



## Servidor Web

É quando você finalmente entende por que as URLs começam com HTTP

### *Resumo:*

*Este projeto é sobre escrever seu próprio servidor HTTP.  
Você poderá testá-lo com um navegador real.*

*HTTP é um dos protocolos mais utilizados na internet.*

*Entender suas complexidades será útil, mesmo que você não esteja trabalhando em um site.*

*Versão: 23.0*

# Conteúdo

I	<b>Introdução</b>	<b>2</b>
II	<b>Regras gerais</b>	<b>3</b>
	<b>Instruções de IA III</b>	<b>4</b>
4	<b>Parte obrigatória IV.1</b>	<b>6</b>
	Requisitos.	8
	IV.2 Somente para MacOS . . . . .	10
	IV.3 Arquivo de configuração . . . . .	10
V	<b>Parte bônus</b>	<b>12</b>
VI	<b>Submissão e avaliação por pares</b>	<b>13</b>

# Capítulo I

## Introdução

O **Hypertext Transfer Protocol** (HTTP) é um protocolo de aplicação para sistemas de informação hipermídia distribuídos e colaborativos.

HTTP é a base da comunicação de dados para a World Wide Web, onde documentos de hipertexto incluem hiperlinks para outros recursos que o usuário pode acessar facilmente, por exemplo, clicando com o botão do mouse ou tocando na tela de um navegador da web.

O HTTP foi desenvolvido para dar suporte à funcionalidade de hipertexto e ao crescimento da World Wide Web.

A função principal de um servidor web é armazenar, processar e entregar páginas web aos clientes. A comunicação cliente-servidor ocorre por meio do Protocolo de Transferência de Hipertexto (HTTP).

As páginas entregues são, na maioria das vezes, documentos HTML, que podem incluir imagens, folhas de estilo e scripts, além do conteúdo de texto.

Vários servidores web podem ser usados para um site de alto tráfego, dividindo o tráfego entre várias máquinas físicas.

Um agente de usuário, geralmente um navegador ou rastreador web, inicia a comunicação solicitando um recurso específico via HTTP, e o servidor responde com o conteúdo desse recurso ou com uma mensagem de erro, caso não consiga. O recurso normalmente é um arquivo real no armazenamento do servidor ou o resultado de um programa. Mas isso nem sempre acontece e pode, na verdade, ser muitas outras coisas.

Embora sua função principal seja servir conteúdo, o HTTP também permite que os clientes enviem dados. Esse recurso é usado para enviar formulários da web, incluindo o upload de arquivos.

## Capítulo II

### Regras gerais

- Seu programa não deve travar em nenhuma circunstância (mesmo que fique sem memória).  
ória) ou terminar inesperadamente.  
Se isso ocorrer, seu projeto será considerado não funcional e sua nota será 0.
- Você deve enviar um Makefile que compila seus arquivos de origem. Ele não deve executar relinkagem desnecessária.
- Seu Makefile deve conter pelo menos as regras:  
\$(NAME), todos, limpos, fclean e re.
- Compile seu código com c++ e os sinalizadores -Wall -Wextra -Werror
- Seu código deve estar em conformidade com o **padrão C++ 98** e ainda deve ser compilado quando adicionando o sinalizador -std=c++98.
- Certifique-se de aproveitar o máximo possível de recursos do C++ (por exemplo, escolha <cstring> em vez de <string.h>). Você pode usar funções em C, mas sempre prefira suas versões em C++, se possível.
- Qualquer biblioteca externa e bibliotecas Boost são proibidas.

## Capítulo III

### Instruções de IA

#### • Context

Durante sua jornada de aprendizado, a IA pode auxiliar em muitas tarefas diferentes. Reserve um tempo para explorar os diversos recursos das ferramentas de IA e como elas podem auxiliar seu trabalho. No entanto, sempre aborde com cautela e avalie os resultados de forma crítica. Seja código, documentação, ideias ou explicações técnicas, você nunca pode ter certeza absoluta de que sua pergunta foi bem formulada ou de que o conteúdo gerado é preciso. Seus colegas são um recurso valioso para ajudá-lo a evitar erros e pontos cegos.

#### • Mensagem principal

- ÿ Use IA para reduzir tarefas repetitivas ou tediosas.
- ÿ Desenvolver habilidades de estímulo — tanto de codificação quanto de não codificação — que irão beneficiar sua carreira futura.
- ÿ Aprenda como os sistemas de IA funcionam para melhor antecipar e evitar riscos, vieses e problemas éticos comuns.
- ÿ Continue desenvolvendo habilidades técnicas e de potência trabalhando com seus colegas.
- ÿ Use apenas conteúdo gerado por IA que você entenda completamente e possa assumir responsabilidade para.

#### • Regras do aluno:

- Você deve reservar um tempo para explorar ferramentas de IA e entender como elas funcionam, para que possa usá-las de forma ética e reduzir possíveis vieses.
- Você deve refletir sobre o seu problema antes de fazer o prompt — isso ajuda você a escrever de forma mais clara, prompts mais detalhados e relevantes usando vocabulário preciso.
- Você deve desenvolver o hábito de verificar, revisar, questionar e testar sistematicamente qualquer coisa gerada pela IA.
- Você deve sempre buscar revisão por pares — não confie somente na sua própria validação.

## • Resultados da fase:

- Desenvolver habilidades de orientação tanto para fins gerais quanto para áreas específicas.
- Aumente sua produtividade com o uso eficaz de ferramentas de IA.
- Continuar a fortalecer o pensamento computacional, a resolução de problemas, a adaptabilidade e colaboração.

## • Comentários e exemplos:

- Você encontrará regularmente situações — provas, avaliações e muito mais — em que precisará demonstrar real compreensão. Esteja preparado e continue desenvolvendo suas habilidades técnicas e interpessoais.
- Explicar seu raciocínio e debater com colegas muitas vezes revela lacunas em sua compreensão compreensão. Priorize a aprendizagem entre pares.
- As ferramentas de IA geralmente não levam em conta o seu contexto específico e tendem a fornecer respostas genéricas. Seus colegas, que compartilham o seu ambiente, podem oferecer insights mais relevantes e precisos.
- Enquanto a IA tende a gerar a resposta mais provável, seus colegas podem fornecer perspectivas alternativas e nuances valiosas. Conte com eles como um ponto de verificação de qualidade.

### ÿ Boas práticas:

Pergunto à IA: "Como testo uma função de classificação?" Ela me dá algumas ideias. Eu as testo e reviso os resultados com um colega. Refinamos a abordagem juntos.

### ÿ Má prática:

Peço à IA para escrever uma função completa e copiá-la e colá-la no meu projeto. Durante a avaliação por pares, não consigo explicar o que ela faz ou por quê. Perco credibilidade — e reprovo meu projeto.

### ÿ Boas práticas:

Uso IA para ajudar a projetar um analisador. Depois, analiso a lógica com um colega. Identificamos dois bugs e reescrevemos juntos — melhor, mais limpo e totalmente compreendido.

### ÿ Má prática:

Deixei o Copilot gerar o código para uma parte essencial do meu projeto. Ele compila, mas não consigo explicar como ele lida com pipes. Durante a avaliação, não consigo justificar e meu projeto é reprovado.

# Capítulo IV

## Parte obrigatória

<b>Nome do programa</b>	servidor web
<b>Entregar arquivos</b>	Makefile, *.{h, hpp}, *.cpp, *.tpp, *.ipp, arquivos de configuração
<b>Makefile</b>	NOME, tudo, limpo, fclean, re
<b>Argumentos</b>	[Um arquivo de configuração]
<b>Funções externas.</b>	Todas as funcionalidades devem ser implementadas em C++ 98. execve, pipe, strerror, gai_strerror, errno, dup, dup2, garfo, socketpair, htons, htonl, ntohs, ntohl, selecionar, pesquisar, epoll (epoll_create, epoll_ctl, epoll_wait), kqueue (kqueue, kevent), soquete, aceitar, ouvir, enviar, receber, chdir, vincular, conectar, getaddrinfo, freeaddrinfo, setsockopt, getsockname, getprotobynumber, fcntl, fechar, ler, escrever, waitpid, matar, sinalizar, acessar, stat, abrir, opendir, readdir e fechado.
<b>Libft autorizado</b>	n / D
<b>Descrição</b>	Um servidor HTTP em C++ 98

Você deve escrever um servidor HTTP em C++ 98.

Seu executável deve ser executado da seguinte maneira:

./webserv [arquivo de configuração]



Mesmo que poll() seja mencionado no assunto e na folha de avaliação,  
você pode usar qualquer função equivalente, como select(), kqueue() ou  
epoll().



Leia os RFCs que definem o protocolo HTTP e realize testes com telnet e NGINX antes de iniciar este projeto.

Embora você não seja obrigado a implementar todos os RFCs, lê-los ajudará você a desenvolver os recursos necessários.

O HTTP 1.0 é sugerido como ponto de referência, mas não é imposto.

## IV.1 Requisitos

- Seu programa deve usar um arquivo de configuração, fornecido como um argumento na linha de comando ou disponível em um caminho padrão.
- Você não pode executar outro servidor web.
- Seu servidor deve permanecer sem bloqueio o tempo todo e lidar adequadamente com o atendimento ao cliente. conexões quando necessário.
- Deve ser não bloqueante e usar apenas 1 poll() (ou equivalente) para todas as operações de E/S entre os clientes e o servidor (incluindo escuta).
- poll() (ou equivalente) deve monitorar a leitura e a escrita simultaneamente.
- Você nunca deve fazer uma operação de leitura ou gravação sem passar por poll() (ou equivalente).
- É estritamente proibido verificar o valor de errno para ajustar o comportamento do servidor após executar uma operação de leitura ou gravação.
- Você não é obrigado a usar poll() (ou equivalente) antes de read() para recuperar seu arquivo de configuração.



Como você precisa usar descritores de arquivo não bloqueantes, é possível usar funções de leitura/recebimento ou gravação/envio sem poll() (ou equivalente), e seu servidor não estaria bloqueando.

Mas isso consumiria mais recursos do sistema.

Portanto, se você tentar ler/receber ou escrever/enviar em qualquer descritor de arquivo sem usar poll() (ou equivalente), sua nota será seja 0.

- Ao usar poll() ou qualquer chamada equivalente, você pode usar todas as macros ou função auxiliar (por exemplo, FD\_SET para select()).
- Uma solicitação ao seu servidor nunca deve ficar travada indefinidamente.
- Seu servidor deve ser compatível com os **navegadores** padrão de sua escolha.
- O NGINX pode ser usado para comparar cabeçalhos e comportamentos de resposta (preste atenção às diferenças entre as versões HTTP).
- Seus códigos de status de resposta HTTP devem ser precisos.
- Seu servidor deve ter **páginas de erro padrão** se nenhuma for fornecida.
- Você não pode usar fork para nada além de CGI (como PHP, Python e assim por diante).
- Você deve ser capaz de **servir um site totalmente estático**.
- Os clientes devem poder fazer **upload de arquivos**.

- Você precisa de pelo menos os métodos GET, POST e DELETE.
- Faça um teste de estresse no seu servidor para garantir que ele permaneça disponível o tempo todo.
- Seu servidor deve ser capaz de escutar várias portas para entregar conteúdo diferente (consulte [o arquivo de configuração](#)).



Optamos deliberadamente por oferecer apenas um subconjunto do RFC HTTP. Neste contexto, o recurso de host virtual é considerado fora do escopo. Mas você tem permissão para implementá-lo se desejar.

## IV.2 Somente para MacOS



Como o macOS lida com write() de forma diferente de outros sistemas operacionais baseados em Unix, você tem permissão para usar fcntl().

Você deve usar descritores de arquivo no modo não bloqueador para obter comportamento semelhante ao de outros sistemas operacionais Unix.



Entretanto, você tem permissão para usar fcntl() somente com os seguintes sinalizadores:

F\_SETFL, O\_NONBLOCK e FD\_CLOEXEC.

Qualquer outra bandeira é proibida.

## IV.3 Arquivo de configuração



Você pode se inspirar na seção "servidor" do arquivo de configuração do NGINX.

No arquivo de configuração, você deve ser capaz de:

- Defina todos os pares interface:porta nos quais seu servidor escutará (definindo vários sites atendidos pelo seu programa).

- Configure páginas de erro padrão.

- Defina o tamanho máximo permitido para corpos de solicitação do cliente.

- Especifique regras ou configurações em uma URL/rota (nenhum regex necessário aqui), para um site, entre os seguintes:

  - Lista de métodos HTTP aceitos para a rota.

  - Redirecionamento HTTP.

  - Diretório onde o arquivo solicitado deve estar localizado (por exemplo, se a URL /kapouet estiver enraizada em /tmp/www, a URL /kapouet/pouic/toto/pouet pesquisará por /tmp/www/pouic/toto/pouet).

  - Habilitar ou desabilitar a listagem de diretórios.

  - Arquivo padrão a ser servido quando o recurso solicitado for um diretório.

  - O upload de arquivos dos clientes para o servidor é autorizado e o local de armazenamento é fornecido.

ÿ Execução de CGI, com base na extensão do arquivo (por exemplo, .php). Aqui estão algumas observações específicas sobre CGIs:

\* Você se pergunta o que é um CGI é?

ÿ Observe atentamente as variáveis de ambiente envolvidas na comunicação entre o servidor web e o CGI. A solicitação completa e os argumentos fornecidos pelo cliente devem estar disponíveis para o CGI.

ÿ Lembre-se de que, para solicitações em partes, seu servidor precisa desmembrá-las, o CGI esperará EOF como o fim do corpo.

ÿ O mesmo se aplica à saída do CGI. Se nenhum content\_length for retornado do CGI, EOF marcará o fim dos dados retornados.

ÿ O CGI deve ser executado no diretório correto para acesso ao arquivo de caminho relativo. ÿ Seu servidor deve suportar pelo menos um CGI (php-CGI, Python e assim por diante).  
adiante).

Você deve fornecer arquivos de configuração e arquivos padrão para testar e demonstrar que todos os recursos funcionam durante a avaliação.

Você pode ter outras regras ou informações de configuração em seu arquivo (por exemplo, um nome de servidor para um site se você planeja implementar hosts virtuais).



Se você tiver alguma dúvida sobre um comportamento específico, poderá comparar o comportamento do seu programa com o do NGINX.  
Disponibilizamos um pequeno testador. Seu uso não é obrigatório se tudo estiver funcionando bem com o seu navegador e os testes, mas pode ajudar a encontrar e corrigir bugs.



Resiliência é fundamental. Seu servidor deve permanecer operacional o tempo todo.



Não teste com apenas um programa. Escreva seus testes em uma linguagem mais adequada, como Python ou Golang, entre outras, ou até mesmo em C ou C++, se preferir.

# Capítulo V

## Parte bônus

Aqui estão alguns recursos adicionais que você pode implementar:

- Suporte a cookies e gerenciamento de sessão (forneça exemplos simples).
- Lidar com vários tipos de CGI.



A parte bônus só será avaliada se a parte obrigatória for concluída sem problemas. Se você não atender a todos os requisitos obrigatórios, sua parte bônus não será avaliada.

## Capítulo VI

# Submissão e avaliação por pares

Envie sua tarefa para o seu repositório Git como de costume. Somente o conteúdo do seu repositório será avaliado durante a defesa. Certifique-se de verificar os nomes dos seus arquivos para garantir que estejam corretos.

Durante a avaliação, uma breve **modificação no projeto** pode ser ocasionalmente solicitada. Isso pode envolver uma pequena mudança de comportamento, algumas linhas de código para escrever ou reescrever, ou um recurso fácil de adicionar.

Embora esta etapa possa **não ser aplicável a todos os projetos**, você deve estar preparado para ela se for mencionada nas diretrizes de avaliação.

Esta etapa serve para verificar sua compreensão real de uma parte específica do projeto.

A modificação pode ser realizada em qualquer ambiente de desenvolvimento que você escolher (por exemplo, sua configuração usual) e deve ser viável em poucos minutos — a menos que um prazo específico seja definido como parte da avaliação.

Você pode, por exemplo, ser solicitado a fazer uma pequena atualização em uma função ou script, modificar uma exibição ou ajustar uma estrutura de dados para armazenar novas informações, etc.

Os detalhes (escopo, meta, etc.) serão especificados nas **diretrizes de avaliação** e podem variar de uma avaliação para outra para o mesmo projeto.



16D85ACC441674FBA2DF65190663F42A3832CEA21E024516795E1223BBA77916734D1  
26120A16827E1B16612137E59ECD492E46EAB67D109B142D49054A7C281404901890F  
619D682524F5