**Министерство науки и высшего образования РФ**

**Государственное образовательное учреждение высшего профессионального образования**

**Белгородский Государственный Технологический Университет им В. Г. Шухова**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем.

Лабораторная работа №3 по дисциплине

«Технологии Web-программирования»

Тема: Серверное программирование.

Выполнила: ст. группы ПВ-41  
Ярыгина И. Ю.  
Проверил: Картамышев С.В.

Белгород 2020 г.

**Цель работы:** познакомиться с основами backend разработки web-приложений на языке Java. Познакомиться с основами работы docker. Познакомиться с фреймворком Spring-Boot 2 и научиться разворачивать проект, производить его настройку. Научится работать с API в приложении Postman.

**Выполение**

Стандартная структура spring приложения с системой сборки maven

Изображение выглядит как текст

Автоматически созданное описание

pom.xml – файл в котором будут находится все необходимые нам зависимости

Dockerfile – файл, в котом будет находится конфигурация нашего Docker-образа

App – класс, точка запуска приложения (настройка конфигурации и многое другое)

Controller – класс, контроллер.

Далее был настроен файл конфигурации Dockerfile

Изображение выглядит как текст

Автоматически созданное описание

Все необходимы зависимости были подключены в pom.xml

Изображение выглядит как текст

Автоматически созданное описание

В данной зависимости находятся все необходимые библиотеки для работы контроллеров, а так же Tomcat (HTPP сервер, работающий по принципу создания пула потоков для принятия запросов).

Для того чтобы запустить проект необходимо создать точку входа

Изображение выглядит как текст

Автоматически созданное описание

Так выглядит базовый класс запуска приложения. Аннотация @SpringBootApplication которая автоматически конфигурирует Tomcat, внедряет зависимости и т.д.

Был добавлен API модуль версии 1 со следующими контроллерами:

Изображение выглядит как текст

Автоматически созданное описание

Аннотация @RestContoller – указывает что данный класс является частью sprint-context и указывает что данный класс будет принимать http запросы и по умолчанию возвращать данные в формате Json.

@RequestMappint – ulr контроллера

@GetMappint – указывает какой тип запроса принимает данный метод. В нашем случае это Get запрос. Так же он содержит url, к которому идет обращение. (/user/1)

Модели представлены в виде классов:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Так же в контроллерах объявлены сервисы(в них описана бизнес логика).

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

проверены запросы в Postman:

Изображение выглядит как текст

Автоматически созданное описание

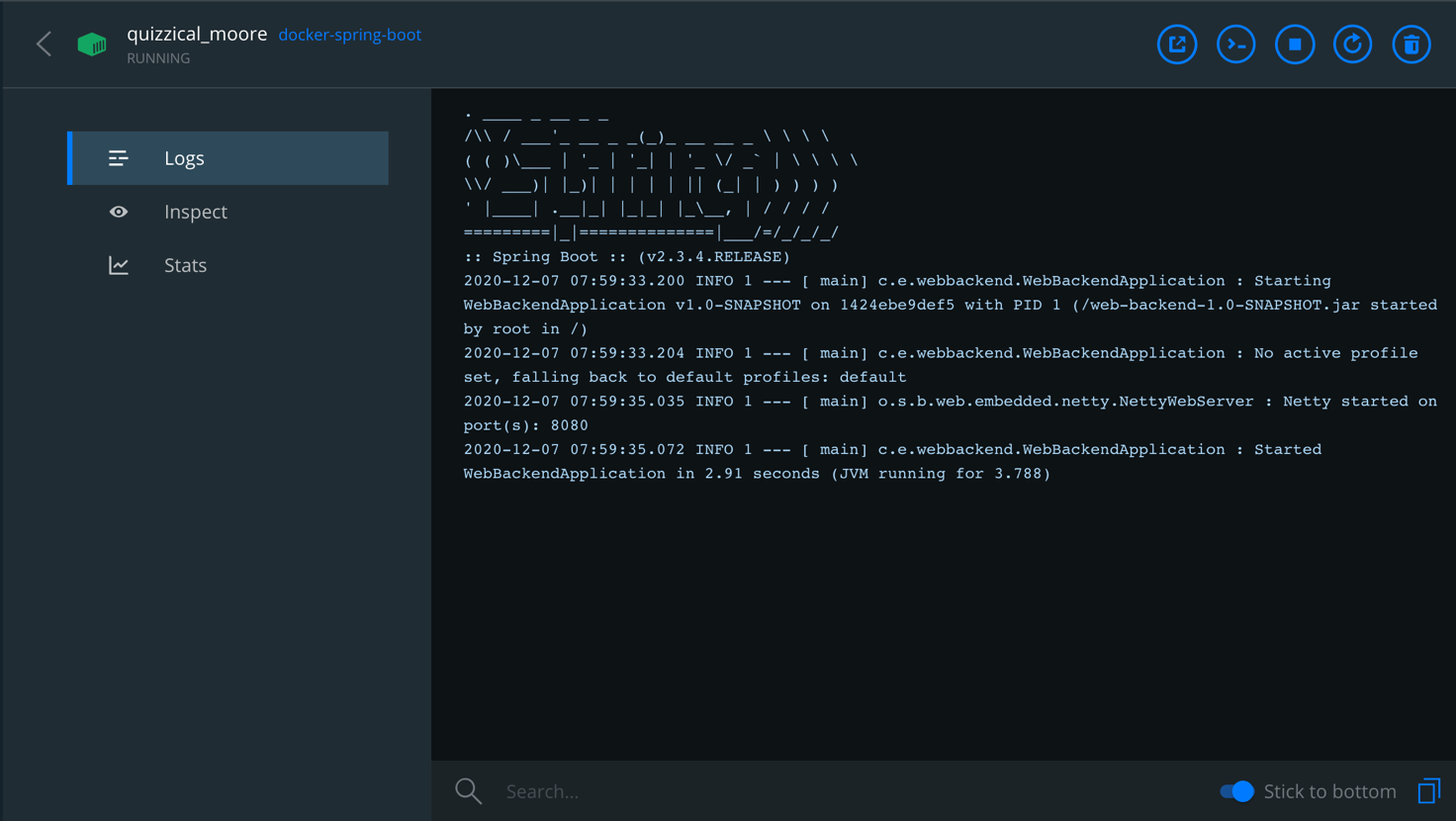
Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Скриншот запущенного Docker образа:



**Вывод:** В результате выполнения работы были рассмотрены основы работы docker с фреймфорком Sprint-Boot.