BOGAZICI UNIVERSITY

Concurrent and Distributed Programming

by

Salih Sevgican

Instructor : Alper Sen

CMPE 436
Term Project Report

in the
Engineering Faculty
Computer Engineering

December 2017

Contents

Chapter 1

Abstract

1.1 Abstract

In today's daily life, every profession has some difficulties. Starting and developing a career is very important for individuals, especially if those individuals are fresh graduates. Choosing a starting point is more likely to be harder for those graduated from medicine faculties in Turkey. When doctors are graduated, it is obligatory for them to work for government facility (hospitals etc.) for two years at least. However there is an assignation process which is generally very complex and rough period because every hospital in different cities have limited capacity. Freshman doctors have to choose a few hospitals across Turkey to determine a starting destination. If a doctor cannot go to one of his/her choices among hospitals because of exceeding the capacity of those hospitals, s/he automatically assigned to a random hospital with free quota(which are generally eastern city hospitals). In order to prevent such complexity and unwanted assignment results among freshman doctors, a platform which gives information about all hospitals, their capacity and their registered users(doctors who are planning to select that hospital) has become a must. Inspired by a website [1] this android application has been developed to grant such platform to freshman doctors.

Chapter 2

Introduction

2.1 State-of-the-art

In Turkey(I don't know about other countries well), there is a person who voluntarily established a website [1], granting a solid communication platform for doctor assignation process in Turkey. Although he was doing this job free of charge, later on due to cost of servers, a small amount of fee is requested in order to join this communication network.

2.2 Motivation

An assignation process which affects thousands of life is a very important point to deal with. It is not only for doctors, this communication network can be implemented to any other professions which have similar processes or requirements. On the other hand, voluntarily joining this kind of movement would improve the self. I've contacted to the website owner before starting the project and learned there is no mobile application version of this platform so I've decided to start. At the end of this term project, I could send an example of this product.

Chapter 3

Approach

3.1 Application-Client

We want to implement an application which establishes a communication through socket based connection, and shows information about hospitals and their registry status. App should have a Login and Signup page for user registration and authorization. User should be able to see his/her registered hospitals. If s/he wants user can show all hospitals in the database then selects and registers an appropriate hospital. While showing hospital list, all detailed information related to hospital should be shown with it. User also can delete him/herself from registered users of that hospital.

3.2 Server

Server should establish database connection if database is preferred. For every client, server should able to grant multi-thread process for each client. Since not all database queries consists write operation, server should use a multiple reader single writer lock in order to implement mutual exclusion.

Chapter 4

Experimental Methodology and Results

While planning the project, I've thought a few ways of implementation all of which consisting a server based database system. Then decided on an implementation path as follows

- Server - Client Connection
  Before beginning to the project it was essential to establish a solid TCP connection between a Server java code and Client java code. By using port number : 50000 server can get and give strings to its client without problem. The main point in Server - Client connection is that server should be able to server more than one client at the same time. For this, creating threads for every single accepting socket is essential.
  In order to implement multithreading, everytime socket accepts a new client it creates a thread by using my thread class and starts execution.

- Server - Database Connection
  While using threads, we need to be careful about shared resources, in this case it is our database. I've preferred using mysql database because it is easy to establish connection with Connector/J driver. Database consists three essential table, users contains user credentials, hospitals contains detailed hospital information and userselection contains which user selected which hospital.

In order to grant mutual exclusion among threads, I've initialized a Read-Write [2] lock which allows multiple reader at the same time but not writer or single writer but no reader. According to SQL query coming from client, server checks whether there is a insert,update or delete operation (write operations) and locks for writer, for other cases before connection database, readers locks reader lock.
All these Server, Database functions are written by using Eclipse Oxygen.

- Processing incoming query from client and returning results to client
  Clients sends their query as a normal string, Server processes it and returns a Result Set from database connection. Parser method [3] in server class, parses Result set to Json array. Threads sends Json array to clients by converting it to string.

- Android client
  Our client is an Android application. When it is necessary Client runs Async task, which is invisible to user, checks for authorization. Whenever a client wants to have data, they run their spesific Async tasks and gets their Json arrays, so how do I extract data from this Json array. For extracting data from Json Arrays I've followed the instructions as explained in the reference [4]. Since I don't know the number of incoming items, I've used RecylerView and CardView and Adapter. Adapter can inflate TextViews into cardview according to their number. For adding new hospital, register/unregister to hospital functions are listened via ImageViews(They seem like floating action buttons). These are the methods I've used, for developing the app, Android Studio 3.0 is used.

- Android studio test challanges
  Android applications are hard to debug compared to server part. To improve debugging I've used logs in activity classes. Since it is a kind of new environment for me to work on, connecting and testing it on a real device helped me a lot.

- Design problems

  During developing I've used different layouts. For unknown number of incoming queries, I've used RecyclerView,CardView and TextView with layout inflator. Most of the parts I've used Vertical Linear Layout In the parts of adding and deleting user, using Frame Layout eased my job a lot.

  By playing with the visibilities of Image Views, I completed a nice view of add and delete button Choosing the right colors might be the most significant design issue :)

As a result, Server-Client, Server-Database and all incoming sql queries worked very well. However adjusting layouts and adapters when sql queries returned null is hard and seems there is a bug related to setting adapter and updating view of main activity.

Chapter 5

Related Work

Only related work which has been done is the website that I'm inspired of. That website have lots of functionality unlike ours. On the website, users can give advertisements, can join to the chatrooms, prepare a selection of choices list, discuss about the hospitals etc.

The main objective of the website is to see the preferences about hospitals and to see how many people selected that place, my app is also driven from the same idea, and it completes its basic and necessary functions.

Chapter 6

Conclusions

6.1 Server-Database side

Database queries can run with infinite number of queries which means in one thread, you can do as many query as possible without terminating thread or closing the socket connection. However, some queries return null objects, in my structure there is no method for handling null result sets. With this handlers, client side reactions may become more stabilized.

6.2 Android-Client side

Client side is starting with login page with a ling to signup page, these activities are running without problem (if don't count design issues as a problem :)). Main activities where user can show the details of choices and other users or hospitals, because of the usage of complex structures such as adapters and different text views in different xml files, unfortunately these activites are capable of producing bug very easily. As much as I tested it with different use cases there was no clear bug, still it may surprise us.

6.3 Overview

Since the application does what it is intended and built for, it is a great candidate for becoming the official mobile application of state of the art website. With database which can work concurrent and TCP connection in which no mistakes are allowed during the communication, there is no reason for this app not to be a great app.
As a future work, design of the app can be improved and new actions and futures with more interactive user interface can be implemented.

References

[1] URL: https://www.doktoruz.com/.

[2] URL: http://tutorials.jenkov.com/java-concurrency/read-write-locks.html#simple.

[3] URL: https://stackoverflow.com/questions/6514876/most-efficient-conversion-of-resultset-to-json.

[4] URL: https://stackoverflow.com/questions/20058240/extracting-data-from-json-array.