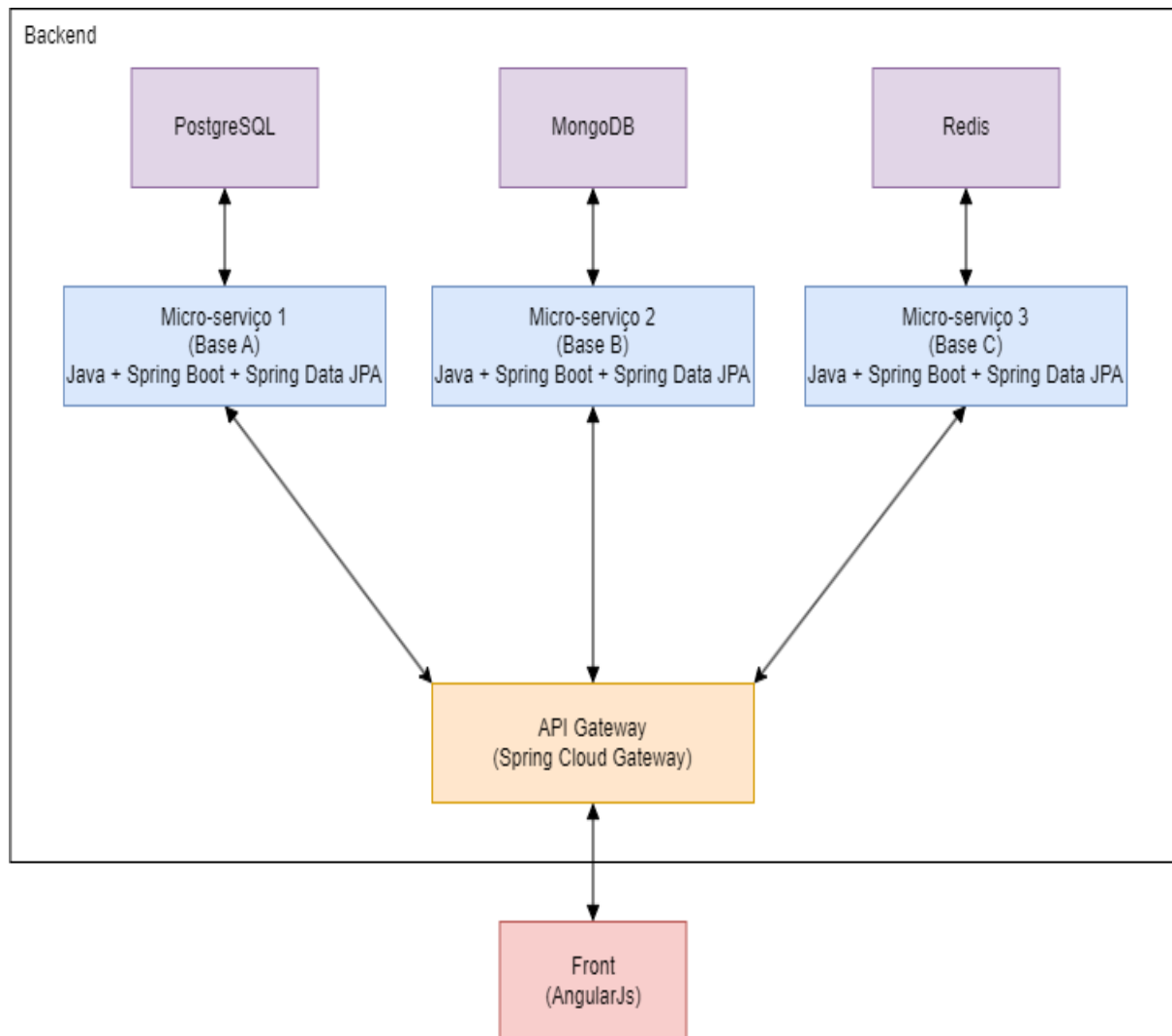


Soluções / Tecnologias para o problema

Com a arquitetura a seguir, teremos um sistema escalável, com segurança e desempenho adequados para cada uma das bases de dados. O API Gateway e os micro-serviços garantem a integração entre as diferentes partes do sistema, enquanto o front-end garante uma experiência de usuário agradável e intuitiva.



Armazenamento

Ao usar essa estrutura de armazenamento, é possível armazenar os dados em diferentes tecnologias de banco de dados, otimizadas para atender às necessidades específicas dos dados armazenados. Isso pode resultar em melhor segurança, desempenho e escalabilidade do sistema como um todo.

Usaremos três bancos de dados:

- Base A: PostgreSQL

PostgreSQL é usado para dados altamente confiáveis e críticos. Pode ser configurado com várias camadas de segurança, incluindo autenticação avançada e criptografia de dados. Além disso, o PostgreSQL é altamente escalável e pode ser configurado para suportar cargas de trabalho pesadas.

- Base B: MongoDB

Podemos usar o MongoDB devido à sua flexibilidade e capacidade de suportar operações de leitura/gravação rápidas. O MongoDB é ideal para armazenar dados não estruturados ou semiestruturados, como documentos JSON. Ele também suporta operações de leitura/gravação rápidas e pode ser facilmente escalável usando clusters distribuídos.

- Base C: Redis

A Base C pode ser armazenada no Redis como um cache em memória para melhorar a velocidade de acesso aos dados. O Redis é ideal para armazenar dados que precisam ser acessados rapidamente e com baixa latência, como chaves e valores simples. Ele suporta várias estruturas de dados em memória, incluindo listas, conjuntos e mapas.

Back End / Micro-serviços

A arquitetura é baseada em micro-serviços, implementados em Java (Spring Boot, Spring Data JPA), com o objetivo de fornecer escalabilidade, manutenção e desenvolvimento mais fácil. Os micro-serviços são expostos como APIs RESTful e podem ser implantados em contêineres como Docker.

Temos 3 micro-serviços:

Micro-serviço 1: Acesso à Base A

Esse micro-serviço será responsável por acessar o PostgreSQL e retornar informações como CPF, nome, endereço e lista de dívidas.

Por ser uma base com dados mais sensíveis e que requer um nível maior de segurança, utilizaremos o Spring Security que oferece suporte a diferentes tipos de autenticação, como login e senha, token de autenticação, integração com provedores de autenticação externos, etc.

Micro-serviço 2: Acesso à Base B

Esse micro-serviço será responsável por acessar o MongoDB e retornar informações como idade, lista de bens, endereço e fonte de renda. Além disso, esse serviço também será utilizado para extração de dados por meio de algoritmos de aprendizado de máquina.

Micro-serviço 3: Acesso à Base C

Esse micro-serviço será responsável por acessar o Redis e rastrear eventos relacionados a um CPF, como a última consulta em um Bureau de Crédito, movimentação financeira e última compra com cartão de crédito.

Deve ser configurado com uma latência extremamente baixa para suportar o rastreamento de eventos em tempo real e, para isso, podemos usar o Spring Cloud

Stream, que oferece recursos de processamento de fluxo para dados em tempo real.

API Gateway

Para gerenciar as requisições e a comunicação entre os micro-serviços e o front-end, usaremos o Spring Cloud Gateway que é fácil de configurar e implantar.

Ele gerencia o tráfego e protege as APIs dos micro-serviços. Fornece roteamento dinâmico, balanceamento de carga, controle de acesso, autenticação e alto desempenho.

Configuramos as dependências, a rota para cada micro-serviço e criamos uma classe principal para executar o Spring Cloud Gateway.

Front End

Usaremos o AngularJS para consumir as APIs fornecidas pelos micro-serviços. Características:

- Arquitetura MVC (Model-View-Controller): Permite que o código seja organizado de maneira mais eficiente e fácil de manter.
- Comunicação de dados bidirecional: Como a arquitetura do AngularJS combina JavaScript e HTML, os códigos de ambos já estão sincronizados. Por isso, o framework poupa muito tempo dos desenvolvedores na hora da criação de aplicações e sites.
- Injeção de dependência: Para gerenciar as dependências do aplicativo. Isso torna o código mais modular e fácil de testar.

- Diretivas: Permitem criar elementos personalizados e reutilizáveis na interface do usuário. Isso ajuda a reduzir a quantidade de código e aumenta a legibilidade.
- Performance: O AngularJS é um framework de alto desempenho que permite criar aplicativos web rápidos e responsivos.
- Compatibilidade com Desktop e Mobile – O AngularJS pode ser executado na maioria dos navegadores. E não apenas em computadores pessoais, mas também em dispositivos móveis. Flexibilidade e versatilidade são grandes características da ferramenta.