Basic Application Documentation

**System Requirements:**

|  |
| --- |
| Java OpenJDK 11 |
| Spring Boot 2.7.7 |
| Apache Maven 3.8.6 |
| MySQL |
| Spring Security |

**Data Base schema:**

Data Base name: stocksdb

|  |  |
| --- | --- |
| Table Name: stocks | |
| Fields | Syntax |
| id | bigint **NOT** **NULL** AUTO\_INCREMENT  **primary** **key** |
| name | **varchar**(255) |
| current\_price | **double** |
| last\_update | **timestamp** **not** **null** |

|  |  |
| --- | --- |
| Table Name: users | |
| Fields | Syntax |
| id | bigint **NOT** **NULL** AUTO\_INCREMENT  **primary** **key** |
| user\_name | **varchar**(255) |
| current\_password | **varchar**(255) |

**Schema file:** 

**REST API details:**

----------------------

APIs are secured, token need to be passed in header for each resource access.



Swagger UI link: **http://<HOSTIP>:<PORT>/swagger-ui.html**

**Docker Details:**

Docker image name: payconiq-app

**Application Docker file content:**

FROM openjdk:11-jre-slim-buster

MAINTAINER sravana.pullivendula@gmail.com

COPY target/assignment-0.0.1-SNAPSHOT.jar payconiqApp.jar

ENTRYPOINT ["java","-jar","/payconiqApp.jar"]

Note: Dockerfile available in project repository

**Docker Compose file content:**

version: "1"

services:

  payconiq-app:

    image: payconiq-app

    ports:

      - "9090:8080"

    networks:

      - app-network-bridge

    depends\_on:

      - mysqlappdb

  mysqlappdb:

    image: mysql:8

    ports:

      - "3306:3306"

    networks:

      - app-network-bridge

    environment:

      - MYSQL\_ROOT\_PASSWORD=root

      - MYSQL\_DATABASE=bootdb

networks:

  app-network-bridge:

Note: Docker compose file available in project repository.

**How to deploy Application into Container based:**

Step-1: Build docker image

1. Goto project path
2. Run: docker image build -t payconiq-app .

Step-2: Run Docker-compose

1. Goto project path.
2. Run: docker-compose up

**Token Generation:**

**Request:**

**Method: post**

**URL:** http://localhost:8080/user?password=password&user=testuser

**Response:**

**{**

**"userName": "testuser",**

**"token": "Bearer eyJhbGciOiJIUzUxMiJ9.eyJqdGkiOiJzb2Z0dGVrSldUIiwic3ViIjoidGVzdHVzZXIiLCJhdXRob3JpdGllcyI6WyJST0xFX1VTRVIiXSwiaWF0IjoxNjc1MjY4MDcyLCJleHAiOjE2NzUyNjg2NzJ9.DpzCUpYxL0AvwsGem77LniUMFdDW5NqU7rd1W6hHih8x2\_42ykkKSi6UxqjWUWxSrC6vVVuJhIlq3gYpcEjTLA",**

**"timestamp": "2023-02-01 16:14:32.533UTC"**

**}**

This token need to be passed header section for each resource access.

Eg:

url: **http://localhost:8080/api/stocks?pageNo=0&pageSize=5&sortBy=id&sortDir=asc**

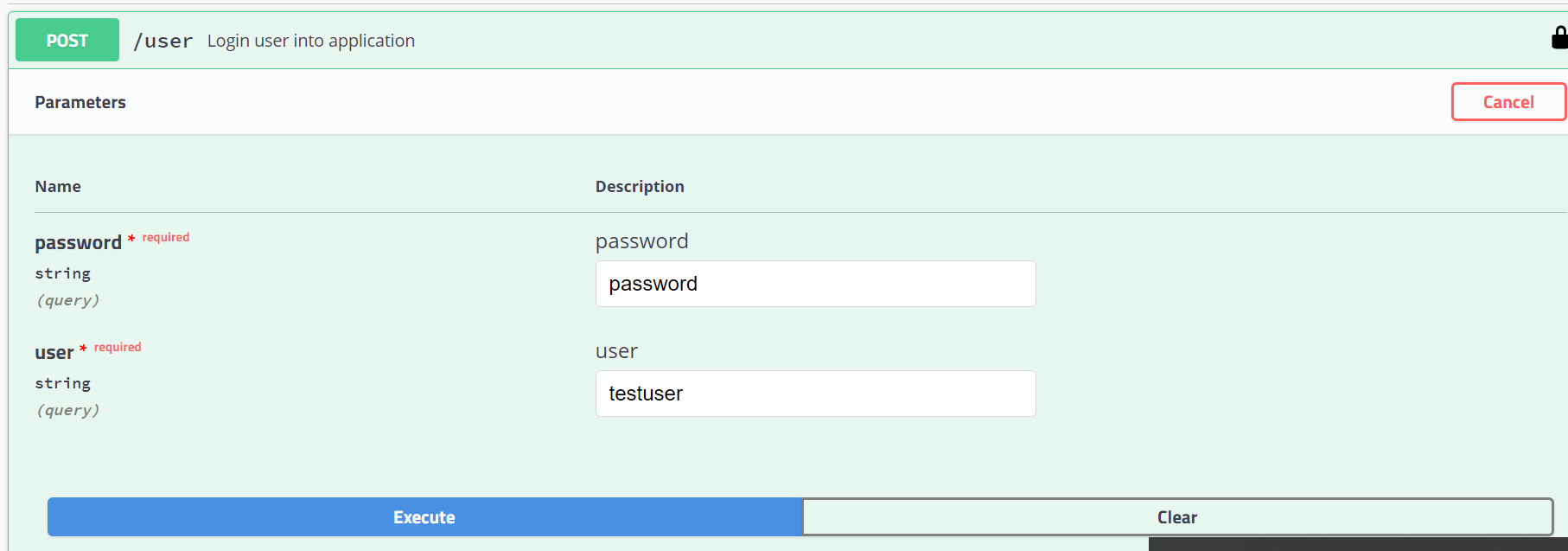
header: "Authorization: Bearer eyJhbGciOiJIUzUxMiJ9.eyJqdGkiOiJzb2Z0dGVrSldUIiwic3ViIjoidGVzdHVzZXIiLCJhdXRob3JpdGllcyI6WyJST0xFX1VTRVIiXSwiaWF0IjoxNjc1MjY4MDcyLCJleHAiOjE2NzUyNjg2NzJ9.DpzCUpYxL0AvwsGem77LniUMFdDW5NqU7rd1W6hHih8x2\_42ykkKSi6UxqjWUWxSrC6vVVuJhIlq3gYpcEjTLA"

Swagger UI:

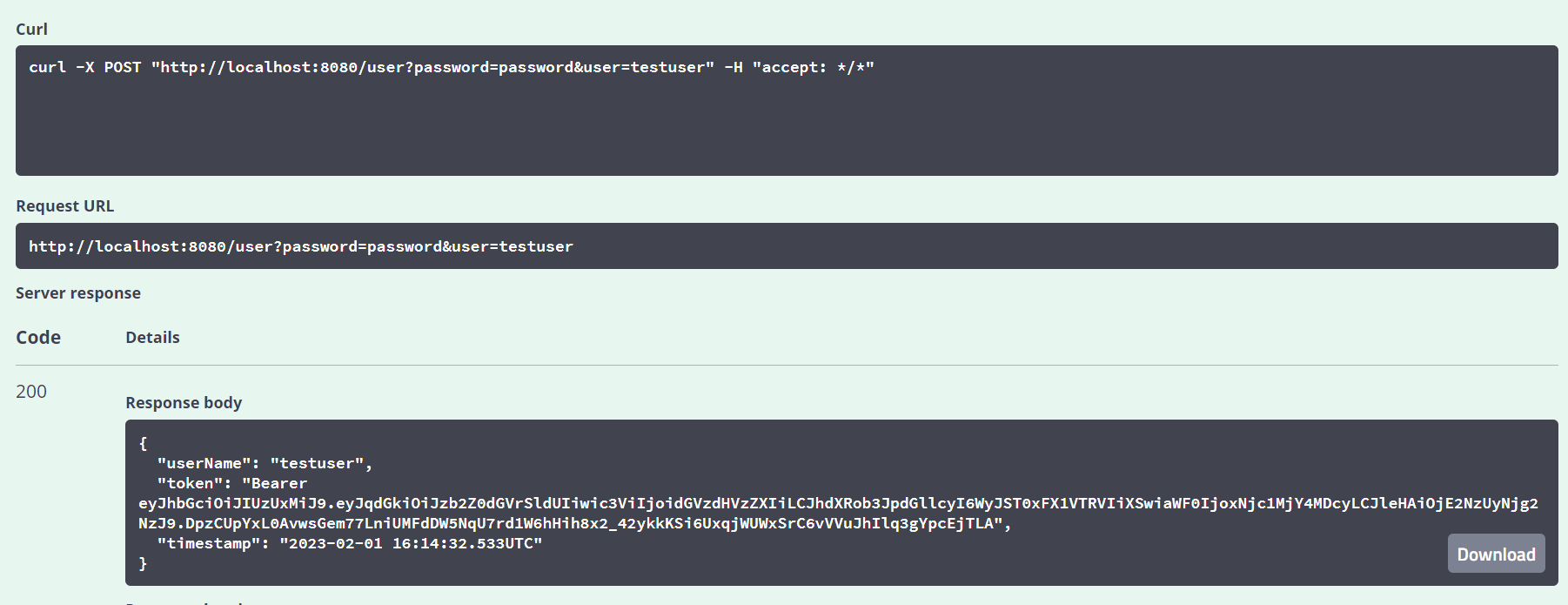
url: http://<HOST:PORT>/swagger-ui.html

step-1: Token Generation:

request:



Response:

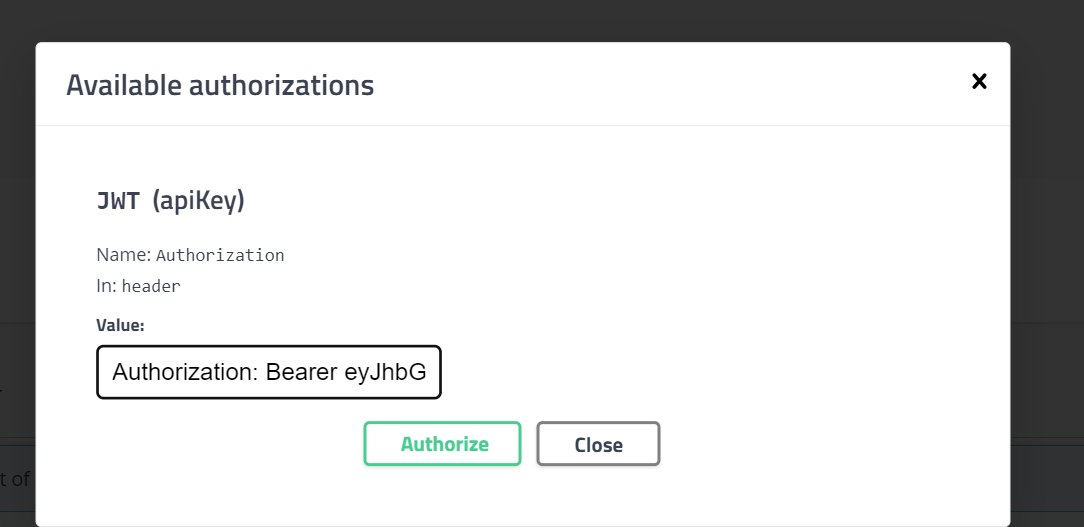


Step-2: Apply token in Swagger UI.

Click Authorize Button



Paste token value in value field.



Click Authorize button.

Step-3:

Continue using Application APIs.