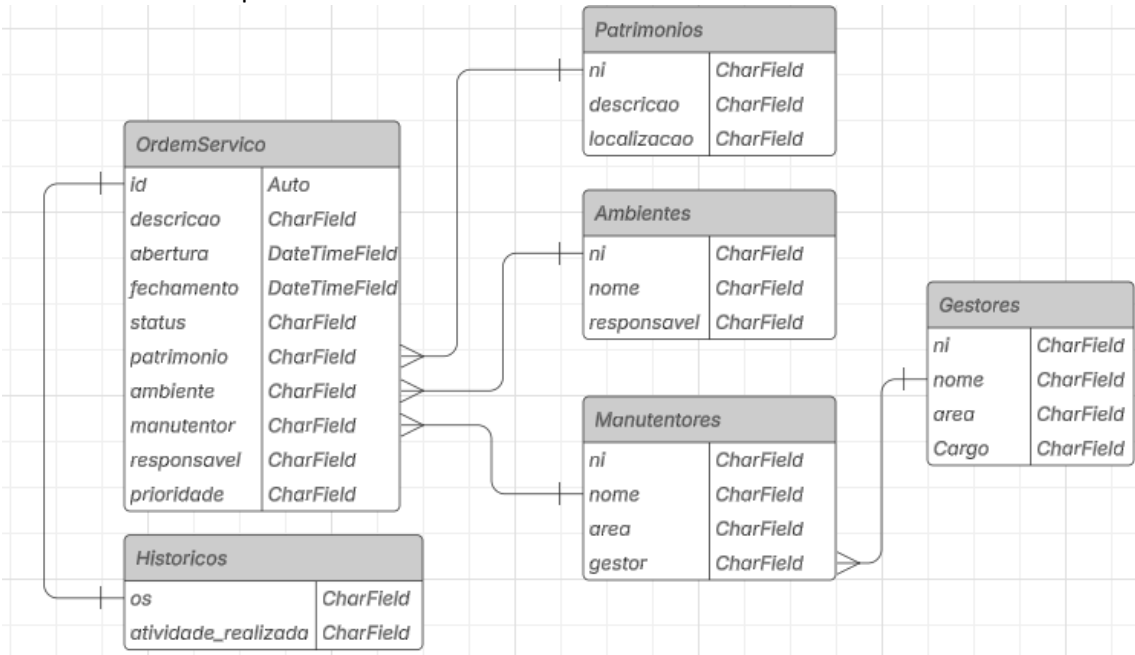


ATIVIDADE	Situação Problema																																																				
<p><b>Contexto</b></p> <p>Você foi contratado para desenvolver o <b>Sistema de Ordem de Serviço da TechEdu</b>, utilizando as tecnologias <b>Django no back end</b> e <b>React no front end</b>. O sistema deve incluir as seguintes funcionalidades:</p> <ol style="list-style-type: none"> <li><b>Cadastro e Autenticação de Usuários:</b> <ul style="list-style-type: none"> <li>Implementar um sistema de login e logout com autenticação via <b>JWT</b>.</li> <li>Somente usuários autenticados podem criar e visualizar ordens de serviço.</li> <li>Implementar <b>sign_in</b>, <b>sign_up</b> e <b>logoff</b> utilizando Django Rest Framework com <b>JWT</b>.</li> <li>Criar um sistema de permissões para diferenciar <b>técnicos</b>, <b>chefes de manutenção</b> e <b>administradores</b>.</li> </ul> </li> <li>Tabelas necessárias para o sistema:</li> </ol>  <pre> graph LR     OS[OrdemServico] --&gt; P[Patrimonios]     OS --&gt; A[Ambientes]     OS --&gt; M[Manutentores]     OS --&gt; H[Historicos]     G[Gestores] --&gt; M     </pre> <p><b>OrdemServico</b></p> <table border="1"> <tr><td>id</td><td>Auto</td></tr> <tr><td>descricao</td><td>CharField</td></tr> <tr><td>abertura</td><td>DateTimeField</td></tr> <tr><td>fechamento</td><td>DateTimeField</td></tr> <tr><td>status</td><td>CharField</td></tr> <tr><td>patrimonio</td><td>CharField</td></tr> <tr><td>ambiente</td><td>CharField</td></tr> <tr><td>manutentor</td><td>CharField</td></tr> <tr><td>responsavel</td><td>CharField</td></tr> <tr><td>prioridade</td><td>CharField</td></tr> </table> <p><b>Patrimonios</b></p> <table border="1"> <tr><td>ni</td><td>CharField</td></tr> <tr><td>descricao</td><td>CharField</td></tr> <tr><td>localizacao</td><td>CharField</td></tr> </table> <p><b>Ambientes</b></p> <table border="1"> <tr><td>ni</td><td>CharField</td></tr> <tr><td>nome</td><td>CharField</td></tr> <tr><td>responsavel</td><td>CharField</td></tr> </table> <p><b>Manutentores</b></p> <table border="1"> <tr><td>ni</td><td>CharField</td></tr> <tr><td>nome</td><td>CharField</td></tr> <tr><td>area</td><td>CharField</td></tr> <tr><td>gestor</td><td>CharField</td></tr> </table> <p><b>Gestores</b></p> <table border="1"> <tr><td>ni</td><td>CharField</td></tr> <tr><td>nome</td><td>CharField</td></tr> <tr><td>area</td><td>CharField</td></tr> <tr><td>Cargo</td><td>CharField</td></tr> </table> <p><b>Historicos</b></p> <table border="1"> <tr><td>os</td><td>CharField</td></tr> <tr><td>atividade_realizada</td><td>CharField</td></tr> </table>		id	Auto	descricao	CharField	abertura	DateTimeField	fechamento	DateTimeField	status	CharField	patrimonio	CharField	ambiente	CharField	manutentor	CharField	responsavel	CharField	prioridade	CharField	ni	CharField	descricao	CharField	localizacao	CharField	ni	CharField	nome	CharField	responsavel	CharField	ni	CharField	nome	CharField	area	CharField	gestor	CharField	ni	CharField	nome	CharField	area	CharField	Cargo	CharField	os	CharField	atividade_realizada	CharField
id	Auto																																																				
descricao	CharField																																																				
abertura	DateTimeField																																																				
fechamento	DateTimeField																																																				
status	CharField																																																				
patrimonio	CharField																																																				
ambiente	CharField																																																				
manutentor	CharField																																																				
responsavel	CharField																																																				
prioridade	CharField																																																				
ni	CharField																																																				
descricao	CharField																																																				
localizacao	CharField																																																				
ni	CharField																																																				
nome	CharField																																																				
responsavel	CharField																																																				
ni	CharField																																																				
nome	CharField																																																				
area	CharField																																																				
gestor	CharField																																																				
ni	CharField																																																				
nome	CharField																																																				
area	CharField																																																				
Cargo	CharField																																																				
os	CharField																																																				
atividade_realizada	CharField																																																				

- **Ordem de Serviço** (*OrdemServico*)
  - **id**: gerado automaticamente pelo banco de dados e será o número da Ordem de Serviço.
  - **descrição**: local onde serão descritos os problemas.
  - **abertura**: data e hora da abertura da OS.
  - **fechamento**: data e hora do fechamento da OS.
  - **status**: serão 3 status (iniciada, em andamento, finalizada, cancelada).
  - **patrimônio**: número do patrimônio que virá da tabela *patrimônios*, esse campo não deve ser obrigatório já que nem toda OS é para equipamento ou algo que tenha número, exemplo alvenaria.
  - **ambiente**: número do ambiente que virá da tabela *ambientes*.
  - **manutentor**: número do manutentor que virá da tabela *manutentores*.
  - **responsável**: número do responsável que virá da tabela *responsáveis*, esse campo não deve ser obrigatório.
  - **prioridade**: serão 3 prioridades (alta, media e baixa).
- **Patrimônios**
  - **ni**: número do equipamento, quando houver.
  - **descrição**: detalhes sobre o patrimônio.
  - **localizacao**: local físico.
- **Ambientes**
  - **ni**: código do ambiente, exemplo: LabA105.
  - **nome**: exemplo: Laboratório de Informática.
- **Manutentores**
  - **ni**: código alfanumérico do funcionário, exemplo: sn1021328.
  - **nome**: nome do manutentor, exemplo: Lindomar José Batistão.
  - **area**: exemplo: Informática
  - **gestor**: nome do gestor desse manutentor que virá da tabela *Gestores*.
- **Responsaveis**
  - **ni**: código alfanumérico do funcionário, exemplo: sn1021328.
  - **nome**: nome do manutentor, exemplo: José da Silva.
- **Gestores**
  - **ni**: código alfanumérico do funcionário, exemplo: sn1021328.
  - **nome**: nome do manutentor, exemplo: José da Silva.
  - **area**: área de atuação do gestor, exemplo: elétrica, informática etc.
  - **cargo**: exemplo: Operador de Práticas Profissionais, Coordenador Técnico, Coordenador Pedagógico.
- **Histórico**
  - **os**: número da ordem de serviço realizada.
  - **atividade\_realizada**: descrição da tarefa que foi feita.

**Observações:**

- pode-se simplificar os nomes dos campos, mas se fizer coloque por extenso nos comentários.
- Algumas planilhas serão disponibilizadas para popular o banco de dados, crie um código em Python para isso, se preferir pode ser com endpoints.

**3. Relacionamento entre tabelas**

- Os relacionamentos deverão ser aplicados nas tabelas conforme diagrama já mencionado acima, lembrando que não foram explicados em aula, portanto você deverá fazer pesquisas para implementação.
- No front-end, dados de tabelas relacionadas deverão ser listados nos campos relacionados.

**4. Gerenciamento de Ordens de Serviço:**

- Em todas as páginas os elementos deverão ser listados com as opções de CRUD para cada registro.
- Desenvolva opções de localização de dados.
- Atualizar o status da OS (pendente, em andamento, concluído).

**5. Gerenciamento de Acesso:**

- Como pode ser observado, todas as tabelas acima são de uso administrativo, ou seja, somente o gestor de manutenção que deverá ter todas essas opções. Você deverá pesquisar como limitar os acessos aos mantenedores e funcionários nas tarefas.
- Funcionários podem **criar** ordens de serviço.
- Técnicos de manutenção podem **visualizar e atualizar** o status das OS.
- O administrador tem acesso total.

**6. Integração entre Front End e Back End:**

- Utilizar **Axios** no React para consumir a API Django.
- Criar uma interface intuitiva para cadastro e acompanhamento das OS.

**Critérios de Avaliação – Back End (Django)**

<b>Critério</b>	<b>Descrição</b>	<b>Peso (%)</b>
<b>Autenticação e Permissões</b>	Implementação de <b>sign in</b> com JWT no Django Rest	<b>10</b>
	Desenvolver página de cadastro de usuários <b>sign up</b>	<b>5</b>
	Fazer <b>logout</b> através de ícone inserido no cabeçalho retornando a página de login e limpando o token	<b>5</b>
<b>Modelagem de Dados (Django)</b> Criação correta dos modelos	OrdemServico	<b>2</b>
	Patrimonios	<b>2</b>
	Ambientes	<b>2</b>
	Manutentores	<b>2</b>
	Históricos	<b>2</b>
	Responsaveis	<b>2</b>
	Gestores	<b>2</b>
	Relações apropriadas (ForeignKeys) e validações.	<b>10</b>
<b>API Rest (Django Rest Framework)</b>	Implementação dos endpoints <b>CRUD</b> para todas as páginas que possua dados. Observação: não precisa ter modal.	<b>20</b>
	Incluir filtros pelo menos em Ordem de Serviço e Patrimônio.	<b>5</b>
<b>Consumo da API (Axios e React)</b>	Comunicação correta entre o front end e o back end usando Axios para listar, criar e atualizar OS.	<b>10</b>
<b>Funcionalidades Extras 1</b>	Implementação de <b>histórico</b> e <b>exportação</b> de relatórios em Excel(XLSX ou CSV), alertas e notificações para mudanças no status das OS.	<b>10</b>
<b>Funcionalidades Extras 2</b>	Desenvolvimento de código para popular o banco a partir de planilhas disponibilizadas.	<b>5</b>
<b>Organização do Código e Boas Práticas</b>	Estrutura do código, modularidade e organização do código Django.	<b>6</b>

**Critérios de Avaliação – Front End (React)**

<b>Critério</b>	<b>Descrição</b>	<b>Peso (%)</b>
<b>Interface e Experiência do Usuário (UX/UI)</b>	O design da aplicação é intuitivo e de fácil navegação. Botões, formulários e listagens são bem organizados.	15%
<b>Funcionalidade e Dinâmica</b>	As funcionalidades implementadas atendem aos requisitos do sistema. O usuário consegue cadastrar, visualizar e atualizar ordens de serviço sem erros.	20%
<b>Página Inicial</b>	Desenvolva a página Home com todas as opções de navegação.	10%
<b>Integração com Back End (API Django)</b>	Consumo da API via Axios, envio e recebimento de dados corretamente. Os estados da aplicação são bem gerenciados.	10%
<b>Cabeçalho e Rodapé</b>	Desenvolva apenas 1 cabeçalho e 1 rodapé para todas as páginas, exceto login e cadastro de usuário. O título de cada cabeçalho deverá ser de acordo com a página, exemplo: página de Ordem de Serviço deverá ter o mesmo tema.	15%
<b>Código e Organização</b>	Código limpo, bem estruturado, uso adequado de componentes reutilizáveis e boas práticas de desenvolvimento.	15%
<b>Autenticação e Controle de Acesso</b>	Implementação correta do login/logout, proteção de rotas e armazenamento seguro do token JWT.	15%

**Nota** = BackEnd x 0,8 + FrontEnd x 0,2

**Aulas Não Presenciais**

<b>Nota</b>	<b>Presença</b>
<b>0 - 33</b>	<b>5</b>
<b>34 - 66</b>	<b>10</b>
<b>67 - 100</b>	<b>15</b>