**Meal App Task for DevOps (meal-app-final)**

This is a brief documentation of meal app task by Nokia Budapest.

**Description**

The application is a REST API for managing meals.

**Technology Stack Specification**

* **Spring Framework** was used for the API design
* **PostgreSQL** for data persistence
* **Docker** - A docker image of the application was built for a local docker environment.
* **Helm -** Helm chart (mealchart) was created containing the deployment and services for the meal-app-final application.
* **JUnit** dependency was used for Unit tests
* **Swagger** for creating the REST API documentation
* **Flyway** for db migration

**Unit Tests**

Unit tests for the four CRUD methods was created and can be found in the ***src/test/java*** directory.

**Database Migration**

Flyway dependency have been added to provide migration for the postgres database. The migration files are available in the src/main/resources/db.migration directory. The migrations are as follows:

* V1\_0\_0\_\_init.sql – Contains script for creating the initial database schema
* V1\_0\_1\_\_seeders.sql – contains script for inserting initial records into the database.

**Deployment**

The API could be deployed to a Kubernetes cluster either using basic deployment.yml file or vial helm install.

The files are specified as follows:

**Without Helm Chart**

* postgres-configmap.yml – contains the configuration for the postgres database
* postgres-credentials.yml – contains the username, password etc for the postgres database
* postgres-deployment.yml – deployment config for postgres
* deployment.yml – deployment configuration for the meal-app-final API

So without helm we can apply the configurations via:

kubectl apply -f <yml file>

**With Helm Chart**

All the files are available in the ***mealchart*** directory. So the meal-app-final can be deployed by running the command:

helm install newchart mealchart

Note that this helm chart was created for deploying just the API and therefore, postgres database should already by deployed to the cluster.

**API Specification**

The apis are as follows:

**GET**[**​/**](http://127.0.0.1:59515/swagger-ui/#/meal-controller/indexUsingGET)

Index

curl -X GET "http://127.0.0.1:59515/" -H "accept: \*/\*"

**GET**[**​/meal​/{id}**](http://127.0.0.1:59515/swagger-ui/#/meal-controller/getMealUsingGET)

getMeal

curl -X GET "http://127.0.0.1:59515/meal/2" -H "accept: \*/\*"

**PUT**[**​/meal​/{id}**](http://127.0.0.1:59515/swagger-ui/#/meal-controller/saveUsingPUT)

Save

curl -X PUT "http://127.0.0.1:59515/meal/2" -H "accept: \*/\*" -H "Content-Type: application/json" -d "{ \"id\": 2, \"meal\_name\": \"Beans and Break\", \"price\": 56.0, \"ingredients\": null, \"spicy\": null, \"gluton\_free\": null, \"description\": null, \"kcal\": null, \"imageUrl\": null}"

**DELETE**[**​/meal​/{id}**](http://127.0.0.1:59515/swagger-ui/#/meal-controller/deleteMealUsingDELETE)

deleteMeal

curl -X DELETE "http://127.0.0.1:59515/meal/3" -H "accept: \*/\*"

**GET**[**​/meals**](http://127.0.0.1:59515/swagger-ui/#/meal-controller/getMealsUsingGET)

getMeals

curl -X GET "http://127.0.0.1:59515/meals" -H "accept: \*/\*"

**POST**[**​/meals**](http://127.0.0.1:59515/swagger-ui/#/meal-controller/addMealUsingPOST)

addMeal

curl -X POST "http://127.0.0.1:59515/meals" -H "accept: \*/\*" -H "Content-Type: application/json" -d "{ \"description\": \"Garry and Soup\", \"gluton\_free\": true, \"id\": 0, \"imageUrl\": \"/photo\", \"ingredients\": \"Pepper\", \"kcal\": 0, \"meal\_name\": \"Rice\", \"price\": 0, \"spicy\": true}"

**API Documentation**

I have included a Swagger API documentation which is available via the route:

/swagger-ui/