Homework: 2-Task 2
Message-Board

Contents

Contributors:

Rajeswari Nikhila S: UIN 927001562
Akshay Gajanan Hasyagar: UIN 924002957

Time taken:

-3 Hrs to build the prototype
-4 Hrs to come up with system architecture
-2 Hr to write the report

Github link:

Task1: https://github.tamu.edu/akshay-gh/678-18-c/tree/master/HW2/Task1

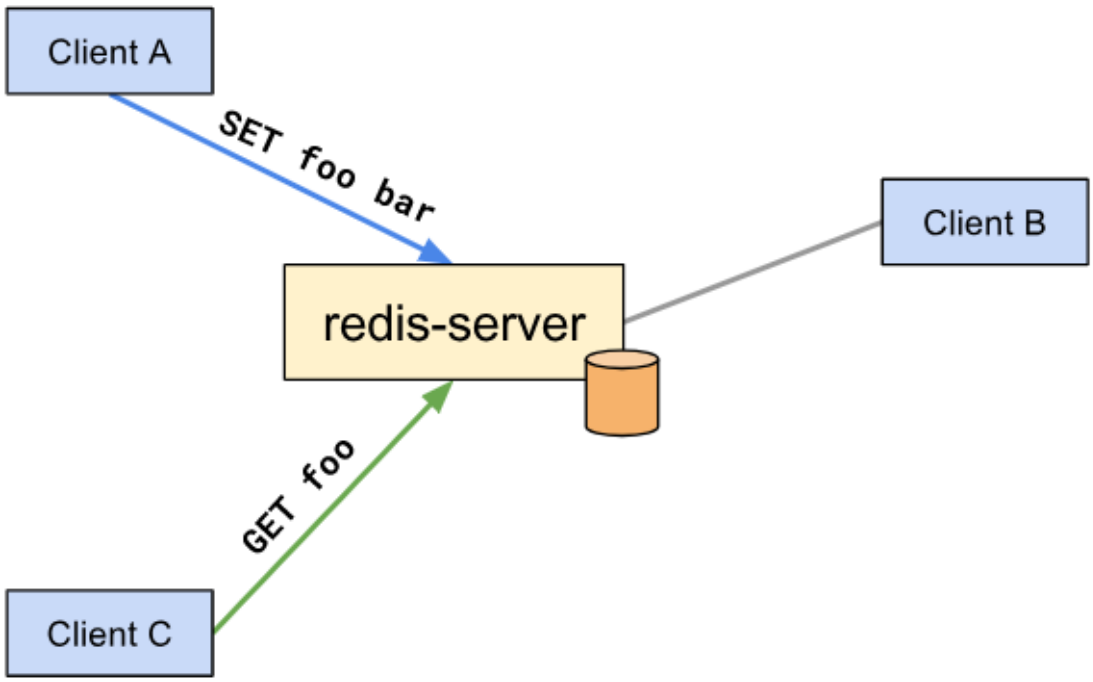
Task2: https://github.tamu.edu/akshay-gh/678-18-c/tree/master/HW2/Task2

Task-2 To Do:

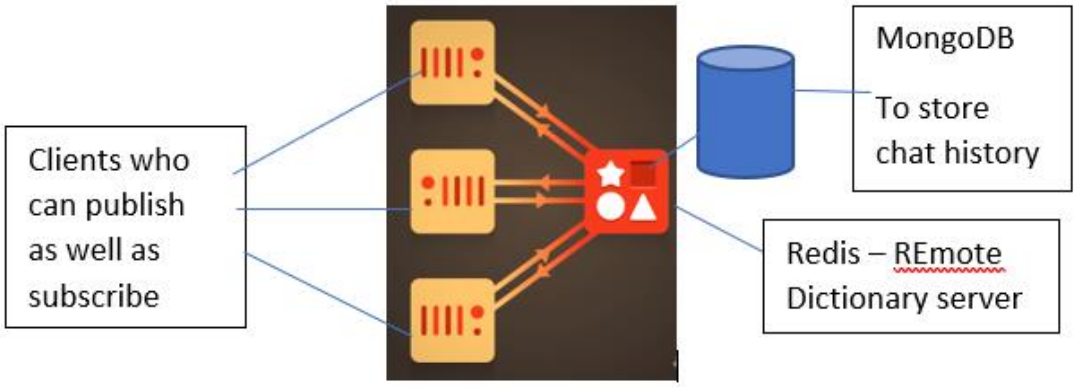
To build a prototype for message board where employees can interact and exchange information with each other.
The message boards are categorized by topics.
Employees can choose select a message board topic, listen to the conversations, write to the boards or read the entire chat history.

System Architecture:

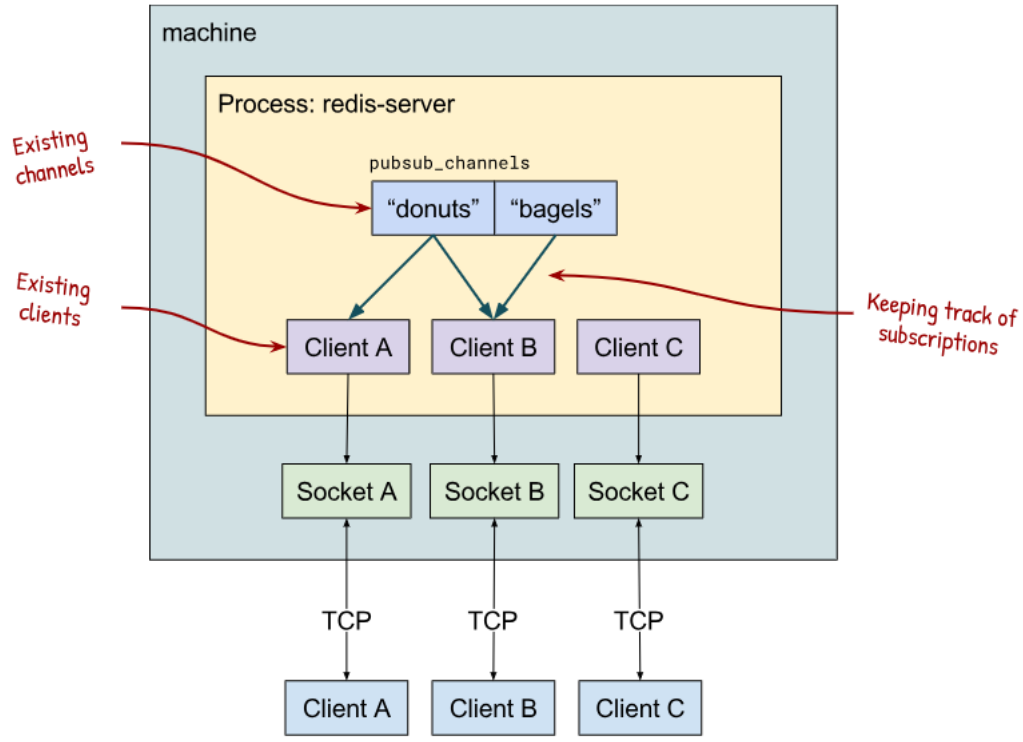
Our message boards are nothing but a real time chat application. Something similar to :



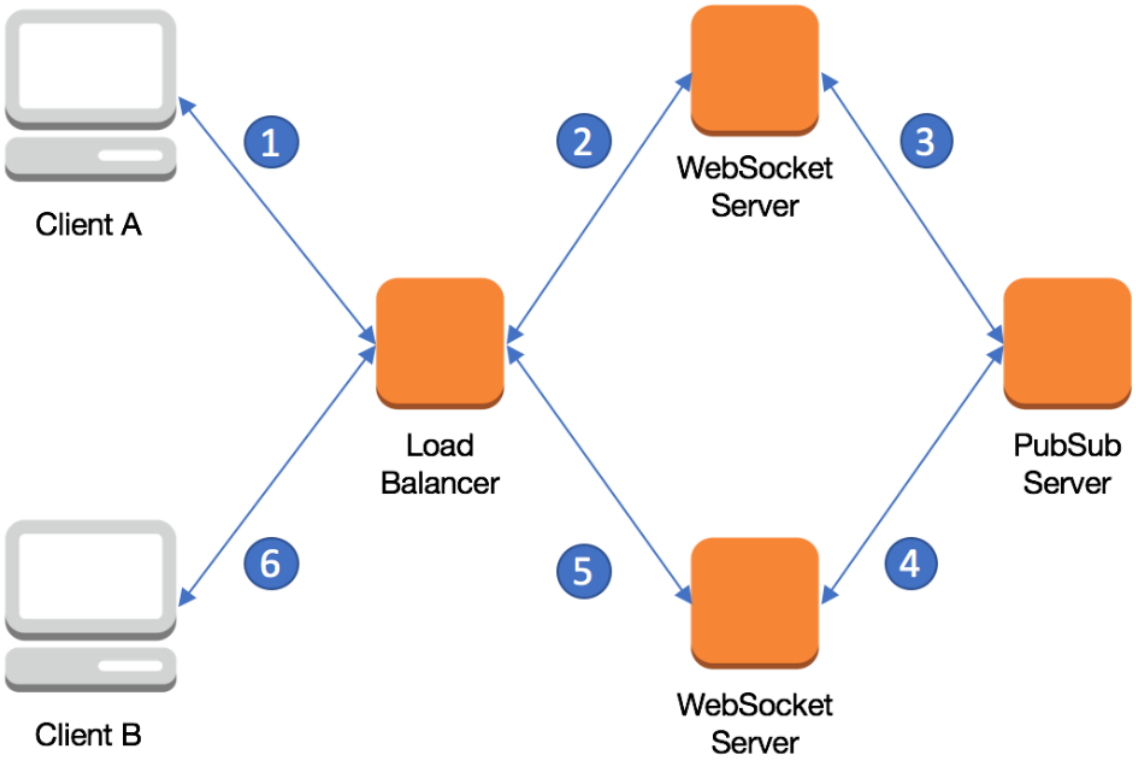
To build this application, we need a communication channel where a publisher can send messages in a chat room and these are distributed to all the subscribers of the chat room.
The publish-subscribe pattern available in redis is a way of passing messages to an arbitrary number of senders. The senders of these messages (publishers) do not explicitly identify the targeted recipients. Instead, the messages are sent out on a channel on which any number of recipients (subscribers) can be waiting for them.
The bidirectional communication facilitated by redis pubsub is represented below:



The Pub/Sub implementation lets clients send three new kinds of command: PUBLISH, SUBSCRIBE, and UNSUBSCRIBE. To track subscriptions, Redis uses a global variable pubsub_channels which maps channel names to sets of subscribed client objects. A client object represents a TCP-connected client by tracking that connection's file descriptor.



If this setup has to scale then a load balancer can be added with web-socket servers and load balancers :



Why MongoDB?

As mentioned earlier, Redis is an in-memory data store. Although it can persist data to disk, using Redis for that purpose probably is not the best way to go. We will use another DB to store uploaded messages and persist data. Also, it's a good idea of using another DB with high risk of losing data with using just Redis as its not highly scalable given that its in-memory data store.
So, our choice here was MongoDB.
MongoDB is a schema-less document-oriented database. It gives us the possibility to store complex data effortlessly and retrieve it straight away, without any additional query. We are using MongoDB to store any persistent data like chat history of the message boards etc.

Why Redis?

Redis, which means REmote DIctionary Server, is an open source, in-memory data structure store. It can be used as a key-value database, cache and message broker.

Pros of redis pubsub:
It is very fast since it makes use of in-memory data stores.
Slow subscribers can not delay publishers from broadcasting messages since it is not queue based.
The simplicity allows users to be flexible and easily scale applications

How code works

The code is built upon the sample code given:

- Connection is made to redis and mongodb server running locally
- In loop keep accepting for input from the user
- Input can be one of these: select <board>, read, write <message>, listen, stop or <Ctrl-C>
- If select then program checks for following words which would correspond to message board name, this is stored in "`mes_board`"
- The user can then select read, write or listen once a message board is selected
- For write, the user must enter the message after the write keyword
- This would be sent out to all the clients listening to this message board
- For listen, select message board and then type "listen"
- One listening the message is automatically delivered if any is written from other clients of the message board.
- To exit listening state use <Ctrl-C>
- When user writes a message it's sent to redis pubsub as well as stored in MongoDB.
- When read is used then the chat history is obtained from MongoDB
- To exit program type "stop"

How to run the code?

Tested on Ubuntu 16.04, python 3.6

Please install the latest version Redis and Mongo DB

Check if Mongo is running on your localhost on port 27017

Check if Redis is running on your localhost on 6379

Install modules of both redis and mongodb :sudo pip install redis, python -m pip install pymongo

To run the code please type the following on command line when inside the Task2 folder :
python message_boards.py

References

https://making.pusher.com/redis-pubsub-under-the-hood/

https://aws.amazon.com/blogs/database/how-to-build-a-chat-application-with-amazon-elasticache-for-redis/