GLOBAL SOLUTION

1TDSPB

Andre Sousa Matuda
RM: 566733
Guilherme Oliveira Feitosa
RM: 566842
Paulo Henrique Muniz Diedirch
RM: 567618

FIAP - Faculdade de Informática e Administração Paulista

2025

Domain Driver Design Using Java

Professor: Leonardo Gasparini Romão

Data da Entrega: 23/11/2025

Sumário

I. Descritivo do Projeto e Objetivos

O projeto **TalentForge** é uma solução de software desenvolvida para enfrentar o principal desafio do futuro do trabalho: a velocidade da **obsolescência de habilidades** causada pela automação e pela Inteligência Artificial. Nossa plataforma simula um sistema de **Requalificação Preditiva** (Reskilling) que utiliza dados para guiar colaboradores e organizações na transição para um mercado de trabalho digital. O Domínio se concentra em entidades centrais como o **Colaborador**, a **Profissão** e a **Trilha** de aprendizado, refletindo a complexidade de RH e educação.

O sistema foi modelado utilizando a metodologia **Domain Driven Design (DDD)** para garantir que a lógica de negócio esteja estritamente ligada às entidades. Utilizamos os conceitos de **Encapsulamento** (em todas as entidades) e **Herança com Polimorfismo** (na classe `Habilidade`) para construir uma estrutura robusta e flexível.

Justificativas e Objetivos

O objetivo central deste projeto em Java é demonstrar, através de uma modelagem de software, a eficácia de uma solução orientada a objetos para o problema da requalificação de talentos.

1. Garantir a Empregabilidade e Redução de Desigualdades (ODS 8 e ODS 10): Nosso sistema busca criar um caminho de aprendizado personalizado, permitindo que o colaborador migre de funções com alto risco de automação (e.g., Atendente de Suporte N1) para novas oportunidades que exigem alta competência humana.
2. Demonstração Completa de POO: O projeto cumpre o objetivo técnico de demonstrar o domínio completo dos conceitos de Orientação a Objetos em Java. Implementamos o Encapsulamento em todas as classes, definindo atributos privados e métodos de acesso públicos (`Getters` e `Setters`).
3. Implementação de Herança e Polimorfismo: Utilizamos a classe abstrata `Habilidade` como Superclasse, da qual derivam as subclasses `HabilidadeTecnica` e `HabilidadeHumana`. Isso permite que a lista `List<Habilidade>` na classe `Trilha` aceite objetos de diferentes tipos especializados (Polimorfismo).
4. Modelagem com Atributos de Referência: As sete classes (seis entidades + classe principal) utilizam atributos de referência (Ex: `Colaborador` referencia `TrilhaAtiva`), garantindo a integridade dos dados e refletindo o fluxo de negócio de nossa solução.
5. Simulação da Lógica de Negócio: Os quatro métodos funcionais implementados (testados na classe `Main`) simulam ações críticas do sistema, como o cálculo de risco (`classificarRisco`) e o registro de progresso (`registrarConclusaoModulo`), comprovando a lógica de negócio da TalentForge.

II . Modelagem de Classes e Relacionamentos

A solução contém sete classes, incluindo a classe abstrata `Habilidade`, o que atende ao requisito mínimo de seis classes e demonstra o uso de herança**.**

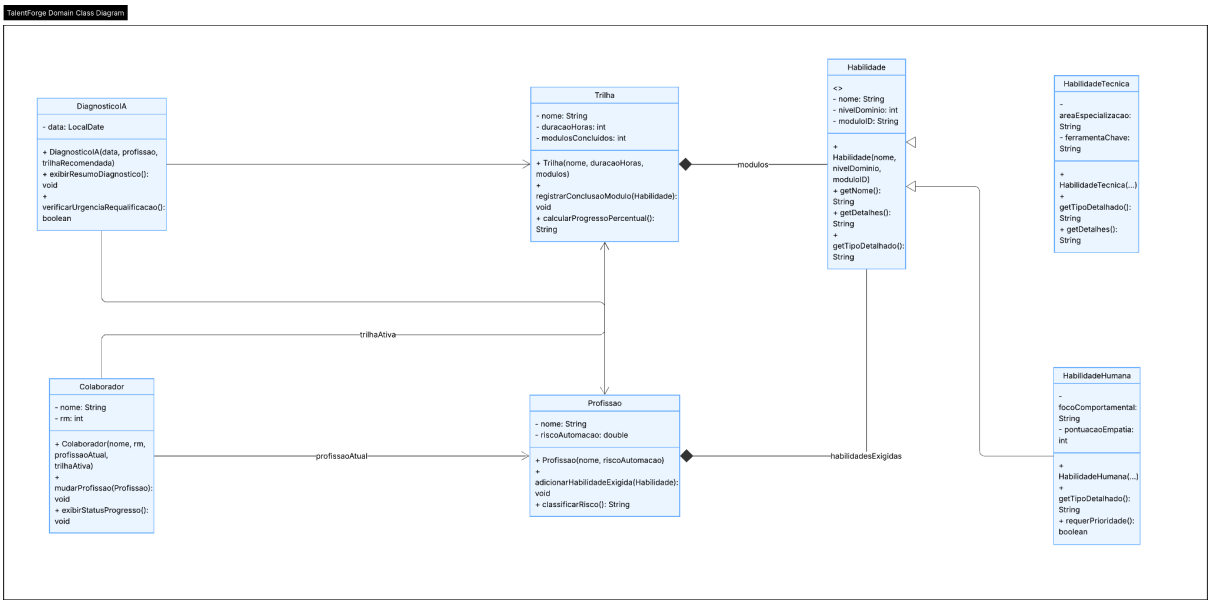


Tabela de Classes e Relacionamentos Chave:

Classe	Tipo	Relações Chave	Atributos Chave
Habilidade	Abstrata (Superclasse)	Nenhuma	nome, nivelDominio, moduloID
HabilidadeTecnica	Subclasse	Herda de Habilidade	areaEspecializacao, ferramentaChave
HabilidadeHumana	Subclasse	Herda de Habilidade	focoComportamental, pontuacaoEmpatia

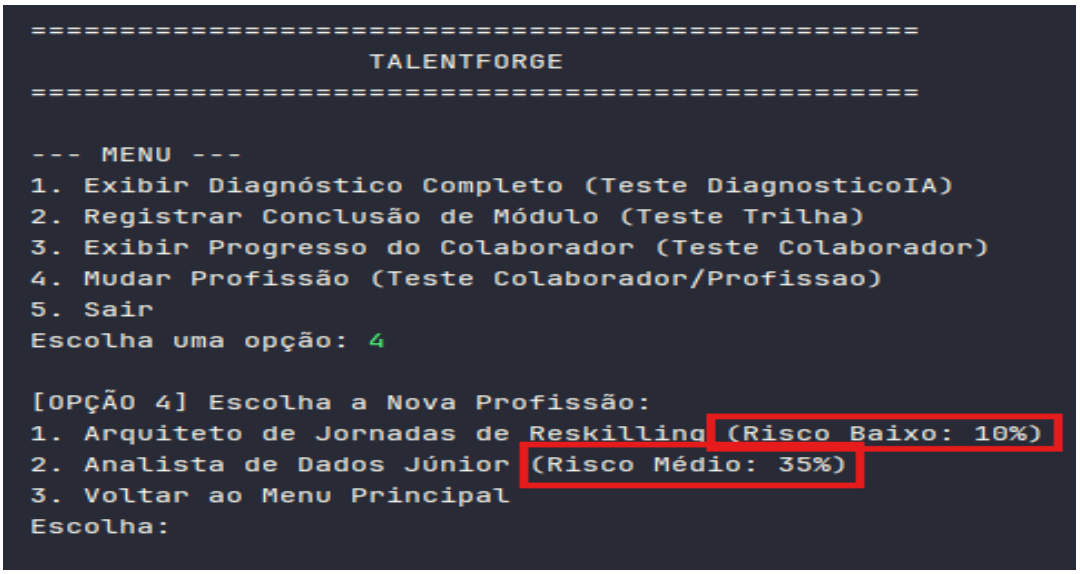
Profissão	Entidade	Contém List<Habilidade>	nome, riscoAutomacao
Trilha	Entidade	Contém List<Habilidade>	nome, duracaoHoras, modulosConcluidos
Colaborador	Entidade	Referencia Profissao, Trilha	nome, rm
DiagnosticoIA	Entidade (Service)	Referencia Profissao, Trilha	data

---

III .Explicação e Demonstração dos Métodos Funcionais

1. Método: `classificarRisco()` (Classe Profissao)

- **Descrição:** Este método implementa a lógica central de classificação de risco de automação. Ele utiliza o `riscoAutomacao` (decimal) para determinar, por meio de condicionais (`if/else`), o nível de risco em texto (Alto, Médio ou Baixo). O método retorna a classificação formatada.



```java
public String classificarRisco() {  3 usages
    Risco nivelRisco = Risco.classificar(riscoAutomacao);

    return nivelRisco.getNome() + " (Risco de Automação: " + String.format("%.0f%%", riscoAutomacao * 100) + ")";
}
```

---

2. Método: `registrarConclusaoModulo()` (Classe Trilha)

- **Descrição:** Este método é fundamental para rastrear o progresso. Ele recebe um objeto da superclasse `Habilidade` como parâmetro (aceitando tanto Habilidade Técnica quanto Humana) e verifica se a habilidade faz parte da lista de módulos. Se o módulo for encontrado, ele incrementa o contador `modulosConcluidos`.

```
======================================================
                    TALENTFORGE
======================================================

--- MENU ---
1. Exibir Diagnóstico Completo (Teste DiagnosticoIA)
2. Registrar Conclusão de Módulo (Teste Trilha)
3. Exibir Progresso do Colaborador (Teste Colaborador)
4. Mudar Profissão (Teste Colaborador/Profissao)
5. Sair
Escolha uma opção: 2

[OPÇÃO 2] Registrar Conclusão de Módulo:
Módulo 'Fundamentos de IA Generativa' concluído!
```

```java
public void registrarConclusaoModulo(Habilidade habilidadeConcluida) {  1 usage
    if (this.modulos.contains(habilidadeConcluida)) {
        this.modulosConcluidos++;
        System.out.println("Módulo '" + habilidadeConcluida.getNome() + "' concluído!");
    } else {
        System.out.println("Erro: Módulo não faz parte desta trilha.");
    }
}
```

---

3. Método: `requerPrioridade()` (Classe HabilidadeHumana)

- **Descrição:** Este método, específico da subclasse `HabilidadeHumana`, implementa uma lógica de prioridade de *soft skill*. Ele avalia a `pontuacaoEmpatia` e retorna um valor booleano (`true`) se a pontuação for insuficiente (baixa), sinalizando que o desenvolvimento dessa habilidade é crítico e deve ser priorizado pelo sistema.

```
--- Teste Habilidade Humana (Método requerPrioridade) ---
Habilidade 'Pensamento Crítico' requer prioridade? true
---------------------------------
```

```java
    public boolean requerPrioridade() {
        return pontuacaoEmpatia < 3;
    }
```

---

4. Método: `exibirResumoDiagnostico()` (Classe DiagnosticoIA)

- **Descrição:** Este método coordena as informações de serviço e domínio para gerar um relatório final coeso. Ele utiliza as referências (`Profissao` e `Trilha`) e **chama métodos funcionais de outras classes** (como `classificarRisco()`) para apresentar o resumo completo do diagnóstico e o status atual do colaborador.

```
[OPÇÃO 1] Detalhes do Diagnóstico:
--- Diagnóstico IA TalentForge ---
Data da Análise: 2025-11-19
Profissão Analisada: Atendente de Suporte N1
Risco de Automação: Médio (Risco de Automação: 65%)
Trilha Recomendada: Transição para Analista de Dados
Progresso na Trilha: 33,33%
---------------------------------
Urgência de Requalificação: BAIXA
```

```java
public void exibirResumoDiagnostico() {  1 usage
    System.out.println("--- Diagnóstico IA TalentForge ---");
    System.out.println("Data da Análise: " + this.data);
    System.out.println("Profissão Analisada: " + this.profissao.getNome());
    System.out.println("Risco de Automação: " + this.profissao.classificarRisco());
    System.out.println("Trilha Recomendada: " + this.trilhaRecomendada.getNome());
    System.out.println("Progresso na Trilha: " + this.trilhaRecomendada.calcularProgressoPercentual());
    System.out.println("---------------------------------");
}
```