


1. نحوه راه اندازی ردیس و اتصال آن به کلاینت:

اتصال سرور:

ابتدا مطابق شکل زیر فایل redis-server.exe را اجرا میکنیم تا سرور ما روی لوکال هاست بالا بیاید سپس فایل redis-cli.exe را اجرا میکنیم تا بتوانیم وضعیت کلاینت خود را با دستوراتی مدیریت کنیم.

 redis-check-aof.pdb

 redis-cli.exe

 redis-cli.pdb

 redis-server.exe



صفحه مربوط به redis-cli.exe :

```
C:\Users\Shahin\Desktop\داده\پایگاه\ترم دانشگاه\ترم دانشگاه\Redis\redis-cli.exe
127.0.0.1:6379>
```

صفحه مربوط به redis-server.exe :

```
C:\Users\Shahin\Desktop\داده\پایگاه\ترم دانشگاه\ترم دانشگاه\Redis\redis-server.exe
[21048] 13 Mar 11:04:34.546 # Warning: no config file specified, using the default config. I
file use C:\Users\Shahin\Desktop\??? ??? ???????\??? ??? ???????\??? ???\?????? \????\Redis\
is.conf

Redis 3.2.100 (00000000/0) 64 bit

Running in standalone mode
Port: 6379
PID: 21048

http://redis.io

[21048] 13 Mar 11:04:34.566 # Server started, Redis version 3.2.100
[21048] 13 Mar 11:04:34.570 * DB loaded from disk: 0.003 seconds
[21048] 13 Mar 11:04:34.570 * The server is now ready to accept connections on port 6379
```

اتصال کلاینت:

سپس نوبت به کلاینت پایتونی از طریق بک اند پایتون میرسد که کانکشن خود را آنجا ایجاد کنیم و با یک سری API دیتابیس خود را مدیریت کنیم. تکه کد زیر اتصال کلاینت را نشان میدهد. (روی پورت 6379)

```
from Cli.cli import CLI
#from Cli.actions.redis_manager import RedisManager
import redis

def main():
    CLI().execute()
    r = r = redis.Redis(host='localhost', port=6379, decode_responses=True)
```

2. توضیح پیاده سازی هر یک از متد ها:

متد CREAT :

پس از اتصال با دیتابیس ، در این متد ، name , description , encrypted_key , expiration_time در قالب فایل json و با set کردن زمان انقضای موجودیت و با یک key که name است به صورت بایتری در دیتابیس ذخیره میشوند(در اینجا value ما همان فایل json است).

با دستورات :

```
serialized_password = json.dumps(password_object)
print(f"Serialized password: {serialized_password}") # Debug statement
self.r.setex(name, exp, serialized_password)
```

متد READ :

در این متد با صدا زدن متد get با دادن به name به عنوان key و تبدیل کردن فایل بایتری json به فایل متنی مشخصات دیکشنری ایجاد شده متشکل از فیلدهای name , description , encrypted_key , expiration_time را در کنسول چاپ میکنیم.(کد این متد در فایل code موجود است)

متد UPDATE:

در این متد با صدا زدن متد `get` با دادن `name` به عنوان `key` و تبدیل کردن فایل باینری `json` به فایل متنی می آییم و یک رمز جدید وارد کرده و پس صدا زدن متد :

```
AESCipher().encrypt(new key)
```

آن را رمزنگاری کرده و مقدار جدید را به فیلد `encrypted_key` در فایل `json` می‌دهیم و سپس دوباره در دیتابیس ذخیره می‌کنیم.

متد DELETE :

در این متد با صدا زدن متد `get` و با دادن `redis_key=name` به عنوان `key` و سپس اجرای خط کد:

```
self.r.delete(redis_key)
```

کل موجودیت آن `json` را حذف می‌کنیم.

متد LIST :

در این با صدا زدن متد:

```
all_keys = self.r.keys()
```

کل کلید های دیتابیس را لود کرده و سپس در یک آرایه همه را پیمایش کرده و اطلاعات مربوط به کل رکورد هارا نمایش میدهیم.

متد GET_REMAINING_TIME :

در این متد با صدا زدن متد get با دادن به name به عنوان key و صدا زدن متد:

```
remaining_time = self.r.ttl(name)
```

زمان باقی مانده تا انقضا را بدست می اوریم.

اینجا r همان شی دیتابیس redis ماست .


3. اسکرین شات از اجرای برنامه و دستورهای پیاده سازی شده:

(پوزش بابت کوچک بودن متن اسکرین شات ها)

متد CREATE : کاربری به اسم kosar ساختم.

```
PS C:\Users\Shahin\Desktop\db-project1\codes> python main.py create -n kosar -c portal password -key 123465 -exp 200
Serialized password: {"name": "kosar", "description": "portal password", "encrypted_key": "tiwKvz0WZgFgsd0IdD5suyiXFcU816S7eiloM/ypVvA=", "expiration_time": "200"}
your Password is encrypted to timKvz0WZgFgsd0IdD5suyiXFcU816S7eiloM/ypVvA=
```

تصویر زیر با دستور EXISTS KEY اگر 1 برگرداند یعنی در دیتابیس وجود دارد.

 C:\Users\Shahin\Desktop\داده\ترم دانشگاه\ترم ششم\پایگاه داده\Redis\redis-cli.exe

```
127.0.0.1:6379> EXISTS kosar
(integer) 1
127.0.0.1:6379>
```

متد READ : کاربری به اسم ahmad که قبلا درست کرده بودیم خواندیم.

```
PS C:\Users\Shahin\Desktop\db-project1\codes> python main.py read ahmad
{"name": "ahmad", "description": "portal password", "encrypted_key": "JKGwwMC0WagrS3FizpYBKwwGNPk9xZwrGw4giJNvPGc=", "expiration_time": "200"}
Name: ahmad
Description: portal password
Encryption password: JKGwwMC0WagrS3FizpYBKwwGNPk9xZwrGw4giJNvPGc=
Expiration Time (seconds): 200
```

تصویر زیر log کلاینت را نشان میدهد: (با دستور MONITOR)

```
crypted_key\": \"JKGwwMC0WagrS3FizpYBKwwGNPk9xZwrGw4giJNvPGc=\", \"expiration_time\": \"200\"}"
1710356071.252030 [0 [::]:57021] "GET" "ahmad"
```

متد UPDATE :

```
PS C:\Users\Shahin\Desktop\db-project1\codes> python main.py update ahmad -key 123445567
your new encryption value is : c8Y6PherJv3y8T8zu5h0RDs+WA8hmpIwN0XjrW/VQeo=
```

متد DELETE :

```
PS C:\Users\Shahin\Desktop\db-project1\codes> python main.py delete ali
Password deleted successfully
```

تصویر زیر میگوید علی دیگر در دیتابیس وجود ندارد:

```
127.0.0.1:6379> EXISTS ali
(integer) 0
```


متد LIST :

```
PS C:\Users\Shahin\Desktop\db-project1\codes> python main.py list
Name: ahmad
Description: portal password
Encryipted_password: uSV55Ng+Vddi0T7wZnJ4xL093a2Svn983Qlh1wAdy+Q=
Expiration time: 200

Name: kosar
Description: portal password
Encryipted_password: tiwKvz0WZgFgsdDIdD5suyiXFcU816S7eiloM/ypVvA=
Expiration time: 200

Description: portal password
Expiration time: 300
```

متد GET_REMAINING_TIME: (برای کاربر kosar)

```
Checking remaining time for key: kosar
63
```

پایان