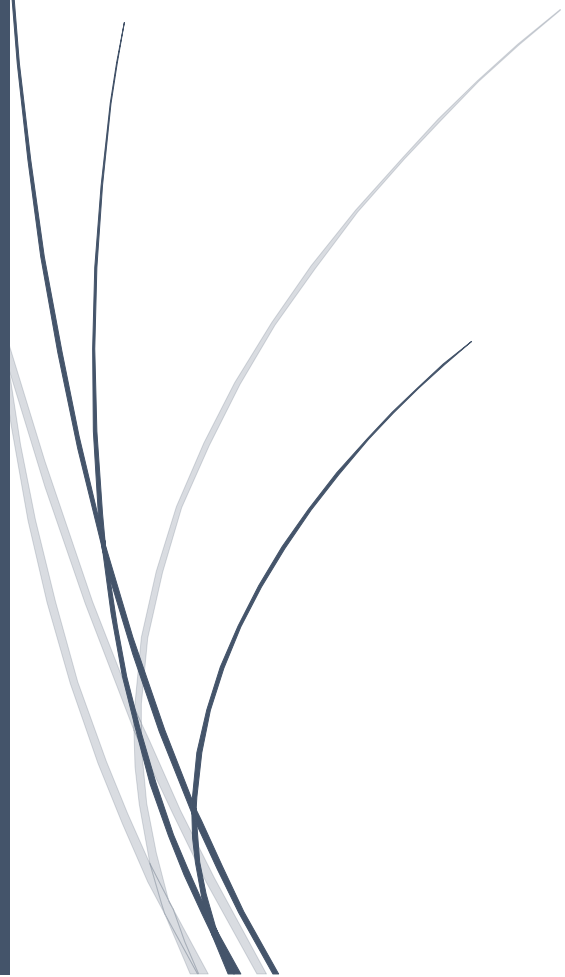
Distributed Operating System

Part 1

4/14/2022

Ameed Omar -11743942

Supervisor: Dr. Samer Arandi



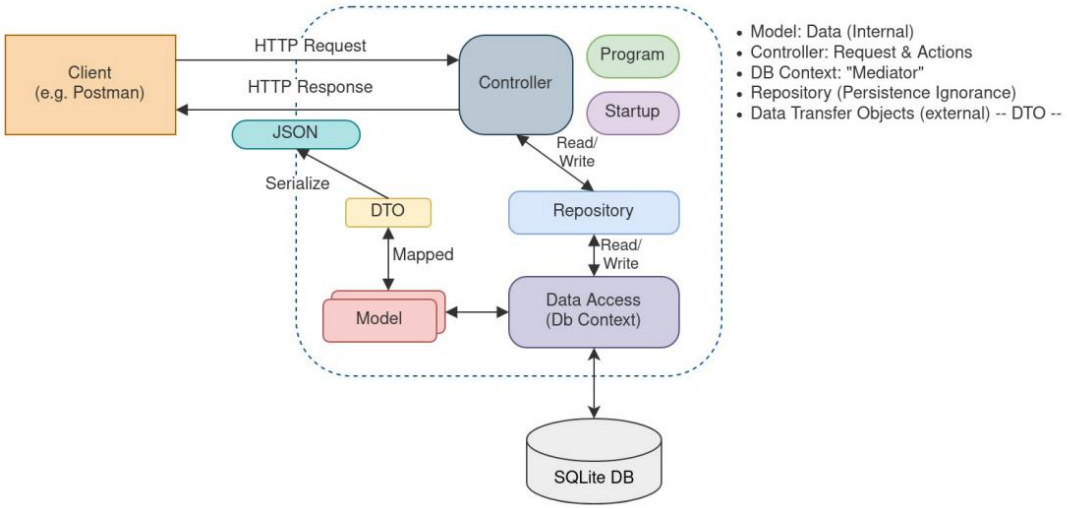
GitHup Link: https://github.com/ameedomar/BazarProject-DOS.git

System Description

First of all I'm going to describe the system , the system consist of two micro services the first one is CatalogServer , this server hold the responsibility of all action related to book (Search, Add and see info) and the second micro service is the OrderServer which hole the responsibility of handling the ordering request like (Purchase and getOrders).

Micro Service Architecture:

I build out system using .NET 5 using ModelViewController (MVC) approach (I implement it using Model and Controller without the View, I implement the View using Flutter) I can briefly describe the Architecture in this picture:



Explanation:

When the client send a request the controller in the server side get that request and handle it , it ask the Repository for the functional things because it contain the implementation and it has the responsibility of talking to the DB Context which represent the DataBase class which deal with the DB.

The Model contain the field that i have in my system , and the DTO contain all my part of these field and we use it to control what the controller can receive and what it can send back as a request response for the client which will be represented in JSON .

How to run the system:

I've implement the virtualization task using Docker so i define a docker image for each back end server and we've implement a automation file which is the docker compose file which can build each docker file to make a container for each server .

To run the docker-compose file you need to run this command using the terminal:

docker-compose up --build

For the front-end as I have mentioned i use flutter to do it and to run the UI service you need to run this command in the terminal:

flutter run

Possible improvement:

By add more functionality for each server like (addBook, deleteBook, showBookDemo)
addBook: give the website admin the ability of adding book to the system.

deleteBook: give the website admin the ability of deleteing book from the system.
ShowBookDemo: let the user to have a look on the book as a demo before he can order it.