Sistema

O sistema inicialmente foi desenvolvido visando a funcionalidade de poder gerenciar seus endereços, porém, sabemos que os sistemas crescem drasticamente ao longo do tempo, e por isso, foi modelado e arquitetado para suportar possíveis mudanças ou aumento de consumo.

A arquitetura implantada foi a MVC(Model, View, Control), uma arquitetura que visa separar as responsabilidades de cada parte do sistema, facilitando manutenção e compreensão do código.

A transmissão de dados é executada por JSON em protocolo HTTP baseado em RestFull.

Como espinha dorsal do sistema, foi usada uma tecnologia bem atualizada no mercado, SpringBoot. Ela visa executar o projeto de forma mais eficaz possível e evitar complicações por diversas anotações que as dependências nos oferecem.

Em uma maneira geral, o código fica muito mais limpo, economizando muitas linhas de ,comando e evitando possiveis bugs.

O banco de dados é o MySql auxiliado ao JPA Hibernate que é responsável por aplicar as persistências na camada de acesso a dados, ele também efetua toda a mapeação das entidades diretamente no sistema, dessa forma, não é necessário scripts de tabelas e etc, através das anotações, à primeira instância que for iniciado o Hibernate efetuará todos os Creates necessários(Tabelas, Sequences)

Para a injeção de dependências, foi utilizado o Maven Repository, uma ferramenta que poupa muito o trabalho do desenvolvedor, basta anotar o que deseja no pom.xml.

Foi usando também Mock, pode-se entendê-lo como uma mapeação de todos os possiveis erros de cada camada da aplicação, programado! Antes de efetuar o deploy, o sistema roda essa classe de testes que irá efetuar todas as verificações anotadas, evitando drasticamente erros em produção.

Na parte de FrontEnd, foi usado AngularJS para fazer a intermediação do Cliente/Servidor com requisições Ajax auxiliado ao HTML para apresentar a interface para o cliente.

A parte de documentação ficou responsável pela biblioteca springfox, utilizando a ferramenta Swagger2 com o auxílio da swagger-ui. Uma rica biblioteca que gera a documentação de todo o projeto através de anotações, fornecendo diversos tipos de informações com uma interface bem agradável a quem necessita consumir a API.

Para acessa-lá: Basta entrar no caminho:

http://localhost:8080/swagger-ui.html

Plano de Desenvolvimento

Para realizar o projeto, foi utilizado o modelo Desenvolvimento Cascata, mapeado as regras de negócios e as necessidades do “cliente”, análise as possiveis tecnologias que poderiam alcançar o que o sistema necessita, codificação das unidades e entrega do projeto.

O Desenvolvimento partiu-se inicialmente do beckEnd, começando pela criação das entidades do banco, passando para a camada de negócios e criando o Controller da aplicação. Os testes de GET/PUT/DELETE/POST eram realizados através do postMan até que a camada do cliente estivesse pronta.

Assim que foi concluído, colocou-se em prática o esboço já pré-definido do template que atenderia as necessidades e iniciou-se o desenvolvimento do FrontEnd.

Após fechar o fluxo de desenvolvimento, reforçou-se a todas as funcionalidades pretendidas e regras de negócio aplicadas para verificar se obtinha o retorno o esperado.