UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO

**Roteiro de Aula Prática – Criação de Servidor HTTP**

DISCIPLINA: DCA0130 – Redes de Computadores

PROFESSOR: Carlos Manuel Dias Viegas

Esta prática consiste em uma introdução ao desenvolvimento de uma abstração de um servidor HTTP utilizando Sockets. O objetivo deste trabalho é colocar em prática os conhecimentos apresentados em aula a respeito da camada de aplicação e do protocolo HTTP. Deverá ser criado um servidor HTTP capaz de responder ao comando GET enviado por um cliente.

- Esta prática consiste em realizar as tarefas descritas abaixo e desenvolver o respectivo código fonte.

- O código fonte desenvolvido e os arquivos .html deverão ser enviados em uma tarefa específica no SIGAA até quarta-feira 17/05/2023 às 23h59;

- Esta prática pode ser realizada em duplas.

O código fonte (em Python) necessário para iniciar esta prática está disponível na seguinte página:

https://www.dca.ufrn.br/~viegas/disciplinas/DCA0130/files/Sockets/HTTPserver/

**Observação 1:** Obrigatoriamente deverá ser utilizado o código fonte disponibilizado acima como base para o desenvolvimento do servidor HTTP!

**Observação 2:** Não devem ser utilizadas bibliotecas “prontas” que implementam o HTTP.

**Observação 3:** Utilize o terminal MobaXterm para utilizar o telnet.

Nome do discente (1): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Nome do discente (2): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**VERIFICAÇÃO**

Nesta prática, deverão ser utilizados como clientes um navegador de Internet e um terminal com a aplicação telnet para acessar o servidor HTTP desenvolvido. Este servidor deverá ser capaz de responder aos pedidos dos clientes, sem distinção entre eles.

1. Execute o código fonte do servidor HTTP fornecido e entenda o seu funcionamento:
   1. [Acesso pelo navegador]

Abra um navegador e acesse o endereço do servidor:

http://127.0.0.1:8080

Repare que a página exibirá sempre a mensagem “Hello, World!”, independentemente do que for solicitado pelo cliente.

* 1. [Acesso pelo telnet]

Abra um terminal e acesse por meio de telnet o endereço do servidor:

telnet 127.0.0.1 8080

GET / HTTP 1.1

Após o GET, proceda com dois “Enter”.

Será exibido o cabeçalho da mensagem (HTTP/1.1 200 OK) e também “Hello, World!”. Repare que independentemente do comando digitado, o servidor retornará sempre a mesma resposta.

1. Após realizar os testes acima, analise o código fonte da aplicação com o uso de um editor de texto ou IDE de programação e proceda à execução das tarefas a seguir:

**TAREFAS**

1. Faça as alterações necessárias no código para que o servidor seja capaz de processar um pedido GET de um cliente e retornar um arquivo .html para o mesmo;
   1. Crie um arquivo index.html (na mesma pasta em que o servidor estiver sendo executado) com o código abaixo (como exemplo) para ser usado como resposta:

<html>

<head><title>Este é o meu servidor!</title></head>

<body>

<h1>Olá mundo!</h1>

O meu servidor funciona!

</body>

</html>

* 1. Quando um pedido é feito ao servidor, a variável request (linha 35 do código fonte) recebe os dados solicitados. Esta variável deve ser inspecionada para que se possa analisar o que está sendo pedido. É a partir dela que devem ser tratados os casos abaixo solicitados.

**DICA:** Esta variável request pode ser segmentada em pedaços menores de informação e consultados por índice (vetor).

* 1. A sintaxe do pedido GET deve seguir o padrão especificado pelo protocolo HTTP:

GET /caminho HTTP/versão

Exemplo de uso 1:

GET /arquivo.html HTTP/1.1

Exemplo de uso 2 (caso o arquivo esteja em uma pasta):

GET /pasta/arquivo.html HTTP/1.1

Exemplo de uso 3:

GET / HTTP/1.1

Neste caso, quando não é especificado o arquivo no pedido, o servidor deve procurar pelo index.html.

**IMPORTANTE**: Ao fazer um pedido com GET, nunca se deve inserir o caminho do arquivo no sistema de arquivos do sistema operacional, pois o HTTP não sabe interpretar esse caminho (exemplo a **NÃO** usar: C:\Documentos\pasta\arquivo.html ou /home/usuario/arquivo.html). Deve-se inserir apenas o caminho do arquivo em relação à pasta em que o servidor HTTP está rodando (caminho relativo), conforme nos exemplos anteriores.

* 1. Após o envio do comando GET pelo cliente, caso o arquivo solicitado exista, o servidor abrirá o mesmo e retornará o conteúdo para o cliente. Entretanto, antes de enviar o conteúdo do arquivo, o servidor deverá enviar/retornar (na primeira linha) o comando

HTTP/1.1 200 OK\r\n\r\n

E em seguida retornar o conteúdo do arquivo solicitado (.html ou outro). É importante destacar que esse comando não deve ser salvo no conteúdo do arquivo .html, pois trata-se de um comando do protocolo HTTP e não faz parte da linguagem HTML. Portanto, deve ser enviado antes do conteúdo do arquivo;

**Observação**: Os códigos \r e \n indicam, respectivamente, o fim de linha (ou fim do comando) e a quebra de linha.

* 1. Caso o arquivo/caminho solicitado não exista, o servidor deve retornar o código 404 - página não encontrada. Este retorno deve seguir a especificação do protocolo HTTP, conforme explicado anteriormente. A primeira linha de retorno deve ser

HTTP/1.1 404 Not Found\r\n\r\n

E em seguida um código .html com uma mensagem de erro (como no exemplo abaixo):

<html>

<head></head>

<body>

<h1>404 Not Found</h1>

</body>

</html>

* 1. Caso algum outro comando diferente de GET seja digitado, o servidor deve tratar a exceção e retornar “comando desconhecido” e continuar aguardando por novos comandos. Neste caso, a primeira linha de retorno deve ser:

HTTP/1.1 400 Bad Request\r\n\r\n

E em seguida um código .html com uma mensagem de pedido inválido (como no exemplo abaixo).

<html>

<head></head>

<body>

<h1>400 Bad Resquest</h1>

</body>

</html>

* 1. Sempre que um cliente enviar um comando GET para o servidor, este deve imprimir na tela (do terminal) todo o comando enviado pelo cliente. Ou seja, a variável request deve ser apresentada na tela do terminal que estiver executando o servidor;
  2. É importante lembrar que, por omissão, quando se faz um pedido diretamente à raiz (isto é: GET / HTTP/1.1), sem especificar uma página, o servidor deve retornar a página padrão (geralmente o index.html). Atentem que o cliente pode (e deve) pedir qualquer nome de arquivo e o servidor deve ser capaz de atender ao pedido (caso exista) ou enviar mensagem de erro (não encontrado).

1. Façam os testes do funcionamento do servidor usando tanto o telnet quanto um navegador.
2. O objetivo ao final da implementação é que o servidor receba e responda às solicitações dos clientes, independentemente do arquivo solicitado e do *software* (navegador) que esteja sendo utilizado para o pedido.