+-----------------------------------------------------------------------------------+

| StockMaster |

| Arquitetura: Cliente-Servidor |

+-----------------------------------------------------------------------------------+

| CLIENTE |

| - ReactJS: Interface de utilizador |

| - Screens (Componentes): |

| - AdicionarProdutoScreen.jsx |

| - CriarArmazemScreen.jsx |

| - EditarArmazemScreen.jsx |

| - EditarProdutoScreen.jsx |

| - HomeScreen.jsx |

| - LoginScreen.jsx |

| - MeusArmazensScreen.jsx |

| - ProdutosScreen.jsx |

| - SignupScreen.jsx |

| -> Comunicação com o servidor via API (HTTP/REST) |

+-----------------------▲---------------------------------------------------------+

| Requisições e Respostas (JSON)

+-----------------------▼---------------------------------------+

| SERVIDOR (Backend) |

| - Node.js com ExpressJS: Servidor Web |

| - Estrutura de Pastas: |

| - app.js e server.js |

| - controllers: |

| - armazemController.js |

| - authController.js |

| - empresaController.js |

| - produtoController.js |

| - middlewares: Funções auxiliares |

| - models: |

| - Armazem.js |

| - Empresa.js |

| - Produto.js |

| - routes: |

| - empresaRoutes.js |

| -> Ligações com MongoDB via Mongoose |

+-----------------------▲-----------------------------------------+

| Ligações de Base de Dados

+-----------------------▼-------------------------------------------------+

| BASE DE DADOS (MongoDB) |

| - Coleções: |

| - Empresas: Dados das empresas registadas |

| - Armazéns: Informações dos armazéns |

| - Produtos: Dados dos produtos de cada armazém |

+---------------------------------------------------------------------------+

**Explicação:**

1. **Cliente (Frontend):**
   * Desenvolvido em **ReactJS**.
   * Componentes no diretório Screens representam diferentes páginas da aplicação (ex.: registo, login, armazéns, produtos).
   * Comunica-se com o servidor através de **APIs REST**, enviando e recebendo dados em formato **JSON**.
2. **Servidor (Backend):**
   * Construído com **Node.js** e **ExpressJS**.
   * Estrutura organizada em controllers, middlewares, models e routes.
   * Usa **Mongoose** para interagir com a base de dados MongoDB.
3. **Base de Dados (MongoDB):**
   * Armazena dados estruturados em coleções (empresas, armazéns e produtos).
   * Relacionamentos definidos nos modelos (ex.: produtos pertencem a armazéns e empresas).