Optei por utilizar o Docker para empacotar a aplicação em contêineres, garantindo portabilidade e consistência do ambiente de desenvolvimento para produção. Essa escolha permite que a aplicação seja executada de forma isolada e independente do ambiente subjacente.

Para o banco de dados, escolhi o PostgreSQL devido à sua confiabilidade, desempenho e robustez, além da facilidade de integração com o Spring Boot. A compatibilidade com o Docker também foi um fator determinante, facilitando a configuração e o gerenciamento do banco de dados em contêineres.

Utilizei o Flyway para o versionamento do banco de dados. O Flyway é uma ferramenta de migração de banco de dados que permite controlar e versionar as alterações no esquema do banco de dados de forma automatizada. Com o Flyway, podemos gerenciar as alterações no banco de dados de maneira consistente e reproduzível em diferentes ambientes.

No Dockerfile, configurei duas etapas: uma para compilar e construir a aplicação Java usando Maven e outra para empacotar a aplicação compilada em uma imagem Docker. Isso permite criar imagens mais leves e otimizadas, reduzindo o tempo de implantação e o tamanho dos contêineres.

Por fim, expus as portas necessárias para acessar a aplicação, como a porta 8080 para a API REST. Essa configuração é feita tanto no Dockerfile quanto no Docker Compose, garantindo que a aplicação seja acessível externamente.

Essas escolhas foram feitas considerando os requisitos do projeto e as melhores práticas de desenvolvimento, visando criar uma solução robusta, escalável e de fácil manutenção.

Versionamento da API

Para versionamento da API, adotei o padrão de versionamento na rota, utilizando o prefixo "/v1" para todas as rotas. Isso permite que diferentes versões da API coexistem no mesmo servidor, facilitando a manutenção e a evolução da API ao longo do tempo.

Ao adotar esse padrão, as rotas relacionadas às pautas terão URLs como "/v1/pautas" para operações CRUD relacionadas às pautas. Se futuras versões da API forem introduzidas, elas poderão seguir um padrão semelhante, como "/v2/pautas", mantendo a consistência e a organização das rotas.

Essa abordagem de versionamento na rota oferece clareza e transparência na comunicação da API, permitindo que os desenvolvedores identifiquem facilmente a versão da API que estão utilizando e adaptem suas integrações de acordo com as mudanças. Além disso, ela evita conflitos de nomeação e facilita a manutenção de diferentes versões da API.