1. Introdução

Este relatório apresenta a fundação técnica da plataforma "Mão na Massa", detalhando duas áreas cruciais: a **Modelagem de Dados** e a **Arquitetura Tecnológica**. O objetivo deste documento é servir como um guia central para a equipe de desenvolvimento, estabelecendo a estrutura do banco de dados, as tecnologias a serem utilizadas e o fluxo de interação entre os componentes do sistema.

A estrutura aqui definida visa garantir uma solução robusta, escalável e segura, alinhada às melhores práticas de desenvolvimento de software moderno.

2. Modelo de Dados

A modelagem de dados define a estrutura lógica e física para o armazenamento de todas as informações da plataforma. O modelo foi projetado para ser centralizado e relacional, garantindo integridade e consistência dos dados.

2.1. Modelo Conceitual

O modelo conceitual da plataforma é composto por três entidades fundamentais:

- **Pessoas (Entidade Forte):** Representa qualquer indivíduo cadastrado no sistema, seja ele um cliente ou um prestador de serviço. Esta entidade centraliza os dados cadastrais básicos e únicos, como CPF, nome e informações de contato.
- **Serviços (Entidade Forte):** Atua como um catálogo mestre, descrevendo cada tipo de serviço que pode ser oferecido na plataforma (ex: Eletricista, Marceneiro, Faxina Residencial).
- **Prestadores (Entidade Associativa):** É a entidade que conecta uma Pessoa a um Serviço, efetivamente definindo que um usuário é um ofertante de um serviço específico. Contém informações comerciais como preço e condições para aquela oferta.

Relacionamentos Principais:

- Uma Pessoa pode oferecer zero ou múltiplos Serviços (relação 1:N, através da tabela Prestadores).
- Um Serviço pode ser oferecido por uma ou múltiplas Pessoas (relação 1:N, através da tabela Prestadores).

2.2. Diagrama de Entidade-Relacionamento (DER)

O diagrama abaixo ilustra visualmente a estrutura e os relacionamentos entre as entidades do banco de dados.

2.3. Modelo Físico (Script SQL)

O modelo físico é a implementação concreta da estrutura do banco de dados em MySQL. O script a seguir define as tabelas, colunas, tipos de dados e chaves estrangeiras.

SQL

```
-- Tabela para armazenar dados cadastrais de todas as pessoas (clientes e prestadores).
CREATE TABLE Pessoas (
    -- CPF da pessoa, chave primária que a identifica unicamente.
    CPF CHAR(11) PRIMARY KEY,
    -- Nome completo, campo obrigatório.
    Nome VARCHAR(100) NOT NULL,
    -- Email para contato.
    Email VARCHAR(100),
    -- Telefone para contato.
    Telefone VARCHAR(15),
    -- CEP do endereço.
    CEP CHAR(8),
    -- Número do endereço.
    Numero VARCHAR(10),
    -- Complemento do endereço (ex: Apto 101, Bloco B).
    Complemento VARCHAR(100)
);

-- Tabela para catalogar os tipos de serviços oferecidos.
CREATE TABLE Servicos (
    -- Código numérico único para cada serviço, gerado automaticamente.
    Codigo INT AUTO_INCREMENT PRIMARY KEY,
    -- Nome do serviço (ex: 'Eletricista', 'Marceneiro').
    Nome VARCHAR(100) NOT NULL,
    -- Breve descrição do que o serviço inclui.
    Descricao VARCHAR(255)
);

-- Tabela para vincular uma pessoa a um serviço que ela presta.
CREATE TABLE Prestadores (
    -- ID numérico único para cada registro de prestador, gerado automaticamente.
    Cod_Prestador INT AUTO_INCREMENT PRIMARY KEY,
    -- CPF da pessoa que oferece o serviço, vindo da tabela Pessoas.
    CPF_Pessoa CHAR(11),
    -- Código do serviço oferecido, vindo da tabela Servicos.
    Codigo_Servico INT,
    -- Preço cobrado pelo serviço. O formato DECIMAL(12,2) suporta até 12 dígitos, com 2 casas decimais.
    Preco DECIMAL(12,2),
    -- Condições específicas do serviço (ex: 'Pagamento 50% adiantado').
    Condicao VARCHAR(500),

    -- Define a ligação com a tabela de Pessoas.
    FOREIGN KEY (CPF_Pessoa) REFERENCES Pessoas(CPF),
    -- Define a ligação com a tabela de Serviços.
    FOREIGN KEY (Codigo_Servico) REFERENCES Servicos(Codigo)
);
```

3. Arquitetura da Solução e Tecnologias

A arquitetura da aplicação segue o modelo cliente-servidor, com uma separação clara de responsabilidades entre a interface do usuário (front-end) e a lógica de negócios (back-end).

3.1. Pilha de Tecnologias (Technology Stack)

A tabela a seguir detalha o conjunto de tecnologias selecionadas para o projeto.

Dimensão	Tecnologia	Descrição e Finalidade
SGBD	**MySQL**	Sistema de Gerenciamento de Banco de Dados relacional para armazenamento persistente e seguro de todas as informações.
Front-end	**HTML5, CSS3, JavaScript (ES6+)**	Tecnologias padrão para estruturar, estilizar e adicionar interatividade à interface do usuário no navegador.
	(Framework CSS: **Bootstrap**)	Para acelerar o desenvolvimento de um design responsivo, moderno e consistente em diferentes dispositivos.
Back-end	**Java (versão 17 ou superior)**	Linguagem de programação robusta e de alta performance que servirá como base para toda a lógica de negócio.
	Spring Boot	Framework para acelerar a criação da API RESTful, que expõe os dados e funcionalidades para o front-end.
	Spring Data JPA & Hibernate	Módulos que abstraem e facilitam a comunicação com o banco de dados MySQL, mapeando tabelas para objetos Java.
Ferramentas de Desenvolvimento	**Visual Studio Code / IntelliJ IDEA**	IDEs para o desenvolvimento eficiente do front-end e back-end, respectivamente.
	Git & GitHub	Sistema de controle de versão para gerenciar o código-fonte e plataforma de hospedagem para colaboração.
	Postman / Insomnia	Ferramentas para testar e validar a API RESTful do back-end de forma isolada.
	MySQL Workbench	Ferramenta visual para modelar, administrar e interagir com o banco de dados.

3.2. Arquitetura e Fluxo de Interação do Usuário

O diagrama abaixo ilustra a arquitetura da solução e o fluxo de uma requisição típica, desde a interação do usuário até a resposta do sistema.

Detalhamento do Fluxo:

1. **Requisição do Usuário:** O usuário interage com a interface (front-end) em seu navegador. Uma ação (ex: um clique de botão) dispara uma função JavaScript.
2. **Chamada à API:** O JavaScript cria e envia uma requisição HTTP (ex: `GET`, `POST`) para um endpoint específico da API back-end.
3. **Processamento no Back-end:** A aplicação Java/Spring Boot recebe a requisição, aciona a lógica de negócio correspondente e prepara a interação com o banco de dados.
4. **Consulta ao Banco de Dados:** Através do Hibernate/JPA, a aplicação executa uma consulta SQL no banco de dados MySQL para ler ou escrever informações.
5. **Retorno dos Dados:** O MySQL retorna o resultado da consulta para a aplicação back-end.
6. **Resposta JSON:** O back-end formata os dados no padrão JSON e os envia de volta ao navegador como resposta à requisição inicial.
7. **Atualização da Interface:** O JavaScript no front-end recebe os dados JSON e atualiza dinamicamente a página para exibir as novas informações ao usuário, sem a necessidade de um recarregamento completo.

4. Conclusão

Este relatório fornece uma base técnica abrangente para o desenvolvimento da plataforma "Mão na Massa". A modelagem de dados e a pilha de tecnologias selecionadas formam um alicerce sólido para a construção de uma aplicação moderna, segura e de alta performance. Recomenda-se que este documento seja utilizado como referência principal pela equipe de desenvolvimento durante todo o ciclo de vida do projeto.