BRD - Sistema de Gerenciamento de Freelancers(ToFree)

Versão: 1.0
Data: 04.10.25
Autor: Enzo Spíndola
Para: Empresa genérica, exemplo usado: serviços de brinquedos infláveis

1. Visão Geral e Objetivos de Negócio

1.1. Propósito do Documento:

- Este documento define os requisitos de negócio, escopo e funcionalidades para o desenvolvimento de um sistema de gerenciamento de freelancers (monitores).
- O objetivo é centralizar e otimizar todo o processo, da contratação ao pagamento, aumentando a eficiência operacional e a confiabilidade do serviço.

1.2. Objetivos de Negócio:

- Reduzir o tempo e a burocracia no processo de contratação e escalação de monitores.
- Aumentar a confiabilidade e disponibilidade da mão de obra freelance, evitando faltas em eventos.
- Melhorar a comunicação entre a empresa e os freelancers.
- Otimizar o controle financeiro com pagamentos e relatórios claros.
- Garantir a qualidade do serviço através de um sistema de feedback e controle de qualificações.

2. Descrição do Problema de Negócio

Atualmente, o processo provavelmente é manual e propenso a erros, envolvendo:

- Planilhas soltas ou anotações para cadastro de freelancers.
- Comunicação via WhatsApp ou telefone, gerando ruído e perdendo o histórico.
- Dificuldade em lembrar as habilidades, confiabilidade e avaliações de cada freelancer.
- Controle financeiro complexo, com valores variáveis por evento e tipo de serviço.
- Risco de o freelancer esquecer o compromisso ou aceitar o trabalho e não comparecer.

3. Escopo do Projeto

3.1. O QUE ESTÁ DENTRO DO ESCOPO (In-Scope):

- Módulo de Cadastro e Perfil de Freelancers.
- Módulo de Gestão de Eventos (Clientes que alugam os brinquedos).

- Módulo de Alocação e Convites para Freelancers. ● Módulo de Confirmação de Presença.
- Módulo Financeiro (Cálculo de pagamentos, aprovação e relatórios).
- Painel administrativo (Dashboard) para visão geral do negócio.

3.2. O QUE ESTÁ FORA DO ESCOPO (Out-of-Scope) - PARA UMA FASE 1:

- Sistema de pagamento automático integrado a gateways (ex: Pix, TED). (Pode ser um objetivo futuro).
- App móvel nativo para os freelancers (iniciamos com uma versão web responsiva).
- Rastreamento em tempo real via GPS dos freelancers.
- Sistema de gestão de estoque dos brinquedos (focamos apenas nos recursos humanos por agora).

4. Partes Interessadas (Stakeholders)

- Empresa de eventos: Dona do negócio
- Gestor/Coordenador de Eventos: Responsável por escalar as equipes.
- Freelancers (Monitores): Usuários finais que recebem e executam os trabalhos.

5. Requisitos de Negócio Detalhados (Funcionalidades)

Vamos dividir por módulos, pensando como um desenvolvedor que entende de negócio:

- **Módulo 1:** Gestão de Freelancers (O "Banco de Talentos")
  - BR-**001:** O sistema deve permitir o cadastro de freelancers com os seguintes dados: Nome, CPF, Contato, Endereço, Dados Bancários e Disponibilidade (dias da semana, períodos).

- **Módulo 2:** Gestão de Eventos
  - BR-**004**: O sistema deve permitir cadastrar um evento (Cliente, Endereço, Data/Horário, Lista de Brinquedos Infláveis contratados).

- **Módulo 3:** Alocação Inteligente e Convites
  - BR-**006:** O sistema deve permitir ao gestor, a partir de um evento, buscar freelancers disponíveis na data, filtrando por proximidade geográfica (baseado no CEP).
  - BR-**007:** O sistema deve permitir o envio de um "convite" para o freelancer (WhatsApp ou no próprio site), com os detalhes do evento e o valor a ser pago.
  - BR-**008:** O freelancer deve poder responder o convite via link no mensagem ("Aceitar" ou "Recusar").
  -
- **Módulo 4:** Confirmação e Check-in
  - BR-**009**: O sistema deve enviar um lembrete automático 24h antes do evento.(email site)
  - BR-**010:** No dia do evento, o freelancer deve realizar um "check-in" via sistema (web) para confirmar sua presença. Isso gera um registro irrefutável.

- **Módulo 5:** Gestão Financeira
    - BR-**011:** O sistema deve gerar uma "folha de pagamento" por evento ou por período (semanal/mensal) para aprovação do gestor.
    - BR-**013:** O sistema deve gerar relatórios financeiros de custos com mão de obra por período.

- **Módulo 6:** Sistema de Reputação
    - BR-**014:** Após o evento, o gestor deve poder avaliar o freelancer (nota de 1 a 5 e comentários). Essa avaliação deve impactar sua "pontuação de confiabilidade" no sistema.

- **Módulo 7:** Dashboard e Relatórios

    - BR-**016:** O sistema deve fornecer um painel principal com KPIs (Indicadores-Chave): Nº de Eventos do Mês, Taxa de Ocupação de Freelancers, Custo Total com Mão de Obra, Top Freelancers, etc.

6. Requisitos Não-Funcionais

- RNF-**001** (Usabilidade): A interface deve ser intuitiva e de fácil uso para freelancers com baixa familiaridade com tecnologia.
  .
- RNF-**002** (Disponibilidade): O sistema deve estar disponível 24/7, especialmente em horários de pico (finais de semana).

- RNF-**003** (Segurança): Os dados pessoais e bancários dos freelancers devem ser armazenados de forma segura e criptografada.

- RNF-**004** (Performance): O processo de busca e alocação de freelancers deve ser rápido (< 3 segundos).

7. Métricas de Sucesso

- Redução em X% do tempo gasto para montar uma equipe para um evento.
- Redução para próximo de 0% das faltas de freelancers não justificadas.
- 90%+ dos freelancers ativos utilizando o sistema para confirmações. - Geração de relatórios financeiros em minutos, e não horas.

DIAGRAMA DE SEQUENCIA:

 Diagrama de Sequência – Cadastro e Consulta de Freelancers

Participantes

- **Freelancer**
- **Gestor**
- **Frontend (Web)**
- **Backend API**
- **Banco de Dados**

🧠 **Fluxo 1 – Cadastro de Freelancer**

1. **Gestor** acessa a **página de cadastro** no **Frontend (Web)**.
2. O **Frontend** envia uma requisição:
    - **GET /api/freelancers/check?cpf=123...**
    - Objetivo: verificar se o CPF informado já existe.
3. O **Backend API** consulta o **Banco de Dados**:
    - Verifica se o CPF já está cadastrado.
    - Retorna o resultado ao **Backend API**.
4. O **Backend API** responde ao **Frontend**:
    - “CPF disponível”.
5. **Gestor** preenche os dados do freelancer (**RF-001**) e **submete o formulário**.
6. O **Frontend** envia:
    - **POST /api/freelancers**
    - Com os dados do freelancer.
7. O **Backend API**:
    - Valida os dados e criptografa informações sensíveis (**RNF-003**).
    - Insere o novo freelancer no **Banco de Dados**.
    - Banco confirma a criação.
8. O **Backend API** responde ao **Frontend**:
    - “Sucesso + ID do novo freelancer”.
9. O **Frontend** mostra a **confirmação de cadastro** ao **Gestor**.

📋 **Fluxo 2 – Consulta de Freelancers**

10. O **Gestor** acessa a **lista de freelancers**.
11. O **Frontend (Web)** envia:
    - **GET /api/freelancers**
12. O **Backend API** consulta o **Banco de Dados**:
    - Busca todos os freelancers cadastrados.
    - Retorna a lista para o **Backend API**.
13. O **Backend API** envia os **dados dos freelancers** ao **Frontend (Web)**.
14. O **Frontend** exibe a **lista** para o **Gestor**.

Requisitos Referenciados

- **RF-001:** Cadastro de freelancer.
- **RF-003:** Consulta e atualização de dados de freelancers.
- **RNF-003:** Criptografia de informações sensíveis.

ENTIDADE RELACIONEMENTO

MODELO ENTIDADE-RELACIONAMENTO (DER) – SISTEMA DE EVENTOS E FREELANCERS
FREELANCER
Descrição: Armazena os dados dos profissionais cadastrados no sistema
Atributos:

- **id (int, PK) Identificador único do freelancer**

- **nome (string) Nome completo**
- **cpf (string, UK) CPF do freelancer (único)**
- **telefone (string) Contato telefônico**
- **cep (string) CEP do endereço**
- **endereco (string) Endereço completo**
- **dados_bancarios (json) Informações bancárias**
- **disponibilidade (json) Dias e horários disponíveis**
- **pontuacao_confiabilidade (decimal) Média de avaliações**
- **status (string) Situação atual (ativo, inativo etc)**
  Relacionamentos:
- **Possui DOCUMENTO (1:N)**
- **Participa ALOCAÇÃO (1:N)**

DOCUMENTO
Descrição: Guarda arquivos ou documentos vinculados a cada freelancer
Atributos:

- **id (int, PK)**
- **freelancer_id (int, FK → FREELANCER.id)**
- **tipo (string) Tipo do documento (RG, CPF, certidão etc)**
- **caminho_arquivo (string) Caminho do arquivo armazenado**
- **data_upload (datetime) Data de envio**
  Relacionamento:
- **Cada FREELANCER pode possuir vários DOCUMENTOS (1:N)**

EVENTO
Descrição: Representa um evento a ser realizado
Atributos:

- **id (int, PK)**
- **nome_cliente (string) Nome do cliente que contratou o evento**
- **contato_cliente (string) Telefone ou e-mail do cliente**
- **endereco_evento (string) Local de realização**
- **cep_evento (string) CEP do evento**
- **data_hora_inicio (datetime) Início do evento**
- **data_hora_fim (datetime) Fim do evento**
- **status (string) Situação atual do evento (planejado, em andamento, concluído, cancelado)**
  Relacionamentos:
- **Contém ALOCAÇÃO (1:N)**
- **Inclui EVENTO_BRINQUEDO (1:N)**

BRINQUEDO
Descrição: Catálogo de brinquedos disponíveis para uso nos eventos
Atributos:

- **id (int, PK)**
- **nome (string) Nome do brinquedo**
- **descricao (string) Descrição do brinquedo**
- **monitores_necessarios (int) Quantidade mínima de monitores necessários**
  Relacionamento:
- **Utilizado_em EVENTO_BRINQUEDO (1:N)**

ALOCAÇÃO
Descrição: Associação entre um freelancer e um evento, indicando quem trabalha em qual evento e com quais condições
Atributos:

- **id (int, PK)**
- **evento_id (int, FK → EVENTO.id)**
- **freelancer_id (int, FK → FREELANCER.id)**
- **valor_pagamento (decimal) Valor acordado**
- **status_convite (string) Exemplo: pendente, aceito, recusado**
- **data_hora_convite (datetime) Data e hora do convite**
- **data_hora_resposta (datetime) Data e hora da resposta**
- **status_checkin (string) Exemplo: confirmado, ausente**
- **data_hora_checkin (datetime) Data e hora do check-in**
  Relacionamentos:
- **Recebe AVALIAÇÃO (1:N)**

AVALIAÇÃO
Descrição: Avaliação do desempenho do freelancer em um evento
Atributos:

- **id (int, PK)**
- **alocacao_id (int, FK → ALOCAÇÃO.id)**
- **nota (int) Avaliação numérica**
- **comentario (text) Feedback textual**
- **data_avaliacao (datetime) Data da avaliação**
  Relacionamento:
- **Cada ALOCAÇÃO pode receber várias AVALIAÇÕES (1:N)**

EVENTO_BRINQUEDO
Descrição: Relação entre eventos e brinquedos utilizados
Atributos:

- **id (int, PK)**
- **evento_id (int, FK → EVENTO.id)**
- **brinquedo_id (int, FK → BRINQUEDO.id)**
- **quantidade (int) Quantos brinquedos desse tipo foram utilizados**
  Relacionamentos:
- **Um EVENTO pode incluir vários BRINQUEDOS**
- **Um BRINQUEDO pode ser usado em vários EVENTOS**
- **Relação N:N resolvida pela tabela EVENTO_BRINQUEDO**

Resumo dos Relacionamentos Principais
FREELANCER 1:N DOCUMENTO Um freelancer pode ter vários documentos
FREELANCER 1:N ALOCAÇÃO Um freelancer pode participar de vários eventos
EVENTO 1:N ALOCAÇÃO Um evento pode ter vários freelancers alocados
ALOCAÇÃO 1:N AVALIAÇÃO Uma alocação pode ter várias avaliações
EVENTO N:N BRINQUEDO Associação feita via EVENTO_BRINQUEDO