Nome: Eduardo da Silva Barbosa **DRE:** 116150432

Atividade 1

Camadas:

1. **Funcionalidades de camadas de interface com o usuário:** recebe do usuário o nome do arquivo e a palavra de busca e exibe na tela o resultado do processamento. O resultado do processamento poderá ser: *(i) uma mensagem de erro indicando que o arquivo não foi encontrado ; ou (ii) o número de ocorrências da palavra no arquivo.* Além disso, as possíveis mensagens serão editadas e tratadas nessa camada para então serem apresentadas ao usuário.
2. **Funcionalidades da camada de processamento:** solicita o acesso ao arquivo texto. Se o arquivo for válido, realiza a busca pela palavra informada e prepara a resposta para ser devolvida para a camada de interface. Se o arquivo for inválido, responde com a mensagem de erro. Vale ressaltar que as mensagens serão entregues para a camada de interface através de uma estrutura de json.
3. **Funcionalidades da camada de acesso aos dados:** verifica se o arquivo existe em sua base. Se sim, devolve o seu conteúdo inteiro. Caso contrário, devolve uma mensagem de erro.

Atividade 2

1. **Lado cliente:** implementa a camada de interface com o usuário. O usuário poderá solicitar o processamento de uma ou mais buscas em uma única execução da aplicação: o programa espera pelo nome do arquivo e da palavra de busca, faz o processamento, retorna o resultado, e então aguarda um novo pedido de arquivo e palavra ou o comando de finalização. O comando de finalização será o carácter “#” e o mesmo poderá ser digitado no momento da inserção do nome do arquivo ou da palavra que será buscada.

2. **Lado servidor:** implementa a camada de processamento e a camada de acesso aos dados. Projete um servidor iterativo, isto é, que trata as requisições de um cliente de cada vez, em um único fluxo de execução (estudaremos essa classificação depois). Terminada a interação com um cliente, ele poderá voltar a esperar por nova conexão. Dessa forma, o programa do servidor fica em loop infinito (depois veremos como lidar com isso).

Implementação

O processamento não faz distinção entre caracteres maiúsculas e minúsculas e desconsidera pontos, vírgulas e caracteres como aspas, parentes chaves e colchetes. Além disso, caso o arquivo testado esteja na raiz do servidor, não é necessário informar o caminho do mesmo, caso contrário, é necessário informar todo o path.

Comunicação

Cliente e servidor irão se comunicar utilizando mensagens que estarão estruturadas em JSON. Inicialmente, o cliente deve enviar uma requisição para o servidor solicitando o processamento de determinado arquivo de texto, tal mensagem de requisição deve conter o nome do arquivo que será processado e a palavra que será usada para a contagem de ocorrência.

Após o processamento, por parte do servidor, o mesmo irá enviar uma mensagem de resposta para o cliente, tal resposta deve informar se o processamento foi bem sucedido ou falhou. No caso de um processamento bem sucedido a mensagem informará a palavra que foi utilizada no processamento e o número de ocorrência da mesma. Por fim, no caso do processamento ter falhado a mensagem conterá o nome do arquivo e uma mensagem informando o motivo da falha. Vale ressaltar que todos os campos do json são todos strings com exceção do número de ocorrências que será um inteiro.

A seguir é apresentado o diagrama de sequência da aplicação, com as diferentes possibilidades de resposta.

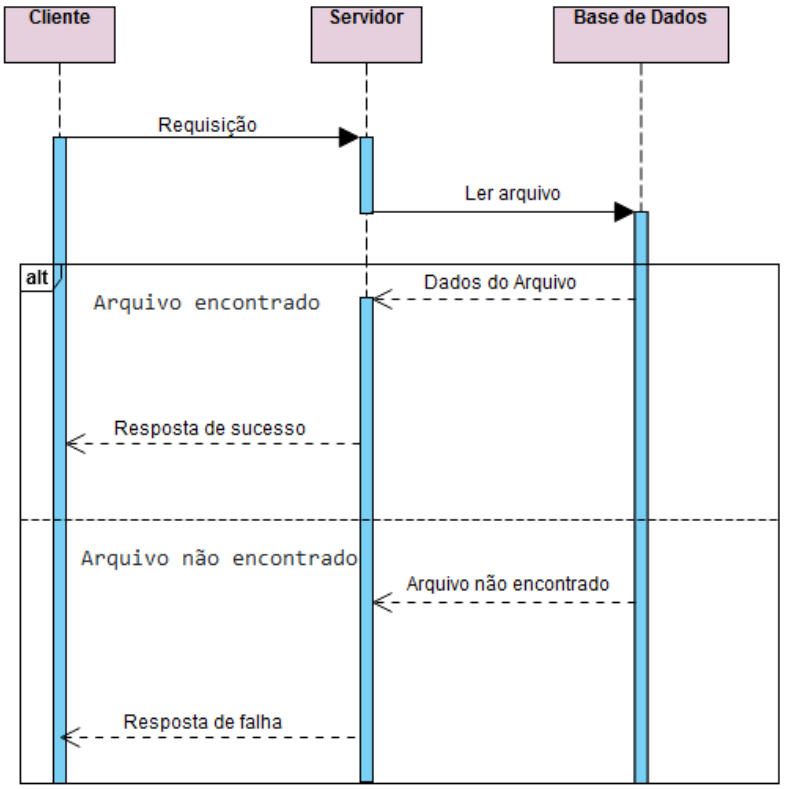

Diagrama de sequência

A seguir é apresentado a estrutura do json para cada uma das possíveis situações de troca de mensagem.

Requisição:

```
{
   “file_name”: "Experimento.txt",
   “word”:”Laranja”
}
```

Resposta no caso de sucesso:

```
{
   “status” : "success",
   “data” : {
      “word”: “Laranja”,
      "occurrences" : 50
    }
}
```

Resposta no caso de falha:

```
{
    “status” : "fail",
    “data” : {
        “file_name”:”Experimento.txt”,
        "message" : “file not found”
     }
}
```