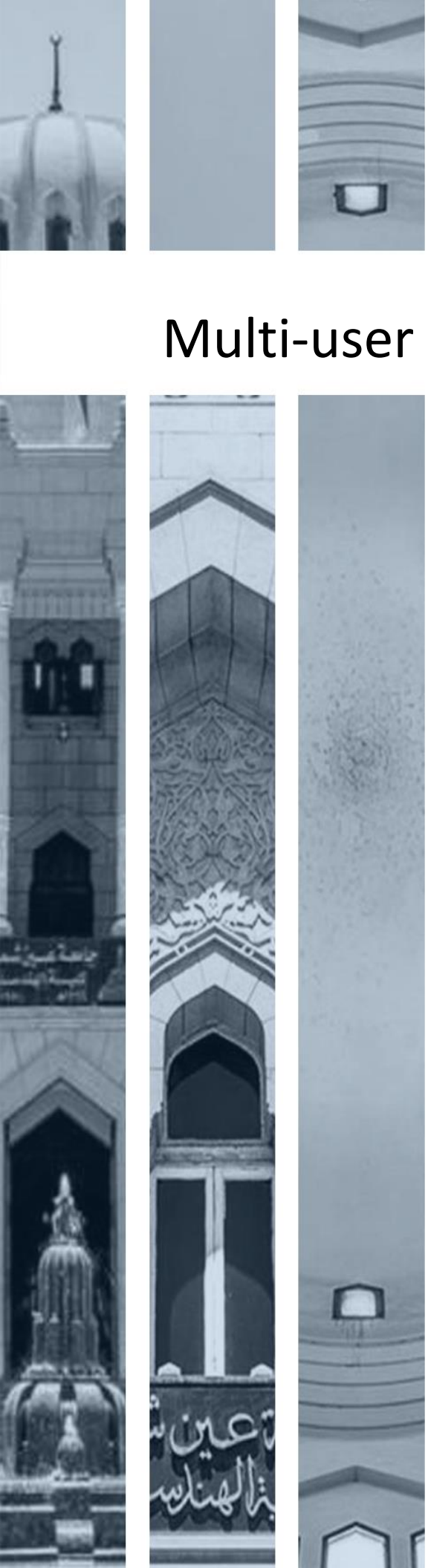

Multi-user Distributed Text Editor

Program: CESS

Course Code: CSE-354

Course Name: Distributed Computing

Presented To:

Prof. Ayman Bahaa
Eng. Mostafa Ashraf

Presented By:

Ahmed Emad Anwer	18P3597
Kareem Mohamed Abdelhamid	18P2027
Madonna Bassem	18P5194
Remon Emad Francis	18P1679

Team Number:

22

Ain Shams University
Faculty of Engineering

Spring Semester – 2022

Table of Content

List of Figures

Introduction

Distributed computing is simply allowing several computed to work together in order to achieve a certain output for the user to benefit of. This cooperation of individual computers ensures the concurrency of the whole system at all times specially when one of this computes fails. The main importance behind developing a distributed computing system is improving the efficiency and the performance.

During this report, we will take about distributed text editor, where more than one user can edit on the same file at the same time. Moreover, how the server manages the workload and what will happen it the server fail, or the client edited the file he/she is offline.

Detailed Project Description

This project aims for developing a multi-user distributed text editor. This text editor will allow several agents weather humans or automated computers to share the resources of a certain document granting the validity of editing to all the users of the shared document at the same time. Apart from the resource sharing, the text editor should be able to perform the main tasks any text editor could simply handle as providing the user the ability to input almost an infinite number of letters and numbers and adjust the fonts, sizes and colors of the text entered.

It is a web-based project where the admin can view details about the servers.

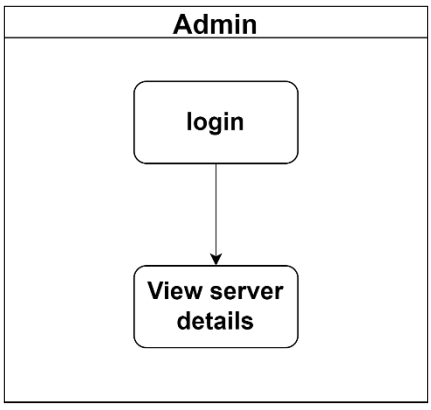

Fig 1. Admin viewpoint

The user needs to register first then login to be able to have access. After login, the user has three options to choose from which are creating new file, open recent file or join an existing file through file ID.

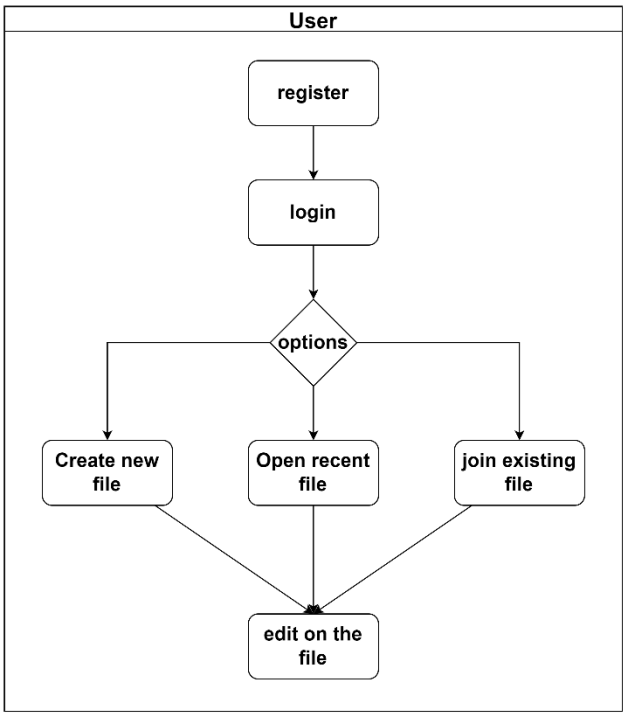

Fig 2. User viewpoint

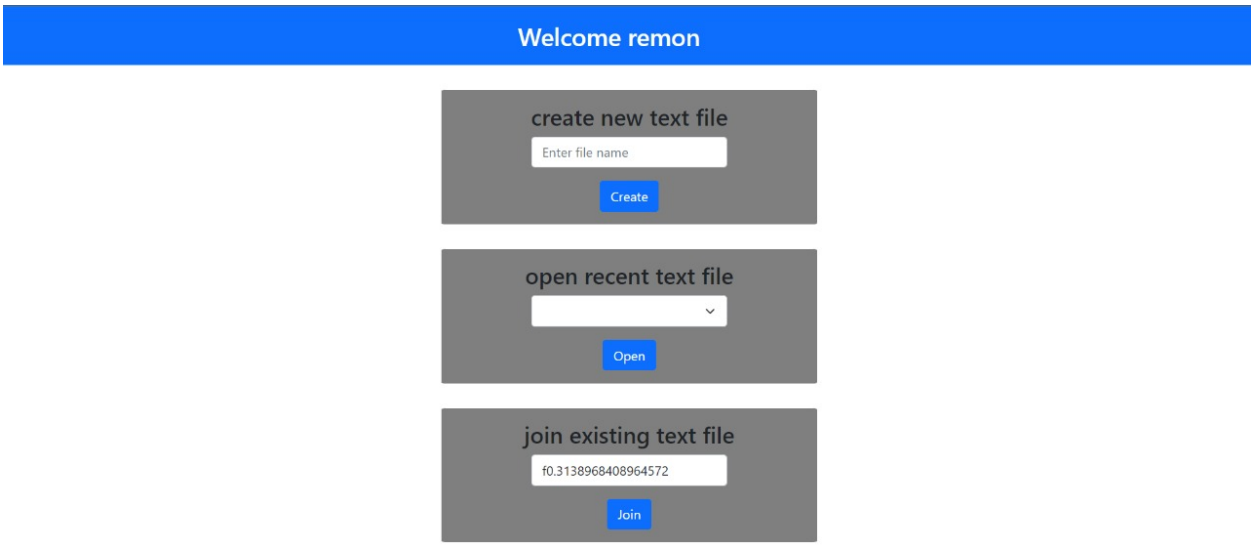

Fig 3. User Page

For example, if the user opens existing file, he/she will view the text in the file coupled with some details which are his/her state (show whether online or offline if disconnected from the network). This will tell the user that other users viewing the file cannot view the edits. Moreover, the number of online users.

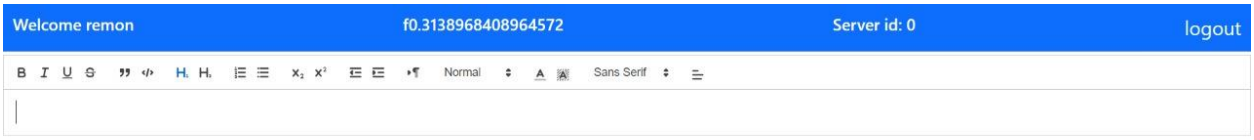

Fig 4. Opening existing file

Beneficiaries

The main advantage behind developing a multi-user distributed text editor is saving the time consumed on gathering a certain group of people for writing a collaborative document, instead, each one of them could access the multi-user distributed text editor using a different computer and start collaborating on writing a certain document. In addition to this, the multi-user distributed text editor could be easily accessed using a smart device giving the user the advantage of the document portability and availability at any time.

Examples:

- Employees in a department of a company
- Group of students working on a project

Not only for groups, but also any individual person can benefit from it.

Example:

- No need to have a copy of the file because you can easily access and edit it from anywhere through smartphone, tablet, laptop, etc.

Detailed Analysis

The structure mainly consists of two parts which are the Dispatcher and the server. The Dispatcher have a list of all files names and the servers storing them, while the servers hold the files. Each file has three copies which are distributed on three servers.

When client try to access a file. Firstly, the client tries to login to the system which will send HTTP request to dispatcher. If the entered data is correct the dispatcher will send HTTP respond to the client which contains the client home page. Secondly, the user sends HTTP request to access certain file. When the dispatcher receives the request, and search in the list it have for the servers holding this file then select one of the three servers and allow the user to open Socket with the server.

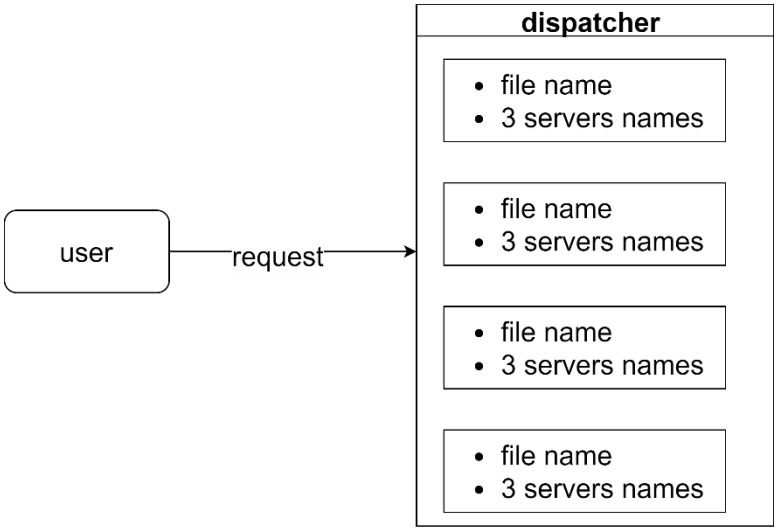

Fig 5. Client sending request to dispatcher

By applying this architecture, the distributed system achieves **Replication Transparency** as the user do not that there are 3 copies of his/her files. Furthermore, the **failure transparency** if one of the three servers is down due to any problem, the client still can access the file without knowing that one of three servers is down.

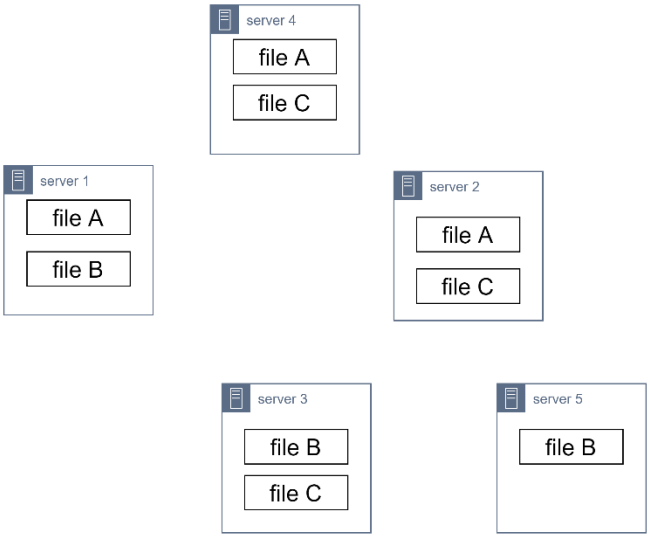

Fig 6. File backup on different servers

Both the server and the client have a show which used to compare the different versions of a file. To illustrate, when the client edits in the file, the edits are sent to the server to compare it with the shadow then update the server stored in the server. After this step, the server sends a copy of the new shadow to other clients. At the client side, the client compares the received shadow with its shadow then update it. By performing this structure, the shadow of the server and clients must be the same.

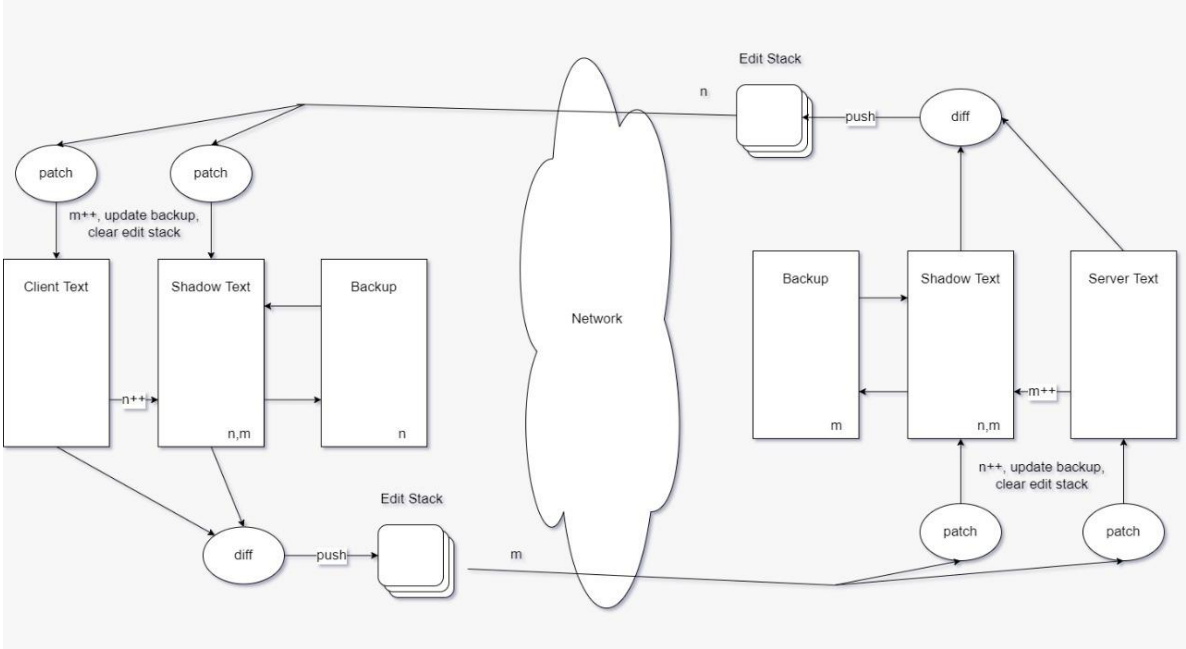

Fig 7. Updating file after user's edits

Moving to another part, how the three servers sync the file? It is similar to the structure the client and server user. When server receives edits from the user and update the shadow, it then sends its shadow to the other two servers, and they compare their shadow with it.

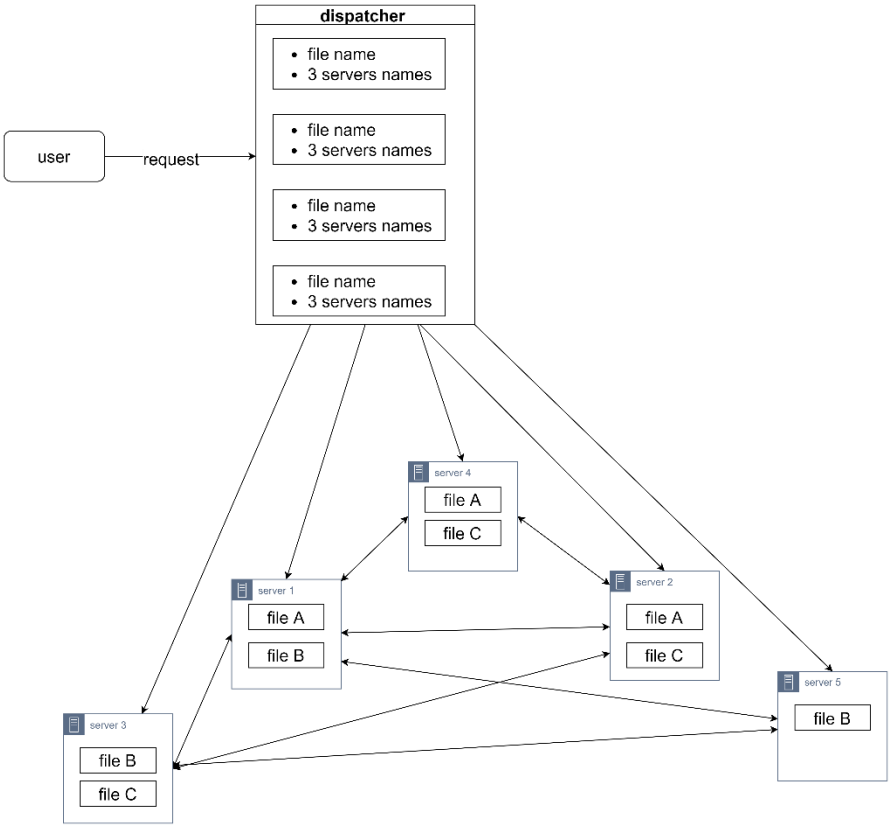

Fig 8. Updating backup files on other servers

By applying this structure, the system achieves **consistency** and **robustness** as well as **distribution transparency**. One more advantage, it the scalability. Now, we can **horizontality scalability** on the system if we need to add new server.

Task Breakdown Structure

In general, we followed the agile methodology but with some modifications. These points illustrating how we moved through the project (in order).

1. Studied the structure and the architecture of sharable files
2. Learn more about real time distributed systems

3. List the system features we need to implement
4. List the libraries/packages needed to implement these features
5. Discuss how Admin and User can access the system and what they do
6. List all the .html pages needed
7. Design the overall structure and architecture which best fit the needed features
8. Design the structure of the database
9. Divide all the system into modules and start coding
10. Test each module after finishing it
11. Integrate the modules at the end and test the whole system

Role of each Team Member

From point 1 to 9 in Task Breakdown Structure, we worked on them together in a meeting then we divided the modules on the four of us.

System architecture and design

the servers communicate with each other using Requests and response.

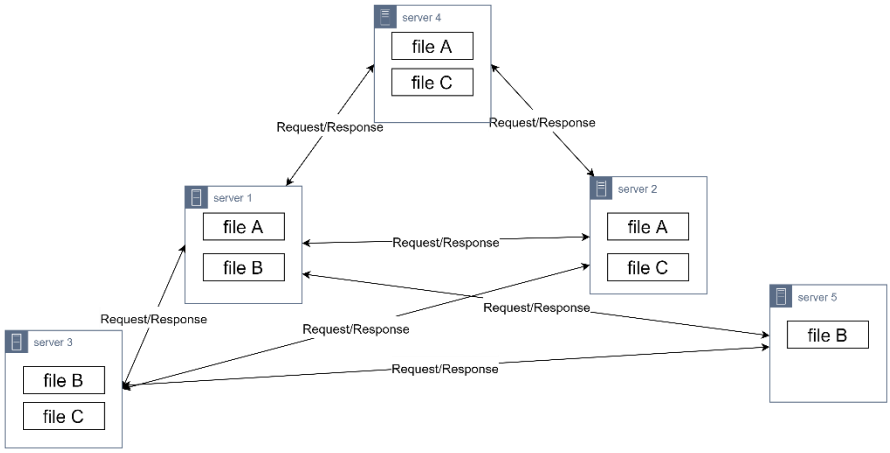

Fig 9. Servers Communications

the client communicates with the server using sockets

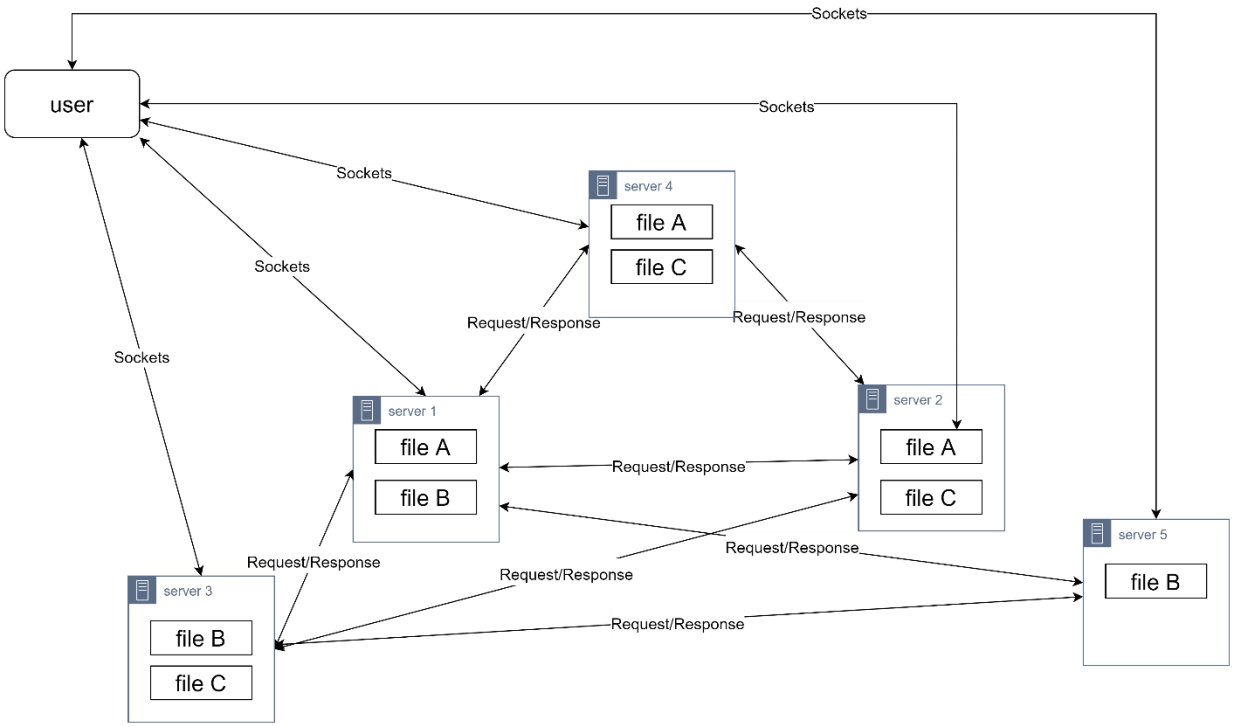

Fig 10. Server and client communicating

Testing scenarios and results

Test1:

Scenario: many clients typing at the same time.

Result: the file is always updating it content and the clients view the update at real time

Test2:

Scenario: a client edits in the file while offline

Result: the edits are sent to the server when become online again and other clients view the changes

Test3:

Scenario: more than one client edits in the file while offline

Result: if they are typing in different lines then no problem will happen and when become online again, the server handles all the changes and clients view all the changes. If more than one client edits in the same place in the line, then the server will take the last edits it receives

Test4:

Scenario: server is down will clients typing

Result: when it became online again, it will receive all the edits and sync it with all the clients

End-user guide

1. Register
2. Login

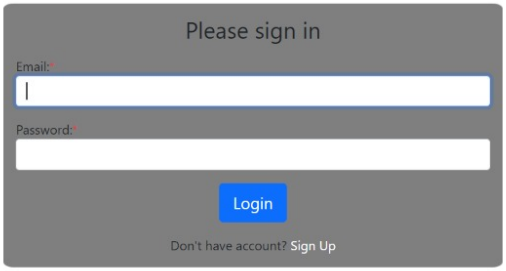

Fig 11. Login Page

3. Create new file, open recent file, or join existing file

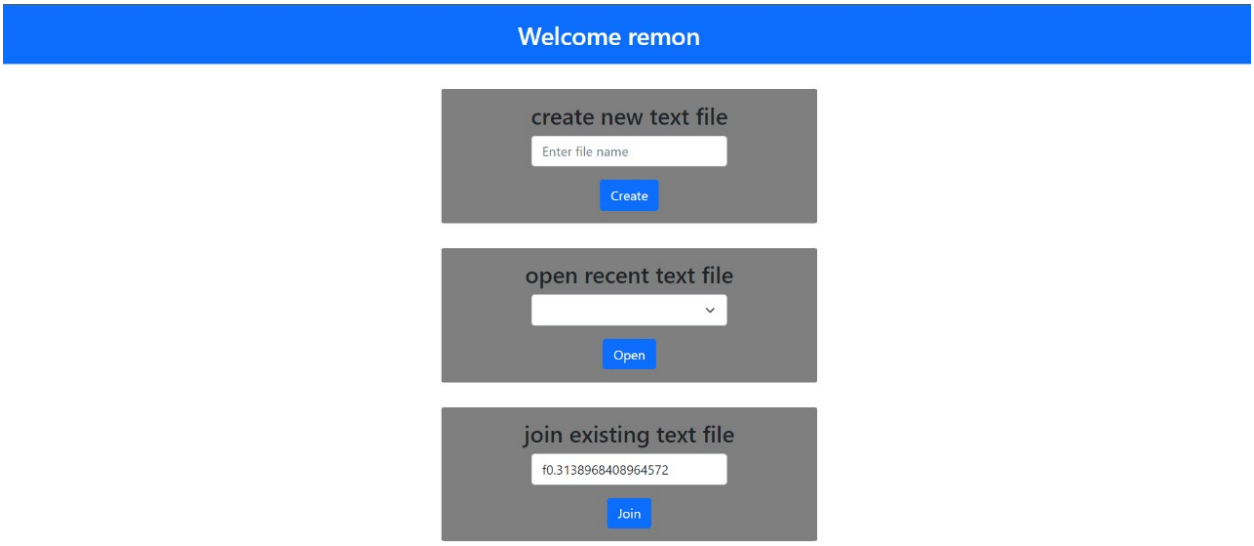

Fig 12. User Page

4. Typing in the file

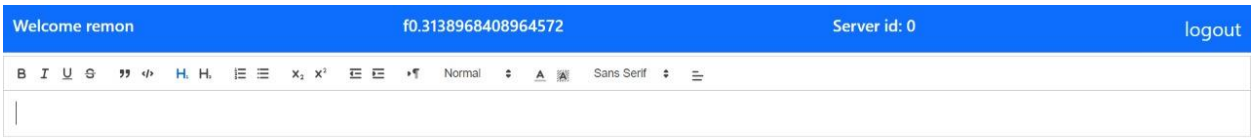

Fig 13. Editing Screen

Conclusion

To sum it all up, it is a fully real-time text editor with low percentage of failure because there is network of distributed servers to handle the request so no load balance problem. Moreover, each file has 3 copies so the user cannot lose his/her files.

References

[1] Brendan Burns, Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services, United States: O'Reilly Media, Inc., February 20, 2018

[2] "Introduction to Distributed System Design", https://bit.ly/3xOucE4 Accessed in June 26th, 2022

[3] "Introduction to distributed Systems", https://bit.ly/3QOXUle Accessed in June 26th, 2022

Video & Code Drive Link

https://drive.google.com/drive/folders/1nS9Dw7Vv_0RAhRfwnMvNL9awnqPdWtTu?usp=sharing

GitHub Link

https://github.com/RemonEmad93/distributed-text-editor-/