

# Projeto Gestão de Horários Escolares — Parte 3

**Aluno:** Igor Carlos Santos Cruz (A043765)

**Grupo:** 10

**Unidade Curricular:** Desenvolvimento Web I

**Parte:** 3 — Integração Full-Stack (LoopBack 4 + React-Admin + Docker)

## 1. Introdução

A fase final do projeto consistiu na evolução da API básica para uma infraestrutura empresarial utilizando **LoopBack 4** no backend e uma interface administrativa moderna com **React-Admin**. O objetivo foi criar um sistema completo onde a gestão de dados (Alunos, Professores, Salas, Aulas e Horários) é feita de forma visual e intuitiva, mantendo a robustez de uma base de dados relacional.

## 2. Arquitetura do Sistema

O sistema é composto por três camadas principais que comunicam em rede isolada via Docker:

- Frontend (React-Admin):** Interface Single Page Application (SPA) para gestão CRUD e visualização de dashboards.
- Backend (LoopBack 4):** API robusta que implementa os padrões Model-Repository-Controller.
- Database (MySQL):** Armazenamento persistente com integridade referencial.

## 3. Tecnologias Utilizadas

Tecnologia	Função
<b>LoopBack 4</b>	Framework TypeScript para criação da API e camada de dados.
<b>React-Admin</b>	Framework de frontend para painéis administrativos.
<b>Recharts / MUI</b>	Utilizados para a criação do Dashboard e gráficos.
<b>Nginx</b>	Servidor web para servir o build de produção do frontend.
<b>Docker / Compose</b>	Orquestração de todos os serviços.

## 4. Backend: LoopBack 4

A API foi reconstruída utilizando LoopBack 4.

### Geração do Projeto

O projeto LoopBack foi criado através do comando:

```
lb4 app
```

Após a geração inicia, foram realizadas adaptações para integração com MySQL e com o frontend React-Admin.

## DataSource MySQL

Configurado para ser dinâmico, permitindo a ligação local ou via Docker através de variáveis de ambiente:

```
const config = {
  name: 'db',
  connector: 'mysql',
  host: process.env.DB_HOST || 'localhost',
  user: process.env.DB_USER || 'root',
  password: process.env.DB_PASSWORD || 'Igor1234!',
  database: process.env.DB_NAME || 'school'
};
```

### Foram efetuadas algumas alterações nos:

- **Models**
- **Application (Cors)**
- **Controllers**

## 5. Frontend: React-Admin

O frontend foi desenvolvido com **React-Admin**, consumindo diretamente a API LoopBack.

### DataProvider Customizado

O maior desafio nesta camada foi a comunicação entre o React-Admin (que espera um formato de dados específico) e o LoopBack 4. Para resolver isto, implementámos um **dataProvider** customizado que:

- **Mapeamento de IDs:** Como a base de dados utiliza chaves primárias específicas (ex: `id_aluno`, `id_professor`), o provider foi programado para injetar uma propriedade `id` genérica em cada registo, garantindo a compatibilidade com os componentes do React-Admin.
- **Comunicação Direta:** Configurámos o `fetch` para apontar diretamente para a porta exposta pelo Docker (`http://localhost:8080`), permitindo operações assíncronas em tempo real.

### Funcionalidades principais:

- **Listagens (List):** Utilização de `Datagrid` com filtros integrados para pesquisa rápida.
- **Criação e Edição (Create/Edit):** Formulários estruturados com `SimpleForm` e validação de campos.
- **Relações (ReferenceFields):** Implementação de chaves estrangeiras visuais. Por exemplo, na listagem de **Horários**, o sistema utiliza o `id_professor` para procurar e exibir o **Nome** do docente diretamente da tabela de Professores, demonstrando a relação 1:n.
- **Visualização Detalhada (Show):** No recurso de Professores, implementámos um `TabbedShowLayout` que permite ver o perfil do docente numa aba e a sua agenda de horários noutra, utilizando o componente `ReferenceManyField`.

## Dashboard e Experiência do Utilizador

Foi incluído um **Dashboard** inicial que apresenta um resumo estatístico do sistema, utilizando componentes da biblioteca **MUI (Material UI)** para manter a coerência visual com o restante backoffice.

## 6. Execução com Docker e Imagens

O projeto cumpre o requisito de aplicação multi-container, utilizando pelo menos duas imagens distintas.

### Imagens no Docker Hub

As imagens foram publicadas no Docker Hub com a tag `m3` para identificar a versão final da Parte 3:

- `inf25dw1g10/react-admin:m3`
- `inf25dw1g10/api-horarios:m3`
- `inf25dw1g10/mysql-db:m3`

### Inicialização do Sistema

Para colocar todo o ecossistema em funcionamento (Base de dados + API + Frontend), deve-se aceder à pasta onde reside o orquestrador e executar o comando:

```
cd express-server
docker compose up -d
```

A Api fica disponível em <http://localhost:8080/explorer> e o Backoffice em <http://localhost:3001>.

## 7. Testes e Verificação de Qualidade

### Conformidade W3C

Para garantir que a interface segue as boas práticas da Web, foi realizada a validação do código gerado:

- HTML5:** O código fonte renderizado pelo React-Admin foi submetido ao W3C Markup Validation Service. A estrutura foi validada com sucesso, respeitando a semântica do HTML5.
- CSS3:** As folhas de estilo (MUI/Custom) foram verificadas no W3C CSS Validator, garantindo a correta interpretação visual em diferentes navegadores.

### Evidências de Funcionamento (Prints)

Nesta secção, apresentam-se as capturas de ecrã que comprovam a integração total do sistema:

Parabéns! Não foram encontrados erros na sua folha de estilo.

Este documento é válido para as CSS nível 3 + SVG !

Para mostrar aos seus leitores que você teve o cuidado de criar uma página web interoperável, você pode inserir um selo nas páginas válidas. Veja a seguir o código XHTML a ser usado para mostrar na sua página o citado selo:

 `<p>
 <a href="https://jigsaw.w3.org/css-validator/check/referer">
 <img style="border:0; width:88px; height:31px;
 src='https://jigsaw.w3.org/css-validator/images/vcss'
 alt="CSS válido!" />
 </a>
</p>`

1. Info Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.  
From line 5, column 5, to line 5, column 28  
`head>  
 <meta charset="utf-8" />`

2. Info Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.  
From line 6, column 5, to line 6, column 43  
`8" />  
 <link rel="icon" href="/favicon.ico" />`

3. Info Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.  
From line 7, column 5, to line 7, column 74  
`o" />  
 <meta name="viewport" content="width=device-width, initial-scale=1" />`

4. Info Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.  
From line 8, column 5, to line 8, column 49  
`1" />  
 <meta name="theme-color" content="#000000" />`

5. Info Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.  
From line 9, column 5, to line 12, column 6



```
<p>
  <a href="https://jigsaw.w3.org/css-validator/check/referer">
    <img alt="CSS válido!" />
  </a>
</p>
```

```
0" /><meta name="description" content="Web site created using create-react-app"/>
<
6. Info Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.
From line 13, column 5 to line 13, column 55
/><link rel="apple-touch-icon" href="/logo192.png" />
7. Info Trailing slash on void elements has no effect and interacts badly with unquoted attribute values.
From line 18, column 5 to line 18, column 49
--><link rel="manifest" href="/manifest.json" />
```

## 8. Utilização de Inteligência Artificial

Foram utilizadas ferramentas de Inteligência Artificial como apoio ao desenvolvimento do projeto, de forma responsável e transparente.

### Âmbito

#### A IA foi utilizada para:

- apoio conceptual
- esclarecimento de dúvidas técnicas
- validação de sintaxe
- auxílio na estruturação inicial de código

Todo o código foi revisto, compreendido e adaptado pelo aluno antes da sua integração no projeto final.

## 9. Conclusão

A Parte 3 do projeto permitiu consolidar conhecimentos de **desenvolvimento Full-Stack**, integrando backend, frontend e base de dados num ambiente profissional e escalável.

A utilização do **LoopBack 4** trouxe maior organização e robustez à API, enquanto o **React-Admin** possibilitou uma interface administrativa funcional e intuitiva. A utilização do **Docker** revelou limitações importantes: certas configurações que funcionam normalmente no meu ambiente local deixaram de funcionar quando executadas dentro de containers.

Apesar destes desafios, permitiu-me compreender na prática as dificuldades reais de portabilidade e compatibilidade entre ambientes.