



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

---

Институт информационных технологий (ИТ)

Кафедра математического обеспечения и стандартизации  
информационных технологий (МОСИТ)

**ОТЧЕТ ПО ИТОГОВОЙ ПРАКТИЧЕСКОЙ РАБОТЕ (ЧАСТЬ 3)**

по дисциплине

«Технология разработки программных приложений»

Тема: «WEB-сервис аренды автомобилей»

Студенты групп

ИКБО-01-21 и ИКБО-20-21

«7» сентября 2023 г.

Матросов Д.Я.

(подпись)

Хитров Н.С.

(подпись)

Преподаватель

«7» сентября 2023 г.

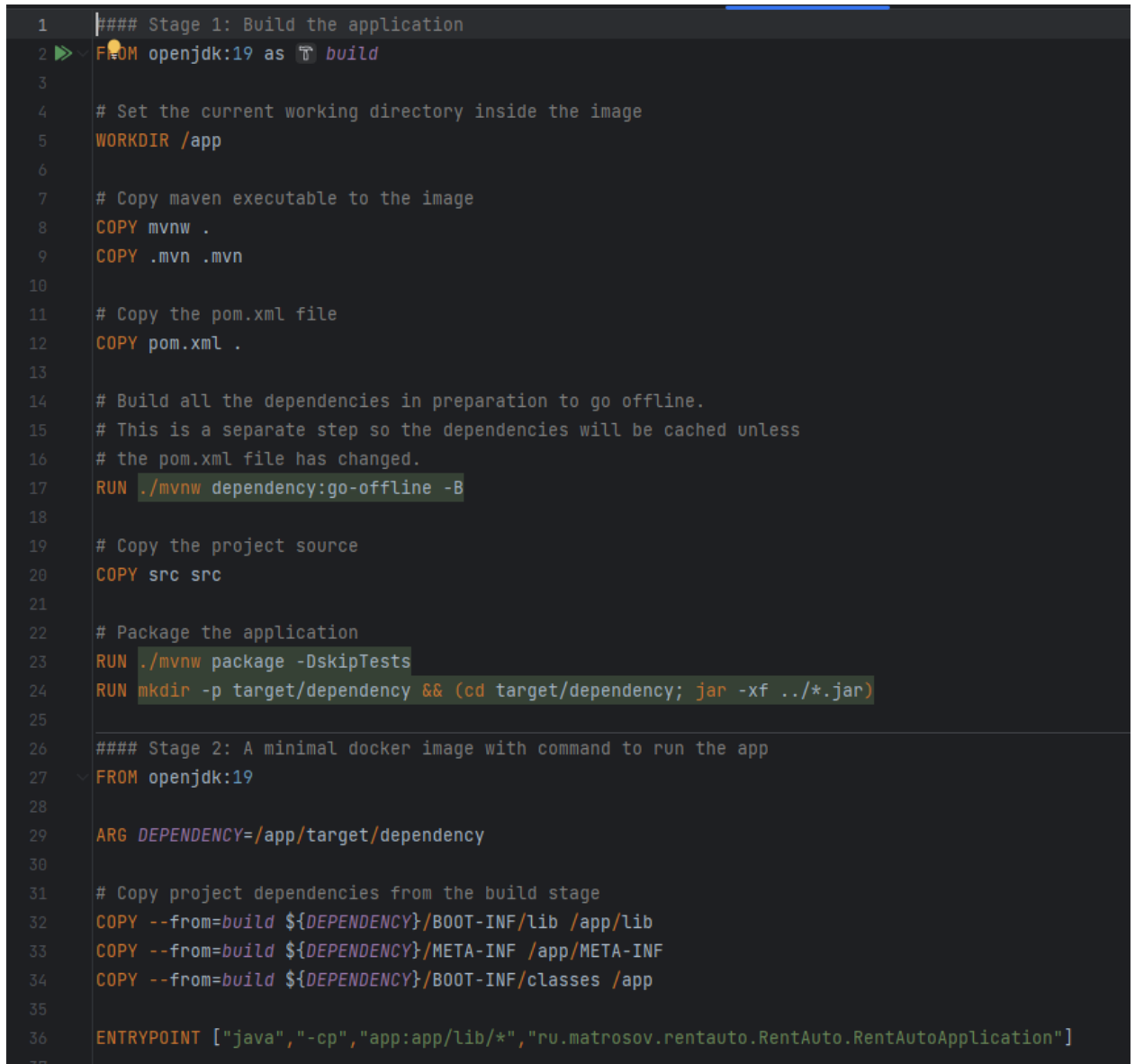
Петренко А.А.

(подпись)

### Часть 3. Развёртывание приложения

1. Написание dockerfile для backend. Выполнил Матросов Данил.

На рисунке 1 показан написанный dockerfile для создания образа с самим приложением.



```
1  ##### Stage 1: Build the application
2  FROM openjdk:19 as build
3
4  # Set the current working directory inside the image
5  WORKDIR /app
6
7  # Copy maven executable to the image
8  COPY mvnw .
9  COPY .mvn .mvn
10
11 # Copy the pom.xml file
12 COPY pom.xml .
13
14 # Build all the dependencies in preparation to go offline.
15 # This is a separate step so the dependencies will be cached unless
16 # the pom.xml file has changed.
17 RUN ./mvnw dependency:go-offline -B
18
19 # Copy the project source
20 COPY src src
21
22 # Package the application
23 RUN ./mvnw package -DskipTests
24 RUN mkdir -p target/dependency && (cd target/dependency; jar -xf ../*.jar)
25
26 ##### Stage 2: A minimal docker image with command to run the app
27 FROM openjdk:19
28
29 ARG DEPENDENCY=/app/target/dependency
30
31 # Copy project dependencies from the build stage
32 COPY --from=build ${DEPENDENCY}/BOOT-INF/lib /app/lib
33 COPY --from=build ${DEPENDENCY}/META-INF /app/META-INF
34 COPY --from=build ${DEPENDENCY}/BOOT-INF/classes /app
35
36 ENTRYPOINT ["java", "-cp", "app:app/lib/*", "ru.matrosov.rentauto.RentAuto.RentAutoApplication"]
37
```

Рисунок 1 – Dockerfile

2. Построение образа серверной части приложения при помощи вышеописанного dockerfile-a. Выполнил Матросов Данил.

На рисунке 2 показан процесс построения образа серверной части приложения, при помощи dockerfile-a.

```

PS C:\Users\Данил\Downloads\RentAuto (1)\RentAuto> docker build -t rentauto .
[+] Building 34.4s (8/8) FINISHED => [internal] load build definition from Dockerfile
=> [internal] load build context
=> => transferring context: 61.98MB
=> [1/2] FROM docker.io/library/openjdk:19@sha256:4123be55fd6853980020c59e7530d017ea08996abbe71741a51c62f7b7586bee
=> => resolve docker.io/library/openjdk:19@sha256:4123be55fd6853980020c59e7530d017ea08996abbe71741a51c62f7b7586bee
=> => sha256:4123be55fd6853980020c59e7530d017ea08996abbe71741a51c62f7b7586bee 1.04kB / 1.04kB
=> => sha256:973fe414a4e1f3e41e291b068183684a88827dd2cb5f78214da26632d5218702 954B / 954B
=> => sha256:2e6f6690e479ce4ad3600d1b87ff79ea5dc6438165902f332ab7f721f7599c6b 4.42kB / 4.42kB
=> => sha256:051f419db9dd9462e8995886d24f592c26cef792cc915dfbc7548e0b19aa55fe 40.59MB / 40.59MB
=> => sha256:aa51c6010a14c1984cbdea1332a5d2f77bf6e0141bc497b44dca611e21f9b391 12.23MB / 12.23MB
=> => sha256:dba785fff917fb7bc8692503ac810691754ab0f6e0cdfbf4941b0de2305ab652 196.18MB / 196.18MB
=> => extracting sha256:051f419db9dd9462e8995886d24f592c26cef792cc915dfbc7548e0b19aa55fe
=> => extracting sha256:aa51c6010a14c1984cbdea1332a5d2f77bf6e0141bc497b44dca611e21f9b391
=> => extracting sha256:dba785fff917fb7bc8692503ac810691754ab0f6e0cdfbf4941b0de2305ab652
=> [2/2] ADD target/RentAuto-0.0.1-SNAPSHOT.jar RentAuto-0.0.1-SNAPSHOT.jar
=> exporting to image
=> exporting layers
=> writing image sha256:2b785281755fcf55bed37220cf48a4aeca18dae446a9ade60de6fa140b77fc5b
=> naming to docker.io/library/rentauto

```

Рисунок 2 – Построение образа

### 3. Написание Dockerfile для frontend. Выполнил Хитров Никита.

На рисунке 3 показан написанный dockerfile для создания образа с клиентской частью приложения.

```

1  ➤ FROM node:latest
2
3  WORKDIR /app
4
5  EXPOSE 3000
6
7  COPY package*.json ./
8
9  RUN npm install
10
11 COPY . .
12
13 CMD ["npm", "run", "start"]

```

Рисунок 3 – Dockerfile

4. Написание файла docker-compose.yml для связывания нескольких контейнеров.

На рисунке 4 показан написанный docker-compose.yml для связывания нескольких контейнеров.

```
1  version: '2'
2
3  > services:
4
5  >   postgres:
6     image: postgres
7     restart: always
8     container_name: db_rentauto
9     environment:
10    POSTGRES_DB: rent_auto_v2
11    POSTGRES_PASSWORD: telino2003
12
13  >   backend:
14     build: ./Backend
15     restart: always
16     container_name: back-end
17     depends_on:
18     - postgres
19     environment:
20     - SPRING_DATASOURCE_URL=jdbc:postgresql://postgres:5432/rent_auto_v2
21     - SPRING_DATASOURCE_USERNAME=postgres
22     - SPRING_DATASOURCE_PASSWORD=telino2003
23     ports:
24     - 8080:8080
25
26  >   frontend:
27     build: ./Frontend
28     restart: always
29     container_name: front-end
30     depends_on:
31     - backend
32     ports:
33     - 3000:3000
```

Рисунок 4 – Dockerfile

## 5. Развёртывание приложения с помощью Maven. Выполнил Матросов Данил.

На рисунках 5-6 показан процесс сборки проекта с помощью maven.

```
[INFO] Scanning for projects...
[INFO]
[INFO] -----< ru.matrosov.rentauto:RentAuto >-----
[INFO] Building RentAuto 0.0.1-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:3.3.0:resources (default-resources) @ RentAuto ---
[INFO] Copying 1 resource
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.10.1:compile (default-compile) @ RentAuto ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:3.3.0:testResources (default-testResources) @ RentAuto ---
[INFO] skip non existing resourceDirectory C:\Users\Данил\Downloads\RentAuto (1)\RentAuto\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.10.1:testCompile (default-testCompile) @ RentAuto ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.22.2:test (default-test) @ RentAuto ---
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running ru.matrosov.rentauto.RentAuto.RentAutoApplicationTests
23:22:35.246 [main] DEBUG org.springframework.boot.test.context.SpringBootTestContextBootstrapper - Neither @Context
23:22:35.251 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Could not detect default
23:22:35.251 [main] INFO org.springframework.test.context.support.AnnotationConfigContextLoaderUtils - Could not det
23:22:35.298 [main] DEBUG org.springframework.boot.test.context.SpringBootTestContextBootstrapper - Using ContextCust
23:22:35.395 [main] DEBUG org.springframework.context.annotation.ClassPathScanningCandidateComponentProvider - Ident
23:22:35.396 [main] INFO org.springframework.boot.test.context.SpringBootTestContextBootstrapper - Found @SpringBoot
23:22:35.526 [main] DEBUG org.springframework.boot.test.context.SpringBootTestContextBootstrapper - Using TestExecu
23:22:35.528 [main] DEBUG org.springframework.test.context.support.AbstractDirtiesContextTestExecutionListener - Bef
```

Рисунок 5 – Сборка проекта

```

[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 6.758 s - in ru.matrosov.rentauto.RentAuto.RentAutoApplicationTests
2023-04-05T23:22:41.779+03:00 INFO 15180 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for
2023-04-05T23:22:41.781+03:00 INFO 15180 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...
2023-04-05T23:22:41.794+03:00 INFO 15180 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.3.0:jar (default-jar) @ RentAuto ---
[INFO] Building jar: C:\Users\Данил\Downloads\RentAuto (1)\RentAuto\target\RentAuto-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:3.0.4:repackage (repackage) @ RentAuto ---
[INFO] Replacing main artifact with repackaged archive
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 22.509 s
[INFO] Finished at: 2023-04-05T23:22:44+03:00
[INFO] -----
Process finished with exit code 0

```

Рисунок 6 – Сборка проекта

На рисунке 7 показан процесс запуска проекта в docker.

```

Данил@TrueCmetanka MINGW64 ~/Downloads/RentAuto (1)
$ docker-compose up
Container db_rentauto Created
Container back-end Created
Container front-end Created
Attaching to back-end, db_rentauto, front-end
db_rentauto | PostgreSQL Database directory appears to contain a database; skipping initialization
db_rentauto |
db_rentauto | 2023-04-06 22:11:36.793 UTC [1] LOG: starting PostgreSQL 15.2 (Debian 15.2-1.pgdg110+1) on x86_64
db_rentauto | 2023-04-06 22:11:36.793 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432
db_rentauto | 2023-04-06 22:11:36.793 UTC [1] LOG: listening on IPv6 address "::", port 5432
db_rentauto | 2023-04-06 22:11:36.799 UTC [1] LOG: listening on unix socket "/var/run/postgresql/.s.PGSQL.5432"
db_rentauto | 2023-04-06 22:11:36.809 UTC [29] LOG: database system was shut down at 2023-04-06 22:11:32 UTC
db_rentauto | 2023-04-06 22:11:36.815 UTC [1] LOG: database system is ready to accept connections
back-end |
back-end |
back-end |
back-end |
back-end |
back-end |
back-end |
back-end | :: Spring Boot :: (v3.0.4)
back-end | 2023-04-06T22:11:38.103Z INFO 1 --- [ main] r.m.r.RentAuto.RentAutoApplication :
back-end | 2023-04-06T22:11:38.106Z INFO 1 --- [ main] r.m.r.RentAuto.RentAutoApplication :
front-end | > rentcar@0.1.0 start
front-end | > react-scripts start

```

Рисунок 7 – Запуск проекта в docker

На рисунке 8 показан запущенный проект в docker.
















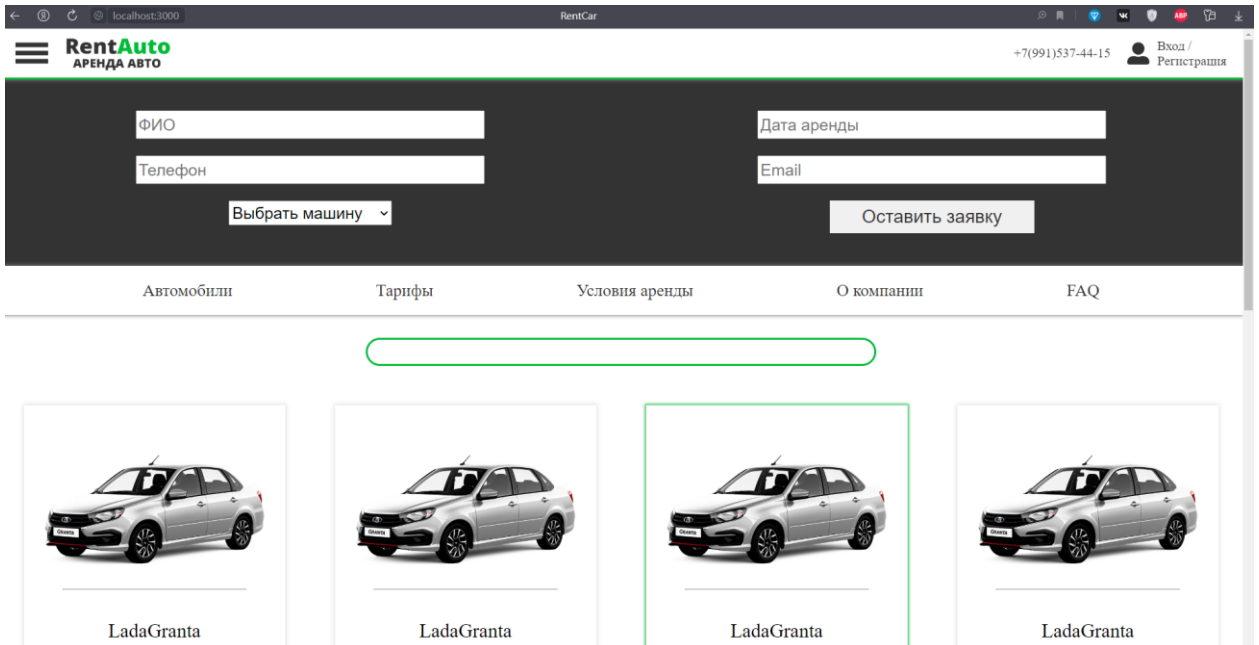
<input type="checkbox"/>	Name	Image	Status	Port(s)	Started	Actions
<input checked="" type="checkbox"/>	 <b>rentauto1</b>	-	Running (3/3)			<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <b>db_rentauto</b> 0ffae88123fb 	<a href="#">postgres</a>	Running		2 minutes ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <b>back-end</b> d4de7bc2c432 	<a href="#">rentauto1-backend</a>	Running	<a href="#">8080:8080</a> 	2 minutes ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <b>front-end</b> 8ae388cd5ee9 	<a href="#">rentauto1-frontend</a>	Running	<a href="#">3000:3000</a> 	2 minutes ago	<input type="checkbox"/> ⋮ 

Рисунок 8 – Запущенный проект в docker

## 6. Тестирование работы приложения (frontend + backend + database)

На рисунке 9 показан результат запуска приложения (<http://localhost:3000/>).



The screenshot shows the RentAuto web application interface. At the top, there's a header with the logo 'RentAuto АРЕНДА АВТО' and a phone number '+7(991)537-44-15'. Below the header is a registration form with the following fields: 'ФИО', 'Телефон', 'Дата аренды', and 'Email'. There are also dropdown menus for 'Выбрать машину' and a button 'Оставить заявку'. Below the form is a navigation bar with links: 'Автомобили', 'Тарифы', 'Условия аренды', 'О компании', and 'FAQ'. The main content area displays four car listings, all labeled 'LadaGranta', with the third one highlighted by a green border.

Рисунок 9 – Запущенный проект в docker