# Introduction

This assignment uses Java version "11.0.14" 2022-01-18 LTS and Spring Boot version 2.7.2.

# How to run the application

## Using IDE

Import the project in IDE and set following properties in application.properties the resources folder

local.beer.repository.path – This specifies absolute path of local repository i.e. beers.json file. The beers.json file can found in the root directory of Git repository.

punk.beers.start.index - Optional. Starting index for Punk beers. Default value is 10000 i.e. index starts at 10001.

Run the application.

## From command line

Run following command, for example

java -Dlocal.beer.repository.path=/beers.json -jar task-0.0.1-SNAPSHOT.jar

The file task-0.0.1-SNAPSHOT.jar can be found in the root directory of Git repository.

## Tests

Due to lack of time, five tests are provided. They are all integration tests. They test success condition for each endpoint. Ideally following tests should have been provided

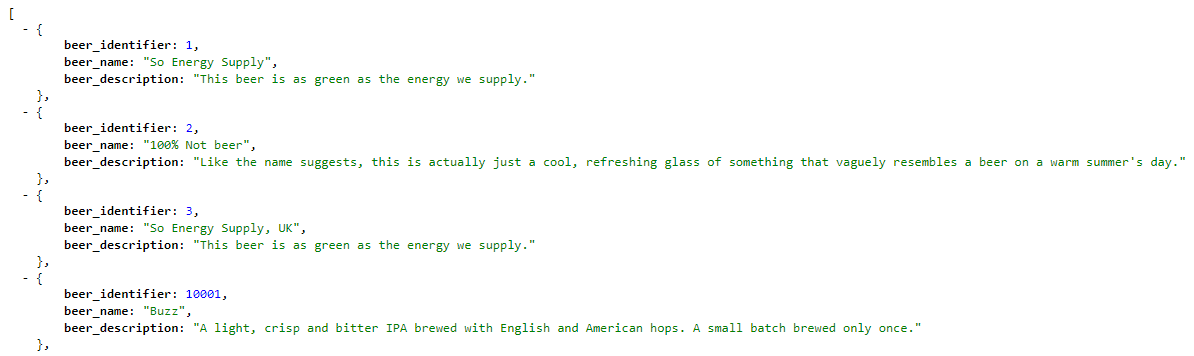
* Unit tests for each layer (Controller, service and repository)
* Integration tests for each endpoint (success and failure condition)

You will also find TestTask.postman\_collection.json file the root directory of Git repository. You can use this file to test the API using Postman.

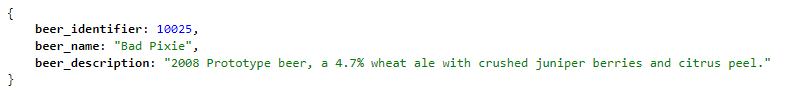
# Endpoints

This application has Following endpoints

/v1/beers (GET) – This endpoint returns all the beers from Punk API and local repository (beers.json)

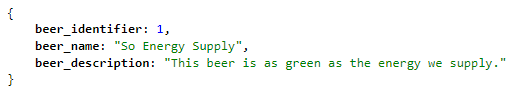


/v1/beers/random (GET) – Returns random beer from Punk API or local repository (beers.json)

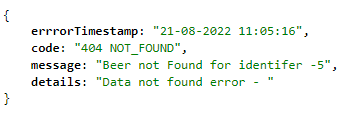


/v1/beers/{id} (GET) – Returns a beer that matches given Id.

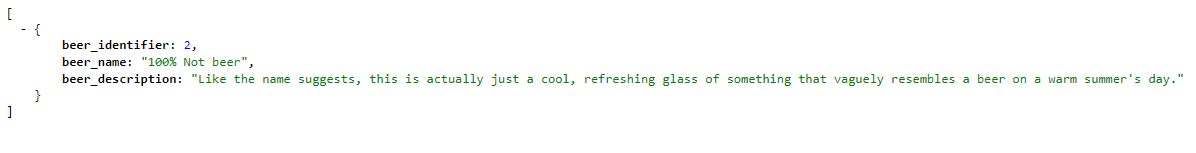
For /v1/beers/1, it returns following



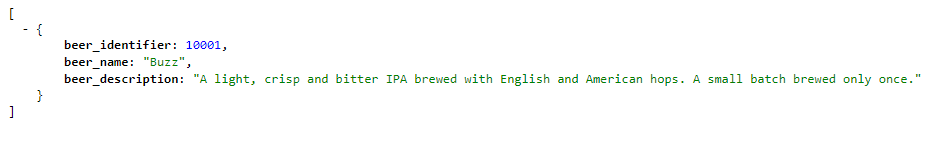
If beer not found for given Id, following error response is returned, for example, for /v1/beers/5, it returns



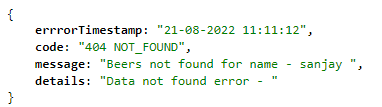
/v1/beers/search?name=beer (GET) – This endpoint accepts a search term and returns all the beer whose name contains said search term. The search is case insensitive. For /v1/beers/search?name=beer, it returns



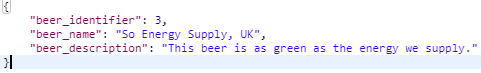
For /v1/beers/search?name=buzz, it returns



For /v1/beers/search?name=Sanjay, it returns following error response



/v1/beers (POST) – This endpoint adds a beer to local repository. It returns following



It returns 201 Created as a status code and URL of newly created resource in the location header

Graphical user interface, application

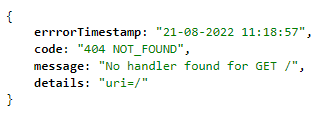
Description automatically generated with medium confidence

# Error handling

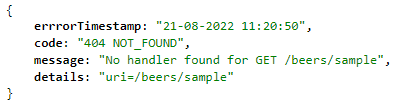
Following scenarios are handled for all the endpoints the application exposes. Also application returns customized response in JSON for any other errors.

If the URI is not specified correctly - Error JSON response is returned.

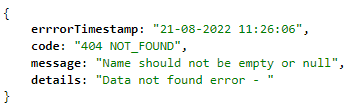
For following URL, <http://localhost:8085> It returns



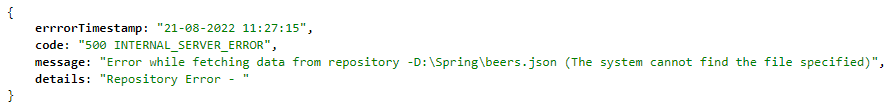
If Id is not specified as integer value for endpoint /v1/beers/{id} – Following error response in JSON is returned, for example, for /v1/beers/sample, it returns



If name is not specified for endpoint /v1/beers/search?name= – Following error response in JSON is returned, for example, for /v1/beers/search?name, it returns



If the local repository (beers.json) path is incorrectly specified in the application.properties or on command line – Following error response is returned



# Components

Main application - TaskApplication

REST Controllers -

BeerController - Main controller which exposes all the endpoints

Repository -

BeerRepository – Main repository

Custom Exception classes -

DataNotFoundException – This class is used when beers are not found for given search term or for given Id etc.

RepositoryException – This class is used when error happens while fetching data from repository.

Beans -

Beer – Bean to represent Beer

ExceptionResponse - To return customized response in case of errors.

Utils -

TaskExceptionHandler – Global exception handler for errors

Tests -

TaskApplicationTests - This class contains following tests

testGetAllBeers()- Tests /v1/beers (GET) endpoint.

testGetRandomBeer()- Tests /v1/beers/random (GET) endpoint.

testGetBeerById()- Tests /v1/beers/{id} (GET) endpoint.

testGetBeerByName()- Tests /v1/beers/search/search?name= (GET) endpoint.

createBeer() – Tests /v1/beers (POST) endpoint.