FAQ

1. I’m using ubuntu to run the system. Do I need Docker Desktop or can I instead just install Docker Engine and do everything fine without any gui support?

Docker engine install on linux should automatically get the docker daemon running in the background. Docker desktop, on the other hand, runs the daemon inside a virtual machine. As I mentioned earlier, it is intended to get linux environment on windows or mac. So if you are already running on linux, you don’t need it.

2. In `shardmaster` we supposed to implement the Join, Leave, Move and Query.
   But in the Join method, the `JoinRequest *request` parameter has only the server but not the shards. How can we access the shards in assignment 1?

Indeed, you cannot (and you don’t have to) send the shards in a JoinRequest request.
The division in shards happens locally to the shardmaster. Basically, the shardmaster has a range of keys (for example 1000) and splits this range based on the number of server joining. So, when it receives the address of the first server, it assigns the full range [0,1000] to that server.
When a second server joins, the shardmaster splits the range into 2 parts and assigns [0,500] to the first and [501,1000] to the second (as described in the README file).
The only time the shards are delivered is when the shardmaster receives a Query RPC.

3. I’m testing the operations in `shardmaster`. But the command:
   `Shard {0, 500} on server first_server_address:1234` does not work in the client. When I’m testing I have two terminals with the client running and the shardmaster.

Terminal 1: runs shardmaster on port 13100

```
./shardmaster 13100
```

Terminal 2: which connects to the shardmaster and then joins two shardmanagers:

```
./client `hostname` 13100
join first_server_address:1234
join second_server_address:8271
query
Shard {0, 500} on server first_server_address:1234
Shard {501, 1000} on server second_server_address:8271
```

`Shard {0, 500} on server first_server_address:1234` is not a command you type in the terminal but the answer to the `query` command. In other words,
when you type in the client terminal `query`, you should get `Shard {0, 500} on server first_server_address:1234`.

4. Should I leave my project open to everyone or private as it is now?

You MUST keep your project private.

5. I'm sending a 'put' RPC for a post (containing post-id and user-id in different kay ranges). Which server is responsible for them?

Let's assume we have two servers, server-1 responsible for key range [0,500] and server-2 for the key range [501, 1000]. I send a 'put' RPC for a post with post_id=600, user_id=2 and post_text="Hello World".
The post should go to server-2 (as post_id=600 is in the range [500,1000]) while server-1 should be responsible for storing the key 'user_2_posts').

6. How could I managed the append case for a post that still doesn't exist? In particular the method ShardkvServer::Append If someone put in the request: "post_200 " "HelloWorld". If the post exists, I just have to change it, but if it doesn't exist to which user I should assign it? Should I still save it?

You can assume that the append is never used to insert a NEW post.

7. I'm dealing with a probable deadlock when running the test.sh script. In particular, it gets stuck in the server_moves test. The problem is that with manual tests it works without problems. I tried some debugging tools but I cannot look at what's happening. Is there a better way to know where the possible deadlock is? I used mutex shared by all threads to lock the common resources in the function and I checked that they get free after the different threads completed the jobs. Furthermore, sometimes a server tries to communicate with other servers, in order to limit the possible trials, I created a maximum number of calls (MAX_TRIALS=1000) that a server can perform. But even with this, it seems that the program stays stuck. Is there any solution?

One reason why it works with manual tests while it doesn't with automatic test can be that the time between each operation is larger in the manual test.
There is not a simple way to debug a deadlock. You can only try running the failing test alone with gdb.

Notice that a deadlock can happen NOT ONLY between threads in the same process (for example inside the same shardkv server) but can also happen between two servers. For example, it can be that a shardkv server owns a lock while it performs a RPC for migrating the keys.

Setting a MAX_TRIALS=1000 works if you get an answer positive or negative for your request from the other server. However, if a server is waiting for any answer from another server, the MAX_TRIALS doesn't avoid the deadlock.