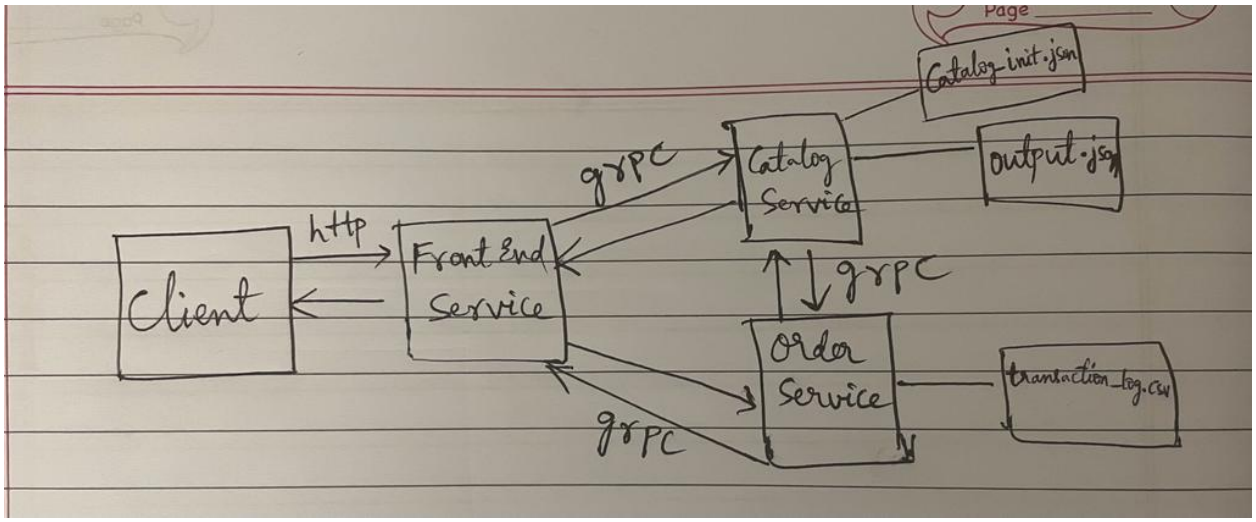
Design Document Lab 2

Architecture:



1. Catalog service uses catalog_init.json to initialize the trade volumes on every new start and output.json contains the final state of the catalog on server exit.
2. Transaction_log.csv maintains the history of transactions.

Front End Service-

The front end service uses simple http.server classes to set up and run the server. The frontend server uses the ThreadingHttpServer class. The request handler for front end service extends to BaseHTTPRequestHandler class. The requirement was to execute GET and POST requests made by the client and this was addressed by overriding the do_GET and do_POST methods of the BaseHTTPRequestHandler by our own customized code as excepted in the problem statement. When a GET request is made, the do_GET method is called up and inside it the run_lookup method is getting executed. The run_lookup call is called through a grpc channel of catalog_service and utitlizes that to generate the appropriate response. Similarly, when a POST request is being made by the client, the do_POST method gets called up and in it the run_order is getting executed. The run_order call is called a grpc channel of order_service and uses its methods to generate and send back the response. On the server code, the ThreadingHttpServer is customised by updating the protocol_versions to HTTP1.1 so as to have persistent connections and implement the thread per session behavior for our service.

Catalog Service:

Implements two GRPC interfaces:

1. Lookup

2. Trade

Lookup Method: Frontend calls this method on catalog service to make a stock lookup

Args: stock_name
Return: {success, stock_details}

Return values:
{-1, {}} if stock_name isn't present
{1, {name:"meta", quantity:"10", status:"0"}} if stock status is active.
{0, {}} if stock status is suspended.

Trade Method: Order Service calls this method on catalog service to execute an order it received from frontend.

Args: stock_name, trade_volume, type
Return: {success}

Success Values:
1 - Successful trade
0 - if trade volume < 0 & if stock is suspended to trade
-1 - if stock_name is invalid
-2 - if trade_volume > stock quantity

Order Service:
Implements one GRPC interface Order

Order Method: Take an order request from Frontend Service

Args:stock_name, trade_volume, type
Return: {success, transaction_id}

Return Values:
{1, 1} - transaction_id is legit only if success is 1
{0, -1} - transaction_id=-1 if order fails.