

PROTÓTIPO DE UM GERADOR DE APLICAÇÕES WEB COM JHIPSTER

Aluno: Ingmar Schmidt de Aguiar

Orientador: Mauro Marcelo Mattos

Roteiro

- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos
- Especificação
- Operacionalidade da implementação
- Resultados e discussões
- Conclusões e sugestões

Introdução

- Em primeiro lugar este trabalho surgiu como um desafio para mim mesmo.
- Então surgiu a pergunta: e se eu criar uma aplicação web a partir de uma base de dados?
- OpenXava x JHipster

Objetivos

- O objetivo principal deste trabalho é desenvolver um protótipo de uma ferramenta de geração automática de uma aplicação web a partir da importação de modelos de bases de dados já existentes.

Objetivos Específicos

- Os objetivos específicos são:
 - conceber e implementar um modelo de mapeamento da estrutura de bancos de dados existentes para o modelo de estrutura JHipster;
 - desenvolver uma API de acesso às funcionalidades disponibilizadas pelo JHipster;
 - construir um conjunto de casos de testes para validar o projeto.

Fundamentação Teórica

- Geradores de código
- Geração de código baseada na técnica de scaffolding
- CRUD
- Linguagem de Domínio Específico (DSL)

Geradores de código

- mecanismo auxiliar ao processo de criação de software
- software que escreve software
- motivação: custo, tempo, produtividade

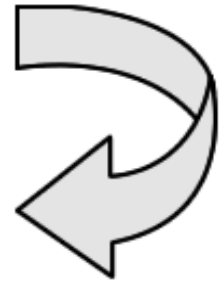
Geração de código baseada na técnica de scaffolding

- alusão à técnica de cimbramento (*scaffolding* em inglês) da engenharia civil
- geração da estrutura fundamental da aplicação
- geração de código funcional de um sistema
- chamados de *scaffolders*

Tradução de template em código fonte

```
1 public static Result edit({{tipoID}} id) {  
2     Form<{{classe}}> {{objeto}}Form =  
3         form({{classe}}.class)  
4             .fill({{classe}}).find.byId(id));  
5     return ok(editForm.render(id, {{objeto}}Form));  
6 }
```

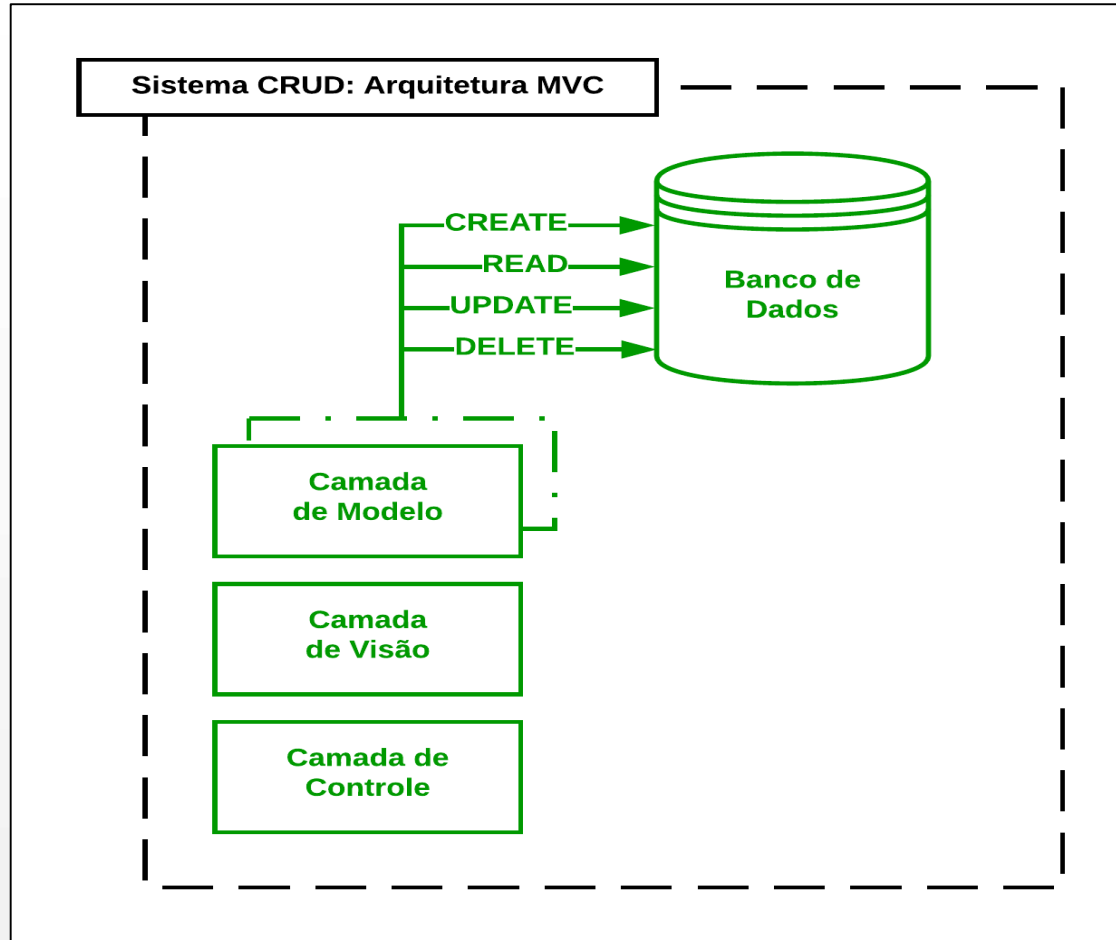
```
1 public static Result edit(Long id) {  
2     Form<Carro> carroForm =  
3         form(Carro.class)  
4             .fill(Carro).find.byId(id));  
5     return ok(editForm.render(id, CarroForm));  
6 }
```



CRUD

- aplicações utilizam sistemas gerenciadores de banco de dados
- necessidade de domínio de operações básicas de acesso aos dados
- criação, recuperação, atualização e remoção de registros
- acrônimo originado das palavras: *Create*, *Read*, *Update* e *Delete*

Estrutura básica de um sistema CRUD



Linguagem de Domínio Específico (DSL)

- linguagem de programação de computadores limitada, focada em um domínio (problema) específico
- classificadas em: interna (Rails) ou externa (desenvolvida)
- motivações: produtividade, criada a nível de negócio
- problemas: custo, podem crescer demasiadamente

JHipster

What Is JHipster?



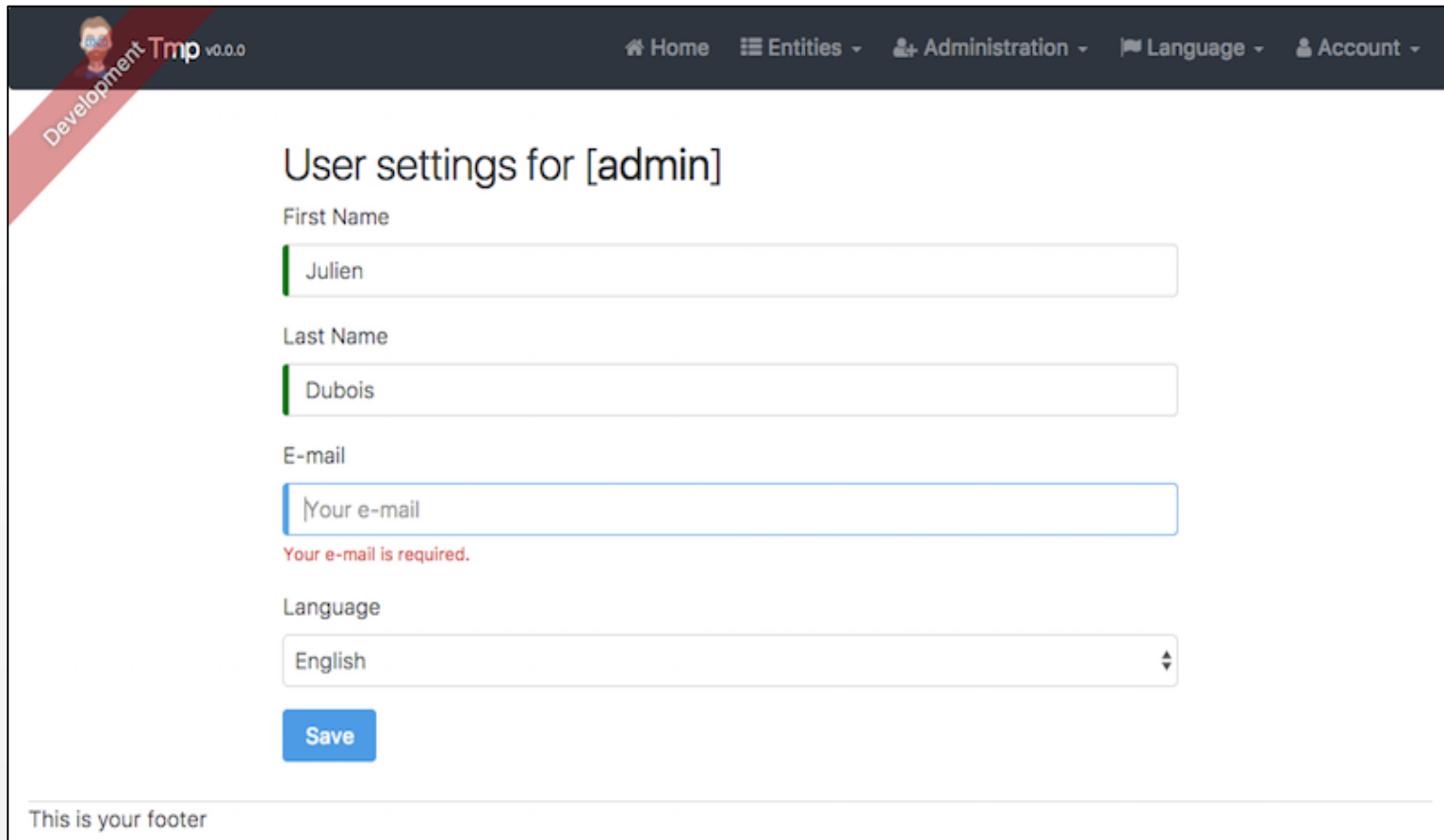
- plataforma de desenvolvimento para gerar, desenvolver e implantar aplicativos Web Spring Boot + Angular / React / Vue e microsserviços Spring

JHipster

Tem como objetivo gerar uma aplicação Web completa ou arquitetura de microsserviços, unificando:

- *stack* Java robusta e de alto desempenho com Spring Boot
- *frontend* elegante, moderno e responsivo
- arquitetura robusta de microsserviços

Exemplo de página gerada pelo JHipster



The screenshot shows a web application interface for user settings. At the top left, there is a logo for 'Development Tmp v0.0.0' with a small cartoon character. The top navigation bar includes links for 'Home', 'Entities', 'Administration', 'Language', and 'Account'. The main content area is titled 'User settings for [admin]' and contains several input fields: 'First Name' with the value 'Julien', 'Last Name' with the value 'Dubois', and 'E-mail' with the placeholder text 'Your e-mail'. Below the email field, a red error message states 'Your e-mail is required.'. The 'Language' field is a dropdown menu currently set to 'English'. A blue 'Save' button is positioned below the form fields. At the bottom of the page, there is a footer area with the text 'This is your footer'.

Development Tmp v0.0.0

Home Entities Administration Language Account

User settings for [admin]

First Name

Last Name

E-mail

Your e-mail is required.

Language

Save

This is your footer

Exemplo de página gerada pelo JHipster

Development Tmp v0.0.0

Home Entities Administration Language Account

Books

+ Create a new Book

ID	Title	Description	Publication Date	Price	Author	
2	Inferno	Inferno is a 2013 mystery thriller novel by American author Dan Brown and the fourth book in his Robert Langdon series, following Angels & Demons, The Da Vinci Code and The Lost Symbol.	May 14, 2013	45	Dan Brown	View Edit Delete
4	King Lear	King Lear is a tragedy written by William Shakespeare. It depicts the gradual descent into madness of the title character.	Dec 30, 1604	10	William Shakespeare	View Edit Delete
5	Gulliver's Travels	It is Swift's best known full-length work, and a classic of English literature.	Oct 26, 1726	15	Jonathan Swift	View Edit Delete

Showing 1 - 3 of 3 items.

« « 1 » »

JHipster

- a criação da aplicação é feita via linha de comando através do comando *jhipster*

```
1. jhipster (node)

JHIPSTER

https://www.jhipster.tech

Welcome to JHipster v5.2.1
Application files will be generated in folder: /Users/mraible/dev/21-points

-----

Documentation for creating an application is at https://www.jhipster.tech/creating-an-app/
If you find JHipster useful, consider sponsoring the project at https://opencollective.com/generator-jhipster

-----

? Which *type* of application would you like to create? Monolithic application (recommended for simple projects)
? What is the base name of your application? TwentyOnePoints
? What is your default Java package name? org.jhipster.health
? Do you want to use the JHipster Registry to configure, monitor and scale your application? No
? Which *type* of authentication would you like to use? JWT authentication (stateless, with a token)
? Which *type* of database would you like to use? SQL (H2, MySQL, MariaDB, PostgreSQL, Oracle, MSSQL)
? Which *production* database would you like to use? PostgreSQL
? Which *development* database would you like to use? H2 with disk-based persistence
? Do you want to use the Spring cache abstraction? Yes, with the Ehcache implementation (local cache, for a single node)
? Do you want to use Hibernate 2nd level cache? Yes
? Would you like to use Maven or Gradle for building the backend? Gradle
? Which other technologies would you like to use? Search engine using Elasticsearch
? Which *Framework* would you like to use for the client? Angular 6
? Would you like to enable *SASS* support using the LibSass stylesheet preprocessor? Yes
? Would you like to enable internationalization support? Yes
? Please choose the native language of the application English
? Please choose additional languages to install French
? Besides JUnit and Jest, which testing frameworks would you like to use? Gatling, Protractor
? Would you like to install other generators from the JHipster Marketplace? No

Installing languages: en, fr
  create package.json
  create README.md
```

JHipster Domain Language (JDL)

- DSL proprietária
- definições de configuração da aplicação
- definição das entidades e relacionamentos do banco de dados

FormGenerate

- trabalho de conclusão de curso de Maicon Klug (2007), com o objetivo de auxiliar o desenvolvedor no acesso ao banco de dados, e na geração de telas e CRUD
- gera os artefatos a partir de uma base de dados existente
- geração de código a partir de templates

OpenXava

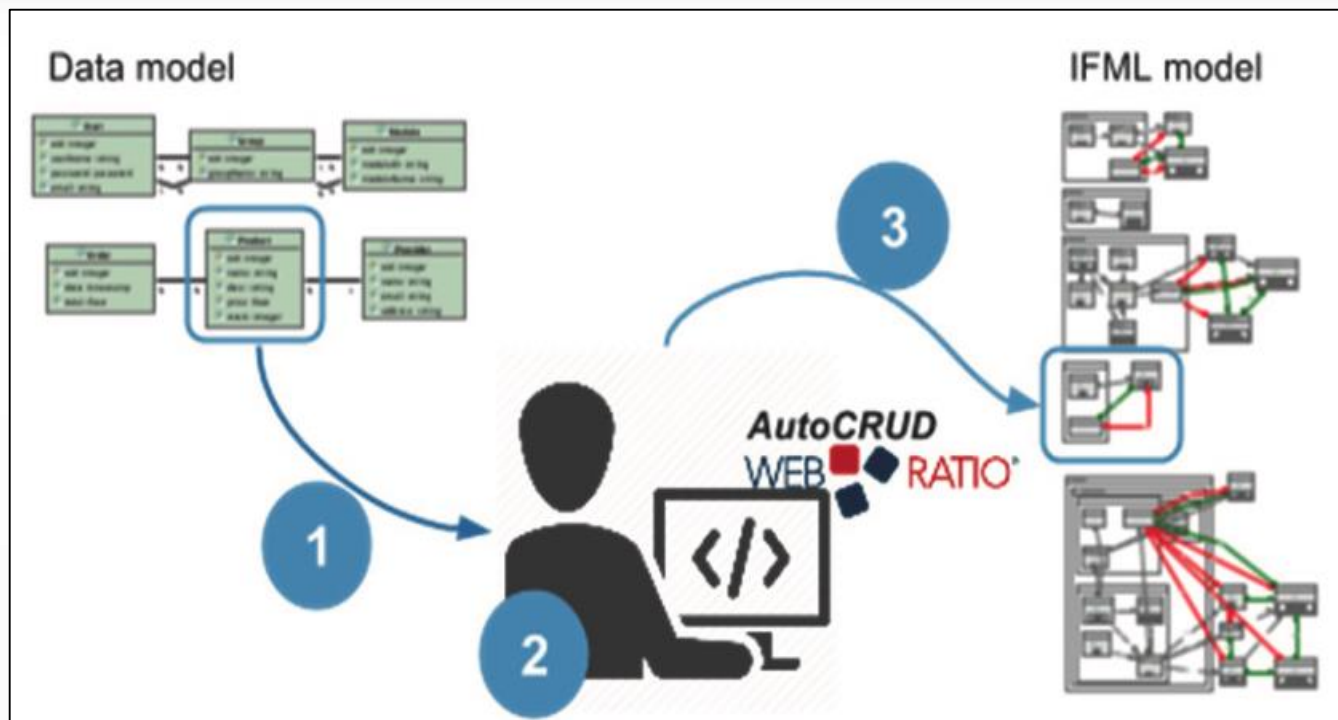
- OpenXava: *framework* para o desenvolvimento de aplicativos a partir de especificações em Java
- o desenvolvedor cria uma classe para a entidade e através de anotações proprietárias, JPA e JavaEE
- a classe é usada em tempo de execução

Flujo de funcionamiento OpenXava



AutoCRUD

- plugin usado dentro da ferramenta WebRatio
- auxilia na criação das operações CRUD dentro de uma aplicação gerada pelo WebRatio



- o desenvolvedor seleciona a entidade no modelo de dados
- define as operações (CRUD) desejadas
- fornece a ligação adequada para os parâmetros da operação CRUD (por exemplo, qual é o elemento a ser excluído).

Requisitos Funcionais

- O protótipo deverá permitir configurar o acesso a uma plataforma de banco de dados.
- O protótipo deverá permitir a importação dos dados das tabelas e relacionamentos do banco de dados.
- O protótipo deverá permitir o ajuste nos relacionamentos entre as tabelas (entidades).
- O protótipo deverá permitir realizar a conversão das tabelas e relacionamentos importados para a linguagem JDL.
- O protótipo deverá permitir a exportação e alteração da conversão gerada.

Requisitos Não Funcionais

- O protótipo deverá ser desenvolvido em Java.
- O protótipo deve suportar os bancos de dados: Postgresql, MySql, Sql Server e Oracle.
- O protótipo deve rodar em sistemas operacionais Windows e Linux e opcionalmente no Mac OS.
- Orientar o usuário no deploy do projeto e na utilização do modelo gerado.
- Chamar (opcional) o JHipster via interface ou linha de comando.

Diagrama de atividades

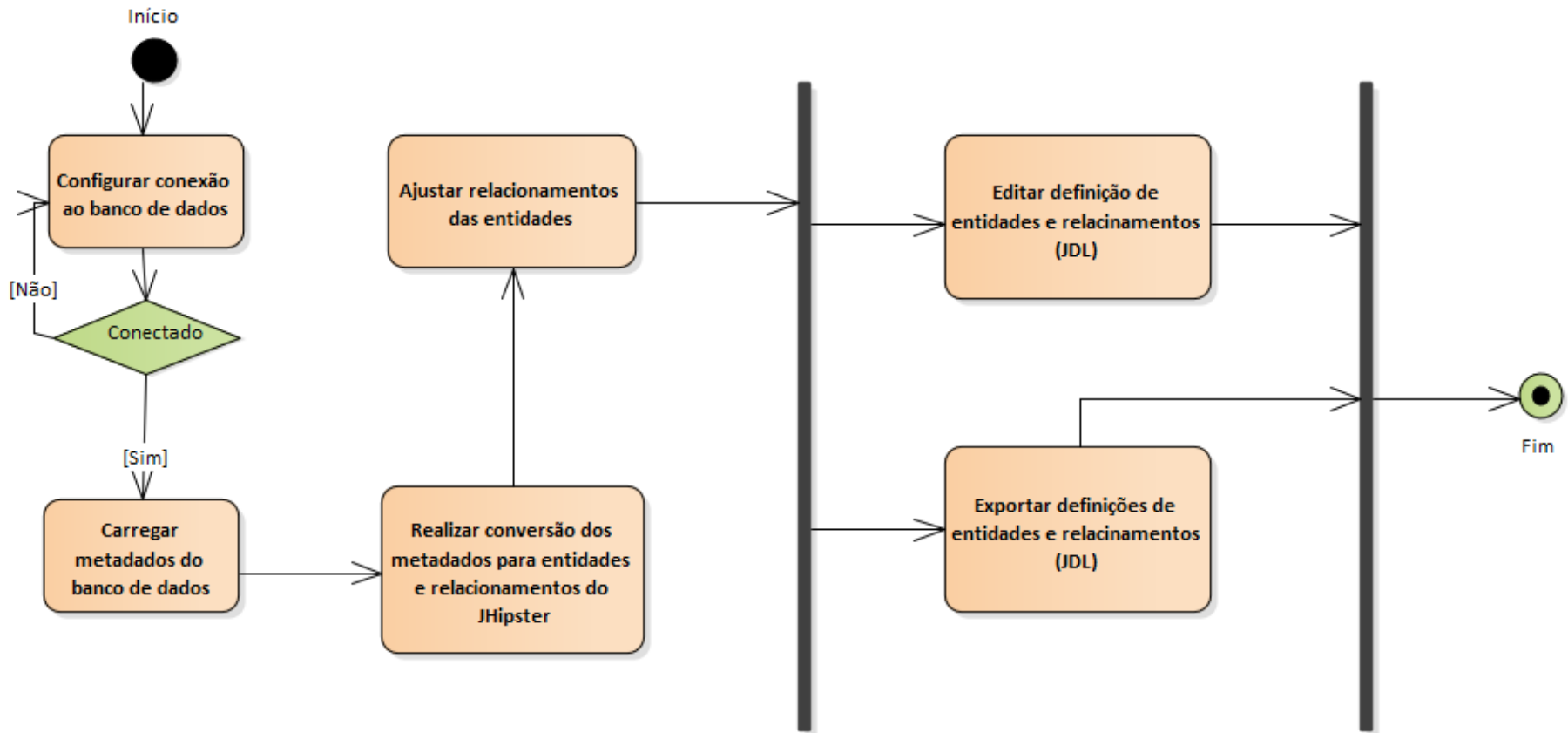


Diagrama de pacotes

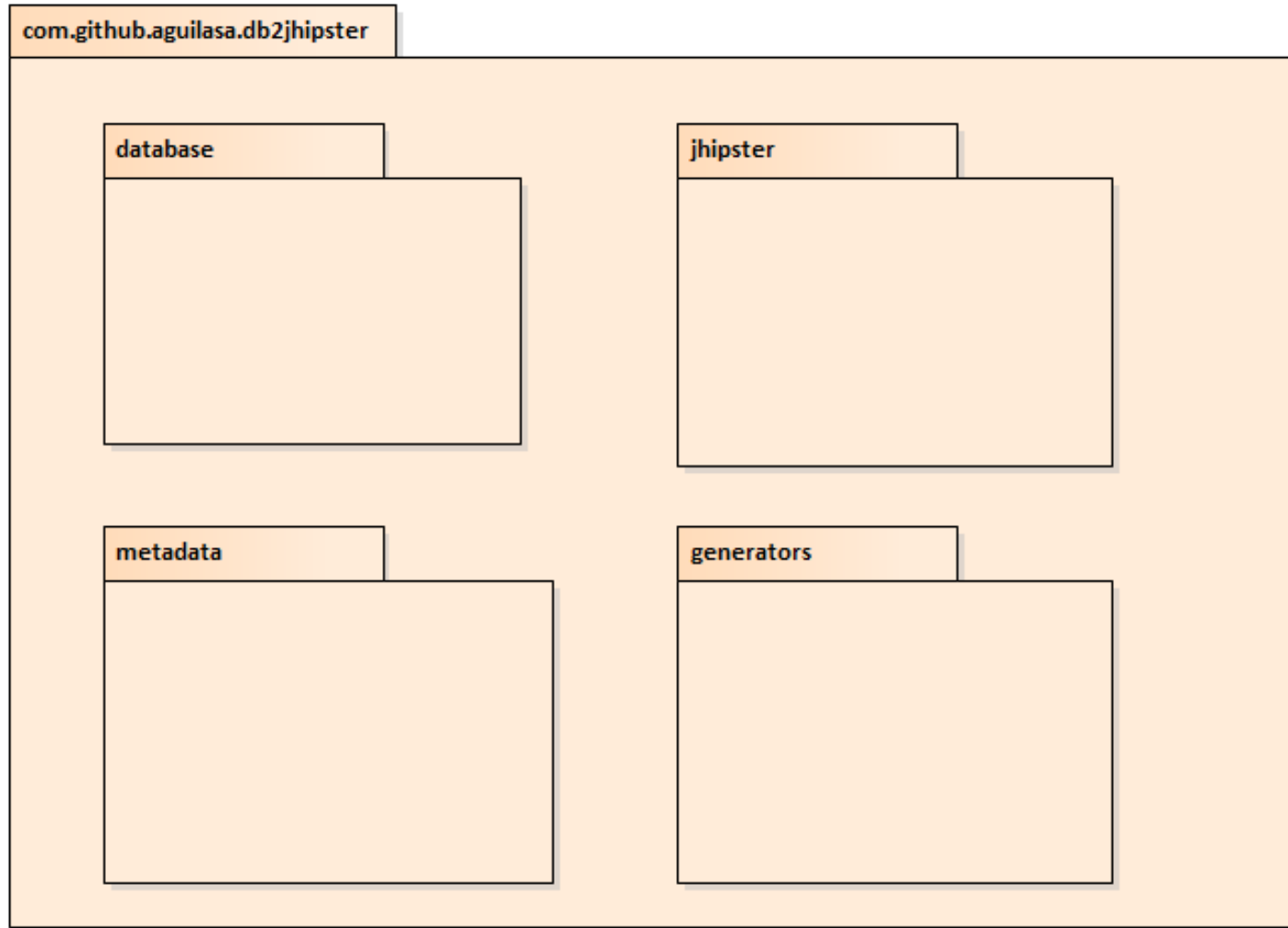
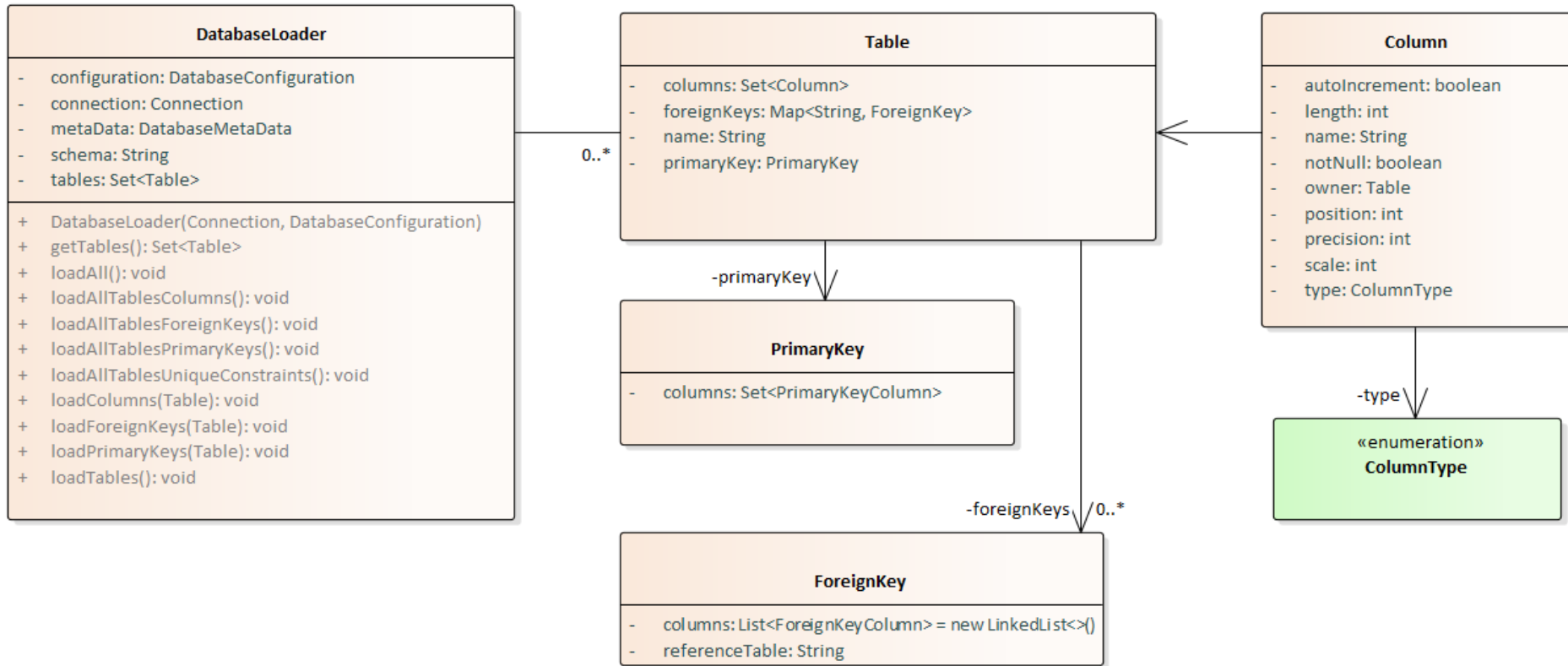
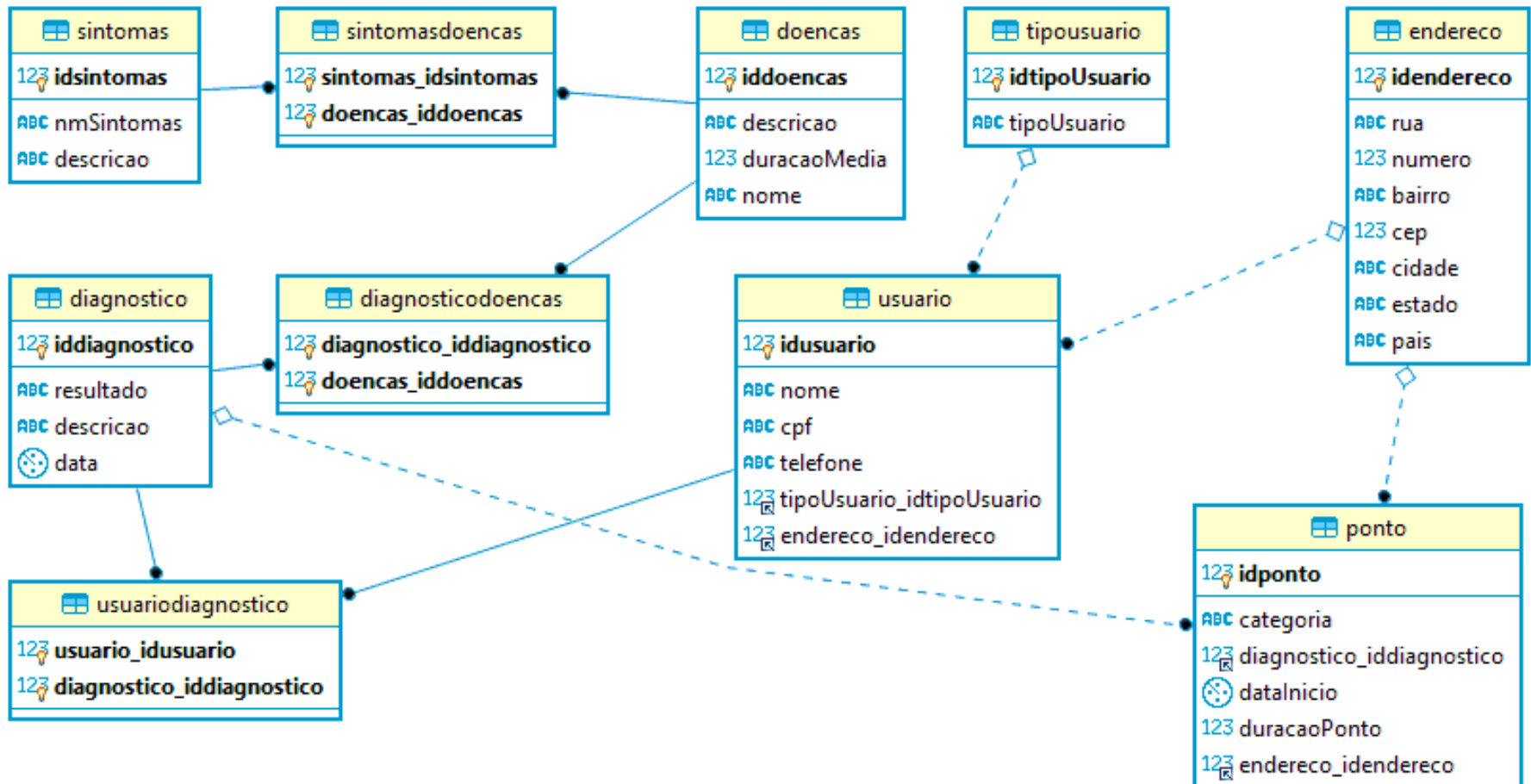


Diagrama de classes



MER



Implementação

- orientação a objetos.
- padrão factory.
- API JDBC: interface DatabaseMetaData.
- Apache Velocity Project.

Ferramentas



Visual Studio Code



Java™



DBBeaver



PostgreSQL

Tela inicial

DB2Hipster

DB to JHipster

MySQL
ORACLE
DATABASE

Microsoft
SQL Server

PostgreSQL

JHipster

Banco de Dados
PostgreSQL

Servidor
localhost

Porta
5432

Database
postgres

Schema
empresa

Usuário
postgres

Senha
●●●●●●●●

Testar conexão

Voltar Avançar

Tela final

DB2JHipster

DB to JHipster



JDL gerada

```
entity EmpresaEntity {
    nome String
}

entity FilialEntity {
    codfil Integer
    nome String
}

entity FuncionarioEntity {
    codfun Integer
    nome String
}

relationship OneToOne {
    FilialEntity{empresaFilial} to EmpresaEntity{filialEmpresa}
    FuncionarioEntity{filialFuncionario} to FilialEntity{funcione
```

JDL Studio

Salvar

Voltar

Avançar

Resultados e Discussões

Aplicação gerada pelo JHipster

Tcc jhipster vDEV

Home Entities Administration Language

Development

Welcome, JHipster!

This is your homepage

You are logged in as user "admin"

If you have any question on JHipster:

- [JHipster homepage](#)
- [JHipster on Stack Overflow](#)

- * Diagnostico Entity
- * Doencas Entity
- * Endereco Entity
- * Ponto Entity
- * Sintomas Entity
- * Tipousuario Entity
- * Usuario Entity

Resultados e Discussões

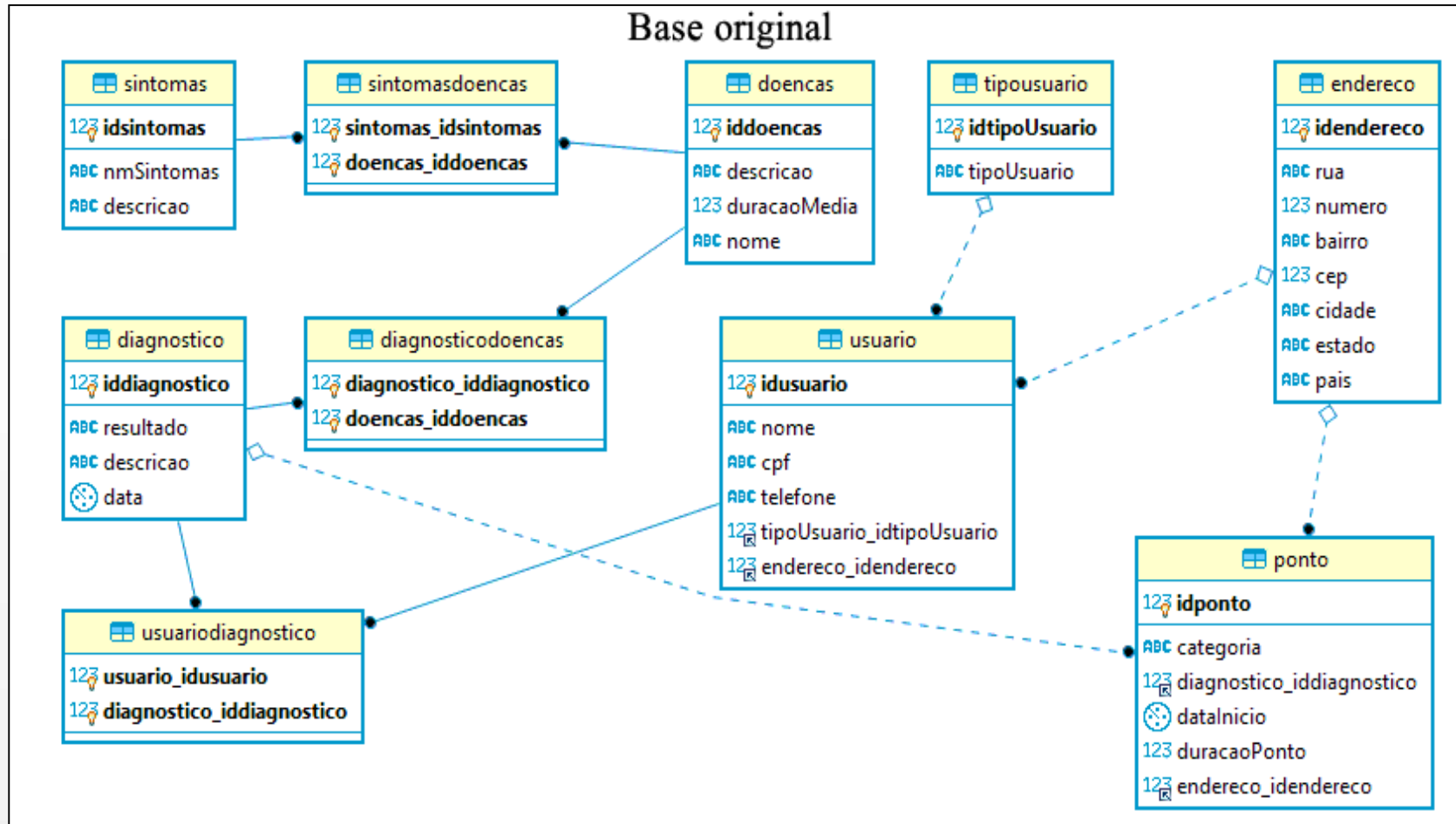
- para validar o código gerado foi criado um novo projeto JHipster.
- após configurar o projeto, foi importado o código gerado a partir de um arquivo.
- foi iniciada a aplicação JHipster: frontend e backend.
- realizado teste exploratório para verificar se todas as entidades possuem suas telas.

Resultados e Discussões

- como o JHipster gera uma nova base de dados, foi feita uma comparação visual da base gerada com a base original.

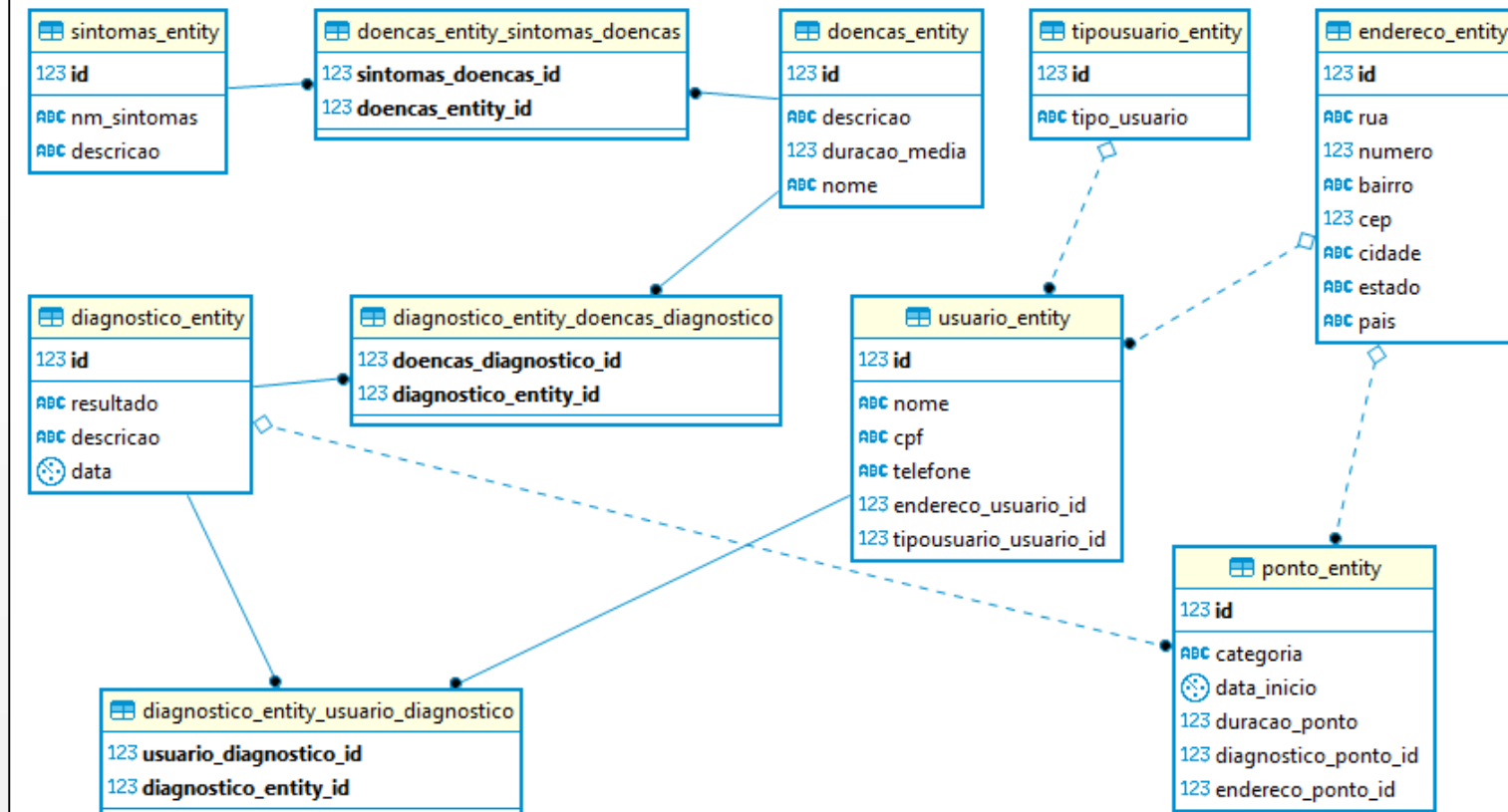
Resultados e Discussões

Base original



Resultados e Discussões

Base gerada pelo JHipster



Resultados e Discussões

Características	FormGenerate	OpenXava	AutoCRUD	DB2JHipster
Geração de aplicação web completa	Não	Sim	Sim	Sim*
Geração a partir de uma base de dados existente	Sim	Não	Não	Sim
Depende de um projeto existente	Sim	Não	Sim	Sim
Módulo de segurança (autenticação e autorização)	Não	Versão paga	Não*	Sim*

Conclusões e Sugestões

- O protótipo atendeu todos os objetivos definidos, possibilitando converter a estrutura de tabelas, campos e relacionamentos de uma base de dados para a linguagem JDL.
- Para que a conversão fosse realizado, foram implementados vários métodos, passíveis de serem reaproveitados por terceiros (API)
- A etapa de validação se mostrou satisfatória.
- Foi possível iniciar uma aplicação funcional, com o CRUD das entidades mapeadas do banco de dados.

Contribuições

- A principal contribuição deste trabalho, é poder escalar uma aplicação web a partir de um modelo de entidade relacionamento em pouco tempo.
- Pode auxiliar na prototipação de uma nova aplicação que já tenha seu modelo de dados definido.
- Pode ser utilizado a nível acadêmico, em disciplinas, como: Banco de Dados, Projeto de Software e outras.

Limitações

- Não aproveita os dados da base original, somente a estrutura.
- Não faz validações de campos obrigatórios.
- Não valida o tamanho de campos do tipo String.
- Não exibe uma descrição intuitiva para os relacionamentos das tabelas no *frontend*

Extensões

- Aproveitar os dados da base original: rodar a partir da mesma base, ou criar uma forma de exportar os dados
- Permitir que o usuário escolha qual campo será exibido nos relacionamentos (*frontend*)
- Gerar o código JDL com as validações dos campos, tamanho de String
- Expandir API, exemplo: rodar o projeto JHipster diretamente do protótipo

PROTÓTIPO DE UM GERADOR DE APLICAÇÕES WEB COM JHIPSTER

Aluno: Ingmar Schmidt de Aguiar

Orientador: Mauro Marcelo Mattos