**Requisitos Funcionais**

1. **Cadastro de Pauta**
   * A solução deve permitir o cadastro de uma nova pauta.
   * Cada pauta será identificada por um título e uma descrição.
2. **Abertura de Sessão de Votação**
   * A API deve permitir a abertura de uma sessão de votação para uma pauta.
   * A sessão de votação ficará aberta por um tempo determinado na chamada ou por 1 minuto por padrão.
   * Se o tempo for ultrapassado, a sessão de votação será fechada automaticamente.
3. **Votação pelos Associados**
   * Cada associado poderá votar em uma pauta apenas uma vez.
   * O voto pode ser "Sim" ou "Não".
   * Cada associado será identificado por um **ID único**.
   * A API deve permitir que o voto seja recebido para uma pauta específica.
4. **Contabilização dos Votos**
   * A API deve contabilizar os votos e fornecer o resultado da votação para cada pauta (quantidade de votos "Sim" e "Não").
   * Quando a votação terminar (após o tempo de duração), a API deve retornar o resultado final da votação.
5. **Validação de CPF para Votação**
   * A API deve integrar uma fachada ou cliente falso que valida se um CPF é válido ou não.
   * Se o CPF for inválido, a API deve retornar **404** com a mensagem: "status": "UNABLE\_TO\_VOTE".
   * Se o CPF for válido, a API deve retornar: "status": "ABLE\_TO\_VOTE".
6. **Exceções de Tempo de Votação**
   * A API deve tratar exceções de tempo de votação.
   * Caso um associado tente votar fora do período estipulado para a votação, a API deve retornar um erro adequado.
7. **Persistência de Dados**
   * As pautas, sessões de votação e votos devem ser persistidos, garantindo que não se percam após o reinício da aplicação.
   * O banco de dados deve ser utilizado para armazenar essas informações.
8. **Desempenho e Escalabilidade**
   * A aplicação deve ser capaz de lidar com cenários de alto volume de votos sem comprometer a performance.
   * Testes de performance devem ser realizados para garantir que a aplicação se comporte bem em cenários com centenas de milhares de votos.
9. **Versionamento da API**
   * A API deve ser versionada para garantir compatibilidade entre versões antigas e novas.
   * A estratégia de versionamento pode ser baseada em prefixos na URL (por exemplo, /v1/)

**Requisitos Não Funcionais**

1. **Manutenibilidade e Legibilidade**
   * O código deve ser bem estruturado e seguir boas práticas de programação, como uso de nomes de variáveis claros, funções pequenas e bem definidas, e comentários explicativos quando necessário.
2. **Segurança e Autorização**
   * Embora a segurança das interfaces seja abstraída para este exercício, é importante garantir que em um cenário real as interações com a API sejam seguras.
3. **Escalabilidade e Performance**
   * A aplicação deve ser projetada para suportar grandes volumes de dados e múltiplos acessos simultâneos, com otimização no processamento de votos e respostas rápidas.
4. **Documentação**
   * A API deve ser bem documentada, incluindo exemplos de requisições e respostas, para facilitar o entendimento e o uso pelos desenvolvedores.
5. **Logs da Aplicação**
   * A aplicação deve gerar logs apropriados, especialmente em situações de erro ou exceção, para facilitar a depuração e o monitoramento da aplicação.
6. **Comunicação em JSON**
   * Toda comunicação entre o servidor e o cliente será feita através de mensagens no formato **JSON**, com especificação clara de como as requisições e respostas serão estruturadas.
7. **Tratamento de Erros e Exceções**
   * A aplicação deve tratar erros de forma robusta, retornando códigos de status HTTP adequados e mensagens claras de erro, para garantir uma boa experiência para o desenvolvedor e usuário.

**Banco de Dados**

O banco de dados pode ser modelado com as seguintes entidades principais:

1. **Tabela pautas**
   * **id** (Chave primária): Identificador único da pauta.
   * **titulo**: Título da pauta.
   * **descricao**: Descrição da pauta.
   * **data\_criacao**: Data e hora em que a pauta foi criada.
2. **Tabela votacoes**
   * **id** (Chave primária): Identificador único da sessão de votação.
   * **id\_pauta** (Chave estrangeira): Relaciona a votação com uma pauta específica.
   * **data\_inicio**: Data e hora de início da sessão de votação.
   * **data\_fim**: Data e hora de término da sessão de votação.
   * **tempo\_votacao**: tempo de votação ou usar tempo default
   * **status**: Status da sessão de votação (aberta, fechada, etc.).
3. **Tabela votos**
   * **id** (Chave primária): Identificador único do voto.
   * **id\_votacao** (Chave estrangeira): Relaciona o voto com uma sessão de votação.
   * **id\_associado**: Identificador único do associado (pode ser CPF ou outro identificador único).
   * **voto**: O voto do associado ("Sim" ou "Não").
   * **data\_voto**: Data e hora em que o voto foi registrado.
4. **Tabela associados**
   * **id\_associado** (Chave primária): Identificador único do associado.
   * **cpf**: CPF do associado.
   * **nome**: Nome do associado.
   * **status\_voto**: Status do associado em relação à votação (se pode ou não votar).