Relatório da Arquitetura do Projeto – Back-End

Para o desenvolvimento deste projeto, optei por utilizar o **Firebase** como solução de back-end e banco de dados, junto com **JavaScript** no front-end. A escolha do Firebase foi estratégica, pois ele oferece várias vantagens que tornam o desenvolvimento mais rápido e prático, principalmente para projetos web de porte pequeno a médio.

1. Escolha do Firebase

O Firebase foi utilizado por meio de **CDN**, o que significa que não precisei instalar nada localmente no servidor ou criar uma infraestrutura complexa. Ele oferece:

Autenticação pronta: Com o Firebase Auth, é possível gerenciar usuários e sessões sem precisar criar toda a lógica de login e senha do zero.

Banco de dados em tempo real: O Firebase permite salvar dados do usuário, como transações, metas e projeções, e recuperar essas informações de forma rápida.

Segurança e escalabilidade: O Firebase já possui regras de segurança que podem ser configuradas para cada coleção, garantindo que cada usuário acesse apenas seus próprios dados.

Essa escolha facilita a manutenção do projeto e diminui o risco de problemas de segurança comuns em bancos de dados tradicionais.

2. Uso do JavaScript

Todo o **front-end é feito com Html/Css e JavaScript**, que se comunica diretamente com o Firebase via CDN. Essa abordagem permite:

Interatividade imediata: O JavaScript permite atualizar a interface do usuário em tempo real, sem precisar recarregar a página.

Simplicidade no projeto: Como não há servidor back-end tradicional, o JavaScript no cliente faz as requisições ao Firebase, economizando tempo e recursos.

Integração direta: A manipulação do DOM para exibir dados como transações, relatórios e projeções financeiras é feita de forma direta, tornando o projeto mais responsivo e fácil de usar.

3. Estrutura de dados

O projeto organiza os dados em **coleções no Firebase**, separadas por tipo de informação, como transacoes, projecoes, relaorios e usuarios. Isso garante que cada funcionalidade tenha seus dados próprios, facilitando consultas e mantendo tudo organizado.

Por exemplo:

transacoes: armazena todas as receitas e despesas do usuário.

projecoes: guarda os cálculos de projeções financeiras.

relatorios: mantém registros dos relatórios gerados pelo usuário.

Essa organização torna o projeto escalável, ou seja, se no futuro houver necessidade de adicionar novas funcionalidades, será fácil integrar novas coleções.

4. Justificativa geral

A combinação **Firebase e o JavaScript** foi escolhida por ser prática, segura e rápida para desenvolver uma aplicação web interativa. Ela elimina a necessidade de configurar um servidor completo e ainda permite salvar e recuperar dados do usuário em tempo real. Para o tipo de projeto que estou desenvolvendo, que é uma **agenda financeira e simulador de projeções**, essa arquitetura atende perfeitamente às necessidades de funcionalidade.