CSCI/ECEN 5673: Distributed Systems
Spring 2023

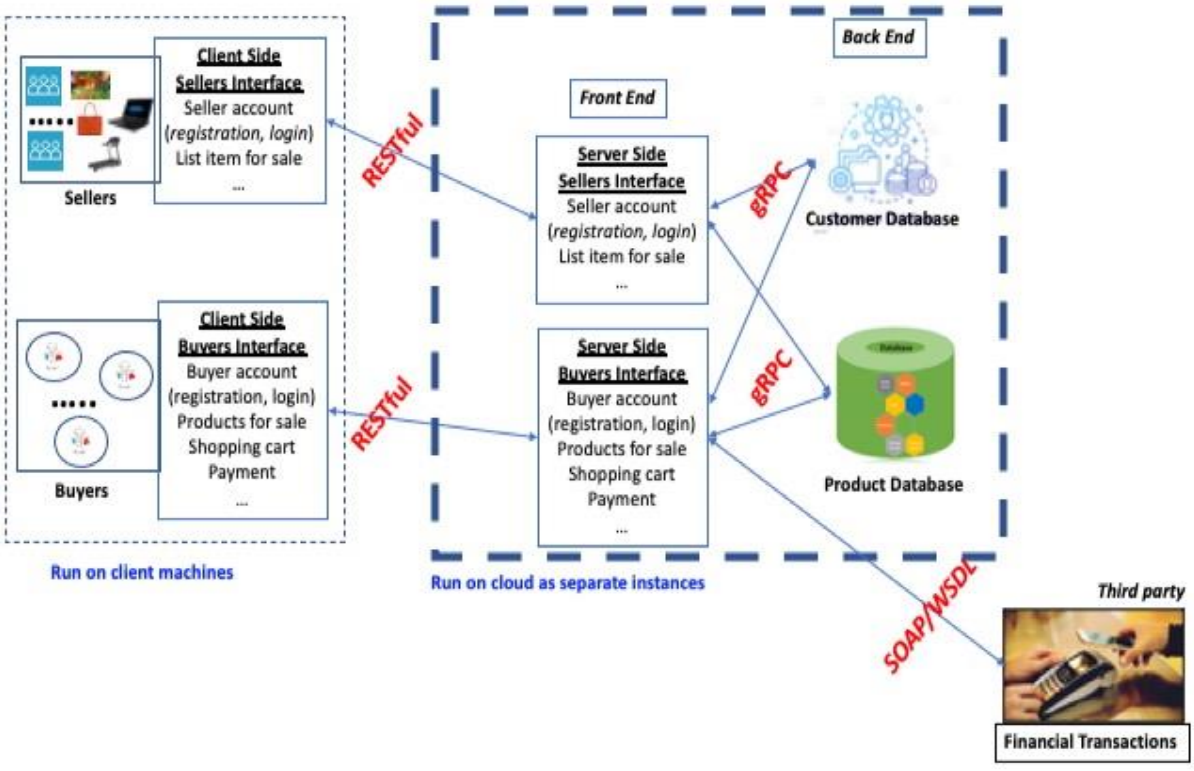
Programming Assignment Two

Due Date and Time: 11:59 PM Sunday, February 19, 2023

Goal: The goal of this programming assignment is to extend the online marketplace system you developed in Programming Assignment One as follows:

- Move all server components (both frontend and backend) to the cloud and run them in separate instances.
- Implement the remaining API function (Make Purchase)
- Make the client-server interactions RESTful (Client Side Seller interface–Server Side Seller interface and Client Side Buyer interface–Server Side Buyer interface)
- Implement all communication between front-end and customer/products databases using gRPC
- Implement a simple prototype of Financial Transactions (to be used as a third-party service) using SOAP/WSDL
- Note: Keep the front-end stateless in your design

You may work in teams of size two students



Characteristics of an item put up for sale (same as programming assignment one)

Seller characteristics (same as programming assignment one)

Buyer characteristics (same as programming assignment one)

CSCI/ECEN 5673: Distributed Systems
Spring 2023

APIs of the logical components (same as programming assignment one)

Client-side sellers interface (same as programming assignment one)

Server-side sellers interface
Same as Client-side Sellers interface

Client-side buyers interface (same as programming assignment one)
Implement the remaining API function:
Make purchase: *credit card details (name, number, expiration date)*

Server-side buyers interface
Same as Client-side buyers interface

Requirements of programming assignment two

Client - Server interactions

Make the client-server interactions RESTful. This includes Client Side Seller interface – Server Side Seller interface interactions and Client Side Buyer interface – Server Side Buyer interface interactions

Front-end – Backend interactions

Implement all communication between the front-end (Server Side Seller interface and Server Side Buyer interface) and backend (customer database and products database) using gRPC

Financial Transactions

Implement a very simple prototype of the Financial Transactions component. It receives a request (user name, credit card number) and returns Yes (90% probability) or No (10% probability). Use SOAP/WSDL to implement this service.

Moving all server components to cloud

Run all server components as separate instances in the cloud. Each student gets $50 Google Cloud Platform credit via the Google Cloud Platform Education Grants program. See the message dated January 30 on Canvas.

Stateless frontend

Design your frontend components to be stateless. Keep any state you need in the backend databases (customer and products databases).

Evaluation

Evaluation is very similar to the evaluation you did in programming assignment one.

Average response time: Response time of a client-side API function is the duration of time between the time when the client invokes the function and time when a response is received from the server. To measure average response time, measure the response time for ten different runs and then take the average.

CSCI/ECEN 5673: Distributed Systems
Spring 2023

Average server throughput: Throughput is the number of client operations completed at the server per second. To measure average throughput, measure the throughput for ten different runs and then take the average. Each run consists of each client invoking 1000 API functions.

Run one instance of each of the four server components as separate processes on possibly different hosts and possibly on cloud. Report the average response time and average throughput for the following scenarios:

Scenario 1: Run one instance of seller and one instance of buyer.
Scenario 2: Run ten instances of buyers and ten instances of sellers concurrently.
Scenario 3: Run 100 instances of buyers and 100 instances of sellers concurrently.

Provide explanations/insights for the performance you observe for each of the three scenarios.
Provide an explanation for the differences in the performances you observe among the three scenarios.
Finally provide a comparison between in performance between your programming assignment one and programming assignment two implementations. Provide an explanation for the differences in the performances you observe.

What to submit

Submit a single zipfile that contains all source code files, makefiles, a README file and a performance report file. In the README file, provide a brief description of your system design along with any assumptions (8-10 lines) and current state of your system (what works and what not). In the performance report file, report all performances measured along with your explanations.