|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3** | |
| **по дисциплине** |  |
| **«**Технология разработки программных приложений**»** | |

|  |  |
| --- | --- |
| **Выполнил:**  Студент группыИКБО-10-23 | Лазаренко С. А. |
| **Проверил:**  Преподаватель к.э.н., доцент | Петросян Л. Э. |

Москва 2025 г.

**СОДЕРЖАНИЕ**

[МОДИФИКАЦИЯ ИСХОДНОГО КОДА ПРОЕКТА 3](#_Toc190951412)

[СКРИНШОТЫ РЕЗУЛЬТАТОВ К ЗАДАНИЯМ 5-8 6](#_Toc190951413)

[КОНТРОЛЬНЫЕ ВОПРОСЫ 8](#_Toc190951414)

[ВЫВОД 10](#_Toc190951415)

# МОДИФИКАЦИЯ ИСХОДНОГО КОДА ПРОЕКТА

1. Нахождение отсутствующей зависимости и указание ее в соответствующем блоке в build.gradle, чтобы проект снова начал собираться

Чтобы конфигурационный файл gradle подавал хотя бы какие-то признаки жизни, пришлось откатить JDK, потому что новые версии не поддерживались Micronaunt и Gradle. Далее добавил недостающую зависимость *implementation("io.micronaut:micronaut-inject"),* который позволял Micronaunt найти модуль для инъекции зависимостей. После внесенных изменений я запустил команду .*/gradlew build* для сборки:



Рисунок 1 – Изменения в файле build.gradle

1. В некоторых классах поправить имя пакета До изменений:

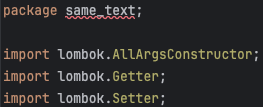


Рисунок 2 – Файл HealthResponse до изменений После изменений:

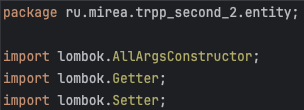


Рисунок 3 – Файл HealthResponse после изменений

1. Собрать документацию проекта, найти в ней запросы состояния и сущности по идентификатору

Выполним команду *./gradlew javadoc*:

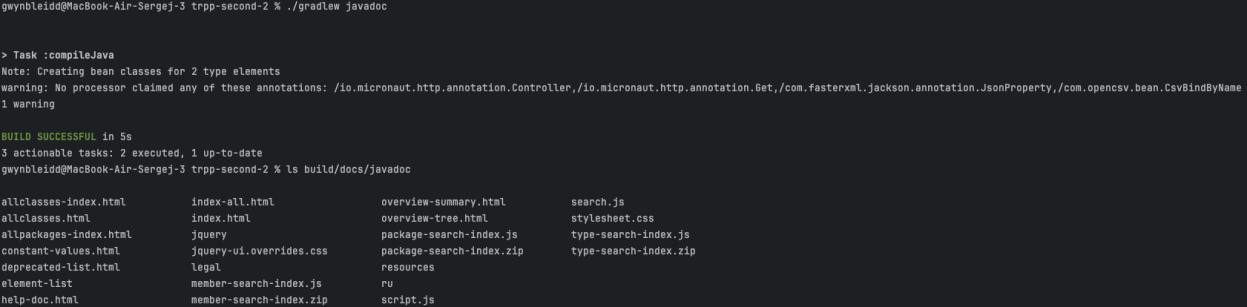


Рисунок 4 – Результат выполнения скрипта

1. Собрать jar со всеми зависимостями(так называемый UberJar), после чего запустить приложение

Выполним команду *./gradlew shadowJar:*

**

Рисунок 5 – shadowJar

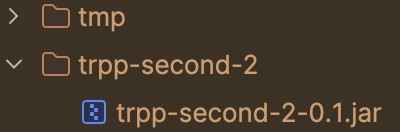


Рисунок 6 – Результат выполнения скрипта

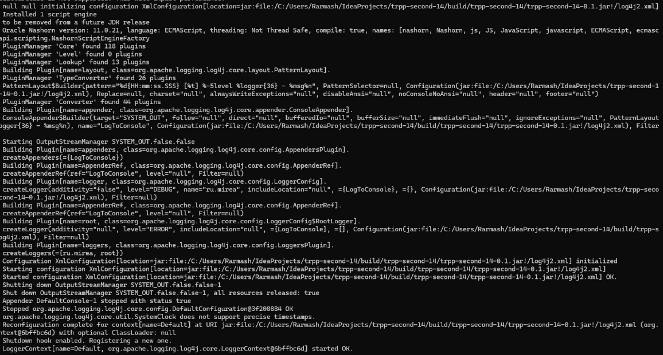


Рисунок 7 – Результат выполнения сборки

# СКРИНШОТЫ РЕЗУЛЬТАТОВ К ЗАДАНИЯМ 5-8

1. Запросить состояние запущенного сервера (GET запрос по адресу <http://localhost:8080>)

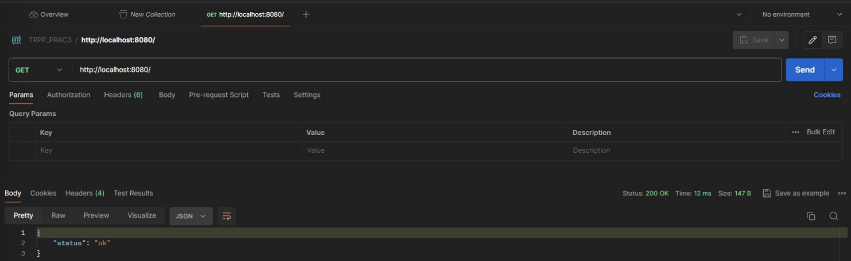
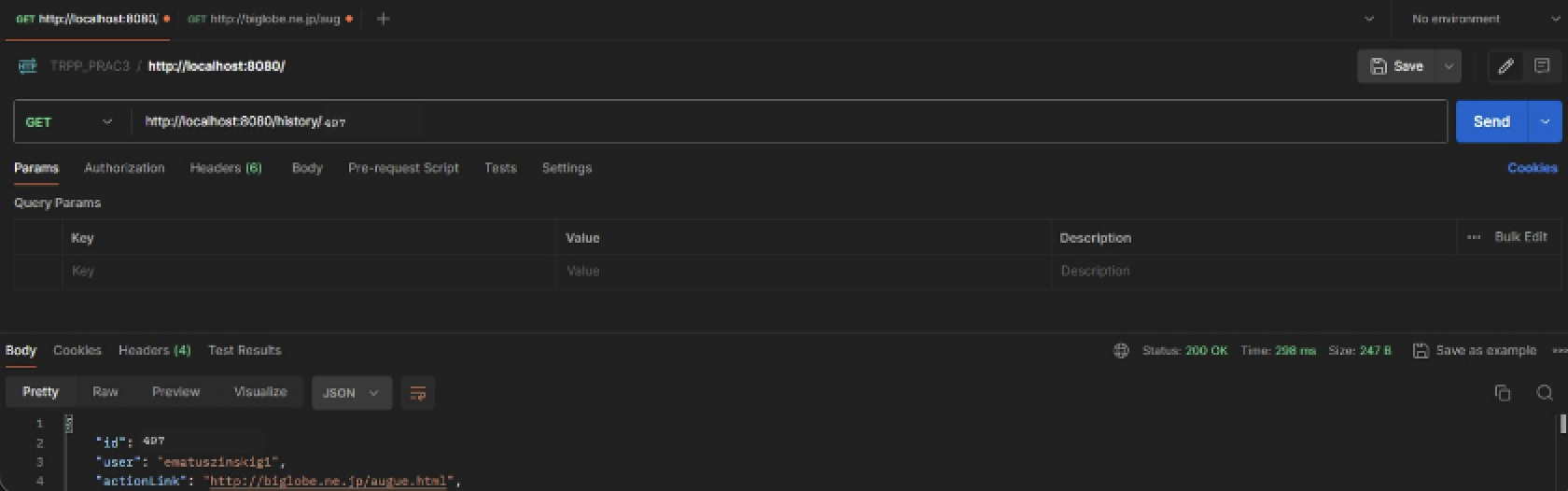


Рисунок 8 - Состояние запущенного сервера

1. Запросить сущность по идентификатору (GET запрос по адресу: <http://localhost:8080/сущность/идентификатор>)

Рисунок 9 – Запрос сущности по идентификатору

1. В задаче shadowJar добавить к jar-файлу вашу фамилию

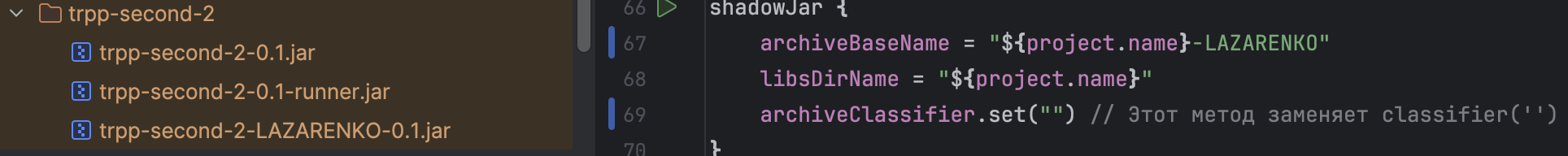


Рисунок 10 – Добавление фамилии к jar-файлу

1. Выполнить задачу checkstyleMain. Посмотреть сгенерированный отчет. Устранить ошибки оформления кода.

Выполним команду *./gradlew checkstyleMain:*



Рисунок 11 – Запуск checkstyleMain

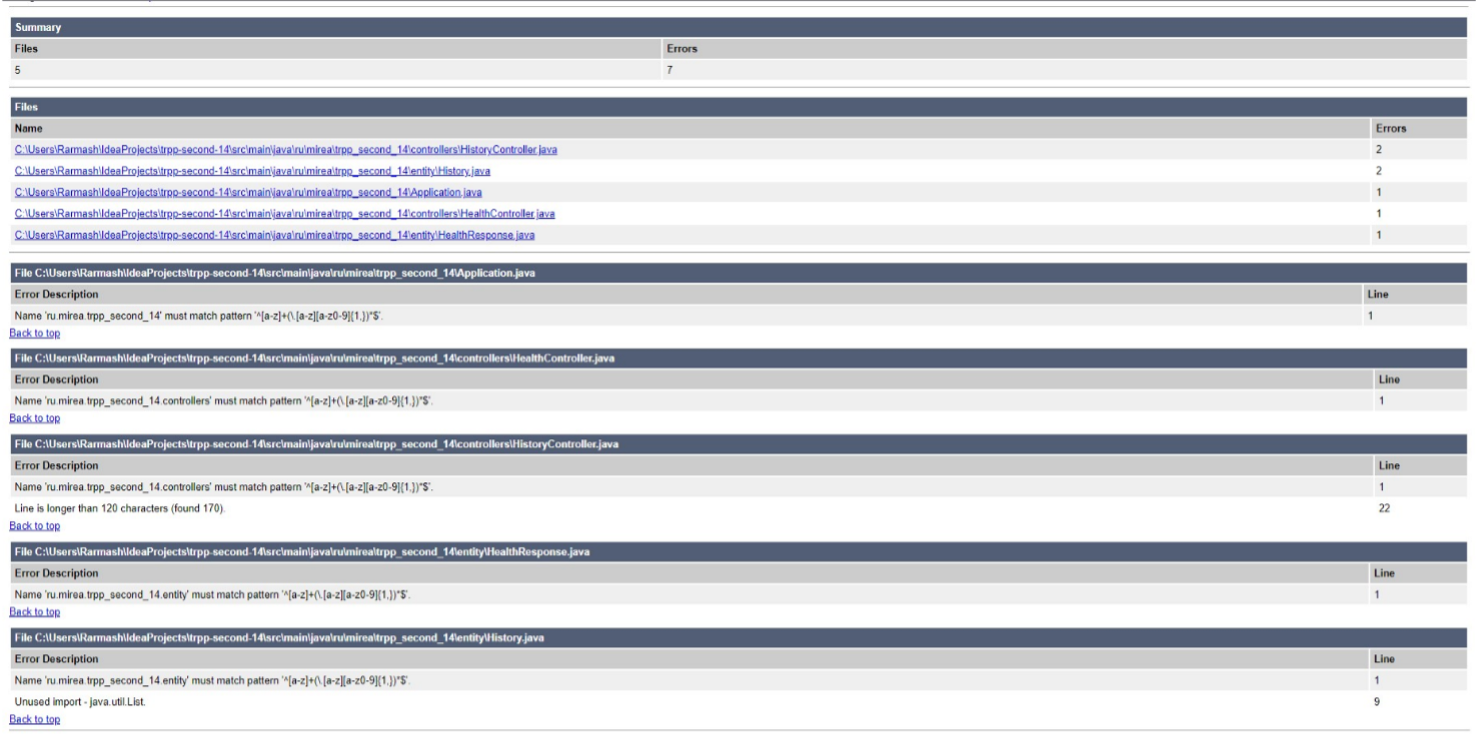


Рисунок 12 – Выявленные ошибки

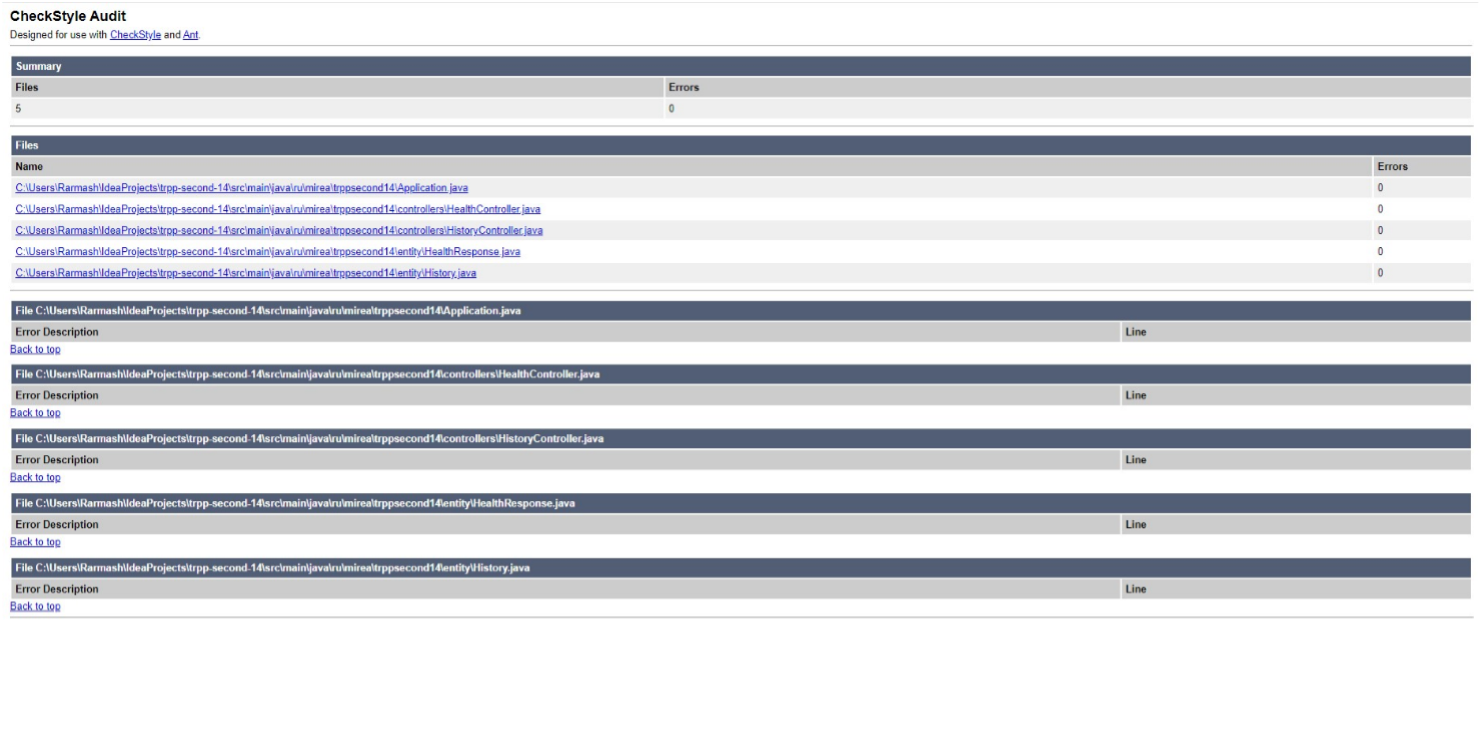


Рисунок 13 – Отсутствие ошибок

# КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое Gradle?

Gradle - это инструмент сборки проектов, который используется в основном для разработки программного обеспечения на языке Java, хотя его также можно использовать и для других языков программирования. Он предоставляет гибкий способ управления проектами и их зависимостями, а также автоматизации сборки, тестирования и развертывания приложений. Gradle использует DSL (Domain Specific Language) на основе Groovy или Kotlin для описания сборочных скриптов, что делает его более удобным и гибким по сравнению с другими инструментами сборки.

1. Что такое Maven?

Maven - это инструмент управления проектами, который обеспечивает сборку, отчетность и управление зависимостями в проектах на Java. Он использует XML-файлы для описания структуры проекта, его зависимостей и сборочных целей. Maven автоматически загружает зависимости из центрального репозитория и обеспечивает стандартизированные структуры проектов, что делает его широко используемым инструментом в сообществе Java.

1. Что делает задача build?

Задача build (сборка) в инструментах сборки, таких как Gradle или Maven, отвечает за компиляцию и сборку исходного кода проекта в исполняемый или дистрибутивный файл. Это включает в себя выполнение различных этапов, таких как компиляция исходного кода, копирование ресурсов, создание исполняемых файлов, запаковку и т. д.

1. Что делает задача compileJava?

Задача compileJava отвечает за компиляцию исходного кода Java в байт-код Java (файлы .class). Она часто является частью процесса сборки проекта и выполняется перед другими этапами сборки, такими как создание исполняемых файлов или дистрибутивов.

1. Что такое javadoc?

Javadoc – это инструмент, который автоматически генерирует документацию API на основе комментариев в исходном коде Java.

1. Что такое checkstyle?

Checkstyle – инструмент статического анализа кода для языка Java, который используется для обеспечения согласованности и соответствия кода определенным стандартам оформления.

# ВЫВОД

В ходе выполнения данной работы были приобретены и освоены навыки работы с gradle, сборкой проекта, работы с сервером.