclusive with the Bearer word.
* Click on **Authorize** button and paste the **token** provided in the previous step in the **value** field. Click on **Authorize** and then on **Close**
* Later, you can begin to test the API.

**Guide to manage the authentication-controller API**

The following guides illustrate how login and logout in the API:

* **/api/v1/login** - It's a **POST** method that generates the token login. It needs an username and password
* **/api/v1/logout** - It's a **GET** method that invalidates the token

**Guide to manage the to-do-controller API**

The following guides illustrate how to manage the to-do objects:

* **/api/v1/todo** - It's a **POST** method that creates a new to-do object
* **/api/v1/todo/{id}** - It's a **GET** method that returns a to-do object
* **/api/v1/todo/{id}** - It's a **PUT** method that updates changes in an existing to-do object
* **/api/v1/todo/{id}** - It's a **DELETE** method that removes a to-do object
* **/api/v1/todo/all** - It's a **GET** method that returns a list of the available to-do objects
* **/api/v1/todo/{id}/summary** - It's a **GET** method that returns the summary of a given to-do id

**Guide to manage the item-controller API**

The following guides illustrate how to manage item objects for each to-do:

* **/api/v1/todo/{toDoId}/item** - It's a **POST** method that creates a new Item of a given to-do id
* **/api/v1/todo/{toDoId}/item/{id}** - It's a **GET** method that returns an Item of a given to-do id
* **/api/v1/todo/{toDoId}/item/{id}** - It's a **PUT** method that updates changes of an existing Item of a given to-do id
* **/api/v1/todo/{toDoId}/item/{id}** - It's a **DELETE** method that removes an Item of a given to-do id
* **/api/v1/todo/{toDoId}/item/all** - It's a **GET** method that returns the list of Items of a given to-do id