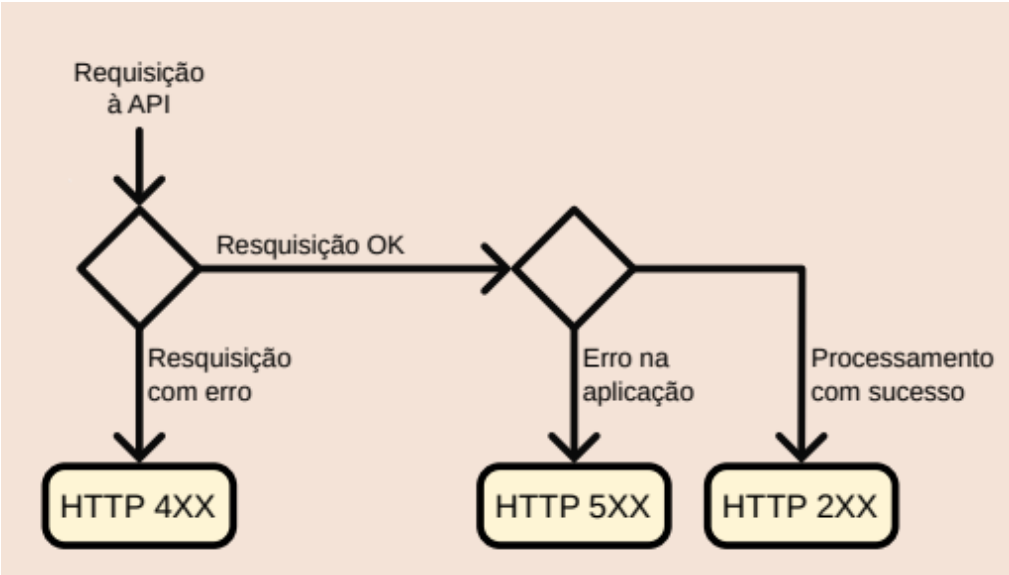
API Rest

Preâmbulo

Este documento descreve a API entre os diversos clientes e o servidor usando o padrão REST sob o protocolo HTTP. Esta API utilizará um sistema proprietário de autenticação através de tokens e um sistema de requisições e respostas em formato JSON. A padronização visa uma melhor integração entre sistemas. As requisições utilizam o protocolo HTTP com respostas que respeitam os códigos de status de retorno. O fluxo segue o diagrama abaixo:



Considerando a RFC-2616, basicamente, os principais códigos de retornos quando o assunto seria API Rest rodando sobre o protocolo HTTP são 2xx para retornos de sucesso, 4xx para processamento não efetuado por conta da request e 5xx para processamento de erro interno na aplicação ou servidor.

Abaixo segue uma pequena tabela com um resumo do que observamos no mercado:

Faixa	Tipo	Caso de uso
1xx	Informacional	Não utilizado para API Rest
2xx	Sucesso	Usado para indicar situação de processamento aceito e concluído com sucesso
3xx	Redirecionamento	Não utilizado para API Rest
4xx	Erro no lado do cliente	Erro ocasionado pela requisição “incorreta” efetuada pelo cliente
5xx	Erro no lado do servidor	Erro impulsionado por qualquer falha provocada por bug na aplicação, falha na dependência ou erro no servidor da aplicação

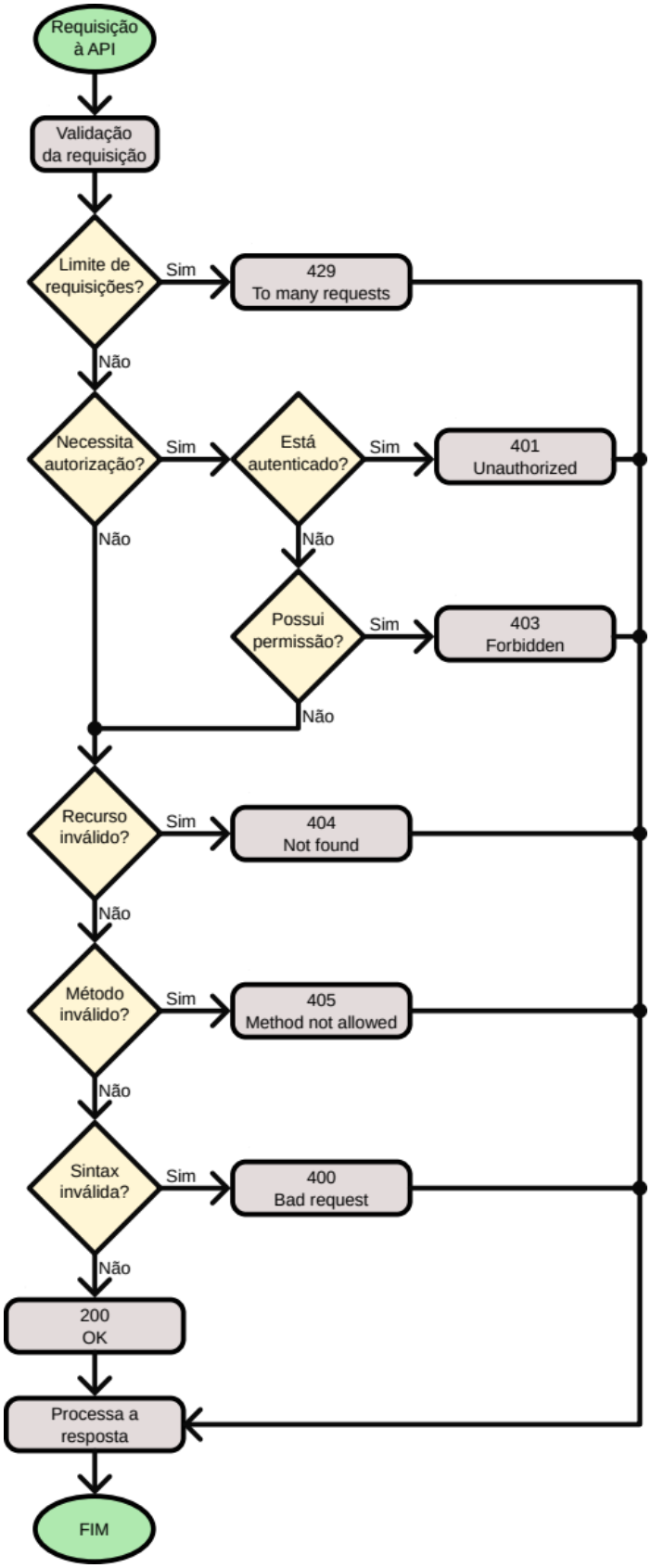
Os códigos de status HTTP retornáveis por esta API serão:

Status HTTP	Descrição
200	Indica que a solicitação foi bem-sucedida
202	Indica que a solicitação foi aceita para processamento, mas o processamento não foi concluído. A solicitação pode ou não ser eventualmente atendida, ou até mesmo ser desaprovada quando o processamento ocorrer. Esta resposta é normalmente usada para ações que demoram muito para serem processadas (envio pra uma fila, agendamento e etc). Sua finalidade é permitir que um servidor aceite uma solicitação para algum outro processo (talvez um processo orientado a lote que seja executado apenas uma vez por dia) sem exigir que a conexão do agente do usuário com o servidor persista até que o processo seja concluído. A entidade retornada com esta resposta DEVE incluir uma indicação do status atual da solicitação e um ponteiro para um monitor de status (local da fila de tarefas) ou alguma estimativa de quando o usuário pode esperar que a solicitação seja atendida.
400	A solicitação não pôde ser compreendida pelo servidor devido à sintaxe ou parâmetros incorretos. O cliente NÃO DEVE repetir o pedido sem modificações. No resultado são incluídos informações que detalham o erro.
401	Indica que a solicitação requer informações de autenticação do usuário. A resposta contém detalhado se o TOKEN informado ainda é válido, se o mesmo tiver sido informado, ou se será necessário requisitar um novo através de nova autenticação.
403	Solicitação não autorizada. O cliente não tem direitos de acesso ao conteúdo requisitado. Ao contrário do 401, a identidade do cliente é conhecida pelo servidor.
404	O servidor não consegue encontrar o recurso solicitado.
405	O método HTTP de solicitação (GET, POST, PUT, etc) é conhecido pelo servidor, mas foi desabilitado e não pode ser usado para esse recurso.
429	O usuário enviou muitas solicitações em um determinado período de tempo (“limitação de taxa”). Neste caso na resposta será informado, se possível, o tempo de espera para uma próxima requisição.
500	Em caso de erro no aplicativo. A resposta contém um detalhamento do erro ocorrido.

Os métodos HTTP suportados por esta API serão:

Método	Descrição
GET	Solicita informações através do recurso especificado. Esta solicitação apenas recupera dados e não têm outro efeito.
PUT	Solicita que a entidade seja armazenada (gravada). Se o URI se referir a um recurso existente, ele será modificado; se o URI não apontar para um recurso existente, a solicitação será rejeitada.
POST	Solicita que o servidor aceite a entidade incluída na solicitação como uma nova instância do recurso identificado pelo URI
DELETE	O método DELETE simplesmente remove o recurso especificado (se existir).

Lógica de processamento da requisição:



As respostas de cada requisição respeitará a seguinte extrutura do formato JSON:

```
{
  "error": "NONE",
  "message": "Usuário autenticado com sucesso",
  "data": {

  }
}
```

O campo error pode conter um dos seguintes valores:

ACCEPTED	A requisição foi aceita para processamento posterior
AUTHENTICATED	O usuário foi autenticado
AUTHORIZED	O usuário foi autorizado a acessar o recurso
CREATED	A entidade cujos valores foram passados foi criada
DELETED	A entidade cujo ID foi informado foi removida
ERROR	A requisição possui algum erro.
FOUND	A consulta retornou dados. Os dados são passados em ‘data’
NOT_ACCEPTED	A requisição não foi aceita para processamento posterior
NOT_AUTHENTICATED	O usuário não foi autenticado
NOT_AUTHORIZED	O usuáiro não foi autorizado à acessar o recurso
NOT_CREATED	A entidade não foi criada
NOT_DELETED	A entidade não foi apagada
NOT_FOUND	A pesquisa não retornou valores
NOT_UPDATED	A entidade cujos valores passamos para atualizar não foi modificada
NOT_VALID	Os valores da entidade informada não são válidos
PROCESSING	A requisição ainda está sendo processada
SUCCESS	A requisição foi executada com sucesso
UPDATED	A entidade cujos valores passamos foi atualizada
VALID	Os valores da entidade informada são válidos

Em message são detalhados os erros ocorridos e/ou mensagens de informação do resultado da solicitação.

Em data são informados os dados retornados, quando necessário.

As requisições seguirão o seguinte padrão:

Nos cabeçalhos deveremos enviar sempre:

- O cliente para o qual estamos fazendo à requisição (contratante)
- O TOKEN obtido na autenticação para identificar o usuário que fez a requisição

Caso a requisição necessite de algum dado adicional, o mesmo será informado no corpo da requisição (quando os métodos forem POST e PUT) e ou na própria URL (quando os métodos forem GET e DELETE).