Como o CRUD Acontecerá para User, Post e Tag

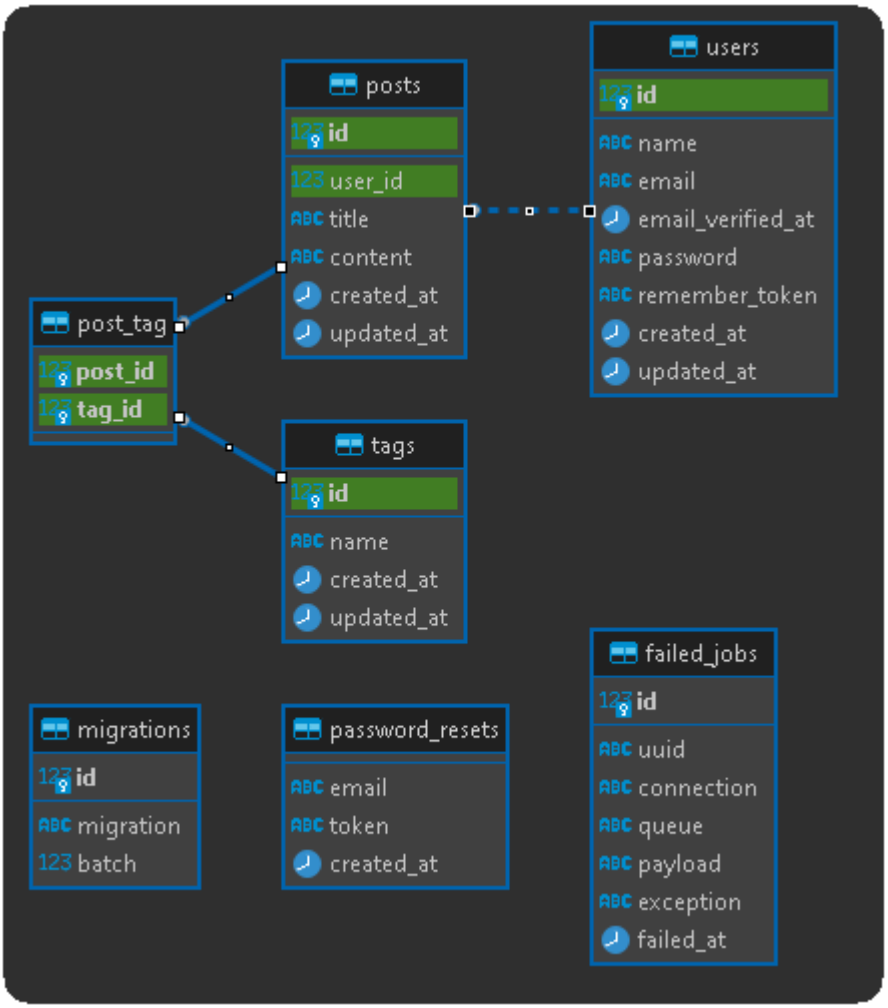
Mapa:

Método HTTP	Rota	Função Controller	Ação
GET	/api/users	`index`	Listar todos os usuários
GET	/api/users/{id}	`show`	Exibir os detalhes de um usuário específico
POST	/api/users	`store`	Criar um novo usuário
PUT	/api/users/{id}	`update`	Atualizar os dados de um usuário específico
DELETE	/api/users/{id}	`destroy`	Excluir um usuário específico
GET	/api/posts	`index`	Listar todas as postagens
GET	/api/posts/{id}	`show`	Exibir os detalhes de uma postagem específica
POST	/api/posts	`store`	Criar uma nova postagem
PUT	/api/posts/{id}	`update`	Atualizar os dados de uma postagem
DELETE	/api/posts/{id}	`destroy`	Excluir uma postagem específica
GET	/api/tags	`index`	Listar todas as tags
GET	/api/tags/{id}	`show`	Exibir os detalhes de uma tag específica
POST	/api/tags	`store`	Criar uma nova tag
PUT	/api/tags/{id}	`update`	Atualizar uma tag existente
DELETE	/api/tags/{id}	`destroy`	Excluir uma tag específica

Com o apiResource, as rotas correspondentes às operações de CRUD para **User**, **Post** e **Tag** são automaticamente geradas e protegidas pela autenticação **JWT** (exceto no end point **User** para login e cadastro) mais detalhe JWT verifique última página.

É aconselhável realizar a inserção do **CRUD** nessa ordem (**User**, **Post** e **Tag)** , pela entidade e relacionamento do banco, obedecendo a existência de dados no relacionamento de chave estrangeira.

Diagrama entidade relacionamento



Users

id (PK) - Chave primária **name** - Nome do usuário **email** - Email do usuário **password** - Senha criptografada **created_at** - Data de criação **updated_at** - Data de atualização

Posts

id (PK) - Chave primária **title** - Título da postagem **content** - Conteúdo da postagem **user_id (FK)** - referenciando o users(id) **created_at** - Data de criação **updated_at** - Data de atualização

Tags

id (PK) - Chave primária **name** - Nome da tag **created_at** - Data de criação **updated_at** - Data de atualização

Post_tag

post_id (FK) referenciando o posts(id) **tag_id (FK)** referenciando o tags(id) **(PK | post_id e tag_id)** - Chave primária composta

Regras do negócio API

User (Usuário)

- **GET - *<u>/api/users</u>*** – **Listar todos os usuários**
  - A rota executa a função index do UserController, retornando uma lista de todos os usuários no banco de dados.
  - URL: http://localhost:8000/api/users

- **GET - *<u>/api/users/{id}</u>*** – **Exibir um usuário específico**
  - A rota executa a função show do UserController, retornando os detalhes de um usuário específico, identificado pelo seu id.
  - URL: http://localhost:8000/api/users/1

- **POST - *<u>/api/users</u>*** – **Criar um novo usuário**
  - A rota executa a função store do UserController, criando um novo usuário com os dados fornecidos no corpo da requisição (payload).
  - URL: http://localhost:8000/api/users
  - Cabeçalhos (Headers): Content-Type: application/json
  - Ex.: Schema (Dados de entrada) - JSON:
  - { "name": "João Silva", "email": "joao@example.com", "password": "senha123" }

- **PUT - *<u>/api/users/{id}</u>*** – **Atualizar dados de um usuário**
  - A rota executa a função update do UserController, atualizando as informações de um usuário existente, identificando-o pelo id.
  - URL: http://localhost:8000/api/users/1
  - Cabeçalhos (Headers): Content-Type: application/<u>json</u>
  - URL: http://localhost:8000/api/ users /1
  - Ex.: Schema (Dados de entrada) - JSON:
  - { "name": "João Silva2", "email": "joao@update.com", "password": "senhaupdate" }

- **DELETE - *<u>/api/users/{id}</u>*** – **Excluir um usuário**
  - A rota executa a função destroy do UserController, excluindo um usuário específico pelo id.
  - URL: http://localhost:8000/api/users/1

Tag (Tag)

- **GET *<u>/api/tags</u>*** – **Listar todas as tags**
    - A rota executa a função index do TagController, retornando uma lista de todas as tags no banco de dados.
- **GET *<u>/api/tags/{id}</u>*** – **Exibir uma tag específica**
    - A rota executa a função show do TagController, retornando os detalhes de uma tag específica identificada pelo seu id.
    - URL: http://localhost:8000/api/tags/1
- **POST *<u>/api/tags</u>*** – **Criar uma nova tag**
    - A rota executa a função store do TagController, criando uma nova tag com os dados fornecidos, como o nome da tag.
    - Ex.: Schema (Dados de entrada) - JSON:
    - { "**name**": "Laravel" }
- **PUT *<u>/api/tags/{id}</u>*** – **Atualizar uma tag existente**
    - A rota executa a função update do TagController, atualizando uma tag existente identificada pelo id.
    - URL: http://localhost:8000/api/tags/1
    - Ex.: Schema (Dados de entrada) - JSON:
    - { "**name**": "Laravel_updated" }
- **DELETE *<u>/api/tags/{id}</u>*** – **Excluir uma tag**
    - A rota executa a função destroy do TagController, excluindo uma tag específica identificada pelo id.
    - URL: http://localhost:8000/api/tags/1

Post (Postagem)

- **GET - *<u>/api/posts</u>*** – **Listar todas as postagens**
    - A rota executa a função index do PostController, retornando uma lista de todas as postagens no banco, incluindo as tags associadas (usando o with('tags') para carregar as tags relacionadas).
- **GET - *<u>/api/posts/{id}</u>*** – **Exibir uma postagem específica**
    - A rota executa a função show do PostController, retornando os detalhes de uma postagem específica, incluindo suas tags.
    - URL: http://localhost:8000/api/posts/1

- **POST *<u>/api/posts</u>*** – **Criar uma nova postagem**
    - A rota executa a função store do PostController, criando uma nova postagem. O corpo da requisição deverá incluir os dados necessários, como title, content, user_id e, opcionalmente, as tags.
    - Ex.: Schema (Dados de entrada) - JSON:
    - { "**title**": "Meu primeiro post", "**content**": "Este é o conteúdo da postagem.", "**user_id**": 1, "**tags**": "**1, 2**"}

- **PUT *<u>/api/posts/{id}</u>*** – **Atualizar uma postagem existente**
    - A rota executa a função update do PostController, atualizando uma postagem existente identificada pelo id. A requisição pode incluir novas tags associadas à postagem.
    - Ex.: Schema (Dados de entrada) - JSON:
    - { "title": "Meu primeiro post update", "content": "Este é o conteúdo da postagem update.", "user_id": 1, "tags": "**3, 1**" }

- **DELETE *<u>/api/posts/{id}</u>*** – **Excluir uma postagem**
    - A rota executa a função destroy do PostController, excluindo a postagem identificada pelo id.
    - URL: http://localhost:8000/api/posts/1