CENTRO UNIVERSITÁRIO FEI
CIÊNCIA DA COMPUTAÇÃO

PROJETO: SERVIDOR HTTP/1.1

ANDY SILVA BARBOSA – RA: 22.218.025-9
RAFAEL ZACARIAS PALIERINI - RA: 22.218.030-9
RUBENS DE ARAÚJO RODRIGUES MENDES - RA: 22.218.009-3
VITOR ACOSTA DA ROSA – RA: 22.218.006-9

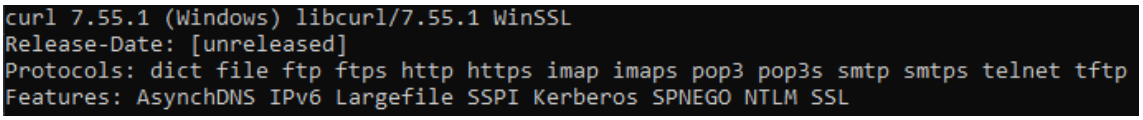
SÃO BERNARDO – SP
2020

1. **Preparação para uso**

Antes de verificar as instruções é necessário que seu computador possua o cURL (https://curl.haxx.se/), um software open source utilizado em scripts ou linhas de comando para transferir dados. Esse software suporta a transferência de arquivos seguindo diversos tipos de protocolos, entre eles, o protocolo HTTP requisitado no trabalho.

Para verificar se seu computador possui o cURL, basta digitar no terminal ou CMD o comando "curl --version". Se existe o software disponível, você receberá uma mensagem como na figura 1.

Figura 1: Resposta ao curl --version



Caso não possua, é possível instalar rapidamente utilizando os comandos:

sudo apt install curl - Para distribuições como Ubuntu.

pacman -Sy curl - Para distribuições como o ArchLinux.

yum install curl - Para distribuições como Fedora/CentOS.

Ou, caso esteja utilizando o sistema operacional Windows, é possível instalar o executável a partir da página de download do cURL (https://curl.haxx.se/download.html).

Após instalar o cURL e certificar que ele está executando corretamente, é possível testar o servidor.

2. **Instruções gerais para uso**

O diretório raiz possui:

4 arquivos .py correspondentes à funcionalidade do servidor.

1 arquivo .html correspondente à página base do servidor.

1 diretório chamado userdata que armazenará os dados enviados pelo cliente ao servidor a partir de uma requisição do tipo PUT.

O arquivo que deverá ser mantido em execução é o **ServerMain.py**, os demais arquivos presentes no diretório raiz são funções auxiliares para que todas as requisições sejam atendidas corretamente. O servidor escuta na porta 8080 do *localhost* (127.0.0.1), qualquer outra porta utilizada para o contatar não

funcionará. Caso deseje outra porta, a partir do arquivo ServerMain.py é possível alterar a variável “self.PORTA”, na linha 6, para a porta desejada.

O servidor HTTP pode responder corretamente aos seguintes cenários:

- A partir de uma requisição GET, retornar (se existir) um arquivo (se for dos tipos: html, jpeg, jpg, png, txt e gif). Utilizando o código 200 OK.
- Caso o recurso buscado não existir, o retorno esperado de um GET será o 404 Not Found.
- A partir de uma requisição PUT, o servidor permite que o cliente envie os dados. Utilizando o código 100 Continue.
- Caso o arquivo enviado pelo cliente através do PUT não exista previamente no servidor, o código 201 Created é retornado.
- Caso o arquivo enviado pelo cliente possua nome (e extensão) similar, o código retornado é o 200 OK.
- Já para a requisição PUT de um arquivo que já exista no servidor, e caso o novo arquivo possua um corpo vazio (isso é, sem dados), o antigo arquivo é sobrescrito e o código 204 No Content é esperado.
- Se o cliente enviar uma requisição PUT para um diretório diferente de /userdata/ o arquivo é criado (ou sobrescrito) nesse diretório e a mensagem 301 Moved Permanently é retornada, informando ao usuário que seu arquivo foi inserido no servidor, porém em um diretório diferente do requisitado.
- Além dessas mensagens, o servidor trata também erros gerais como 400 Bad request e 505 para versões HTTP não suportadas.

3. Adquirindo objetos do servidor com GET

O GET pode ser realizado de duas formas, com o cURL e pelo próprio navegador.

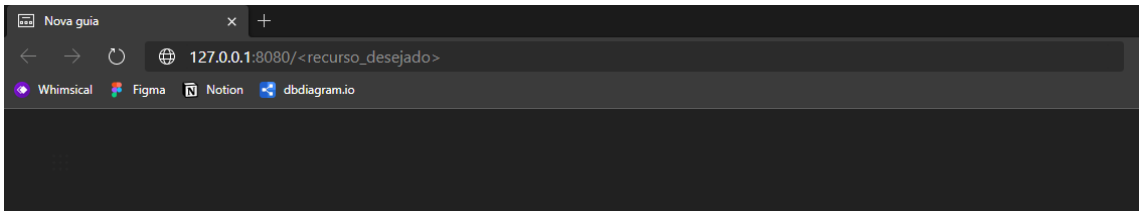
As respostas podem variar de acordo com a disponibilidade do arquivo no diretório requisitado, erros de sintaxe ou até mesmo a versão do http utilizado na requisição.

De forma geral, a sintaxe do GET (caso utilizando o cURL) será:

curl --http1.1 --verbose 127.0.0.1:8080/<objeto_buscado>

O grupo recomenda utilizar as tags --http1.1 para garantir o uso da versão correta requisitada no projeto e o --verbose para analisar com detalhes como a requisição e respostas foram estruturadas e enviadas.

Caso acessado pelo navegador, basta inserir na barra de pesquisa o endereço, porta e o recurso buscado.



4. Inserindo arquivos no servidor através do método PUT

A escolha para inserir objetos no servidor foi utilizar o método PUT. Para esse processo, será necessário ter o cURL em mãos.

O comando de inserção é simples:

curl --http1.1 --verbose -T <diretorio+arquivo>
127.0.0.1:8080/userdata/<nome_do_arquivo>

A tag -T serve para o upload de arquivos em um dado url.

É necessário seguir a composição do arquivo de upload informando todo o caminho até o arquivo.

Já o destino do upload será sempre o 127.0.0.1:8080/userdata. Esse diretório é tratado pelo servidor como o destino de todo arquivo inserido pelo método PUT.

Após executar o comando, para verificar se o dado foi corretamente inserido, podemos realizar um GET.

De acordo com a RFC 2616, caso um arquivo requisitado já exista no servidor, o arquivo deve ser sobrescrito e a resposta retornada deve ser 200 OK indicando a mudança no conteúdo original, ou 204 No Content caso o novo arquivo não possua nenhum dado.

Agora, se o cliente desejar inserir em qualquer diretório que não seja o "userdata", o servidor irá encarregar-se de criar o arquivo na pasta correta e responder com o código 301, avisando ao usuário que seu recurso foi criado, mas não no url informado.

5. Considerações

O grupo aconselha fortemente a visualização do projeto em sua página do GitHub (https://github.com/Clube-do-Enrolado/Redes/tree/main/HTTP), pois no readme existem gifs que exemplificam cada situação as quais o servidor pode lidar.