Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра «Информационной безопасности»

Направление подготовки: «Информационная безопасность автоматизированных систем»

ОТЧЕТ

по вариативной части проектной практики

Студент: Хомутов Матвей Васильевич 241-371

Место прохождения практики: Московский Политех

Кафедра: «Информационная безопасность»

Отчет принят с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дата\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики: Гневшев Александр Юрьевич

Москва 2025

Для прохождения вариативной части проектной практики была выбрана тема: «**Build Your Own Redis**»

Стек технологий

* **Язык программирования**: Python
* **Сетевой протокол**: TCP
* **Сериализация данных**: JSON

План работы

1. **Исследование Redis**
   * Изучить, как работает Redis
   * Разобраться с базовой архитектурой.
2. **Создание упрощенной версии**
   * Реализовать сервер на Python, обрабатывающий команды GET/SET/DELETE.
3. **Документирование**
   * Написать Markdown-руководство с кодом и схемами.
   * Добавить UML-диаграммы (например, последовательности для GET/SET).
4. **Оформление**
   * Оформим репозиторий.

**Исследование Redis**

**1.1. Что такое Redis?**

Redis (Remote Dictionary Server) — это key-value база данных в памяти с возможностью сохранения на диск.

* Хранит данные в оперативной памяти → очень быстрый (микросекунды на запрос).
* Поддерживает разные типы данных: строки, списки, хеши, множества.
* Имеет простые команды (SET, GET, DEL и др.).
* Работает по клиент-серверной модели (TCP-соединения).

**1.2. Основные команды Redis**

| Команда | Описание | Пример |
| --- | --- | --- |
| SET | Запись значения по ключу | SET name "Alice" |
| GET | Получение значения по ключу | GET name → "Alice" |
| DEL | Удаление ключа | DEL name |
| EXISTS | Проверка наличия ключа | EXISTS name → 1 (true) |
| KEYS \* | Получить все ключи | KEYS \* → ["name"] |

**1.3. Протокол RESP (Redis Serialization Protocol)**

Redis использует текстовый протокол RESP для обмена данными между клиентом и сервером.

**Формат запроса:**

\*<число\_аргументов>\r\n$<длина\_аргумента1>\r\n<аргумент1>\r\n$<длина\_аргумента2>\r\n<аргумент2>\r\n...

Пример команды SET name Alice:

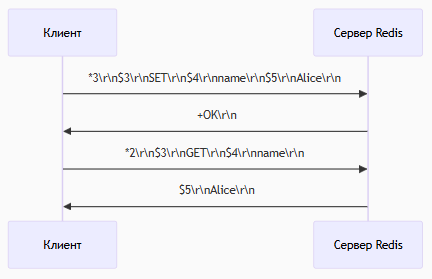
\*3\r\n$3\r\nSET\r\n$4\r\nname\r\n$5\r\nAlice\r\n

**Формат ответа:**

* + — простое строковое сообщение (например, +OK\r\n).
* - — ошибка (например, -Error message\r\n).
* : — целое число (например, :42\r\n).
* $ — бинарная строка (например, $5\r\nhello\r\n).

**2. Архитектура Redis**

Упрощённая схема работы Redis:



1. Сервер слушает TCP-порт (по умолчанию 6379).
2. Клиент подключается и отправляет команды в формате RESP.
3. Сервер парсит команду, выполняет её и возвращает ответ.

**Создание упрощённой версии Redis**

План:

* **Сервер на Python** (используем socket для TCP).
* **Обработка команд** (SET, GET, DEL).
* **Хранение данных** — словарь Python (dict).
* **Сериализация ответов** в формат RESP.

**Код:**

(См. server.py в репозитории)

Так же, необходимо добавить **TTL (время жизни ключей)** — это важная функция Redis, позволяющая автоматически удалять ключи по истечении заданного времени.

**Обновленный код сервера:**

(См. ttl.py в репозитории)

**Документация**

Создаем файл readmy.md, в котором прописываем инструкцию по запуску сервера Redis:

Запуск сервера

Требования

Python 3.7+

Утилита telnet (для тестирования)

1. Установка

Клонируйте репозиторий:

//git clone <https://github.com/matveykh192/Homutov-practice/blob/master/part2/ttlserver.py>

//cd simple-redi

2. Запуск сервера

//python redis\_server.p

Сервер запустится на localhost:6379.

3. Подключение клиента

Откройте новый терминал и подключитесь через telnet:

//telnet localhost 6379

5. Остановка сервера

Нажмите Ctrl+C в терминале с сервером.

6. Пример сессии

SET user\_name "John Doe"

+OK

GET user\_name

$8

John Doe

EXPIRE user\_name 30  # Установит TTL 30 секунд

:1

TTL user\_name

:27  # Осталось 27 секунд

KEYS \*

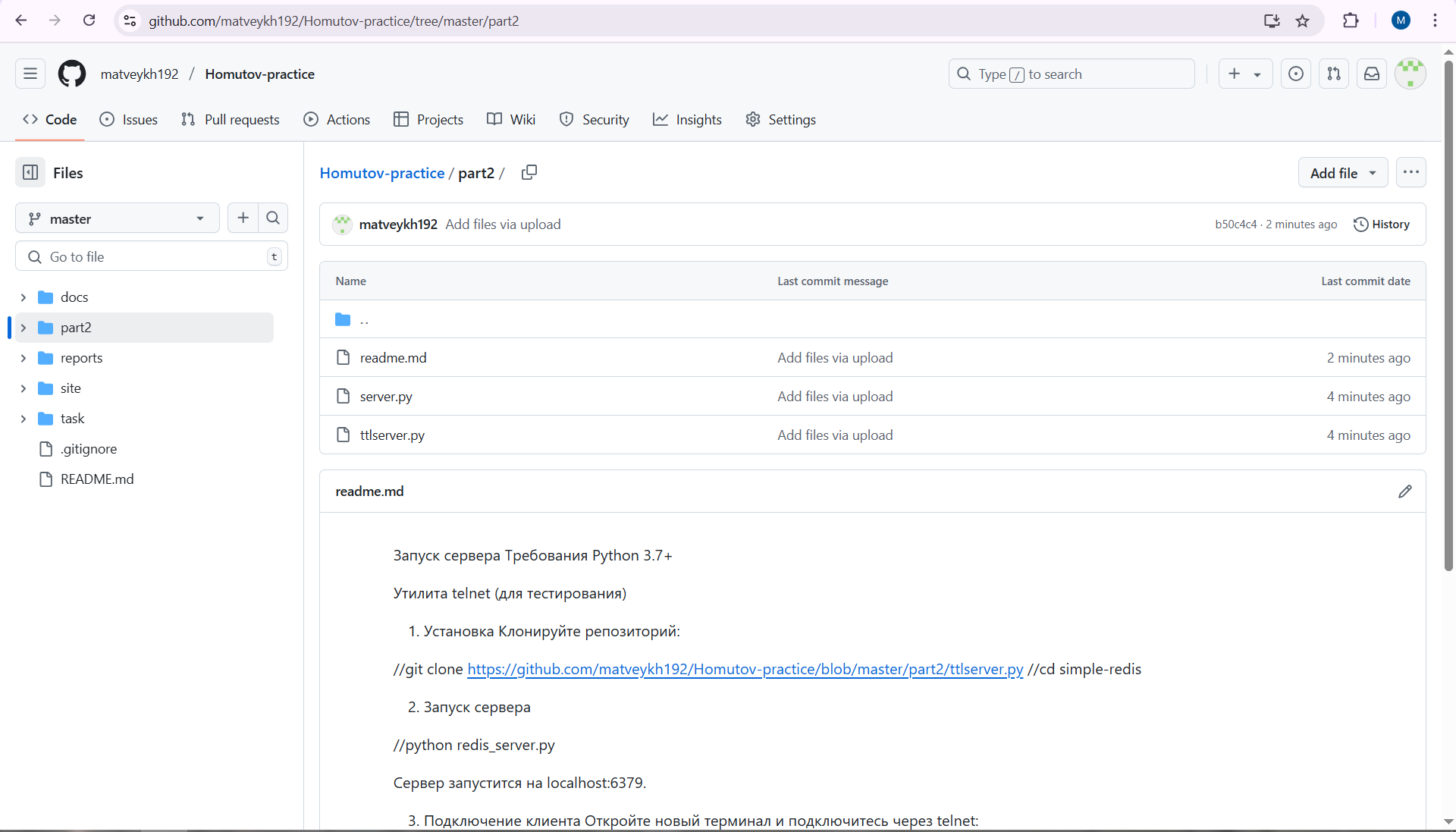
\*1

$9

user\_name

**Оформление**

Загружаем наши файлы в GitHub



**Заключение**

В ходе выполнения проекта была разработана упрощённая реализация Redis-сервера на языке Python, включающая базовый функционал key-value хранилища с поддержкой основных команд: SET, GET, DEL, EXISTS, KEYS, а также механизма TTL.

Проект соответствует следующим критериям:

* Реализован клиент-серверный протокол на основе TCP с использованием сокетов
* Поддерживается текстовый протокол RESP для обмена данными
* Обеспечена потокобезопасность за счёт многопоточной архитектуры

Результаты работы представлены в виде:

1. Исходного кода сервера (redis\_server.py)
2. Документации

Для развертывания достаточно выполнить стандартную процедуру клонирования репозитория и запуска серверного приложения, как указано в сопроводительной документации.