Conexão entre Backend e Frontend

A comunicação entre o frontend (interface do usuário) e o backend (servidor) é essencial para o funcionamento de aplicações modernas. Vamos explorar como essa comunicação ocorre tecnicamente através do estudo de alguns conceitos.

Requests e Responses

Requests (Requisições)

Quando o usuário interage com a aplicação frontend, por exemplo, ao clicar em um botão ou preencher um formulário, o frontend envia uma requisição (request) ao backend. Esta requisição pode conter dados que o backend precisa processar, como informações de login ou dados de um novo cadastro.

Tipos de Dados: JSON e XML

As requisições e respostas entre o frontend e o backend geralmente são feitas usando formatos de dados padronizados como JSON (JavaScript Object Notation) ou XML (eXtensible Markup Language). JSON é o mais comum devido à sua simplicidade e compatibilidade com JavaScript.

Estrutura de uma Requisição

Uma requisição HTTP geralmente inclui:

- **URL**: O endereço do recurso no servidor.
- **Método HTTP**: Define a ação a ser executada (veremos mais detalhes abaixo).
- **Headers**: Informações adicionais, como o tipo de conteúdo.
- **Body**: Dados a serem enviados, geralmente em formato JSON ou XML.

Exemplo de Requisição JSON

```
POST /api/login HTTP/1.1
Host: example.com
Content-Type: application/json
{
  "username": "aluno",
  "password": "senha123"
}
```

Métodos HTTP

Os métodos HTTP indicam a ação desejada para um recurso específico. Os métodos mais comuns são:

- **GET**: Solicita dados do servidor (sem alterar o estado).
- **POST**: Envia dados ao servidor para criar um novo recurso.
- **PUT**: Atualiza um recurso existente com os dados fornecidos.
- **DELETE**: Remove um recurso do servidor.

Responses (Respostas)

Após receber e processar uma requisição, o backend envia uma resposta (response) ao frontend. Esta resposta pode conter os dados solicitados ou informações sobre o resultado da operação.

Estrutura de uma Resposta

Uma resposta HTTP inclui:

- **Status Code**: Indica o resultado da requisição (veremos mais detalhes abaixo).
- **Headers**: Informações adicionais sobre a resposta.
- **Body**: Dados retornados, geralmente em formato JSON ou XML.

Exemplo de Resposta JSON

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "message": "Login bem-sucedido",
  "token": "abcdef123456"
}
```

Status HTTP

Os status HTTP são códigos numéricos que indicam o resultado de uma requisição HTTP. Eles são divididos em faixas, cada uma indicando uma classe geral de resposta. Abaixo, vamos explorar as faixas de status mais comuns e exemplos de cada uma.

Faixas de Status

1xx (Informational - Informativo)

- **Descrição**: Indica que a requisição foi recebida e está sendo processada.

- **Exemplos**:
    - 100 Continue: O servidor recebeu os cabeçalhos da requisição e o cliente pode enviar o corpo da requisição.
    - 101 Switching Protocols: O servidor concorda em mudar o protocolo solicitado pelo cliente.

2xx (Success - Sucesso)

- **Descrição**: Indica que a requisição foi bem-sucedida.
- **Exemplos**:
    - 200 OK: A requisição foi bem-sucedida.
    - 201 Created: O recurso foi criado com sucesso.
    - 204 No Content: A requisição foi bem-sucedida, mas não há conteúdo para retornar.

3xx (Redirection - Redirecionamento)

- **Descrição**: Indica que é necessário tomar uma ação adicional para completar a requisição.
- **Exemplos**:
    - 301 Moved Permanently: O recurso foi movido permanentemente para uma nova URL.
    - 302 Found: O recurso foi encontrado em uma nova URL temporariamente.

4xx (Client Error - Erro do Cliente)

- **Descrição**: Indica que ocorreu um erro por parte do cliente na requisição.
- **Exemplos**:
    - 400 Bad Request: A requisição está malformada.
    - 401 Unauthorized: O cliente não está autorizado a acessar o recurso.
    - 403 Forbidden: A requisição contém dados válidos mas o servidor está negando realizar a ação.
    - 404 Not Found: O recurso solicitado não foi encontrado.
    - 405 Method Not Allowed: O método HTTP utilizado no endpoint não tem efeito.

5xx (Server Error - Erro do Servidor)

- **Descrição**: Indica que ocorreu um erro por parte do servidor ao processar a requisição.
- **Exemplos**:
    - 500 Internal Server Error: Ocorreu um erro interno no servidor.
    - 503 Service Unavailable: O servidor não está disponível no momento.

Processo de Login

1. **Usuário Inicia o Login**: O usuário acessa a página de login da aplicação.
2. **Envio dos Dados**: Ao preencher o formulário, o frontend envia os dados de login para o backend.
3. **Validação dos Dados**: O backend valida as credenciais do usuário. Se as credenciais existirem no banco de dados e estiverem corretas, o backend autentica o usuário e gera um token de acesso. Se as credenciais estiverem incorretas, o backend retorna um erro.
4. **Resposta do Backend**: O backend envia uma resposta ao frontend. Se o login for bem-sucedido, a resposta inclui um status 200 OK e o token de acesso, opcionalmente com uma mensagem no body da requisição. Se o login falhar, a resposta inclui um status 401 Unauthorized.
5. **Armazenamento do Token**: O frontend armazena o token de acesso localmente (por exemplo, em cookies ou localStorage).
6. **Redirecionamento**: O frontend redireciona o usuário para a página inicial da aplicação.
7. **Acesso aos Recursos Protegidos**: O frontend inclui o token de acesso em cada requisição subsequente para acessar recursos protegidos pelo backend. Este é um exemplo simplificado do processo de login em uma aplicação web. O uso de tokens de acesso ajuda a garantir a segurança das informações transmitidas entre o frontend e o backend.

Clique duas vezes (ou pressione "Enter") para editar