САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Практическая работа 4

Выполнил:

Крохин Владимир

БР1.1

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

- реализовать тестирование API средствами Postman;
- написать тесты внутри Postman.

Ход работы

Для выполнения домашнего задания за основу была взята версия проекта из ДЗ3. В ДЗ3 был проведен импорт Swagger-документации в коллекцию Postman (средствами Postman).

В начале выполнения домешнего задания была произведена настройка переменных окружения Postman:

- было создано окружение с названием Range Rookies (название приложения - стрелкового дневника)
- в этом окружении были созданы переменные окружения baseUrl со значением “http://localhost:8000”, token (без инициализации), userId (без инициализации).

Результаты см. в Рис. 1.

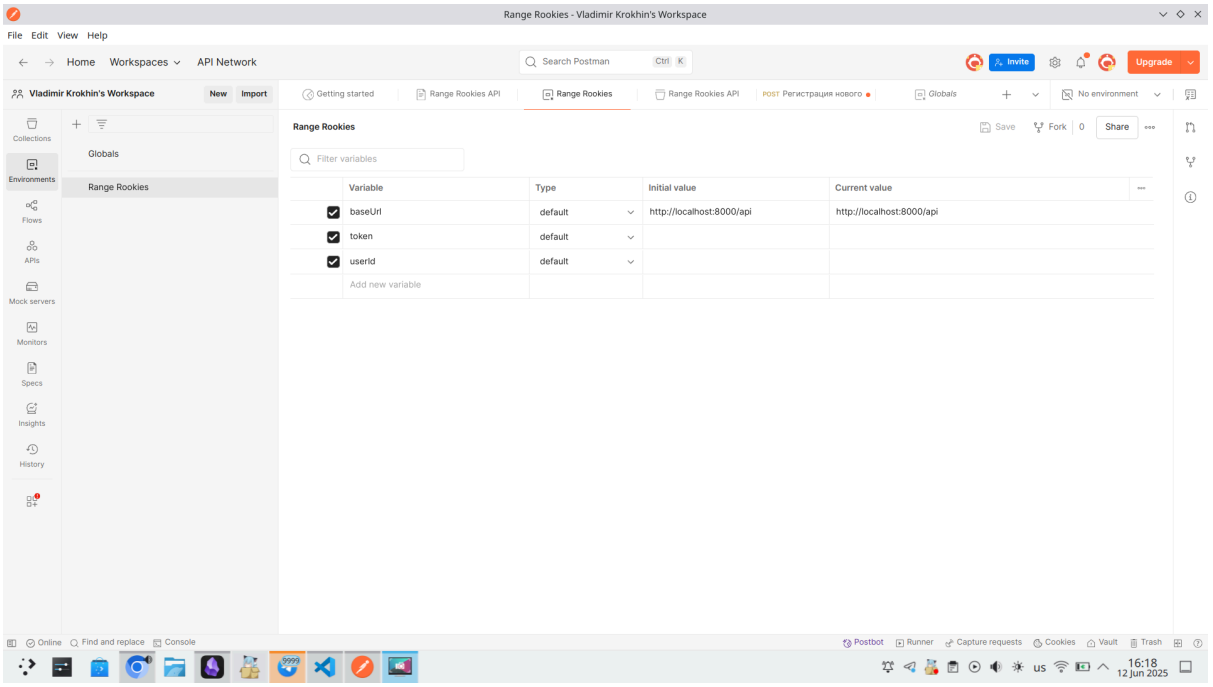

Рис. 1. Переменные окружения Postman проекта

Далее на странице запроса регистрации (/api/auth/register) во вкладке Body было заполнено тело запроса следующим образом. См. рис. 2.

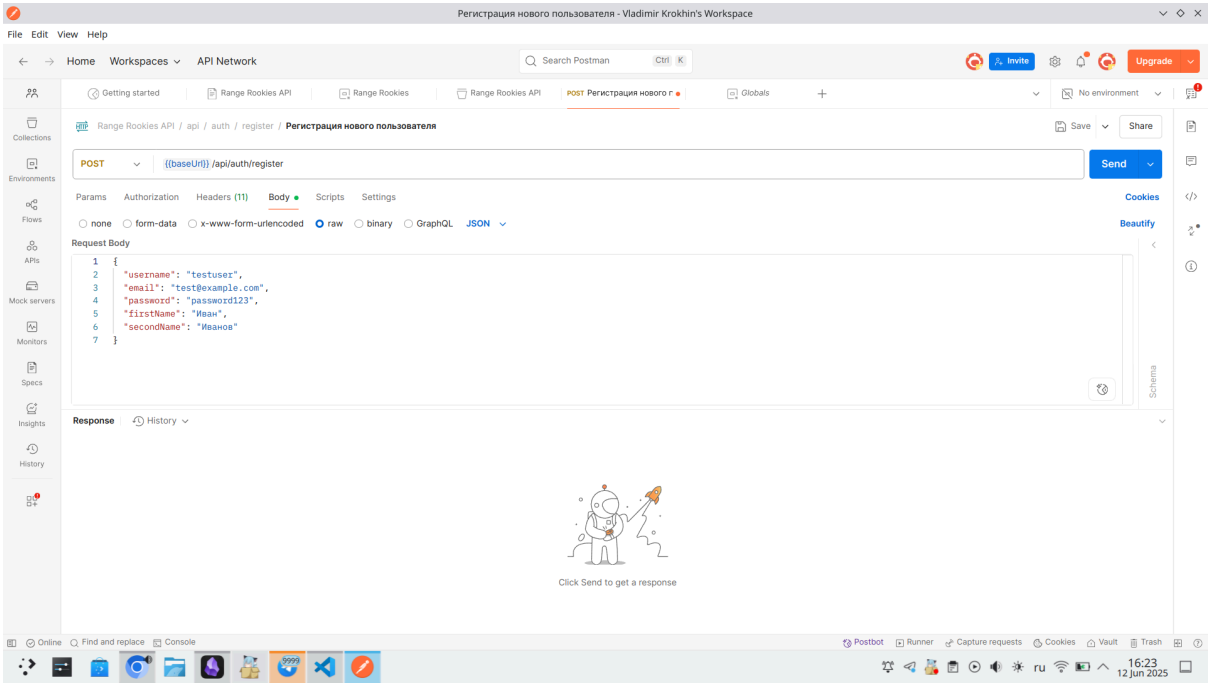

Рис. 2. Страница POST-запроса регистрации пользователя (вкладка Body)

По нажатию на кнопку Send (синяя кнопка в правом верхнем углу), был отправлен запрос. Полученный ответ - 200 OK и тело с accessToken. accessToken далее был записан в переменную окружения token.

См. рис. 3.

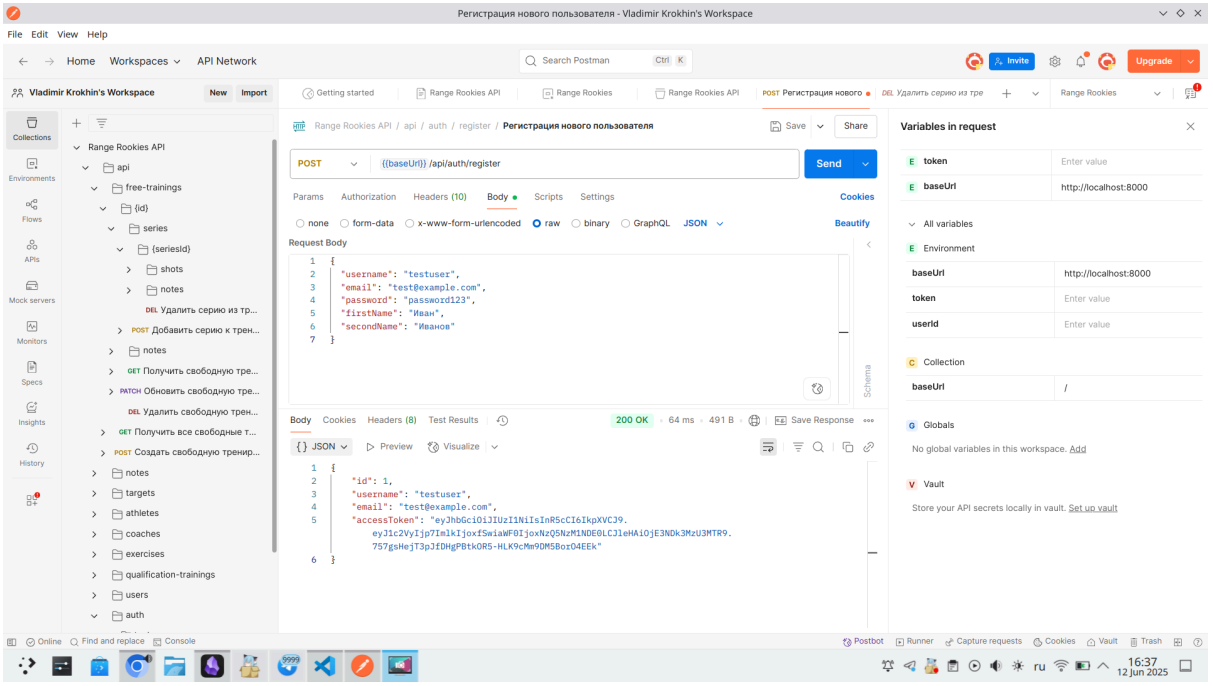

Рис. 3. Ответ на POST-запрос регистрации

Далее по адресу /api/weapon-types/ был послан POST-запрос на создание нового типа оружия.

См. рис. 4.

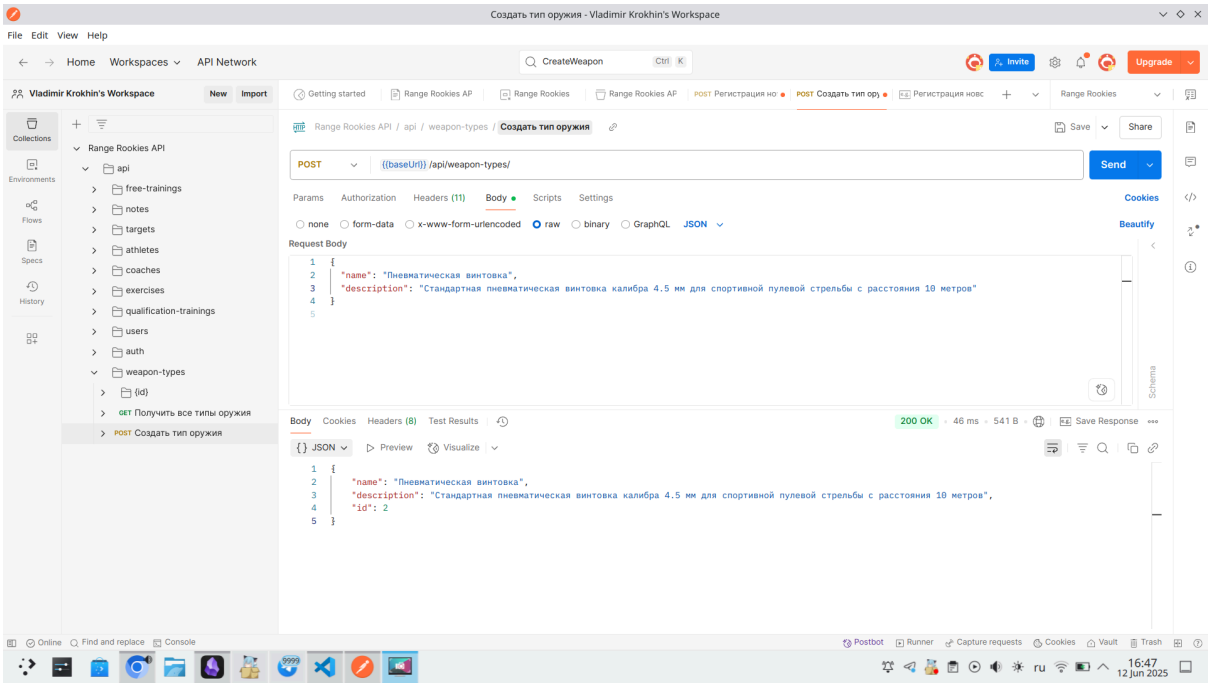

Рис. 4. Ответ на POST-запрос создания вида оружия.

Далее по адресу /api/targets/ был послан POST-запрос на создание нового ресурса мишени. См. рис 5.

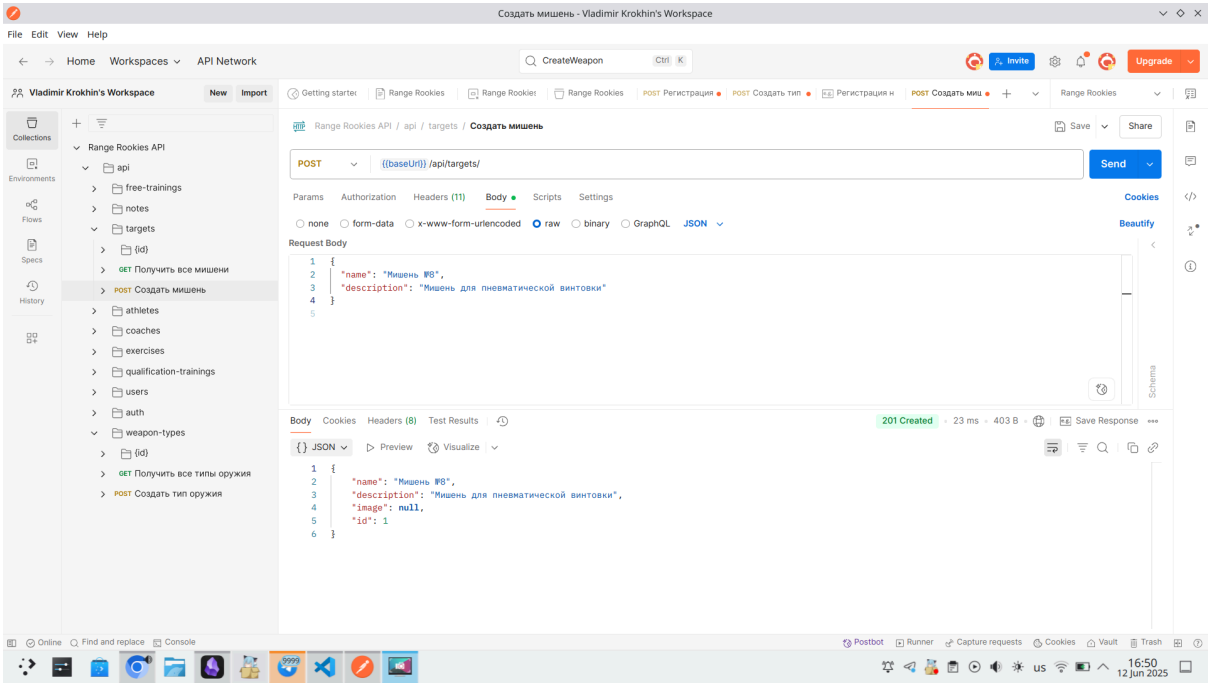

Рис. 5. Ответ на POST-запрос создания новой мишени.

Далее по адресу /api/athletes/ был послан POST-запрос на создание спортсмена. См. рис. 6.

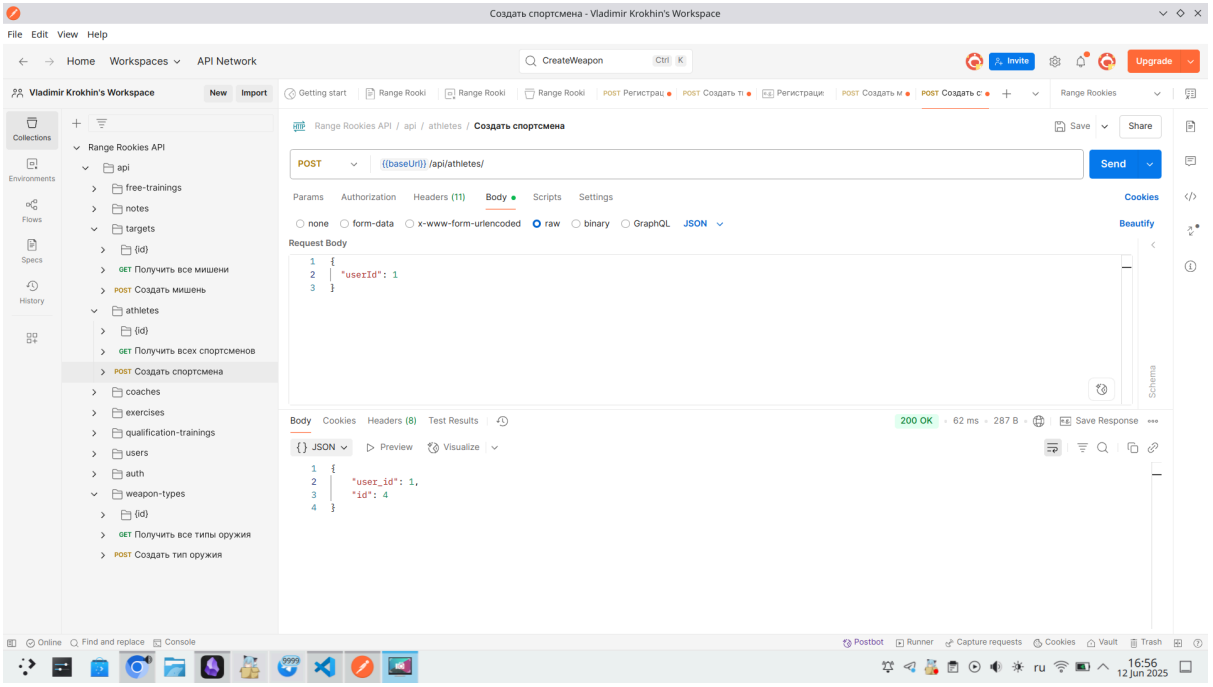

Рис. 6. Ответ на POST-запрос создания спортсмена.

Далее был послан POST-запрос на создание свободной тренировки. См. рис. 7

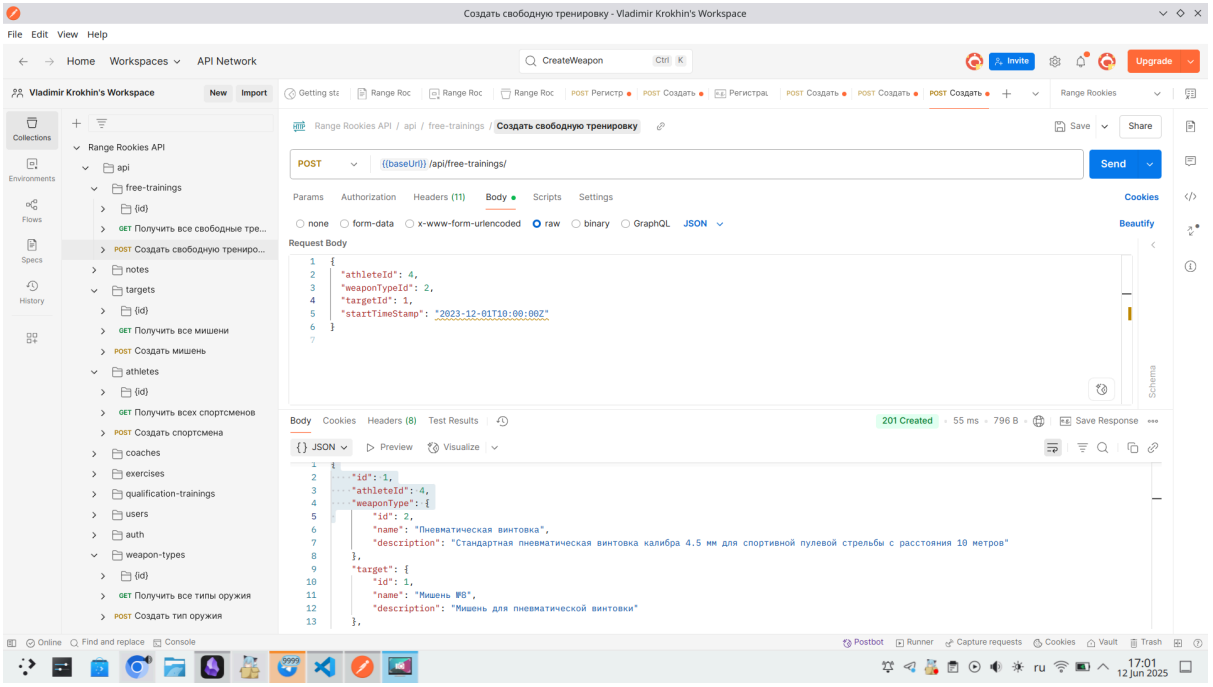

Рис. 7. Ответ на POST-запрос создания свободной тренировки.

Далее по адресу /api/free-trainings/:id/series, где id был указан 1, был послан POST-запрос на добавление новой серии свободной тренировке. См. рис. 8.

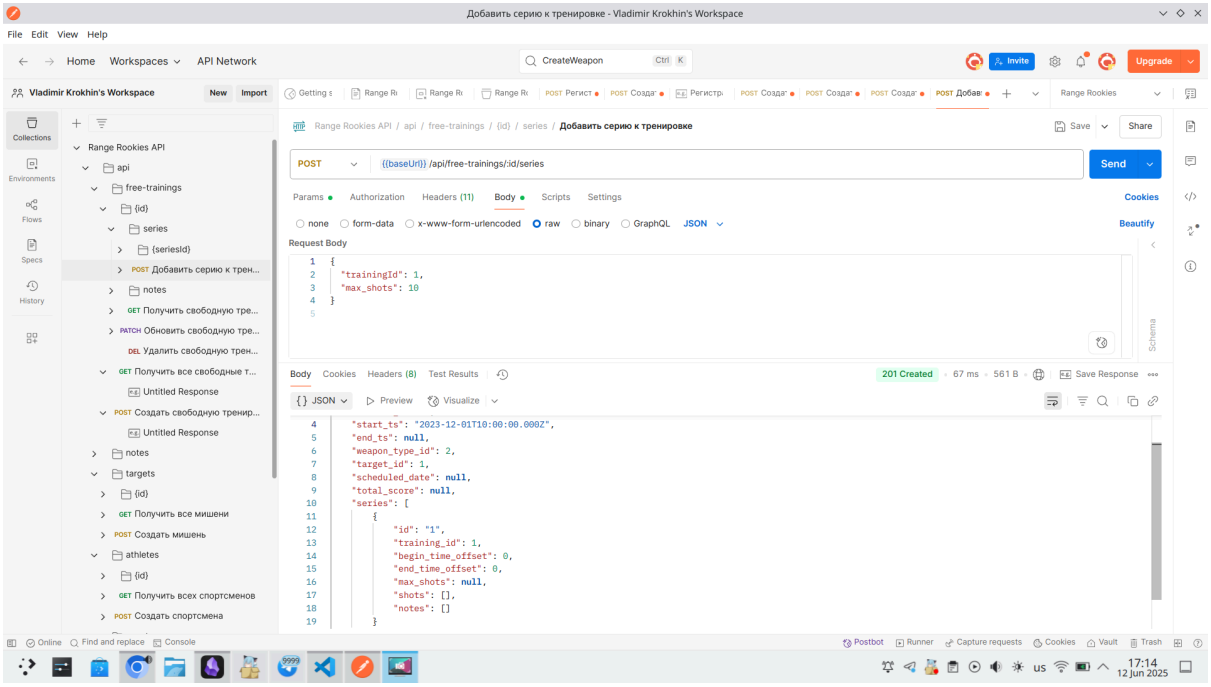

Рис. 8. Ответ на POST-запрос добавления новой серии свободной тренировке

Далее по адресу /api/free-trainings/:id/series/:seriesId/shots был послан POST-запрос на добавление выстрела в серию. id серии и id тренировки был указан в 1 и 1 соответственно. См. рис. 9.

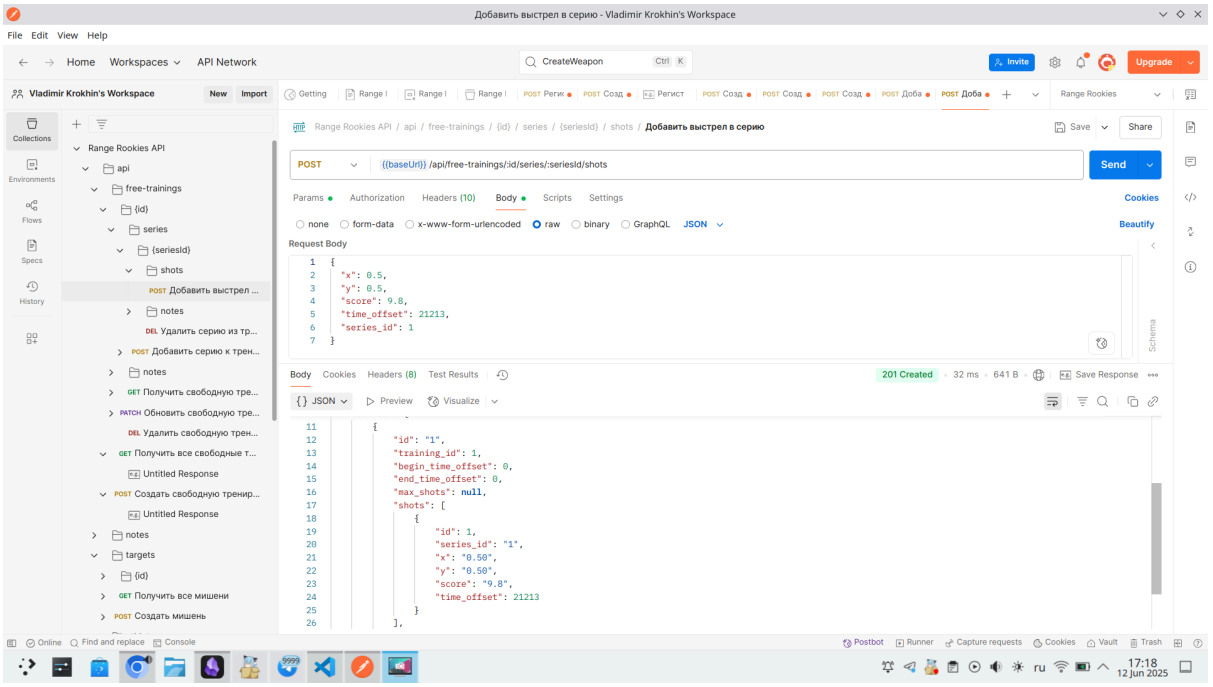

Рис. 9. Ответ на POST-запрос добавления нового выстрела в серию свободной тренировке.

Ввиду большого количества эндпоинтов, все из них не будут рассмотрены в данном отчете.

Приступим к написанию автоматизированных тестов внутри Postman.

В ходе выполнения домашнего задания были написаны 5 тестов Postman для проверки API.

Тест 1: Проверка доступности API

Для запроса "GET /api/targets" (получение списка мишеней)

```
pm.test("Сервер доступен и возвращает статус 200", function() {
    pm.response.to.have.status(200);
});

pm.test("Ответ содержит массив данных", function() {
    var jsonData = pm.response.json();
    pm.expect(Array.isArray(jsonData)).to.be.true;
});
```

Тест 2: Проверка регистрации с обработкой ошибок

Для запроса "POST /api/auth/register":

```
pm.test("Проверка ответа на запрос регистрации", function() {
    // Проверяем, что ответ получен
    pm.response.to.not.be.error;
```

```
    // Проверяем статус ответа (может быть 201 при успехе или 400/500 при ошибке)
  if (pm.response.code === 201) {
    // Если успешно, проверяем структуру ответа
    var jsonData = pm.response.json();
    pm.expect(jsonData).to.have.property('id');
    pm.expect(jsonData).to.have.property('username');
    pm.expect(jsonData).to.have.property('email');
    pm.expect(jsonData).to.have.property('accessToken');

    // Сохраняем токен и ID для использования в других запросах
    if (jsonData.accessToken) {
      pm.environment.set("token", jsonData.accessToken);
    }
    if (jsonData.id) {
      pm.environment.set("userId", jsonData.id);
    }
  } else {
    // Если ошибка, проверяем наличие сообщения об ошибке
    try {
      var jsonData = pm.response.json();
      pm.expect(jsonData).to.have.property('message');
      console.log("Ошибка регистрации: " + jsonData.message);
```

```
        } catch (e) {
            console.log("Ошибка при обработке ответа: " + e);
        }
    }
});
```

Тест 3: Проверка аутентификации

Для запроса "POST /api/auth/login":

```
pm.test("Проверка аутентификации", function() {
    // Проверяем статус ответа
    if (pm.response.code === 200) {
        var jsonData = pm.response.json();
        pm.expect(jsonData).to.have.property('accessToken');

        // Сохраняем токен для использования в других запросах
        if (jsonData.accessToken) {
            pm.environment.set("token", jsonData.accessToken);
            console.log("Токен успешно получен и сохранен");
        }
    } else {
        try {
```

```
        var jsonData = pm.response.json();
            console.log("Ошибка аутентификации: " + (jsonData.message || "Неизвестная ошибка"));
    } catch (e) {
        console.log("Ошибка при обработке ответа: " + e);
    }
  }
});
```

Тест 4: Проверка создания и получения мишени

Для запроса "POST /api/targets":

```
pm.test("Создание мишени", function() {
  if (pm.response.code === 201) {
    var jsonData = pm.response.json();
    pm.expect(jsonData).to.have.property('id');
    pm.expect(jsonData).to.have.property('name');

    // Сохраняем ID мишени для использования в других запросах
    if (jsonData.id) {
      pm.environment.set("targetId", jsonData.id);
      console.log("ID мишени сохранен: " + jsonData.id);
    }
```

```
    } else {
        try {
            var jsonData = pm.response.json();
                console.log("Ошибка создания мишени: " + (jsonData.message ||
"Неизвестная ошибка"));
        } catch (e) {
            console.log("Ошибка при обработке ответа: " + e);
        }
    }
});
```

Для запроса "GET /api/targets/{{targetId}}":

```
pm.test("Получение мишени по ID", function() {
    if (pm.response.code === 200) {
        var jsonData = pm.response.json();
        pm.expect(jsonData).to.have.property('id');

pm.expect(jsonData.id).to.eql(Number(pm.environment.get("targetId")));
        pm.expect(jsonData).to.have.property('name');
    } else {
        try {
```

```
        var jsonData = pm.response.json();
          console.log("Ошибка получения мишени: " + (jsonData.message || "Неизвестная ошибка"));
      } catch (e) {
        console.log("Ошибка при обработке ответа: " + e);
      }
   }
});
```

Тест 5: Проверка создания тренировки и добавления серии

Для запроса "POST /api/free-trainings”:

```
pm.test("Создание свободной тренировки", function() {
   if (pm.response.code === 201) {
      var jsonData = pm.response.json();
      pm.expect(jsonData).to.have.property('id');

      // Сохраняем ID тренировки
      if (jsonData.id) {
         pm.environment.set("trainingId", jsonData.id);
         console.log("ID тренировки сохранен: " + jsonData.id);
      }
```

```
    } else {
        try {
            var jsonData = pm.response.json();
             console.log("Ошибка создания тренировки: " + (jsonData.message
|| "Неизвестная ошибка"));
        } catch (e) {
            console.log("Ошибка при обработке ответа: " + e);
        }
    }
});
```

Для запроса "POST /api/free-trainings/{{trainingId}}/series":

```
pm.test("Добавление серии к тренировке", function() {
    if (pm.response.code === 201) {
        var jsonData = pm.response.json();
        pm.expect(jsonData).to.have.property('id');

        // Сохраняем ID серии
        if (jsonData.id) {
            pm.environment.set("seriesId", jsonData.id);
            console.log("ID серии сохранен: " + jsonData.id);
```

```
    }
  } else {
    try {
      var jsonData = pm.response.json();
       console.log("Ошибка добавления серии: " + (jsonData.message ||
"Неизвестная ошибка"));
    } catch (e) {
      console.log("Ошибка при обработке ответа: " + e);
    }
  }
});
```

После этого был проведен запуск всех тестов. См. рис. 10

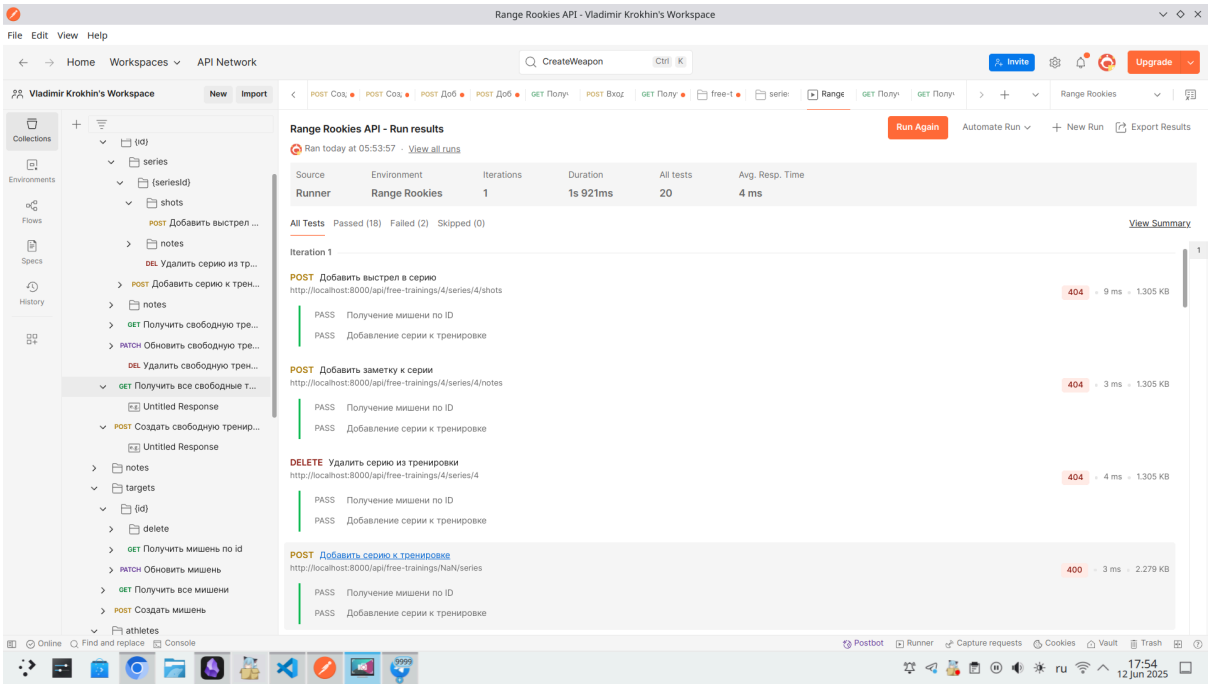

Рис. 10. Запуск тестов Postman

Вывод

В ходе выполнения лабораторной работы был освоне навык тестирования и написания тестов на Postman.