Lab3 System Description

- Chuuhsiang hung, Chuanpu Luo

1 Overall Program Design

The system structure of this lab is based on lab2 where it includes 4 main parts: Client, FrontEndService, CatelogService and OrderService. They are all created by run.sh script and we use docker-compose to manage these services (described in detail in How To Run section). However, in this lab we have 2 catalog servers and 2 order servers.

Client class reads HTTP commands (for example, search/distributed systems) from Command List File, sends HTTP requests to FrontEndService and writes responds into Log File.

FrontEndService receives requests from Client, load balances HTTP requests to CatelogServices and OrderServices using round robin. Also, we implement in-memory cache for LOOKUP so the requests will not be forwarded to the server if the requested item is in the cache. If the server is not responding, the request will be redirected to its replica. The item in the cache will be invalidated later once a BUY request is fulfilled.

CatelogService reads and writes book data (for example, book name, book cost and etc) from and into Book Data File. **OrderService** records all BUY operation into its Log File. To ensure the data consistency, both **CatelogService** and **OrderlogService** sends HTTP requests to their replicas to update the database files. Also, for any update to the database files, the system will first send a DELETE request to the **FrontEndService** to invalidate the item from cache.

We choose consistency over availability which implies that the user will NOT always get the latest number of books but the old one if the replica is not synchronized yet. One thing needs to notice is that we didn't implement RAFT in this lab so the system may run into race condition in the long run as the number of BUY requests increase. FrontEndService, CatelogService and OrderService only provides HTTP service, calls are further handled by FrontEndServer, CatelogServer and OrderServer.

Last but not least, we dockerized these 4 components into images and containers so the users can run them without worrying about the environment.

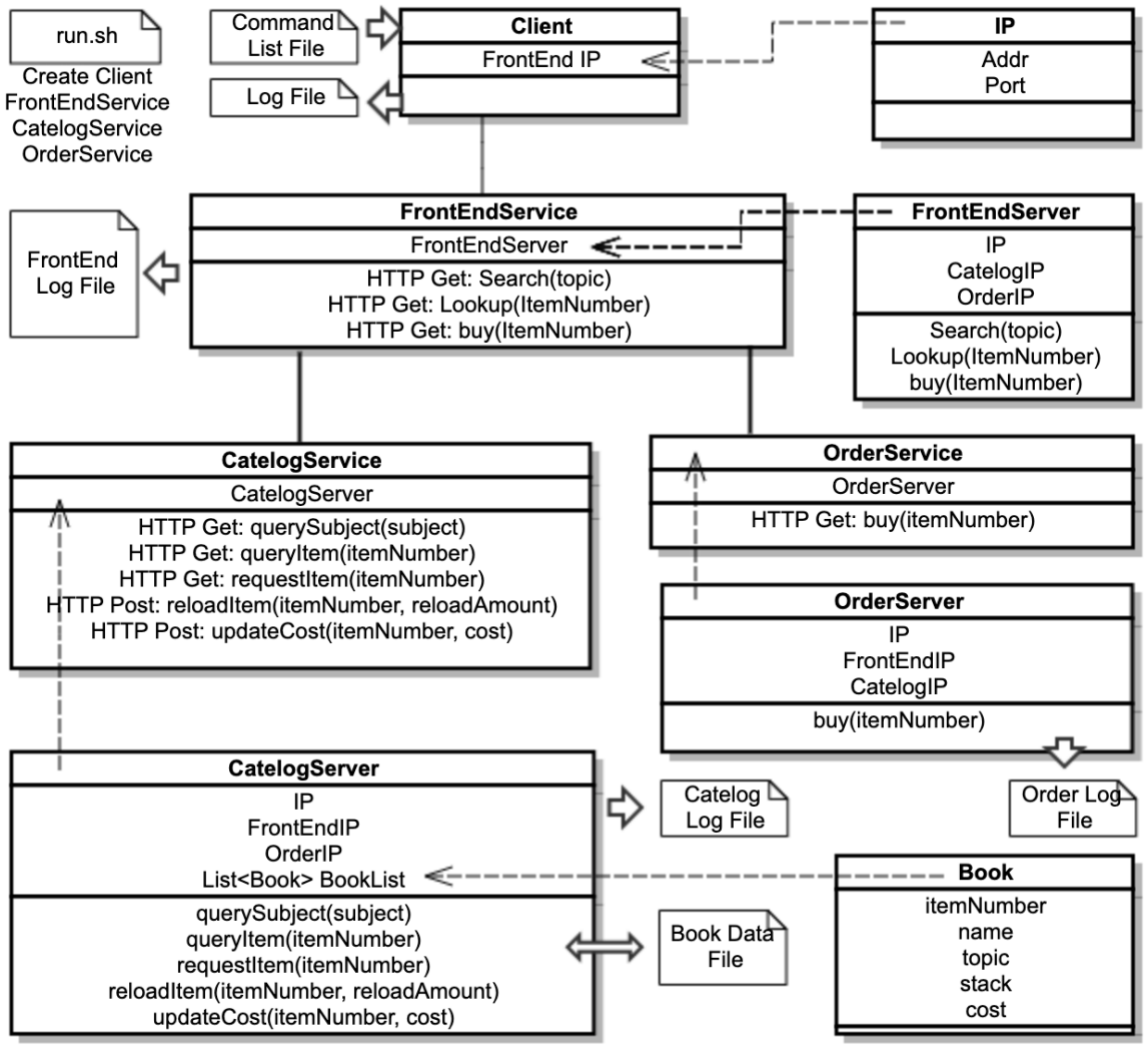

Figure 1. Program Structure of System

2 How it works

a. Rest API Provider

We used Java and Spark to provide HTTP service. The reason is that both of us are more familiar with Java and we are able to reuse some code from Lab1 (for example, IP.java). Spark provides a HTTP framework that is very easy to use. We define all HTTP service route in FrontEndService, CatelogService and OrderService, using both Get and Post methods. After these Service class received a HTTP request, it will call functions in FrontEndServer, CatelogServer and OrderServer to handle it.

b. Book data storage

We used a simple CSV file to store all book data (for example, book name, book itemNumber). In order to make sure the correctness of data under concurrency, after every data operation, we write data update into File. Next time when we need book data, we read it from File again.

3 Design Tradeoff

We choose consistency over availability which implies that the user will NOT always get the latest number of books but the old one if the replica is not synchronized yet. For the containers, we only tested them on the local machine so we don't need to deal with the network issues as we don't have enough time.

4 Possible Improvement

In this System, we didn't implement RAFT in this lab so the system may run into race condition in the long run as the number of BUY requests increase. If time allows, it is better to implement RAFT to ensure the system's consistency and availability and containerize the components and allow them to communicate across machines.

5 How to Run

5.1 Local

Step 1: download project from GitHub

$ git clone https://github.com/ds-umass/lab-3-lab-3-hung-luo.git

Step 2: Enter root directory, make sure file pom.xml is under current directory.

$ cd lab-3-lab-3-hung-luo/

Step 3: Make sure IP address frontend/catalog/order service are correct

$ vim run.sh

```
3  # Define EDLab Multiple Computer Parameters
4  # Need to change before running for TA
5  EDLAB_FRONTEND_IP=128.119.243.164:5018
6  EDLAB_CATELOG_IP=128.119.243.175:5019
7  EDLAB_ORDER_IP=128.119.243.175:5039
```

Step 3: Execute run.sh to run service run.sh [local/edlab] [frontend/catalog/order/client]

$ bash ./run.sh edlab frontend
$ bash ./run.sh edlab catelog
$ bash ./run.sh edlab order
$ bash ./run.sh edlab client

Step 4: Check Output Log File

Input File	Command List File	"./tests/edlab_test_client_command_list.csv"
Output File	Book Data File	"./tests/edlab_test_book_data.csv"
	Client Log File	"./tests/edlab_test_client_log_file.csv"
	FrontEnd Log File	"./tests/edlab_test_frontend_log_file.csv"
	Catelog Log File	"./tests/edlab_test_catelog_log_file.csv"
	Order Log File	"./tests/edlab_test_order_log_file.csv"

5.2 Docker

Step 1: download project from GitHub

$ git clone https://github.com/ds-umass/lab-3-lab-3-hung-luo.git

Step 2: Enter root directory

$ cd lab-3-lab-3-hung-luo/

Step 3: Build images

$ docker-compose build

Step 4: Run containers for each service

$ docker-compose up

Step 5: Check Output Log File in each container

$ docker exec -it [container_name] bash

Input File	Command List File	"./tests/local_test_client_command_list.csv"
Output File	Book Data File	"./tests/local_test_book_data.csv"
	Client Log File	"./tests/local_test_client_log_file.csv"
	FrontEnd Log File	"./tests/local_test_frontend_log_file.csv"
	Catelog Log File	"./tests/local_test_catelog_log_file.csv"
	Order Log File	"./tests/local_test_order_log_file.csv"

Sample Output:

```
catalog    | Catelog Service Running
catalog    | query subject graduate school success
catalog    | query item 0 success
catalog    | query item 1 success
catalog    | query item 2 success
catalog    | invalidate item 0 success
catalog    | reload item 0 5 more, 1763 left
catalog    | invalidate item 1 success
catalog    | reload item 1 5 more, 1934 left
catalog    | request item 3 fail, 5 left
client     | [INFO] Scanning for projects...
client     | [INFO]
client     | [INFO] ----------------------< com.dslab3:com.dslab3 >-----------------------
client     | [INFO] Building com.dslab3 1.0
client     | [INFO] --------------------------------[ jar ]---------------------------------
client     | [INFO]
client     | [INFO] --- exec-maven-plugin:1.6.0:java (default-cli) @ com.dslab3 ---
client     | Client Created
client     | command:http://frontend:5000/search/distributed systems
```

```
catalog     | query subject distributed systems success
frontend    | search distributed systems success
client      | Run command:http://frontend:5000/search/distributed%20systems
client      | Result:"{\n  \"0\": \"How to get a good grade in 677 in 20 minutes a day\",\n  \"1\": \"RPCs for Dummies\"\n}"
client      | command:http://frontend:5000/search/graduate school
frontend    | search graduate school success
catalog     | query subject graduate school success
client      | Run command:http://frontend:5000/search/graduate%20school
client      | Result:"{\n  \"2\": \"Xen and the Art of Surviving Graduate School\",\n  \"3\": \"Cooking for the Impatient Graduate Student\",\n  \"4\": \"How to finish Project 3 on time\",\n  \"5\": \"Why theory classes are so hard\",\n  \"6\": \"Spring in the Pioneer Valley\"\n}"
client      | command:http://frontend:5000/lookup/0
frontend    | http://catalog:5100/queryItem/0
catalog     | query item 0 success
frontend    | lookUp catalog server item number 0 success
client      | Run command:http://frontend:5000/lookup/0
client      | Result:"\"{\\n  \\\"itemNumber\\\": 0,\\n  \\\"name\\\": \\\"How to get a good grade in 677 in 20 minutes a day\\\",\\n  \\\"topic\\\": \\\"distributed systems\\\",\\n  \\\"stack\\\": 1763,\\n  \\\"cost\\\": 12\\n}\""
client      | command:http://frontend:5000/lookup/0
frontend    | lookUp cache item number 0 success
client      | Run command:http://frontend:5000/lookup/0
client      | Result:"\"{\\n  \\\"itemNumber\\\": 0,\\n  \\\"name\\\": \\\"How to get a good grade in 677 in 20 minutes a day\\\",\\n  \\\"topic\\\": \\\"distributed systems\\\",\\n  \\\"stack\\\": 1763,\\n  \\\"cost\\\": 12\\n}\""
catalog     | invalidate item 0 success
catalog     | reload item 0 5 more, 1768 left
order       | reload item 0 5 more success
catalog     | invalidate item 1 fail
catalog     | reload item 1 5 more, 1939 left
order       | reload item 1 5 more success
client      | command:http://frontend:5000/lookup/1
frontend    | http://catalog:5100/queryItem/1
catalog     | query item 1 success
frontend    | lookUp catalog server item number 1 success
client      | Run command:http://frontend:5000/lookup/1
client      | Result:"\"{\\n  \\\"itemNumber\\\": 1,\\n  \\\"name\\\": \\\"RPCs for Dummies\\\",\\n  \\\"topic\\\": \\\"distributed systems\\\",\\n  \\\"stack\\\": 1939,\\n  \\\"cost\\\": 22\\n}\""
client      | command:http://frontend:5000/lookup/1
client      | Run command:http://frontend:5000/lookup/1
frontend    | lookUp cache item number 1 success
```

```
client      | Result:"\"{\\n  \\\"itemNumber\\\": 1,\\n  \\\"name\\\": \\\"RPCs for
Dummies\\\",\\n  \\\"topic\\\": \\\"distributed systems\\\",\\n  \\\"stack\\\": 1939,\\n
\\\"cost\\\": 22\\n}\""
client      | command:http://frontend:5000/lookup/4
client      | Run command:http://frontend:5000/lookup/4
client      | Result:"No item exists, please enter number between 0~3"
client      | command:http://frontend:5000/lookup/2
frontend    | lookUp cache item number 2 success
client      | Run command:http://frontend:5000/lookup/2
client      | Result:"\"{\\n  \\\"itemNumber\\\": 2,\\n  \\\"name\\\": \\\"Xen and the Art of
Surviving Graduate School\\\",\\n  \\\"topic\\\": \\\"graduate school\\\",\\n  \\\"stack\\\":
0,\\n  \\\"cost\\\": 33\\n}\""
client      | command:http://frontend:5000/lookup/2
frontend    | lookUp cache item number 2 success
client      | Run command:http://frontend:5000/lookup/2
client      | Result:"\"{\\n  \\\"itemNumber\\\": 2,\\n  \\\"name\\\": \\\"Xen and the Art of
Surviving Graduate School\\\",\\n  \\\"topic\\\": \\\"graduate school\\\",\\n  \\\"stack\\\":
0,\\n  \\\"cost\\\": 33\\n}\""
client      | command:http://frontend:5000/lookup/5
client      | Run command:http://frontend:5000/lookup/5
client      | Result:"No item exists, please enter number between 0~3"
client      | command:http://frontend:5000/lookup/2
frontend    | lookUp cache item number 2 success
client      | Run command:http://frontend:5000/lookup/2
client      | Result:"\"{\\n  \\\"itemNumber\\\": 2,\\n  \\\"name\\\": \\\"Xen and the Art of
Surviving Graduate School\\\",\\n  \\\"topic\\\": \\\"graduate school\\\",\\n  \\\"stack\\\":
0,\\n  \\\"cost\\\": 33\\n}\""
```