



**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем**

**Лабораторна робота №2
з дисципліни “Бази даних. Частина 2”
тема “Практика використання сервера Redis”**

Варіант 1

**Виконав
студент III курсу
групи КП-81
Янковський Дмитро
Олексійович**

Посилання на репозиторій: <https://github.com/Dizzzmas/db-labs-s2>

Мета

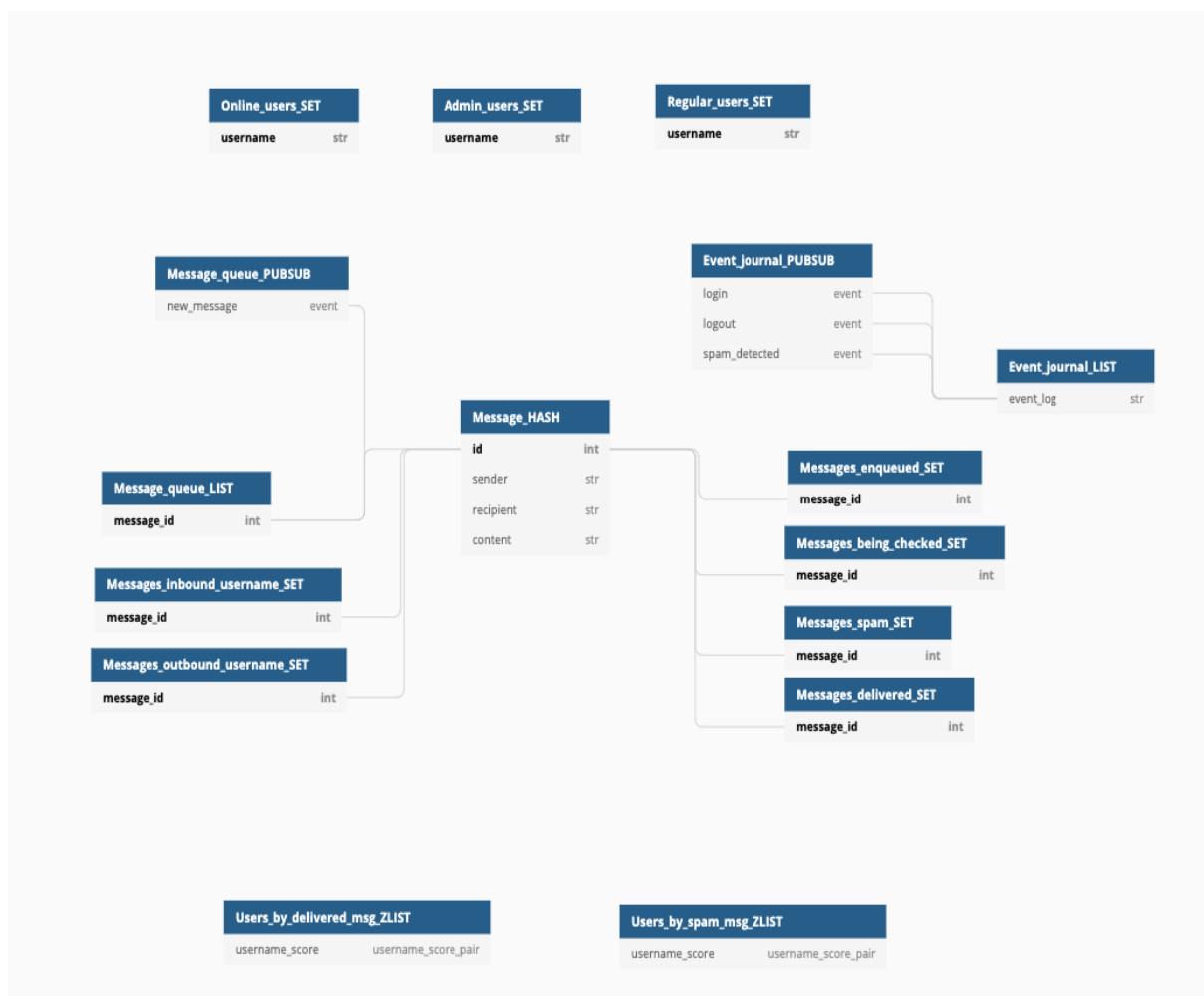
Метою роботи є здобуття практичних навичок створення ефективних програм, орієнтованих на використання сервера Redis за допомогою мови Python.

Постановка завдання

Реалізувати можливості обміну повідомленнями між користувачами у оффлайн та онлайн режимах із можливістю фільтрації спам-повідомлень.

Структура Redis

<https://dbdiagram.io/d/6040b727fcdcb6230b228eda>



SET - Використовується для зберігання унікальних, не впорядкованих даних (імена користувачів). Використовується для зберігання статусу повідомлень для простішого знаходження повідомлень за їх статусом.

HASH - Використовується для зберігання пар ключ-значення, де значення зазвичай об'єкт, що моделює якусь структуру даних. Використовується для зберігання даних про повідомлення до доступу до них за id-ключем.

LIST - Двустороння черга, що дозволяє впорядкований доступ до даних. Використовується для черги повідомлень, де нові додаються у кінець, а worker обробляє їх, беручи з початку черги. Також використовується для збереження подій доданих до журналу у хронологічному порядку.

ZLIST - Дозволяє ранжувати записи за нумеричним ключем.

Використовується для зберігання пар

“користувач->кількість_надісланих_повідомлень” та

“користувач->кількість_спам_повідомлень”. Це дозволяє легко знайти найактивніших та найшкідливіших користувачів.

PUB/SUB - Дозволяє створити канал повідомлень куди можна надсилати та з якого можна слухати повідомлення. Використовується для журналювання подій логін/логаут/знайдення_спам_повідомлення. Також використовується як триггер для сповіщення worker-а про нові повідомлення у черзі та початку їх опрацювання.

Запуск функціоналу

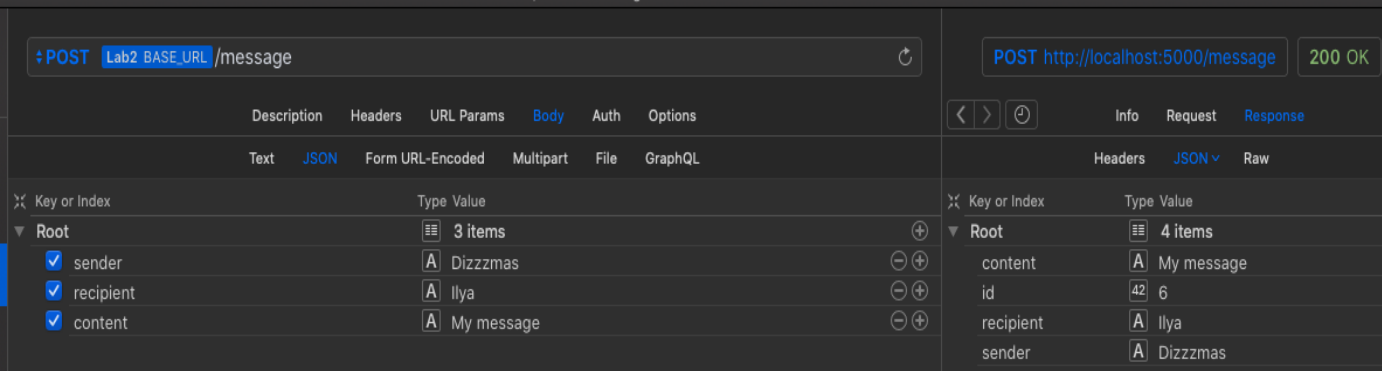
Замість консольного інтерфейсу було створено API з аналогічним функціоналом.

Доступні наступні запити:

- **POST /login** - Авторизувати користувача за ім'ям. Позначити його як online у Redis
- **POST /logout** - Позначити користувача як offline
- **POST /message** - Надіслати повідомлення користувачу recipient від sender з вмістом content.

- **GET** */message/inbound* - Дістати вхідні повідомлення для користувача з username
- **GET** */user-stats* - Дістати кількість повідомлень для username, що були доставлені/в черзі/перевіряються/у спамі
- **GET** */spammer-stats* - Дістати список користувачів ранжований за найбільшою кількістю спам повідомлень що вони надіслали
- **GET** */chatter-stats* - Дістати список користувачів ранжований за найбільшою кількістю успішно надісланих повідомлень
- **GET** */online-users* - Дістати список користувачів, що на даний момент онлайн
- **GET** */event-journal* - Дістати список подій у хронологічному порядку

Скріншоти роботи програми



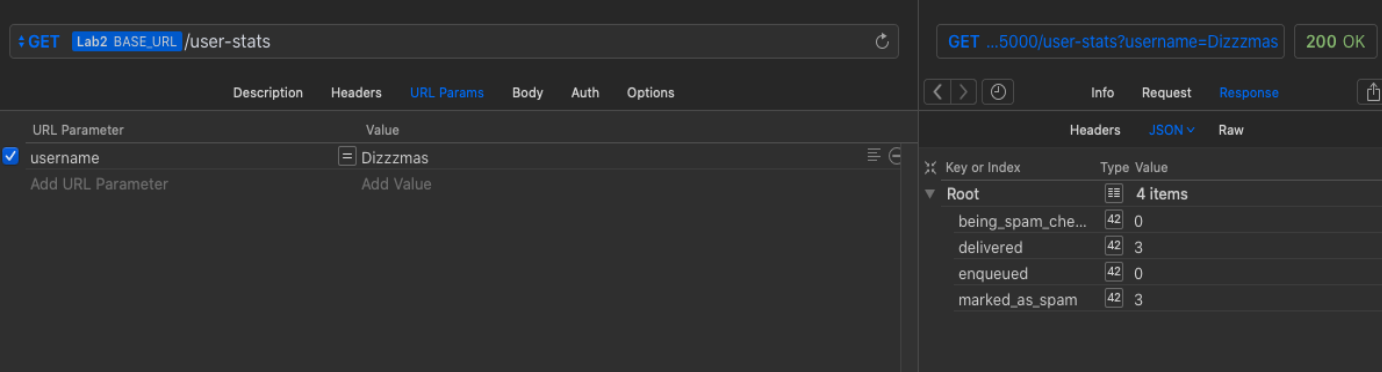
POST Lab2_BASE_URL /message

POST http://localhost:5000/message 200 OK

Key or Index	Type	Value
Root	3 items	
sender	A	Dizzzmas
recipient	A	Ilya
content	A	My message

Key or Index	Type	Value
Root	4 items	
content	A	My message
id	42	6
recipient	A	Ilya
sender	A	Dizzzmas

Приклад відправки повідомлення



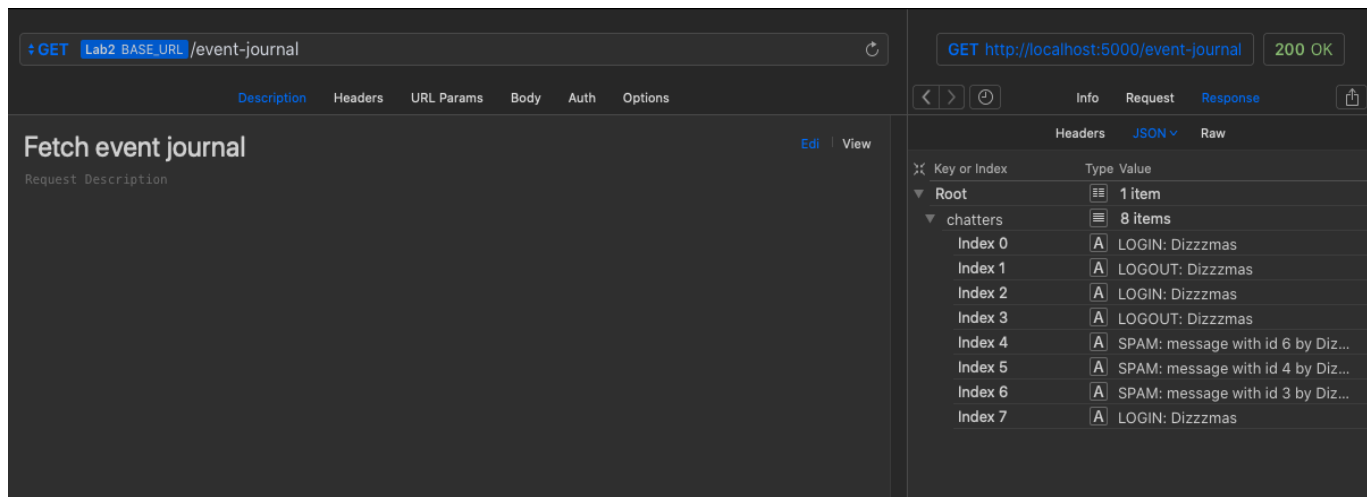
GET Lab2_BASE_URL /user-stats

GET ...5000/user-stats?username=Dizzzmas 200 OK

URL Parameter	Value
username	Dizzzmas

Key or Index	Type	Value
Root	4 items	
being_spam_che...	42	0
delivered	42	3
enqueued	42	0
marked_as_spam	42	3

Приклад отримання статистики повідомлень



Приклад перегляду журналу подій

Висновок

У ході лабораторної роботи було здобуто навички створення ефективних програм, орієнтованих на використання сервера Redis за допомогою мови Python. Отримано досвід використання основних структур та команд Redis.