



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт кибербезопасности и цифровых технологий (ИКБ)

КБ-2 «Информационно-аналитические системы кибербезопасности»

ОТЧЕТ О ВЫПОЛНЕНИИ ИНДИВИДУАЛЬНОГО ЗАДАНИЯ №6
В РАМКАХ ДИСЦИПЛИНЫ «ТЕХНОЛОГИИ
ХРАНЕНИЯ В СИСТЕМАХ КИБЕРБЕЗОПАСНОСТИ»

Выполнил:

Студент 3-ого курса

Учебной группы БИСО-02-22

Зубарев В.С.

Москва 2024

Разверните Redis и RedisInsight (Redis Stack) с помощью Docker Compose.
Файл docker-compose:

```
version: '3.8'
```

```
networks:
```

```
  redis:
```

```
    name: redis
```

```
    driver: bridge
```

```
services:
```

```
  redis-stack:
```

```
    image: redis/redis-stack:latest
```

```
    container_name: redis_zvs
```

```
    networks:
```

```
      - redis
```

```
    ports:
```

```
      - "6305:6379"
```

```
      - "60005:8001"
```

```
    environment:
```

```
      REDIS_ARGS: "--requirepass adminzvs"
```

```
    restart: unless-stopped
```

```
  redis-insight:
```

```
    image: redislabs/redisinsight:latest
```

```
    container_name: insight_zvs
```

```
    networks:
```

```
      - redis
```

```
    ports:
```

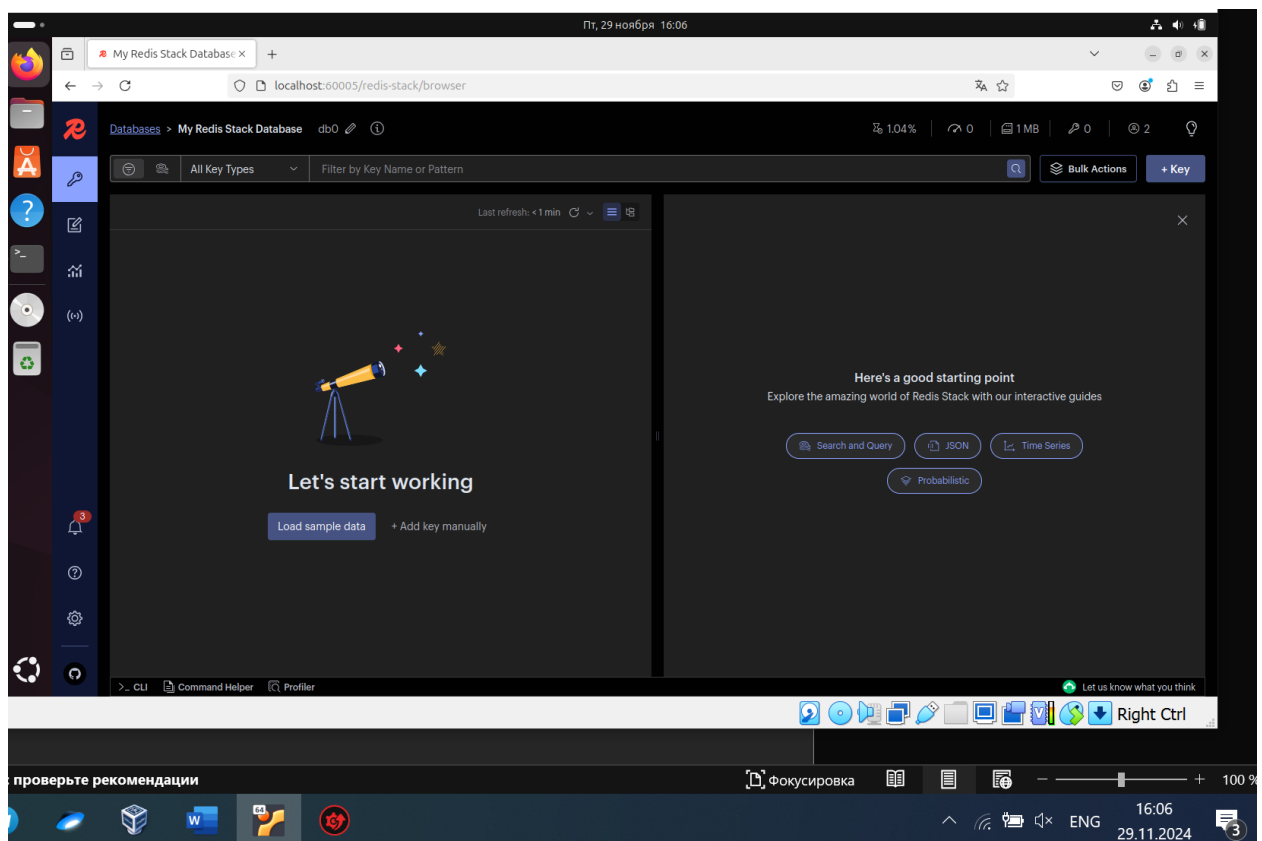
```
      - "6005:8001"
```

```
    restart: unless-stopped
```

```
zvs05@zubarev: ~/docker/redis
WARN[0000] /home/zvs05/docker/redis/docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 2/2
 ✓ Container insight_zvs Started      0.3s
 ✓ Container redis_zvs Started        0.3s
zvs05@zubarev: ~/docker/redis$ sudo docker ps -a
```

| CONTAINER ID | IMAGE | COMMAND | CREATED | STATUS | PORTS |
|--------------|-------------------------------|-------------------------|---------------|--------------|--|
| 37b2c6b63f52 | redislabs/redisinsight:latest | "/docker-entry.sh n..." | 7 seconds ago | Up 6 seconds | 5000/tcp, 0.0.0.0:6005->8001/tcp, [::]:6005->8001/tcp |
| 55198aa5adb2 | redis/redis-stack:latest | "/entrypoint.sh" | 7 seconds ago | Up 6 seconds | 0.0.0.0:6305->6379/tcp, [::]:6305->6379/tcp, 0.0.0.0:6005->8001/tcp, [::]:6005->8001/tcp |

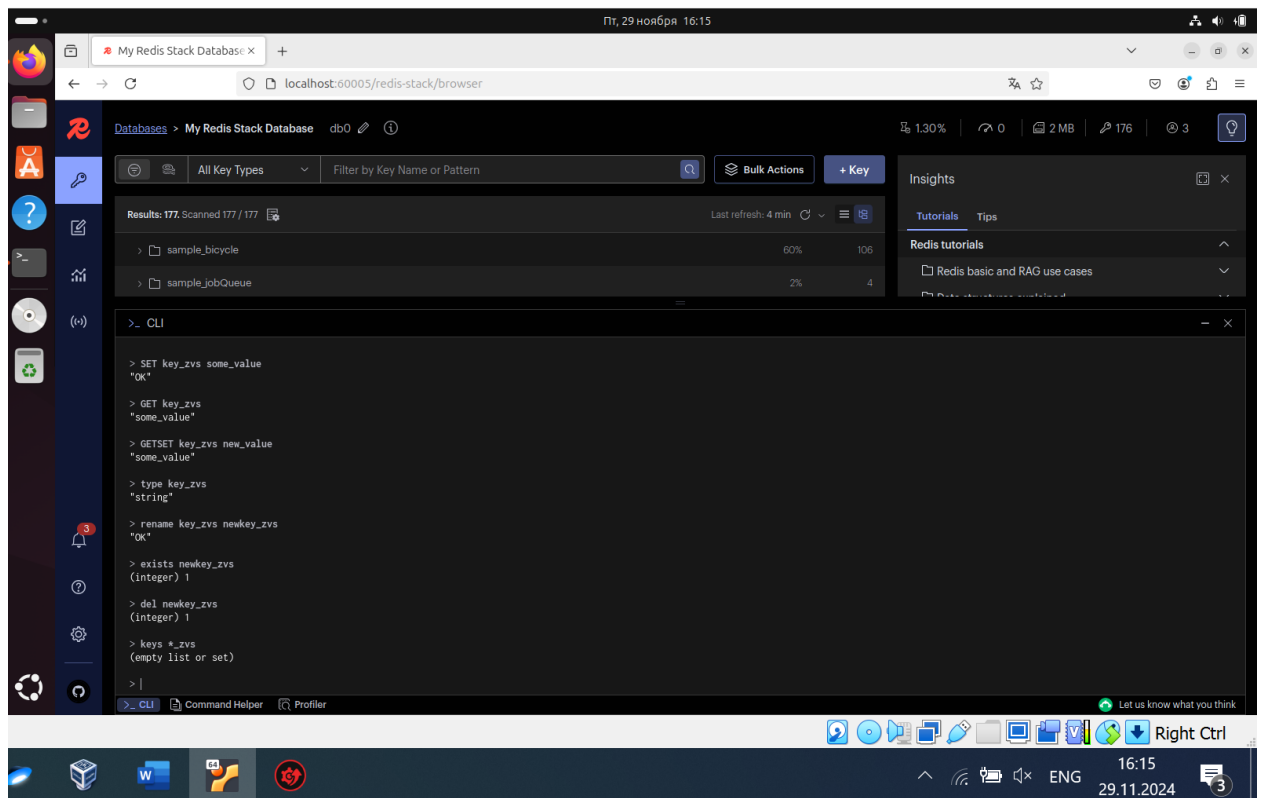
```
zvs05@zubarev: ~/docker/redis$
```



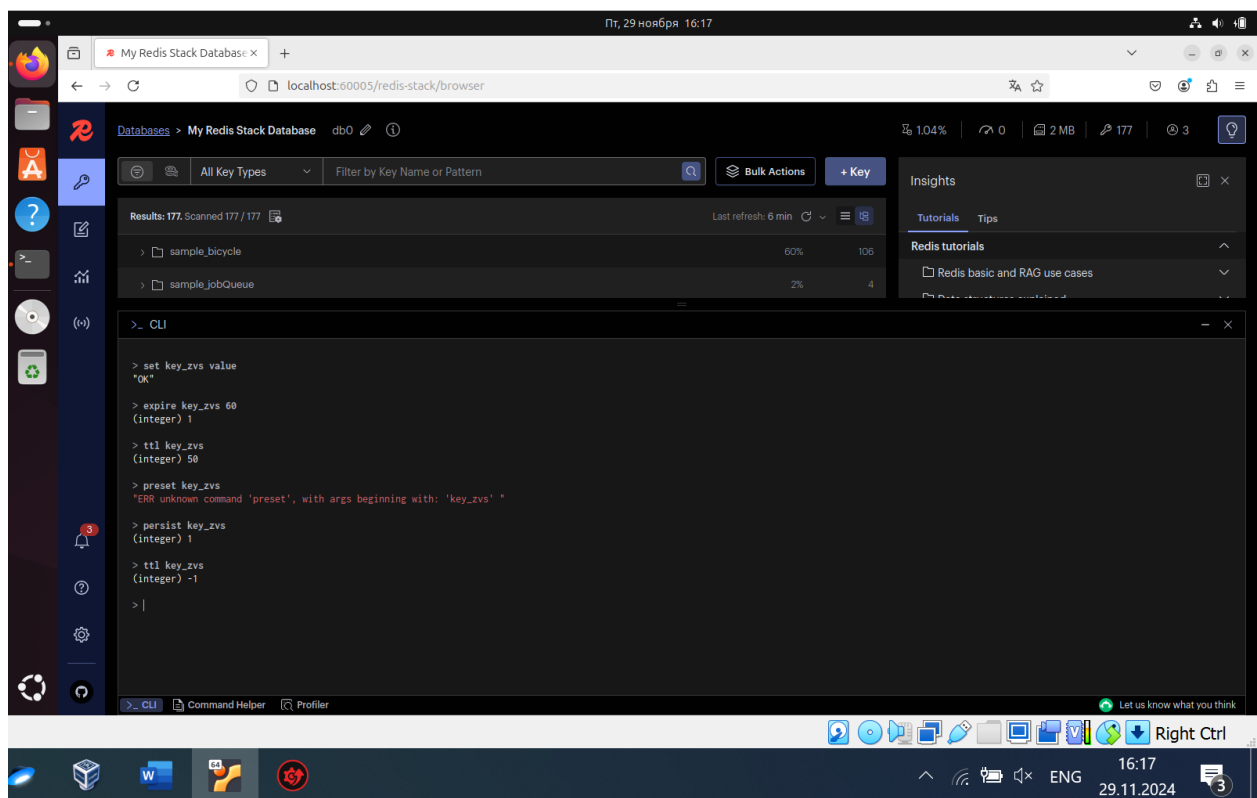
Изучите команды по созданию, выборке, модификации, удалению и получению базовой информации об объектах с использованием интерфейса командной строки (CLI):

– set

- get
- getset
- type
- rename
- exists
- del
- keys

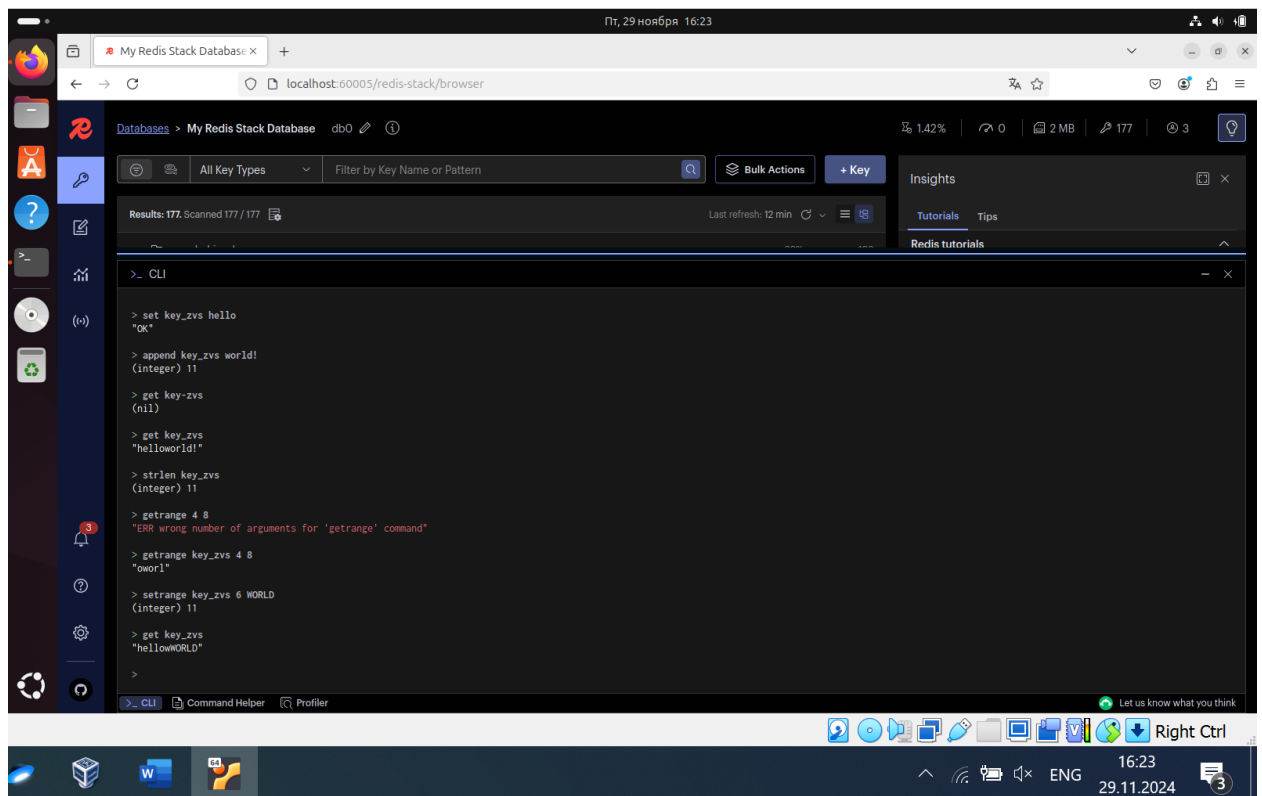


Изучите возможность использования параметра времени жизни объекта (TTL). Изучите команды `ttl` и `expire`



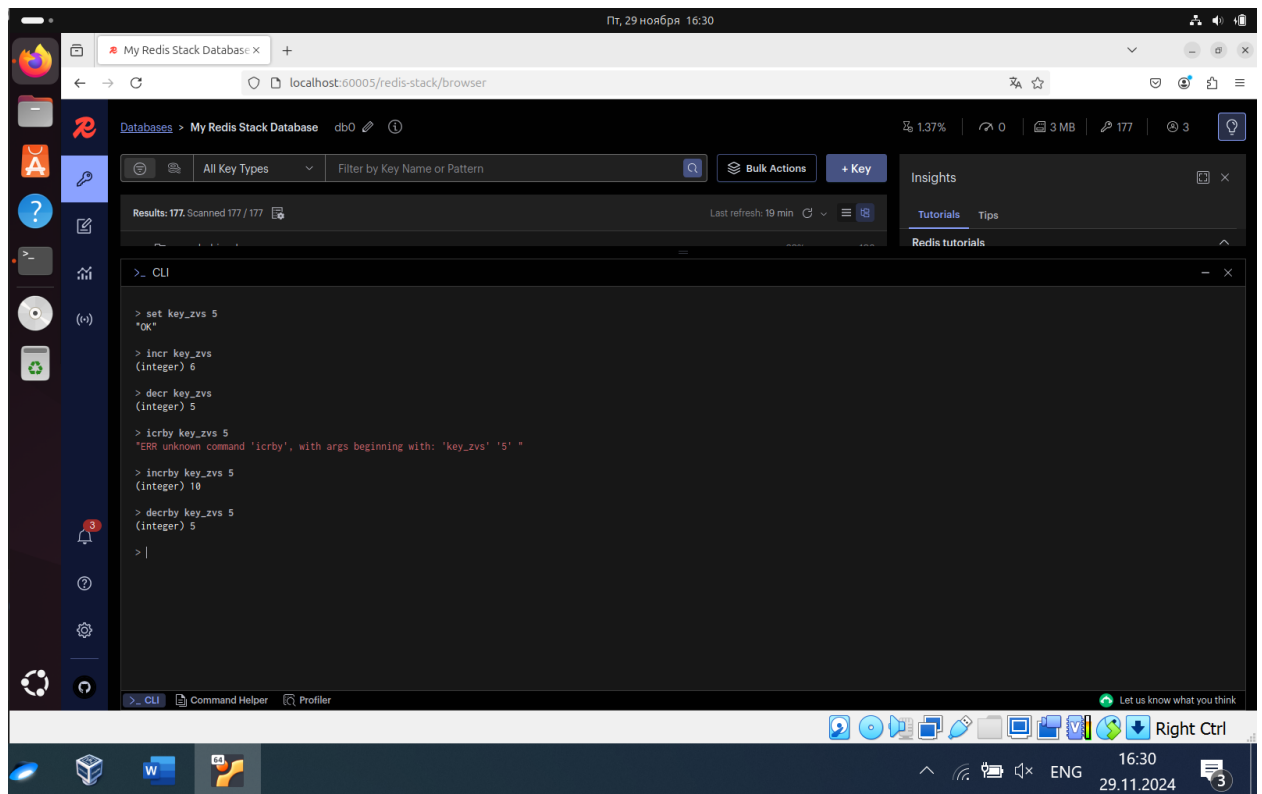
Изучите основные строковые операции:

- append
- strlen
- getrange
- setrange



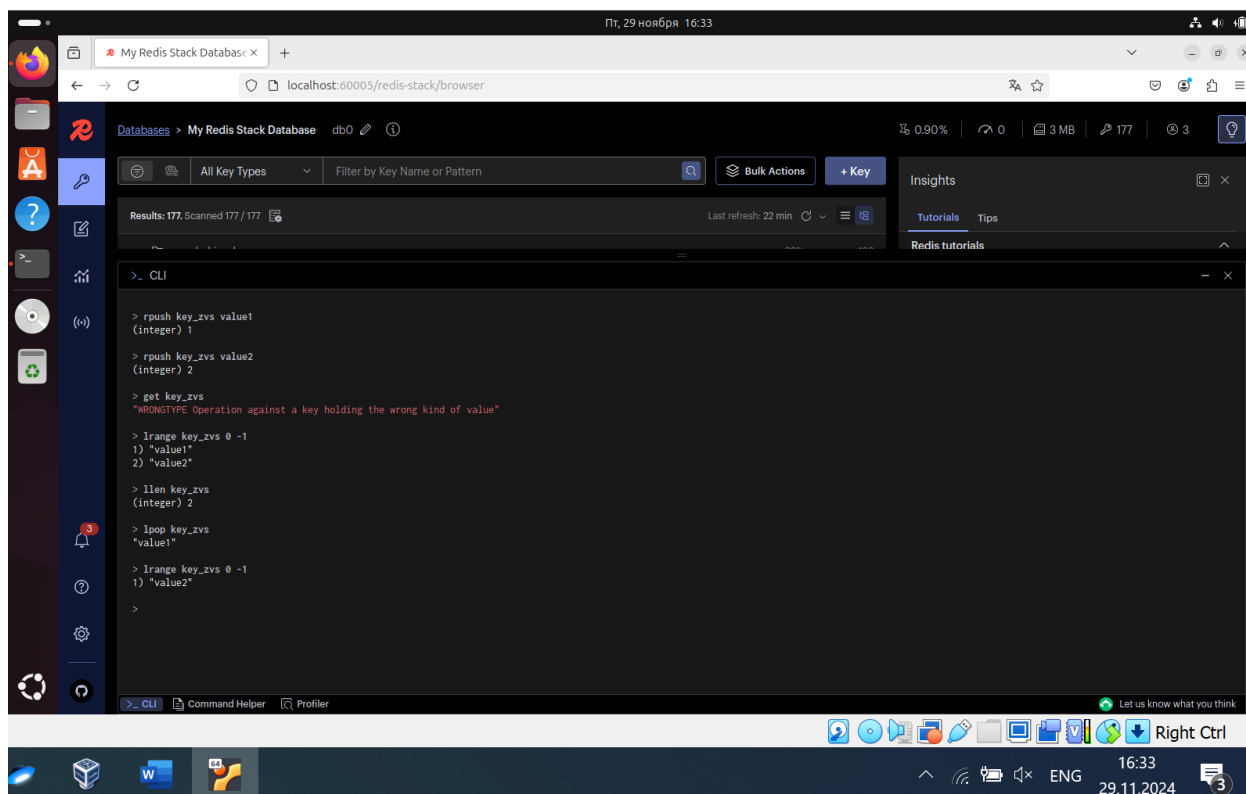
Изучите операции над числами:

- incr
- decr
- incrby
- decrby



Изучите основные операции над списками:

- rpush
- lrange
- llen
- lpop



Разверните тестовый сервис авторизации на веб-странице с хранением информации о количестве неудачных попыток в Redis. При 3 и более неудачных попытках авторизации для конкретного пользователя необходимо на 60 секунд приостанавливать возможность авторизации.

```
zvs05@zubarev:~/docker/redis$ tree
.
├── app.py
├── docker-compose.yml
├── Dockerfile
├── requirements.txt
└── templates
    └── login.html
```

Листинг app.py

```
from flask import Flask, render_template, request
import redis
import time
```

```
app = Flask(__name__)
```

```
# Настройки для подключения к Redis
```



```

redis_host = "redis_zvs" # Имя контейнера Redis из docker-compose.yml
redis_port = 6379
redis_password = "adminzvs"

# Создаем подключение к Redis
r = redis.StrictRedis(host=redis_host, port=redis_port, password=redis_password,
decode_responses=True)

# Данные для авторизации
VALID_USERNAME = "zvs_05"
VALID_PASSWORD = "12345"

# Функция для получения количества неудачных попыток
def get_failed_attempts(username):
    return int(r.get(f"failed_attempts_{username}")) or 0

# Функция для увеличения счетчика неудачных попыток
def increment_failed_attempts(username):
    failed_attempts = get_failed_attempts(username)
    r.set(f"failed_attempts_{username}", int(failed_attempts) + 1, ex=60) #
Сохраняем на 60 секунд

# Функция для сброса счетчика неудачных попыток
def reset_failed_attempts(username):
    r.delete(f"failed_attempts_{username}")

# Функция для регистрации удачной попытки
def register_successful_login(username):
    timestamp = time.strftime('%Y-%m-%d %H:%M:%S', time.localtime())
    r.rpush("successful_logins", f"{timestamp}:{username}") # Добавляем запись
в список

# Функция для проверки, заблокирован ли пользователь
def is_blocked(username):

```

```

failed_attempts = get_failed_attempts(username)
if failed_attempts >= 3:
    # Добавляем пользователя в список заблокированных
    r.sadd("blocked_users", username)
    return True
return False

# Функция для проверки, заблокирован ли пользователь в Redis
def is_user_blocked(username):
    return r.sismember("blocked_users", username)

# Главная страница - форма авторизации
@app.route("/", methods=["GET", "POST"])
def login():
    error = None
    username = request.form.get("username")
    password = request.form.get("password")

    # Проверка на заблокированного пользователя
    if username and is_user_blocked(username):
        error = "Ваш аккаунт заблокирован. Попробуйте позже."
        return render_template("login.html", error=error)

    if request.method == "POST":
        # Проверка правильности логина и пароля
        if username == VALID_USERNAME and password ==
VALID_PASSWORD:
            # Успешный вход - сброс неудачных попыток и регистрация удачной
попытки
            reset_failed_attempts(username)
            register_successful_login(username)
            return "Успешный вход!"
        else:
            # Неверный логин или пароль

```

```

        increment_failed_attempts(username)
        error = "Неверный логин или пароль."

    return render_template("login.html", error=error)

@app.route("/blocked")
def blocked_users():
    blocked = r.smembers("blocked_users")
    return f"Заблокированные пользователи: {' '.join(blocked)}"

@app.route("/successful")
def successful_logins():
    logins = r.lrange("successful_logins", 0, -1)
    return f"Успешные попытки входа:<br>" + "<br>".join(logins)

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=True)

```

Листинг docker-compose.yml

```

version: '3.8'

networks:
  redis:
    driver: bridge

services:
  redis-stack:
    image: redis/redis-stack:latest
    container_name: redis_zvs
    networks:
      - redis
    ports:
      - "6305:6379"

```

```
- "60005:8001"
environment:
  REDIS_ARGS: "--requirepass adminzvs"
restart: unless-stopped
```

```
redis-insight:
  image: redislabs/redisinsight:latest
  container_name: insight_zvs
  networks:
    - redis
  ports:
    - "6005:8001"
  restart: unless-stopped
```

```
web:
  build: .
  container_name: web_auth
  ports:
    - "5000:5000"
  environment:
    - REDIS_URL=redis://redis_zvs:6379
    - REDIS_PASSWORD=adminzvs
  depends_on:
    - redis-stack
  restart: unless-stopped
  networks:
    - redis
```

Листинг Dockerfile

```
# Используем официальный Python образ
FROM python:3.9-slim
```

```
# Устанавливаем рабочую директорию
```

WORKDIR /app

Копируем все файлы в контейнер

COPY . /app

Устанавливаем зависимости

RUN pip install --no-cache-dir -r requirements.txt

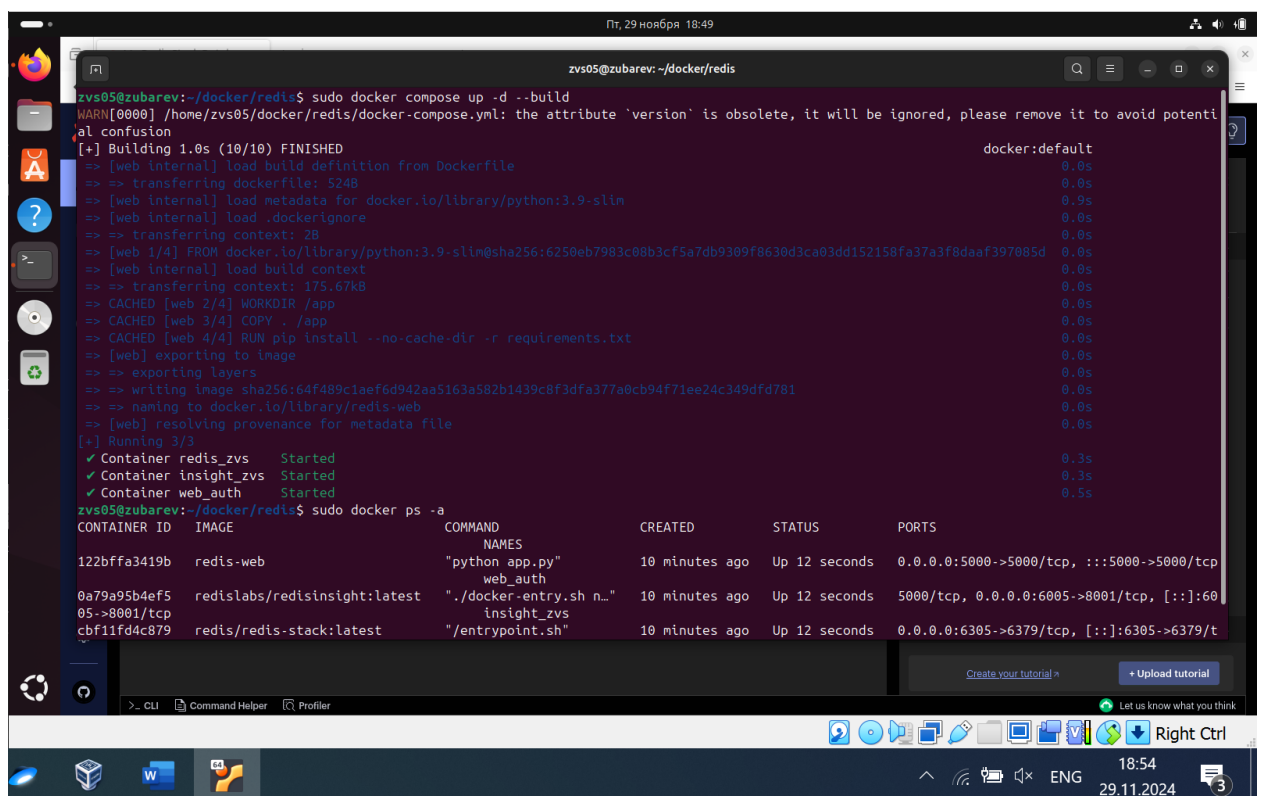
Открываем порт для Flask-приложения

EXPOSE 5000

Запускаем приложение

CMD ["python", "app.py"]

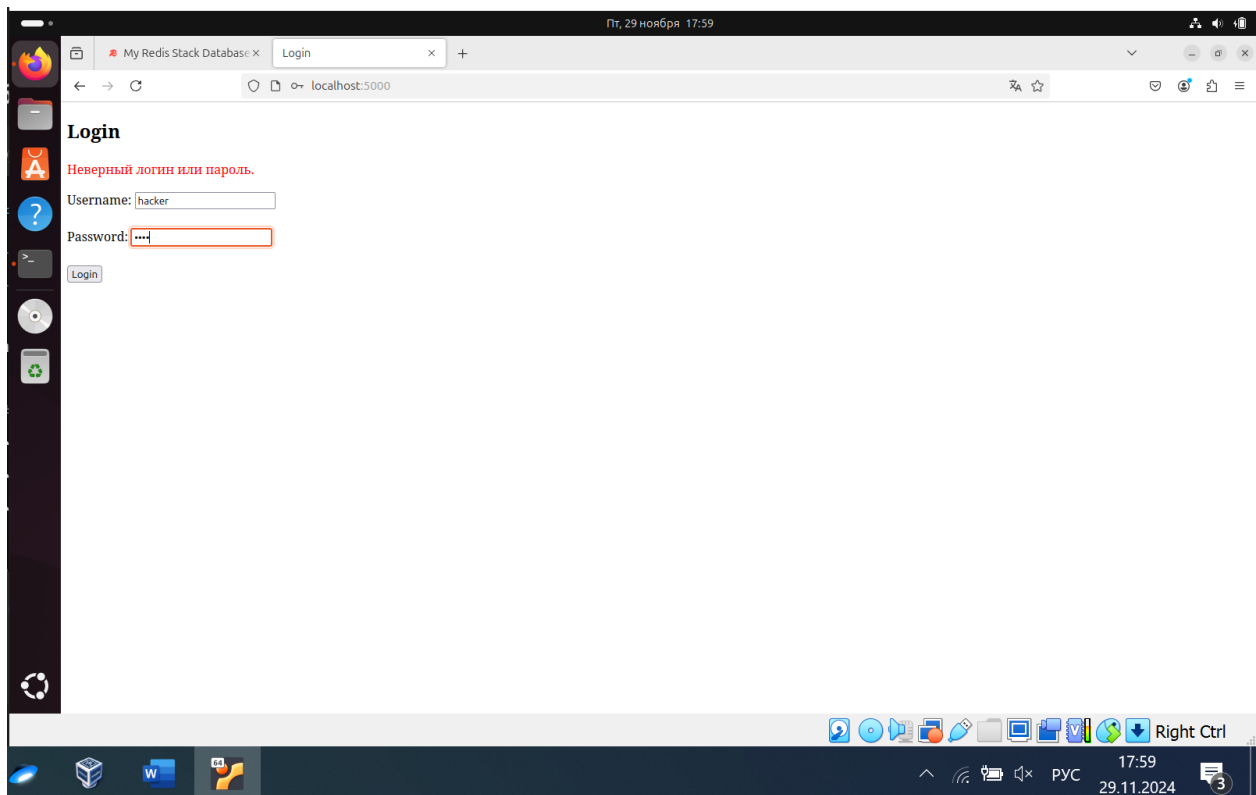
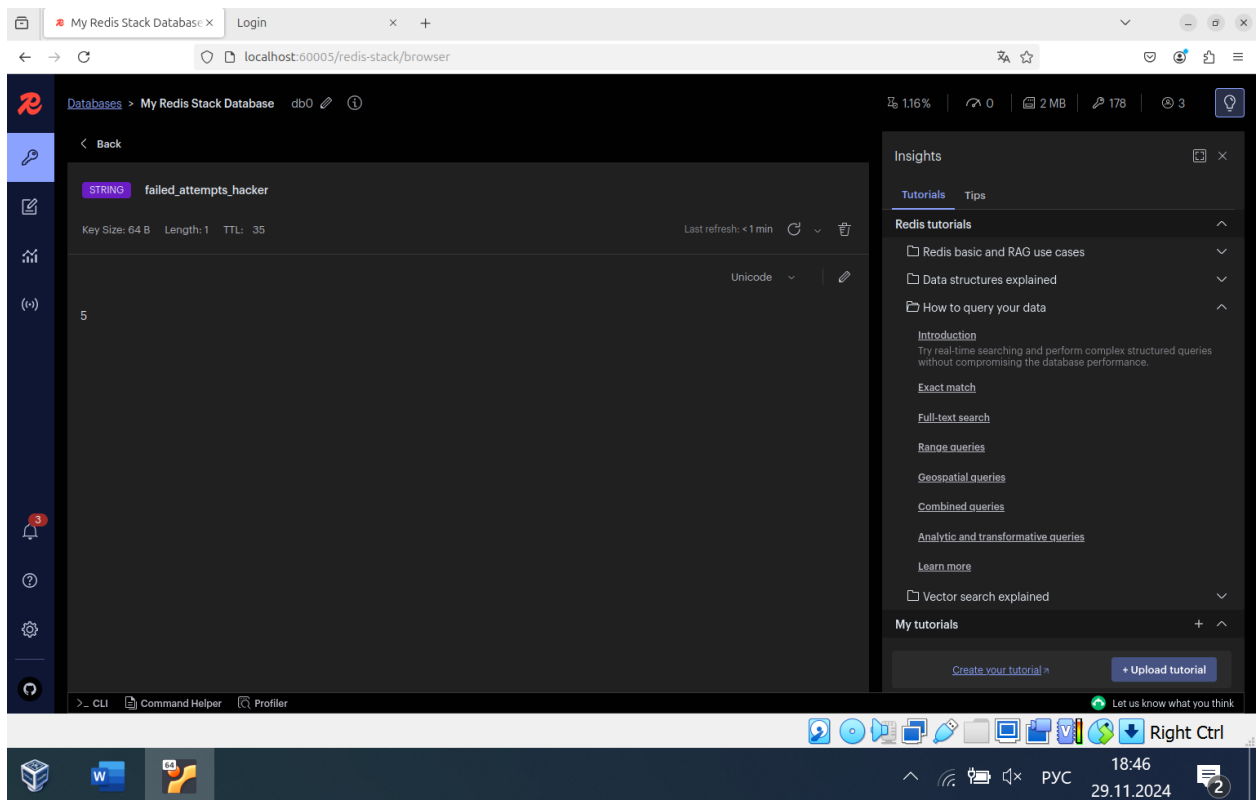
Работа контейнеров

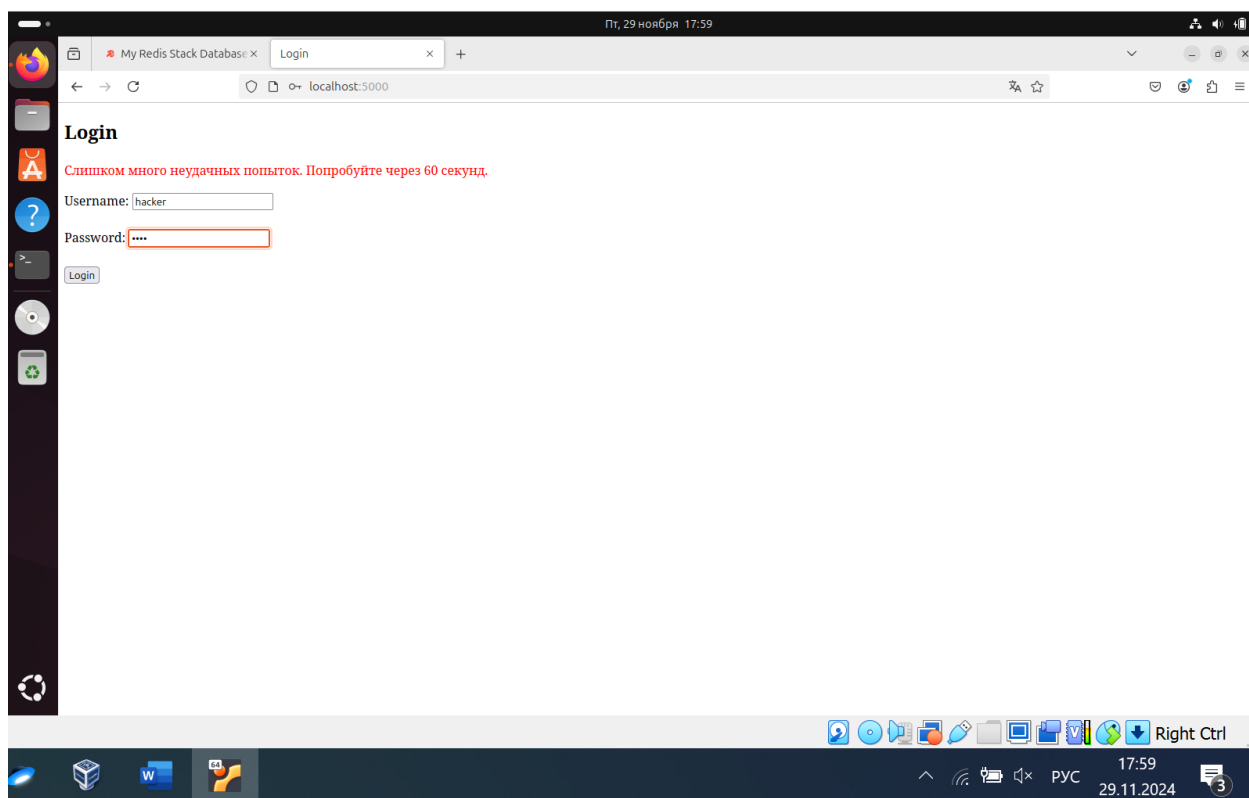


```
zvs05@zubarev: ~/docker/redis
WARN[0000] /home/zvs05/docker/redis/docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Building 1.0s (10/10) FINISHED
=> [web internal] load build definition from Dockerfile
=> [web internal] load metadata for docker.io/library/python:3.9-slim
=> [web internal] load .dockerignore
=> [web internal] load build context
=> [web internal] transferring context: 28
=> [web 1/4] FROM docker.io/library/python:3.9-slim@sha256:6250eb7983c08b3cf5a7db9309f8630d3ca03dd152158fa37a3f8daaf397085d
=> [web internal] load build context
=> [web internal] transferring context: 175.67kB
=> CACHED [web 2/4] WORKDIR /app
=> CACHED [web 3/4] COPY . /app
=> CACHED [web 4/4] RUN pip install --no-cache-dir -r requirements.txt
=> [web] exporting to image
=> [web] writing image sha256:64f489c1aef6d942aa5163a582b1439c8f3dfa377a0cb94f71ee24c349dfd781
=> [web] naming to docker.io/library/redis-web
=> [web] resolving provenance for metadata file
[+] Running 3/3
✔ Container redis_zvs Started
✔ Container insight_zvs Started
✔ Container web_auth Started
zvs05@zubarev: ~/docker/redis$ sudo docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
122bffa3419b   redis-web                           "python app.py"         10 minutes ago Up 12 seconds 0.0.0.0:5000->5000/tcp, :::5000->5000/tcp
0a79a95b4ef5   redislabs/redisinsight:latest       "./docker-entrypoint.sh n... 10 minutes ago Up 12 seconds 5000/tcp, 0.0.0.0:6005->8001/tcp, [::]:6005->8001/tcp
cbf11fd4c879   redis/redis-stack:latest            "/entrypoint.sh"        10 minutes ago Up 12 seconds 0.0.0.0:6305->6379/tcp, [::]:6305->6379/tcp
```

Протестируйте работу сервиса. Проанализируйте данные в Redis с помощью RedisInsight для трех случаев:

- неудачная попытка авторизации;
- превышено количество попыток авторизации;





— успешная авторизация.

My Redis Stack Database x Login x +

localhost:60005/redis-stack/browser

Databases > My Redis Stack Database db0

Back

LIST successful_logins

Key Size: 104 B Length: 1 TTL: No limit Last refresh: now

Unicode Add Elements

| Index | Element |
|-------|----------------------------|
| 0 | 2024-11-29 15:39:23:zvs_05 |

Insights

Tutorials Tips

Redis tutorials

- Redis basic and RAG use cases
- Data structures explained
- How to query your data

Introduction

Try real-time searching and perform complex structured queries without compromising the database performance.

Exact match

Full-text search

Range queries

Geospatial queries

Combined queries

Analytic and transformative queries

Learn more

Vector search explained

My tutorials

Create your tutorial Upload tutorial

Let us know what you think

Right Ctrl

18:46 29.11.2024 3

Пт, 29 ноября 18:00

My Redis Stack Database x localhost:5000/ x +

localhost:5000

Успешный вход!

Right Ctrl

18:00 29.11.2024 3