DESIGN SPECIFICATION

This project implements a simple book store through a client server model using REST APIs to communicate between layers.

There are two tiers in the web-app: a front-end and a back-end.
The front end accepts incoming requests from clients and accordingly processes them, by forwarding the calls to relevant servers.
The backend consists of two servers:
1. Catalog server: The catalog server maintains a catalog (which currently consists of four books) and the relevant information corresponding to those books.
Following is the schema for the catalog server.

```
{
        id: int (primary key),
        title: text
        cost: text
        count: text
        topic: text
}
```

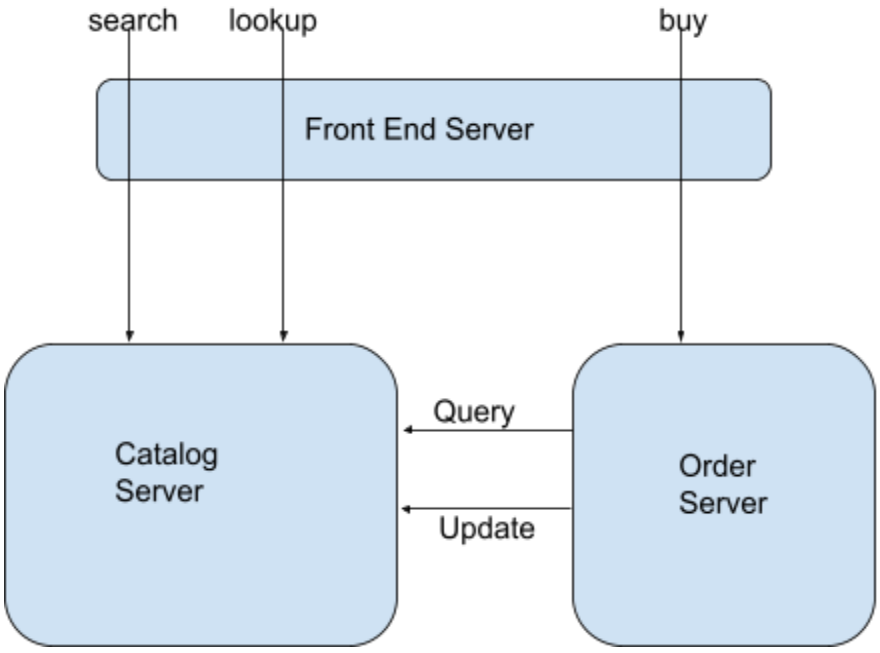
2. Order Server: The order server maintains a list of all orders received for the books.
Following is the schema for order server:

```
{
        id: int (primary key),
        item_id: int,
        Created: date
}
```

The front end server supports three operations:

1. search(topic) - which allows the user to specify a topic and returns all entries belonging to that category (a title and an item number are displayed for each match).
2. lookup(item_number) - which allows an item number to be specified and returns details such as number of items in stock and cost
3. buy(item_number) - which specifies an item number for purchase.

The way the three servers interact with each other is shown below.



For this milestone, we have implemented all three servers to be running on the same machine, but on different ports.

IMPLEMENTATION

Catalog server

The catalog server exposes a GET API (‘item’) which allows for books to be looked up from it’s database, either by id, or by topic. The catalog server also exposes a PUT API on ‘item’ which allows for the altering the items database in the catalog server. For example, This can be used when a book is purchased or if the cost of a book needs to be updated.

Order Server

The order server exposes a GET API(‘buy’) method which looks for a book corresponding to a certain ID and then buy it from the catalog server. When a client invokes the buy function on the front end server to buy a book by an ID number, the frontend server calls the endpoint on the order server with this ID number to buy a book. The order server first queries the catalog server to ensure that the book to be bought is available in stock, and then executes the buy operation and updates the database to reflect it.

Frontend Server

The front end server also allows the user to look for a book either by its ID or by its topic. The lookup function allows the user to look up the details of a particular book by it’s ID. A query is made to the catalog server and specific details of the book corresponding to the ID is returned.

Similarly, to search for all books belonging to a particular topic, the search function is implemented by the front end server which makes a call to the catalog server which returns a list of all the books belonging to that topic.

Client Process

A client process is responsible for making calls to the front_end server. This process takes in as argument a parameter based on which, it iteratively calls the lookup, search and buy methods of the front end server (the count for calling these is passed as a parameter to the function) and logs the results of these operations.